Immediate Feedback as a Supporting Tool in a Web Text-handling Support System

By Nasiroh Omar,
B.Sc (University of Binghamton, USA), M.Sc (Universiti Kebangsaan Malaysia, Malaysia)

Thesis submitted to the University of Nottingham for the degree of Doctor of Philosophy,
November 2008
Abstract

This research investigates the viability of having immediate feedback as a supporting tool in a Web Text-handling Support System. The system provides learners in higher education with an environment that allows them to perform their Web text-handling activities such as search the Web and write free-response texts based on the given Web text-handling task. On request, their Web text-handling outcomes are evaluated and the feedback is presented to them in real-time. Currently, existing Web text-handling applications have not yet implemented immediate feedback as a supporting tool in a real-life task setting. This research is important in determining the usability and accessibility of the proposed system specifically in making use of real-time data (which are produced by learners in a real-life setting) and providing immediate constructive feedback as a support tool in Web text-handling.

This thesis presents research on identifying the complexities of a Web text-handling task. It produces two models of Web text-handling activities. These models have been used as the bases for the Web Text-handling Support System implemented as part of the research. The system is capable of capturing the Web text-handling processes and output and converting them to real-time data. The data is evaluated along two dimensions: verbatim quotient detection and Latent Semantic Analysis. The result of the evaluation is presented to the learners in the form of text and graphical representations. It consists of five items: the total number of words in the rewritten text; total number of words copied verbatim; percentages of the verbatim words taken from each of the extracted Internet resources; Latent Semantic Analysis of sentence-to-sentence cohesion in the rewritten text; and Latent Semantic Analysis of document-to-document similarity of the rewritten text to the given Internet research task.

Evaluation results of the Web Text-handling Support System indicate that the system is usable for capturing, assessing, and providing immediate feedback based on the learners’ real-time data in a real-life setting. This research finds that learners were more controlled of the number of verbatim copied words in their free-response text when they were informed of the existence of the verbatim copying detection mechanism and presented with e-feedback. In addition, this research proves that the Latent Semantic Analysis technique in the proposed system is usable for determining the quality of learner’s free-response texts in terms of its sentence-to-sentence coherence and its similarity with the given Web text-handling task, i.e. in the domain of an informal, unstructured and unconstrained type of free-text automatic assessment.
ACKNOWLEDGEMENTS

The preparation of this thesis has been a long journey for me that began with opportunities, inspired by curiosities, and lead by destiny. Along the way, I often encountered excitement, sometimes challenges, but always interesting experiences. It has been made possible by many other people, who have supported me and were involved in one way or another. I am extremely delighted to have this opportunity to express my gratitude to all of them. I attempt my best and apologize dearly for those whom I omit.

I would like to thank the University Technology Mara, Malaysia for giving me an opportunity to further my studies by providing financial support. Next, a ‘big thank you’ to my first supervisor, Dr. Colin Higgins, head of the Learning Technologies Research Group at the University of Nottingham (whom I first encountered through my Web search one year before I started my PhD) who was willing to accept me as his research student, offer me an opportunity to work with my second supervisor, Professor Colin Harrison, Professor of Literacy at the School of Education, University of Nottingham, and provide me with constant support and ideas. Then, another ‘big thank you’ to my second supervisor for giving me an opportunity to be his first research student in the area of Web-text handling research, who redirected my thinking towards the world of online reading comprehension and brought new dimensions to my professional culture. Then, a ‘big thank you’ to Emeritus Professor David Brailsford at the School of Computer Science, University of Nottingham who provided me twice with opportunities to evaluate my prototype systems with his first year Computer Science students (year 2005 and year 2006). Last but not least, a ‘big thank you’ to Dr. Boriana Koleva for supporting and helping me during my difficult moments, of which I could not repay but will always treasure as part of my PhD achievement.

I would like to acknowledge the support of Viktor Huddleston, Nick Reynolds, Roger Lee, Francois Rossard, Deigo Campo Milan and all the staff at the School of Computer Science Information Technology, University of Nottingham.

I would also like to thank Swe Myo Htwe, Marzieh Ahmadzadeh, Maha Al-Yahya, Marjahan Begum, Brett Bligh, Geoffrey Gray and other current members of Learning Technology Research Group, University of Nottingham with whom I shared plenty of memorable moments.

I would like to extend my special thanks to Joyce Cox who helped me by proofreading my thesis writing.

Lastly, I am ever-thankful to my family for their patience, support (mental, physical and emotional), inspiration, encouragement and understanding: my husband Afifi, my parents Omar and Minah, my children Alia, Sofia, Alif, Aifa and Akif (my PhD baby). This success belongs to all of us.
Table of Contents

Abstract ........................................................................................................................... ii

Acknowledgements........................................................................................................ iii

Table of Contents............................................................................................................iv

List of Figures .................................................................................................................iv

List of Tables...................................................................................................................xi

1. Introduction .................................................................................................................. 1-11
   1.1 Introduction ........................................................................................................... 1
   1.2 Motivation ............................................................................................................. 2
   1.3 Objectives ............................................................................................................. 4
   1.4 Scope .................................................................................................................. 5
   1.5 Approach ............................................................................................................. 7
   1.6 Contributions ...................................................................................................... 8
   1.7 Synopsis of the Dissertation .............................................................................. 9

2. Research Background ............................................................................................... 12-47
   2.1 Introduction ......................................................................................................... 12
   2.2 E-Learning, E-Assessment and E-Feedback ....................................................... 13
   2.3 E-Assessment ...................................................................................................... 14
      2.3.1 An Overview .................................................................................................... 14
      2.3.2 E-Assessment of Essays: A Review ............................................................. 16
         2.3.2.1 PEG™ ...................................................................................................... 17
         2.3.2.2 Criterion™ ............................................................................................... 18
         2.3.2.3 Summary Street™ .................................................................................. 19
         2.3.2.4 BETSY™ ................................................................................................. 22
         2.3.2.5 Discussion ............................................................................................... 23
      2.3.3 E-Assessment of Plagiarisms: A Review ................................................... 27
         2.3.3.1 Search-Based Mechanism ....................................................................... 28
         2.3.3.2 Linguistic Analysis .................................................................................... 30
4.4.1.2 Access to a Web Browser .........................................................79
4.4.1.3 Access to the Assessment Mechanisms of Web Text-handling ........................................80

4.4.2 Real-time Data Collection Component ..................................................80
4.4.2.1 Type of Data to be Collected .........................................................81
4.4.2.2 Identifying a Technique to Collect the Data ....................................81
4.4.2.3 Determining a Technique to Process the History File ......................82
4.4.2.4 Obtaining Personalized Data ..........................................................83

4.4.3 Assessment Mechanisms in Web Text-handling Platform Component ..........................................................84
4.4.3.1 Assessment of the Written Essay ....................................................85
4.4.3.2 Assessment of the List of URL Addresses and their Content ..............86

4.5 The Requirements for Designing Usable System ........................................87
4.6 Summary ........................................................................................................91

5. The Development of the WTSS System ....................................................... 92-141

5.1 Introduction ....................................................................................................92
5.2 General Requirements for the Web Text-handling Support System ..........93
5.3 Design Specification for Learners’ and System Activities ..........................95
5.4 Building a UML Activity Diagram for the Web Text-handling Task .........97
5.5 A Comprehensive Design Plan for the WTSS System .................................99
5.6 A Comprehensive Implementation Plan for the WTSS System .................101
5.6.1 Overview of Data Flow within the WHTS System ...............................101
5.6.2 The Chosen Language for Implementation .........................................104
5.7 The Development of the WTP System .......................................................105
5.7.1 The First Design and Implementation of the WTP System .................106
5.7.2 The Second Design and Implementation of the WTP System ............113
5.8 The Collection of Learner’s Real-time Data ...............................................122
5.9 The Integration of the VQD System ............................................................124
5.9.1 The Distribution of the Functions .........................................................128
5.9.2 The Implementation of the Functions ....................................................130
5.10 The Integration of LSA System .................................................................131
5.10.1 The Distribution of the Functions .........................................................134
5.10.2 The Implementation of the Functions .........................................................138
5.10.3 The Development of Corpus for the LSA System ................................139
5.11 Summary .........................................................................................................140

6. Evaluation ........................................................................................................142-169

6.1 Introduction ....................................................................................................142
6.2 The First System Evaluation ........................................................................144
   6.2.1 Participants .............................................................................................144
   6.2.2 Task Design ............................................................................................145
   6.2.3 Procedures ...............................................................................................146
   6.2.4 Data ..........................................................................................................147
   6.2.5 Results .....................................................................................................148
     6.2.5.1 Participants’ Web Text-handling Strategies .........................................148
     6.2.5.2 Verbatim Quotient Detection .............................................................149
   6.2.6 Findings ....................................................................................................151
     6.2.6.1 From Learners’ perspective ...............................................................151
     6.2.6.2 From Research Perspective ...............................................................152
6.3 The Second System Evaluation ....................................................................153
   6.3.1 Participants .............................................................................................154
   6.3.2 Results .....................................................................................................154
     6.3.2.1 Participants’ Web Text-handling Strategies .........................................155
     6.3.2.2 Verbatim Quotient Detection .............................................................156
     6.3.2.3 LSA ....................................................................................................157
   6.3.6 Findings ....................................................................................................165
     6.3.6.1 From Learners’ perspective ...............................................................166
     6.3.6.2 From Research Perspective ...............................................................167
6.4 Conclusions ....................................................................................................167
6.5 Summary .........................................................................................................168

7. Conclusions and Future Research ..................................................................170-181

7.1 Introduction ....................................................................................................170
7.2 Meeting the Objectives ................................................................................170
7.3 Contributions ................................................................................................174
   7.3.1 World Wide Web ......................................................................................174
# Table of Contents

7.3.2  Learning Technology................................................................. 175
7.3.3  Software Design ................................................................. 175

7.4  Dissemination ............................................................................ 176

7.5  Future Research ........................................................................ 178
   7.5.1  Learning Technology ......................................................... 178
   7.5.2  World Wide Web ............................................................. 179

7.6  Closing Remarks ....................................................................... 180

**References** .................................................................................. 182-192
## List of Figures

1.1  Research Scope........................................................................................................6  
3.1  The setting for Web text-handling.................................................................52  
3.2  The data analysis process ...................................................................................53  
3.3  A model of learner’s behaviour in task-driven Web text-handling...............58  
3.4  The model of processes involved in Web text-handling ...............................61  
3.5  The model of a Web text-handling scenario (the offline materials are derived  
from literature, the online materials are derived from section 3.2).................65  
4.1  Identified parts of Web Text-handling Support System .................................76  
4.2  The screen of Web Text-handling Support System .......................................77  
5.1  Use cases diagram for Web text-handling scenario adapted from FREMA and  
(Millard et al., 2006).........................................................................................94  
5.2  The factoring of the learner use case into a list of learner and system  
activities...........................................................................................................96  
5.3  UML Activity Diagram for the Web Text-handling Support System (WTSS)........98  
5.4  A comprehensive design plan of the WTSS system ...................................99  
5.5  The implementation plan of WTSS System ...............................................101  
5.6  The flow of internal and external data within WTSS system .......................102  
5.7  The packages and stand alone system in the WTSS system .......................103  
5.8  The conceptual design and implementation of WTP system’s screens .......107  
5.9  The first screen in WTP system .................................................................108  
5.10a-e Screens of of the implemented WTP system ........................................112-110  
5.11  The second conceptual design and implementation of WTP system’s screen.114  
5.12  The second screen of the WTP system .......................................................115  
5.13  The ‘Read Me’ screen of the WTSS system .............................................116  
5.14  The ‘Step 1’ screen of the WTSS system ...................................................117  
5.15  The ‘Step 3’ screen of the WTSS system ...................................................118  
5.16  The ‘Step 4’ and ‘Step 5’ screen of the WTSS system .................................119  
5.17  The ‘Step 6’ screen of the WTSS ...............................................................120  
5.18  The process of collecting learners’ real-time data which are captured by the WTP  
and RDC sub-systems in the WTSS system ..................................................124
List of Figures

6.1 Web text handling activities performed by learners in the first evaluation...
6.2 Shows the number of unique URL addresses visited by the learners in the study, with their frequency in the first system evaluation
6.3 Shows the number and frequency of different keywords used in navigation (i.e. most words were used once only; fewer than 20 were used by two participants, etc.) in the first system evaluation
6.4 Graph shows participants with the percentages of words copied in their rewritten text in the first system evaluation
6.5 Shows the number of unique URL addresses visited by the participants in the study, with their frequency in the second system evaluation
6.6 Shows the number and frequency of different keywords used in navigation (i.e. most words were used once only; fewer than 20 were used by two participants, etc.) in the second system evaluation
6.7 Graph shows participants with the percentages of words copied in their rewritten text in the second system evaluation
6.8 Bar graph showing the percentage of participants in different ranges of percentage differences between human marker’s marks and their projected marks after application of four different formulae in the second system evaluation
6.9 Line graph showing the percentage of participants in different ranges of percentage differences between human marker’s marks and their projected marks after application of four different formulae in the second system evaluation
6.10 Bar graph showing the percentage of participants in different ranges of percentage differences between human marker’s marks and their projected marks after application of four different formulae in the first system evaluation
6.11 Line graph showing the percentage of participants in different ranges of percentage differences between human marker’s marks and their projected marks after application of four different formulae in the first system evaluation
List of Tables

2.1  A Summary of the Characteristics of Automatic Writing Assessment Tools ....24
3.1  Samples of evidence provided by the computer to reflect a learner’s activities in handling Web text.................................................................69
6.1  The highest, lowest, average and standard deviation Verbatim Quotient Detection scores for all the participants in experiment one (n= 58) ..................150
6.2  The highest, lowest, average and standard deviation Verbatim Quotient Detection scores for all the participants in experiment two (n= 59) .......................156
Chapter 1

Introduction

1.1 Introduction

The Web has been able to provide students with instant and global access to a massive amount of information. Being a prominent information provider, the Web has established search engines as a major tool for finding information and online documents as a dominant medium of delivery (Tombros et al., 2005). The students’ success in seeking the right text for the right task is based on how they interact with Web tools and the Web medium (Belkin, 1993). Hence, one important skill for the student to possess is the information-seeking skill, which is defined for the purpose of this research as being able to seek, comprehend and use a Web document efficiently.

Unfortunately, in order to acquire the skill, students have to do many task-driven information-seeking exercises and each time produces a free-response short essay or summary. It has been previously shown that by producing such a document, while or after reading, people comprehend better (Duke and Pearson, 2002, Graham and Perin, 2007). In reality, the skill is informally acquired by the students and is dependent on their own initiative to engage in such exercises. However, they tend to constantly face with one repetitive problem, i.e. handling the Web text in written form. In this thesis, handling text is defined as whenever the students want to use texts (in written form) from a Web document, they have to decide either to summarize or cut-and-paste the content in order to accomplish their task.
Therefore, to aid students in this activity, real-time support could be provided. In order to assist real-life exercises, this support ought to be in the form of automated and personalized feedback during or after the exercise, which is referred to as e-feedback. E-feedback, as defined in FREMA (the e-learning framework reference model for assessment), involves “itemized and personalized information to guide”. It may provide students with information such as “whether their response is correct or incorrect, present a score, give suggestions on why they might have chosen that response or point to remedial materials”\textsuperscript{1}.

This thesis presents exploratory research of the application of e-feedback in the domain of Web-based information-seeking. This work involves research and system analysis, design, implementation and evaluation of a support system that facilitates automated feedback in the described domain.

This chapter presents the introduction of the thesis. It highlights the motivation and scope, and gives an overview of this research. The synopsis outlines the structure of this dissertation and how it corresponds to later chapters.

### 1.2 Motivation

The fundamental motivation for this research has been to gain a better understanding of how task-driven students normally handle text from the Web and to explore the possibilities of how e-feedback could be used in the context of such informal and

\textsuperscript{1} FREMA Web page – [http://frema.ecs.soton.ac.uk/wiki/](http://frema.ecs.soton.ac.uk/wiki/) <last accessed: 14/02/2008>
unstructured type of knowledge acquisition – specifically Web-based information-seeking.

The second motivation for this work has been to examine the application of the Latent Semantic Analysis (LSA) technique in the context of supporting students in handling Web text, in a Web-based information-seeking setting. LSA is “a theory and method for extracting and representing the contextual-usage meaning of words by statistical computations applied to a large corpus of text” (Landauer et al., 1997). LSA offers a promising possible solution for supporting students in a Web information-seeking task due to its ability to act as an effective tool for scoring and commenting on a written essay (Foltz et al., 2000) by predicting text coherence and resulting comprehension and simulating human judgment on the similarity between two documents (Kintsch et al., 2000).

The third motivation for the current research has been to examine the ‘copy-and-paste’ type of plagiarism in the context of supporting students in handling Web text. In this thesis, cut-and-paste plagiarism is defined as word-for-word copying or using exactly the same single word. Therefore, this research aims to investigate whether ‘cut-and-paste’ plagiarism detection is usable by task-driven students. Can it be employed in a beneficial way to support the students in how they handle the text retrieved from the Web?

These research questions have not been explored by previous research. Advances in computer hardware and, more importantly, advances in computer software facilities now
make it possible to address the aims of this research – to investigate the complex processes of a Web-based information-seeking exercise, and to make use of the processes and output as real-time data, captured in a real-life setting.

1.3 Objectives

A key objective of the present research is to investigate how task-driven students in higher education handle Web texts in Web-based information-seeking activities and to produce a model that would facilitate the design of a support system that is able to present e-feedback in real-time, in a real-life setting. The thesis research aims to answer the following key questions:

- Is it possible through exploratory and informal investigation to develop a computer-based model to be used as the basis for handling Web texts in Web-based information-seeking activities?
- Can the model be used to identify the requirements for the development of the support tool system?
- To what extent can the system be used to achieve the key objective of this research, i.e. to apply e-feedback in the domain of Web-based information-seeking?

To answer these questions the thesis research must achieve the following specific goals:

- Develop a model to conceptualize information-seeking in the Web-based domain. To achieve this the Goal-Directed Approach will be adopted, which is an informal method of software design developed by Cooper (Cooper, 1999).
• Identify real-time data to be used as input for the e-feedback.

• Use two mechanisms, LSA and cut-and-paste plagiarism detection, as tools to analyze the real-time data in order to generate items in the e-feedback.

• Demonstrate how e-feedback can be incorporated into a support tool system by using the Goal-Directed Approach.

• Evaluate the proposed support system.

1.4 Scope

This thesis draws upon and contributes to the theory and techniques in three areas: The World Wide Web, Learning Technology and Software Design. Figure 1.1 illustrates a high-level view of the scope of this work.
In the Learning Technology domain, the work focuses on the area of providing instant and personalized feedback to students in order to enhance their information-seeking skill.

In the World Wide Web domain, the research focuses on supporting task-driven students in the area of Web-based information-seeking. Two issues are taken into consideration in providing the support. Firstly, how relevant their written texts (as the output of their information-seeking session) are to their task; secondly, how many words have been cut-and-paste plagiarised from the Web resources.
From the Software Design area, the work uses the Goal-Directed Approach to support two research activities: to investigate the students’ information-seeking behaviour; and to analyze, design and implement the information-seeking support system.

1.5 Approach

Initially, an exploratory experiment was designed to enable experienced information seekers to demonstrate their information-seeking skill in a real-life setting. The study involved three types of concurrent tasks: seeking, reading and using the Web text. Based on the collected data in the study, four archetypal student profiles (as information seekers) were developed. Following the Goal-Directed Approach methodology the students were orchestrated in a fictional information-seeking scenario. This scenario was crucial in identifying the information-seeking processes and output. As a result, a new model of Web-based information-seeking was developed. The model is used as the basis for identifying the requirements to develop the support system.

The subsequent stage of the thesis research used the requirements to confidently identify sub-components of the targeted system. The design and implementation of the system were divided into four stages to enable iterative development based on user feedback. The first stage was to enable users to perform the three distinct information-seeking processes (read the task, seek for information on the Internet and write about the found information) through a single environment. The second stage involved capturing and processing relevant real-time data into meaningful information, in a real-life setting. This data would be used in the subsequent stage. The third stage dealt with cut-and-paste
plagiarism. The work involved getting visited URLs from the Internet browser’s history file, retrieving their content, and comparing with the student’s written text in real-time. At this stage, the interface failed to achieve good interactive design and the system did not meet e-feedback criteria in several key aspects, but the experiment was crucial in identifying the shortcomings of the current system, to be considered and modified in the next stage. Finally, the fourth stage modified the interface, built LSA and embedded LSA into the existing system. In this stage, all items in the e-feedback were made available to the students.

Two evaluations of the support system were carried out during the development process – in between stages two and three, and between stages three and four. The evaluations were crucial in improving the system in terms of its screen design, accessories, and performance.

1.6 Contributions

The primary contribution of this thesis is in the area the World Wide Web. Providing e-feedback in real-time while an information seeker is handling Web text in a real-life setting is new to the Web-based information-seeking field. In addition, the initial work of modelling the complex user behaviour during Web-based information-seeking contributes to the understanding of its processes and output. The contribution includes the application of LSA and the cut-and-paste plagiarism detection techniques to a new domain, i.e. Web-based information-seeking. Both techniques use real-time data obtained in a real-life
setting. The contribution has opened various possibilities of where this research could be taken forward, and these are documented in Chapter 7.

The second contribution is in the area of Learning Technology. This research has applied e-feedback to a new domain, where current assessments of the information-seeking skills of student do not address Web text-handling, but focus only on how students search the Web text (e.g. evaluated through Multiple Choice Questions). This research provides a system for assessing Web-based information-seeking skills, focusing on Web text-handling and dealing with real-time data, obtained in a real-life setting. The real-time data includes both information-seeking processes and output.

The third contribution is to the Goal-Directed Approach, which is used for system analysis and design. The thesis work has applied the approach prior to the system analysis and design stage, i.e. in modelling the complex behaviour of Web-based information-seeking activities. This has shown that it is possible to use the Goal-Directed Approach not only to analyse and design a system but also to design a model for complex human behaviour.

1.7 Synopsis of the Dissertation

This chapter presented the motivation, objectives, scope, approach and contributions of the thesis research, the focus of which has been the exploration of the use of e-feedback on handling Web text for information seekers, and its realisation as a support tool system to be used in a real-life setting.
Chapter 2 introduces the background and key concepts in the area of automated feedback for supporting Web-based information seekers in handling Web text. In addition, it presents an overview of the two mechanisms chosen to be used in generating items for e-feedback: LSA and cut-and-paste plagiarism detection. Further, it introduces the Goal-Directed Approach as a design method for the support system. The chapter concludes with a specification for the research problem.

Chapter 3 describes how the design approach has been used in modelling Web-based information-seeking behaviour. Further, it explains how the model has been simplified to enable standard computers to capture the real-time data during the information-seeking session. This model is used in the next chapter as the basis for identifying the requirements of the proposed support system.

Chapter 4 describes the proposed support system and identifies four minimal requirements which will enable the information seeker to perform a Web-based information-seeking task in a unified environment. These minimal requirements include an appropriate system environment for information-seeking activities to take place, a method of collecting its real-time data in terms processes and output, and two chosen mechanisms to assess the collected data in order to produce items for e-feedback.

Chapter 5 reports on a prototype that implements four staged designs. Each design becomes an independent sub-system. The first sub-system sets up a platform for the information-seeking to take place. It provides basic facilities for the task to be displayed
on the screen, to launch the Internet Explorer browser and a text pane for writing. The second sub-system allows the real-time data to be collected and processed upon the information seeker’s request. The third sub-system detects the cut-and-paste plagiarism. Finally, the fourth sub-system provides feedback about sentence-to-sentence coherence in the written text (by the information seeker) and document-to-document similarity between the written text and the task. This chapter documents the role, implementation, and interfaces of each sub-component.

Chapter 6 reports the results and discussions of the prototype support system based on two evaluations. The system has been evaluated by two groups of first-year students at the Department of Computer Science, University of Nottingham. The support system has demonstrated considerable accomplishments in the objectives that have been set out in Chapter 1.

In summary, this thesis supports the argument that the proposed model of Web-based information-seeking behaviour and the development of the e-feedback system are useful and can be used as the basis for future research. Chapter 7 highlights these claims by reviewing the objectives of the thesis as set out in Chapter 1 and Chapter 4, and links the results given in Chapter 6. It also discusses the contributions of this research to the fields of the World Wide Web, Learning Technology and Goal-Directed Approach, while indicating possible extensions for future work.
Chapter 2

Research Background

2.1 Introduction

This chapter presents the background for the present research along three dimensions – providing automated feedback to learners, supporting learners in Web-based information-seeking and applying the Goal-Directed Approach in software engineering for developing the automated feedback support system. The three dimensions provide theory and technology which are useful for developing an e-feedback system as a support tool for learners to promote their skill in handling Web text. More specifically in:

- The field of automated feedback by drawing out techniques for assessing learners’ free-response texts which they obtain through handling Web text and types of possible feedback in relation to the technique of assessments.
- The field of Web-based information-seeking by drawing out possible techniques to support learners while handling Web text in a real-life setting.
- The field of software engineering by drawing out an approach to developing a usable automatic feedback support system.

The chapter starts with an introduction to e-learning, e-assessment and e-feedback. It then presents a brief description of the areas where e-assessment is being applied, focusing on e-assessment of essays, Web-based plagiarism and information seekers. Furthermore, this chapter explains the motivation for developing a constructive e-feedback system, its advantages and barriers. Next, this chapter presents a development approach to increase
system usability. This chapter concludes with a specification of the research problem and a chapter summary.

2.2 E-Learning, E-Assessment and E-Feedback

In the broadest sense, *e-learning* can be best described as *using technology in various areas which include e-assessment and e-feedback to enhance learning*. The term **e-assessment** is used in this thesis in relation to its context in the e-learning area. As defined by Millard et. al. **e-assessment** is *“the process of assessment facilitated by a machine”* which *“includes not only runtime assessment systems, but also planning, quality, analysis, grading and feedback tools”*. In addition, they claim e-assessment is one of the more established areas in the e-learning domain. However, Conole and Warburton who present a detailed review of e-assessment conclude that e-assessment is still in its infancy and is in need of further exploration (2005). This is mostly due to the lack of common tools and standards for e-assessment research, which leads to the existence of a heterogeneity of tools and standards adopted in the area (Millard et al., 2006).

E-feedback is a term tightly related to e-assessment. The term denotes a response or an outcome of an e-assessment process, of which both e-assessment and e-feedback are presented to the learners via computer systems. Basically, Bligh defines **e-feedback as the integral part of the formative e-assessment process due to its central aim to enhance a continuous learning** (Bligh, 2006). Hence, the purpose of having e-feedback is to add a substantial meaning to e-assessment. Malmi et. al. claim that providing e-feedback is far more meaningful to learners compared to providing them with just model solutions. They
report that learners benefit when they are allowed to resubmit their work based upon provided e-feedback (Malmi et al., 2002).

However, there is always a technology gap between e-learning and e-assessment (MacKenzie, 2003, Buzzetto-More and Alade, 2006, Ashton and Thomas, 2006) due to an imbalance in technology advancement in the two areas. The gap is even wider between e-assessment and e-feedback since e-feedback has not been fully implemented within existing e-assessment systems (Lilley et al., 2005, Nicol and MacFarlane-Dick, 2006). This phenomenon is the result of most of the development in e-feedback research taking place only after a lot of the research in e-assessment had been accomplished.

2.3 E-Assessment

2.3.1 An Overview

Starting from the early 1990s, most early attempts to develop e-assessment systems were focused on supporting the area of computer science subjects. Then e-assessment research began to address a wider range of science subjects including physics, mathematics and chemistry. Now e-assessment is applied in many areas of academic and non-academic learning (Bligh, 2006, Tsintsifas, 2002).

The earlier e-assessment systems (Denton, 2003, Bligh, 2006, Tsintsifas, 2002, Sukkarieh et al., 2003, Barker and Britton, 2005, Callear et al., 2001b, Callear et al., 2001a, Davies, 1999, Craig, 2007) can be characterised by three attributes. Firstly, the assessment processes were concentrated on the analysis and marking of the submitted works.
Therefore, the systems lacked feedback mechanisms. Secondly, the systems focused on assessment through simple types of questions, for example in the form of multiple-choice questions. The use of this type of question simplifies the marking processes as simple matching mechanisms can be applied, where the submitted answers are compared directly with the sample answers in order to find a match. Thirdly, the e-assessment systems were developed for the formal and structured type of e-learning areas in formal types of education, mostly in higher level education.

Current computer technologies and software systems allow advancement in the types of question which e-assessment systems can handle. Bligh reports that various types of question are employed in current e-assessment systems (Bligh, 2006). He describes two categories of e-assessment systems: fixed-response and free-response. The fixed-response e-assessment deals with multiple-choice, short-response or graphical hotspot types of question. On the other hand, free-response e-assessment deals with programming, essay or diagrammatic types of assignment.

The type of the e-assessment system has important implications for e-feedback. Firstly, in the case of fixed-response assessment, learners are expected for each question to choose or select from fixed lists of answers provided by the system. Thus, the answers are fully structured and predictable. This implies that the nature and content of e-feedback can be predicted and fully determined prior to the assessment exercises and structured in static form. As a result, the e-feedback items can be prepared and stored in a database. The e-assessment systems retrieve them when the feedback items are needed. Secondly, in the
case of free-response types of assignment, learners are expected to produce various possible answers within the scope of structured learning. Even though learners produce structured answers, the answers are less predictable and non-deterministic prior to the assignments. Thus, the outcomes of the e-assessment are flexible. These imply that the feedback items are very much dependent on the learners’ answers. In this case, providing feedback to the learners requires highly personalized items. As a result, the feedback items are not static and therefore less likely to be stored in database prior to the assessment process. Instead more complex mechanisms need to be employed by the system in order to dynamically generate appropriate feedback.

Since the outcome of a learner’s handling of Web text is in the form of free-response essay-like text, which is likely to include verbatim parts from Web documents, the next two sub-sections will focus on the techniques used for the e-assessment of essays and the e-assessment of plagiarism.

### 2.3.2 E-Assessment of Essays: A Review

An *e-assessment of essay* also known as ‘automatic essay scoring’ is defined in this work as a *computer application that is able to repeatedly perform formative e-assessment on written drafts*. Typically, e-assessment of essays does not employ direct mechanisms, as found in the traditional essay assessment by human graders. Instead, it adopts various assessment mechanisms which basically perform correlations of some intrinsic elements in the essay in order to produce a score for the essay and to provide immediate feedback based on the assessment.
This section reviews 4 approaches employed in the e-assessment of essays which are described in literature: PEG\textsuperscript{TM}, Criterion\textsuperscript{SM}, Summary Street\textsuperscript{TM} and BETSY\textsuperscript{TM}. The systems are chosen based on their popularity (Dikli, 2006, Kakkonen and Sutinen, 2004), and because they have at least one measure of computer-generated scores or they tend to match human expert scores (as frequently as two human scores match each other) (Miltsakaki and Kukich, 2000). Also the systems have been selected to illustrate different assessment methods, which is useful for comparison purposes in order to determine what techniques are appropriate to apply in the support tool developed by this thesis. Generally, each system will be discussed in relation to what assessment mechanism is used and what types of feedback are presented to learners.

2.3.2.1 PEG\textsuperscript{TM}

PEG\textsuperscript{TM} (Valenti et al., 2003, Dikli, 2006, Whittington and Hunt, 1999), the acronym for Project Essay Grader\textsuperscript{TM}, was developed by Ellis Page in 1966 and aimed to promote the effectiveness and practicality of large-scale essay scoring. It uses a statistical method called \textit{trins} and \textit{proxes}. \textit{Proxes} are defined as “computer approximations or measure of \textit{trins}”. \textit{Trins} refer to various intrinsic surface-features of an essay – those which human-graders normally look for while grading an essay. The idea behind this is to simulate the essay assessment attributes of a human grader’s behaviour. The measurements in \textit{proxes} include:

1. Essay length – to represent the intrinsic value of fluency.
2. Counts of prepositions, relative pronouns or other parts of speech – to represent the intrinsic value of the complexity of sentence structure.
3. Word length – to represent the intrinsic value for diction.
Generally, PEG\textsuperscript{TM} evaluates essays solely by using a statistical approach of *proxes* measurements based on the following assumption: “that the quality of an essay is reflected by the measurable proxes” (Valenti et al., 2003).

The major advantage of PEG\textsuperscript{TM} is that it shows positive effectiveness in such a way that it produces results comparable to those of human graders and detects surface-features errors in writing. However, there are three limitations in the method it employs: it lacks syntactic consideration, it lacks semantic measurements; and it has to use a set of human graded essays for the purpose of training the system prior to grading.

### 2.3.2.2 Criterion\textsuperscript{SM}

Criterion\textsuperscript{SM} is a Web-based system for e-assessment of essays, established since 1998 and developed by the ETS (Educational Testing Service) development team. It is aimed as assessing students’ written essays. It provides immediate formative feedback, along with its associative score report, on students’ writing skill, based on their writing style and content (Dikli, 2006, Whittington and Hunt, 1999, Burstein et al., 2001, Lucas, 2005).

Criterion\textsuperscript{SM} adopts a *Natural Language Processing* (NLP) method in its assessment tools. NLP is a complex subset of Artificial Intelligence and linguistics. In order to develop automatic writing assessment intelligence, NLP acquires knowledge by extracting linguistic attributes from large-text corpora. Normally, NLP works in two stages. The first stage, also known as the machine-learning stage, involves a data-entry process in order to train the system in identifying features in human-graded essays and storing these in its database. The second stage is a repetitive process (for each essay submission) where
the submitted essay is evaluated and scored by comparing features in human-graded essays (that have previously been stored in the database in the first stage) with the features in the submitted essay (Dikli, 2006).

Overall, the CriterionSM systems assesses and provides instant feedback in the form of holistic and diagnostic scoring both on the content and style of the submitted essay in the following areas: writing quality, organization and development of the writing, grammar usage, writing mechanics and writing style.

Although CriterionSM has advantages of assessing a wider range of writing areas, it has several disadvantages. Like PegSM, its major disadvantages are that it does not take into account the semantic aspect of the essay and its need of a collection of pre-graded essays for training purposes. Another disadvantage of CriterionSM is that the system costs a lot of money and time that need to be spent on a combination of automatic and manual processing prior to the marking session for any new writing task.

2.3.2.3 Summary StreetTM

Summary StreetTM, previously known as State of Essence (1997 – 1998) is a Web-based e-assessment system for free-response text in the form of summary, developed by the LSA Research Group2 since 1997 (Kintsch et al., 2000). However, its primary goal is to provide content-based support in writing and revision activities by fostering deep reading

---

2 The members of the LSA Research Group are Walter Kintsch, Thomas Landauer, Rogerio De Paula, Eileen Kintsch, Darrell Laham, Missy Schreiner, Gerry Stahl, and Dave Steinhart.
strategies through the use of *Latent Semantic Analysis* (LSA) in providing instant feedback (Franzke et al., 2005).

LSA is an established theory and method of text analysis which involves automatic acquisition and inference of the meaning of a word based on its contextual usage in a large collection of texts called the *corpus* (Landauer and Dumais, 1997). LSA has the following attributes:

1. It automatically gains knowledge from the raw texts that reside in the corpus.
2. It uses the co-occurrence of words in the corpus to establish and represent their meaning. Thus, LSA’s word-meaning representation is corpus dependent.
3. It ignores the order of words in the corpus grammatically or syntactically.

The LSA Research Group has assessed the capability and performance of the LSA method and they indicate positive results in LSA’s capability of predicting text coherence and resulting comprehension (Landauer et al., 1998a). In the LSA method, each term and document of the submitted written work is represented as a vector of length $d$. The similarities of any terms or documents are computed by measuring the angle referred to as a *cosine value* between the terms or documents. The *cosine value* is the contained angle between the vectors in semantic $d$-space - interpreted as the degree of qualitative similarity of meaning based on the selected corpus (Landauer et al., 1998b).

Summary Street™ returns feedback immediately upon any summary submission. This enables students to use its feedback for further improvement in the next succession of
repeated submit-and-receive-feedback cycles (Franzke et al., 2005). Students were found to benefit most from this kind of feedback compared to delayed feedback in traditional assessment (Graham and Harris, 2005).

In addition, Franzke et al. (2005) observed that the immediate feedback provided by Summary Street™ was beneficial to students in the following ways:

1. The system enhances the students’ performance in such a way that they can address their own deficits by repeated trial-and-error sessions at their own pace according to their level of capacity until they meet the pre-defined criteria.

2. The system provides an environment, in which frequent and rich occurrence of formative assessment and feedback is embedded within learning, which is beyond the kind of support they can get from a human-grader.

3. The system alerts the students to their summary problems but at the same time allows them to struggle at their own pace to meet the predefined criteria without penalizing or overcorrecting them. Research done by the Summary Street™ research group proved that this type of learning-and-assessing promotes students’ reading comprehension and writing skills.

One major advantage of Summary Street™ over PEG™ and Criterion™ is that it assesses essays semantically. The second advantage is that the LSA method employed in Summary Street™ does not involve feeding human knowledge manually into the system as in Natural Language Processing’s machine learning stage. The third advantage is that LSA ignores grammatical and syntactical factors of the pre-graded essays as used in the statistical method in PEG™. Thus, it reduces the amount of pre-processing time needed
for every essay task. On the other hand, LSA disadvantages are that it can not measure the
word order, organization and syntax of the essay.

2.3.2.4 BETSY™

The Bayesian Essay Testing Scoring System (BETSY™) is a window-based e-assessment for essay, being developed by Lawrence M. Rudener since early 1992. It is
not only an automatic writing assessment application but it can also be applied to any text
classification. Rudner claims that (Rudner and Liang, 2002) the classification of texts can
be used to make inferences in essay assessment (content and style). The classification is
built on the literature in the field of information science and incorporates Bayesian
Networks techniques.

The text classification technique in the BETSY™ assessment tool is aimed at classifying
text into a three-point or four-point category system such as “extensive, essential, partial,
unsatisfactory” based on trained text materials to form a large set of items. The items are
defined as “a broad set of essay features including content features (specific words,
phrases), and other essay characteristics such as the order in which certain concepts are
presented and the occurrence of specific noun-verb pairs” (Rudner and Liang, 2002). In
the case of automatic writing assessment, essay scoring typically categorizes into groups
of “pass/fail or advanced/proficient/basic/below basic” (Dikli, 2006).

BETSY™ adopts the Multivariate Bernoulli Model and the Multinominal Model in its text classification technique (Valenti et al., 2003). Both models make use of the probabilities of features contained in essays or within text categories.

Rudner and Liang (Rudner and Liang, 2002) claimed that “in theory, this approach to computer grading has the advantages of the best features of PEG and the Bernoulli Model plus several crucial advantages of its own. It can be employed on short essays, is simple to implement, can be applied to a wide range of content areas, can be used to yield diagnostic results and can be adapted to yield classifications on multiple skills”.

Like PEG™ and Criterion™, its major disadvantages are that it cannot measure an essay semantically and that it must have a collection of pre-graded essays by a human grader prior to processing the rest of remaining essays.

2.3.2.5 Discussion

The previous four sub-sections have presented an overview of four different tools for the e-assessment of essays, which implement different methods and produce different feedback. This section highlights their general characteristics in terms of their assessment mechanisms and also their strengths, weaknesses and outputs. Table 2.1 summarizes the characteristics of each of the tools pertinent to the scope of this research.
Table 2.1 – A Summary of the Characteristics of Automatic Writing Assessment Tools

<table>
<thead>
<tr>
<th>The Writing Assessment Tool</th>
<th>What It Evaluates</th>
<th>How It Does</th>
<th>Its Characteristics</th>
</tr>
</thead>
</table>
| **PEG™**                   | • Surface feature. | • Multiple regressions Use pre-graded essays. | **Advantages:** • Measure writing style.  
**Limitations:** • No conceptual measure.  
• Lacks syntactic measure.  
**Output:** • The resulting score is based on the coefficient values of proxes. |
| **Criterion™**             | • Surface feature.  
• Content feature.  
• Order of content. | • NLP  
• Multiple regressions. Use pre-graded essays. | **Advantages:** • Measure syntactic.  
• Measure writing style.  
• Measure essay organization.  
• Measure essay development.  
**Limitations:** • Can be tricked.  
**Output:** • The holistic result.  
• The diagnostic feedback. |
| **Summary Street™**        | • Content feature. | • LSA  
Use both pre-graded essays (if possible) and corpus of text in specific domain. | **Advantages:** • Measure essay content.  
**Limitations:** • No syntactic measure.  
**Output:** • Percentage of content.  
• Percentage of mechanic.  
• Percentage of style. |
| **Betsy™**                 | • Surface feature.  
• Score category Use pre-graded essays. | • Probabilities  
• Score category Use pre-graded essays. | **Advantages:** • Measure mechanics.  
**Limitations:** • No syntactic measure.  
• Lack of content measure.  
**Output:** Category to which the essay belongs. |

As can be seen from column 2 of table 2.1, there are three broad categories of essay feature: surface, content and organizational. The *surface features* deal with the syntactic characteristic of the essay such as grammar, number of prepositions, punctuation and spelling used. The *content features* cover such things as the choice of vocabulary, style and concept used. The *organizational features* are concerned with how the key ideas are developed, organized and presented.
However, a given assessment mechanism may be more capable of evaluating one type of feature than another. For example, multiple regression and natural language processing mechanisms are more capable of evaluating surface and organizational features, whilst LSA is more capable of evaluating content features. However, assessment mechanisms which use multiple regressions and Natural Language Processing must be trained with a specific set of sample essays, pre-graded by a human grader, in order to be functional as an assessment mechanism (see column 3, table 2.1). By using pre-graded essays, computerized model answers are developed to represent the judgments made by humans. In other words, those types of mechanisms are dependent on explicit attributes of the pre-graded essays and do not use inference. On the other hand, LSA is capable of making use of both explicit and implicit attributes of the essays due its inference capability. Therefore, even without pre-graded essays, LSA is capable of performing partial assessments.

Using pre-graded essays to model human judgment is suitable for the formal, structured and domain-specific type of essay since this type of essay is normally given to groups of people (e.g. students) and they produce similar types of essay in respect of the essay features – i.e. the surface, content and organizational. Therefore, the sample set of essays to be pre-graded is easily available. In addition, the obtained model of the computerized answer is applicable for grading the rest of the available essays, since they conform to the same domain of task and topic. On the other hand, in the case of this research, which involves an informal, unstructured and unconstrained type of free-text automatic assessment; a set of pre-graded sample essays is unlikely to exist. Thus, among the
reviewed assessment mechanism/techniques (as summarised in table 2.1), the LSA is the most appropriate choice due to its capability of making use of implicit attributes of the essay and it will be adopted as the assessment mechanism for the Web text-handling support tool developed by this thesis. Further, this work adopts LSA as an assessment technique in order to investigate its potential of assessing free-response text without using any pre-graded essays, which have been used in all reviewed systems. However, this research inherits the weaknesses of LSA, namely that it is incapable of evaluating the surface and organizational aspects of essay features.

Based on the essay features discussed before, there are four broad categories of the essay feedback (last column of table 2.1): surface, content, organizational and category. *Surface feedback* presents feedback which encompasses the syntactic characteristic of the essay, such as the grammar, number of prepositions, punctuation and spelling used. *Content feedback* covers such things as the choice of vocabulary and concept used. *Organizational feedback* concerns how the key ideas are developed, organized and presented. Finally, *category feedback* categorizes the essay into several groups, according to its overall characteristic, ranging from those having poor to those having good essay characteristics. This thesis adopts content feedback since the LSA method is more capable of producing that type of feedback.

In summary, this research adopts the LSA method as the writing assessment mechanism due to its inference capability. Its inference capability allows the assessment mechanism to partially work without being trained with pre-graded essays. This feature is important
to this research since it involves an informal, unstructured and unconstrained type of knowledge acquisition where a set of pre-graded homogeneous answers is unlikely to be available. Even if a set of answers is available, it is likely to be a set of limited model answers to represent potentially unlimited essay contents which can be obtained from the vast amount of Internet material and resources. Therefore, this research investigates the applicability of the LSA technique for assessing a free-response text in the domain of informal, unstructured and unconstrained knowledge acquisition within task-driven Web text-handling. The system will also provide immediate formative feedback in terms of content-related essay features.

2.3.3 E-Assessment of Plagiarisms: A Review

*E-assessment of plagiarism* is defined in this work as a computer application or tool that is able to assess learners’ digitally written text in terms of plagiarised text and provide them with constructive feedback. Park defines plagiarised material as using the private information without crediting the source, which he considers as stealing (Park, 2003). Further, he describes plagiarism as a complex and deep-seated issue in higher education. Bull et. al. identify the two most common types of plagiarism: firstly, learners plagiarising materials from theses or text books and secondly, learners copy-and pasting Web texts (Bull et al., 2001).

Due to the complexity of plagiarism, MacDonald and Carroll suggest that higher education institutions teach learners skills within the context of academic approach in learning. They argue that by providing information and means, learners can develop skills to avoid plagiarism (MacDonald and Carroll, 2006). In addition, they believe that
“students will best understand what constitutes plagiarism if they actively work with whatever definitions they are offered”. However, it is reported that the availability of self-assessment systems for enhancing the skill to combat plagiarism is very low (Stefani and Carroll, 2001).

E-assessment of plagiarism employs plagiarism detection mechanism to assess learners’ digitally written text. The plagiarism detection systems adopt diverse mechanisms for allowing users to submit their text files; they also use diverse techniques for searching plagiarised sources and detecting plagiarised text segments. Basically, the detection mechanism works by comparing the submitted text against the text from possible plagiarised source, which could be from the Web, uploaded essay banks and uploaded databases of papers (McKeever, 2006).

McKeever describes six types of plagiarism detection mechanism, of which three are suitable for adoption to detect plagiarised text from the Web, as follows (McKeever, 2006).

2.3.3.1 Search-Based Mechanism

The search-based mechanism works by comparing the submitted text against the text from a plagiarised source, specifically looking for string overlaps in word-for-word copying, also known as verbatim copying. Its working principle is “two writers are unlikely to use exactly the same sequence of words above and beyond a certain phrase length”.
An example of e-plagiarism tool that uses the search-based mechanism is Turnitin\(^4\). It is a Web-based and commercial plagiarism detection tool, developed by iParadigms LLC\(^5\) (Maurer et al., 2006). It has been used in academic settings since 1996 including by organization such as JISC Plagiarism Detection Service\(^6\). It offers instant feedback on the 'copy-and-paste' type of plagiarism thus allowing tutors to focus more on addressing plagiarism's causes rather than being involved in exhaustive and extensive searching and detecting of documents which might consume much valuable time. Turnitin provides feedback within minutes upon submission of a document. It is in the form of a personalized report, referred to as ‘the originality report’, and is displayed within a user interface. The report presents marked submitted documents along with their percentage of plagiarism. Additionally, it provides links to original resources and enables the user to check the marked part in the submitted document against them.

The advantage of the search-based mechanism is that there are many existing service providers or systems based on this approach and to stay competitive, the service providers or systems need to broaden from specializing in copy-and-paste only to i.e. detecting slight modification in sentences. On the other hand, the disadvantages of the searched-based systems are: Firstly, the written text and the Web text must be compatible in format to ensure that any sentence comparison can take place. Secondly, the search-based mechanism involves exhaustive searching activities to cover most Web texts: Web pages, 


\(^6\) [http://www.submit.ac.uk](http://www.submit.ac.uk) (Last accessed: February 02, 2008>
private essay-banks and private databases which contain journals, books or theses. As a result, turnaround time is increased, especially when the service is used for a large group of users.

2.3.3.2 Linguistic Analysis

The linguistic analysis mechanism works based on a technique known as cloze procedure. Its working principle is “everyone has their own writing style and can recall it if necessary” (McKeever, 2006). Basically, learners’ written texts are submitted to service providers and the texts will be returned with certain patterns of words eliminated (this is the cloze procedure). The learners are asked to fill in the blanks in their text and to return it to the service provider. Next, the service provider will compare with the original text and find any discrepancy. The percentage of discrepancy between the original text and the cloze procedure text will determine whether the text has been plagiarised or not.

An example of a system using this approach is Gatt Plagiarism Services\(^7\). It offers plagiarism services to both academic institutions and learners. Institutions use the services to detect plagiarism, while learners use the services for self-assessment of their own text against plagiarism. The services provide both institutions and learners with instant feedback for them to trouble-shoot plagiarism.

The major advantage of linguistic analysis is that it capable of detecting plagiarism from various resources and is not limited only to Web-based resources (McKeever, 2006). However, one major disadvantage is that besides learners having to submit texts, they also have to become involved in an extra two-way communication with the plagiarism services. Learners are left to prove they are not guilty prior to any plagiarism detection that has taken place. Another, disadvantage is that the service is more suitable for short paragraphs. For longer essays, it is unlikely for learners to participate since the detection process is not only time consuming but also intrusive.

2.3.3.3 Using Search Engines

The most widely used mechanism for detecting plagiarism is the use of search engines or metasearch engines. Basically, it works on the principle that search engines do repetitive try-and-locate operations to locate as many as possible plagiarized sources based on submitted search phrases by end users (normally tutors or markers). The search phrases can be submitted to search engines and their results are checked by end users manually, which has the disadvantage of being a labour- and time-intensive activity.

Due to these disadvantages, it is unlikely for tutors and learners to use it for their own purposes. However, it’s a simple and quick approach that has been adopted as a back-end processor by plagiarism service or system providers.

2.3.3.4 Discussion

The three plagiarism detection mechanisms described in the previous sub-sections have been designed for assessing plagiarism and providing feedback. Derived from literature,
the disadvantages of most plagiarism service or system providers are as follows: Firstly, powerful computer servers are needed to perform the plagiarism detection. Secondly, most of the fully-fledged and reliable plagiarism detection tools are commercial products; therefore learners (or the institutions where the users are based) have to subscribe or buy before using the tools. Hence, the tools are not as openly accessible to the public as the Web is and the services and systems are not tailored to be used by learners. This is because the majority of the detection tools are not meant for learners to use for self-assessment in dealing with plagiarism issues due to their high cost. Learners are unlikely to purchase the system or service in order to find their own plagiarism problems. Thirdly, most of the tools are operated through a Web portal. Although this approach makes mass distribution easy, it may be less reliable as it offers a single entry point for all users.

On the positive side, most of the tools offer immediate feedback in the form of a personalized report. Generally, the report consists of an indication of how much plagiarism is involved (either presented in the form of a percentage or an index); shows the plagiarized sections in the submitted document; and where the original resources are located on the Web or other types of digital resources. The advantage of the personalized report is it can be tailored to provide self-assessment and self-regulated learning of plagiarism issues for learners, which is recommended by authors in the plagiarism field as a suitable way to encourage learners to actively participate in plagiarism prevention (McKenzie, 1998, Barrett and Malcolm, 2005).
Out of the three mechanisms for detecting verbatim copying plagiarism, the search-based mechanism, is most suitable to be adopted here because this research deals with a time-constraint Web text-handling task performed in a real-life setting where learners are required to complete their task by using the same computer throughout the session. Their Web data which is captured automatically by the Web browser residing on the computer is sufficient to be promptly used in tracing their Web navigation trails. Then, the trails (in the form of URLs) are fed as input to the search-based mechanism and the Web texts are retrieved for comparison purposes. In contrast to other plagiarism detection mechanisms, this mechanism involves an explicit Web page retrieval followed by plagiarism detection, which reduces the system’s back-end processing i.e. to identify which Web pages have been visited by learners using learners’ written texts. By reducing the amount of back-end processing, the support system does not need to be run on powerful computer hardware and software.

In summary, the chosen plagiarism detection mechanism makes it possible to adopt the proposed support system using in-house standard computer hardware and software. At the same time, the support system is capable of capturing sufficient data for the purpose of this research i.e. to provide immediate formative feedback to learners in terms of verbatim copying.

### 2.3.4 E-Assessment of Information Seekers

In relation to the World Wide Web, an *information seeker* is defined in this work as *anyone who seeks, reads and retrieves text from the Web*. In relation to e-learning, the information seeker is defined in this work as an *online learner who seeks, reads and
retrieves text from the Web. Therefore, in this work, both terms – information seeker and learner – refer to the same person.

E-assessment of information seekers is defined in this work as a computer application or tool that is able to assess learners’ handling of Web text and provide constructive feedback. Hughes argues that learners, especially online learners are in need of support in handling Web text (Hughes, 2004). The task of handling Web text is closely related to two issues: avoiding Web text plagiarism (as discussed in the previous section) and comprehending Web text.

Publications in defining (Broder, 2002), modelling (Juvina et al., 2005), evaluating (Bruza et al., 2000, Rose and Levinson, 2004, Dalal et al., 2000, Prasetyo et al., 2002) and supporting information-seeking behaviour (Lee et al., 2005, Olston and Chi., 2003, Teevan et al., 2004, Aula et al., 2005) are widespread, as are studies in the fields of reading comprehension (Harrison, 2004b, Harrison, 2004a) and automatic feedback (Tsintsifas, 2002, Higgins and Bligh, 2006), however, Kuiper et. al. report that there is little existing research on developing a support system that provides feedback for information seekers which combines both information-seeking and handling of Web text as one entity. Their review concludes that most learners “like to use the Web but often do not possess the necessary skills to find the right information. And when they do find the right information, it is also difficult for them to use it to pursue an inquiry or solve a problem. Searching for information, usually results in insufficient knowledge, understanding, and insight”. However, they find that “virtually no empirical research”
has focused on these abilities (Kuiper et al., 2005). Conole and Warburton suggest that we need to explore the use of technology in assessment in terms of what we want learners to learn and to consider “what we want to be assessing and how best to do this”. Further, they conclude that “the role of technology and how it might impact on assessment is still in its infancy and we need to develop new models for exploring this” (Conole and Warburton, 2005).

Derived from the work of Leu et al., handling Web text involves synthesizing the text. In their work, synthesizing text is referred to as a phenomenon where learners (referred to as readers) “actually construct the texts they read by the choices they make in the links that they follow, collecting a series of non-continuous texts and synthesizing the essential aspects of each during the comprehension process. ……, readers often skip more information at any single page than they read; the units of text that readers find useful at any single page are often quite small and they seldom read all of the information at a single Web page. Online reading is a continuous synthesis and evaluation process, with readers choosing the information that they read; often with new searches for information in the middle of the reading process, all of which takes place in a recursive manner until the reader determines that they have solved the informational problem or have come up with an answer that is sufficient” (Leu et al., 2007).

The International Reading Association has recognized this unique phenomenon and they describe it as a unique nature of online reading comprehension and they urge the development of suitable assessment to measure it (International Reading Association,
Dowsing and Long argue that one major difficulty with automated assessment for IT skills is to associate learners’ errors with particular skills. They suggest that the complexity of a test should be reduced so that it comprises a series of actions with each one testing the learning of a single skill. Further, Dowsing and Long suggest that capturing a learner’s event processes, while performing a real-time test can provide extra insight. They suggest that a learner’s actions should be part of that learner’s temporal record. The temporal record is regarded as “a simple model” which “relates errors to the current actions being undertaken by a candidate”. Dowsing and Long acknowledge that, unfortunately, *event stream collection as part of data* (referred to as real-time data in this thesis) to be assessed is not the norm in current practices (Dowsing and Long, 1999).

However, Conole and Warburton in their review of computer-assisted assessment state that there is shift in assessment research from “assessment of products or outputs to assessing the process of learning” (Conole and Warburton, 2005)

Nevertheless, Leu et al. find that none of the existing approaches of assessing Web text-handling assess both the process and the product of Web text-handling. To date, Leu et al. identify three approaches for assessing online reading comprehension, of which only two relate to Web text-handling (Leu et al., 2007):

1. **Measuring performance on problem solving tasks within a limited informational space developed to imitate a very small portion of the Web.**

   The advantage of this approach is that the massive amount of Web information is reduced dramatically; therefore, it is easier to monitor learners’ actions. However, the approach uses an artificial setting of Web text-handling, therefore, it cannot be
implemented in a system for real-time self-regulated self-assessment that allows learners to acquire Web text-handling skills in a real-life setting.

2. **Measuring performance on problem solving tasks on the Web.** This approach uses Camtasia\(^8\) video recording software to record the learners’ event streams. Even though this approach captures real-time data in a real-life setting, the processes of assessing the captured data and providing feedback to learners are time and labour extensive. Further, the approach is not suitable for providing instant feedback for a self-assessed and self-regulated type of learning, as explored in this research.

A review of the literature shows that currently there are no existing systems that provide an integrated environment for learners to self-assess their handling of Web text and to receive instant feedback in a real-life setting. This thesis focuses on the development and evaluation of just such a novel system.

### 2.4 Motivation for E-Feedback

The fundamental motivation for incorporating e-feedback tools in e-assessment systems has been that in most institutions of higher education the increase in the number of learners has been more than that of the tutors. As a result, learners are getting less personalized feedback from tutors (Lilley et al., 2005). While getting timely and effective feedback is the essence of any learning or assessment (Blayney and Freeman, 2003), the tutors are neither capable of assessing the learners (Dalziel, 2001) nor able to provide support at any time and place (Malmi et al., 2002).

---

The second motivation has been the need to follow the principal attributes of the ultimate e-assessment engine identified by FREMA research group. MacKenzie indicates two principal attributes of an e-assessment system required by e-assessment or e-learning that is suitable for learners. The first principle is to deliver “informative, detailed and context or score sensitive feedback to student”. The second principle is to provide “information feedback on student and question performance to the tutor so that tutorial help may be targeted and improvement may be made for the future” (MacKenzie, 2003).

This work will utilise an e-assessment approach with e-feedback that focuses on the first principal attribute, i.e. to deliver useful information to learners regarding their performance in their e-assessment. While, at the same time it will also fulfil the fundamental requirements, i.e. to provide personalized and timely e-feedback.

2.4.1 A Constructive View of Feedback

Feedback is useful for effective assessment. Different approaches to teaching and learning view feedback with different purposes. The most cited view of feedback by the present e-assessment community is constructive (Askew and Lodge, 2000, Nicol and MacFarlane-Dick, 2006, Nicol and Macfarlane-Dick, 2004, Tsintsifas, 2002). In this work, constructive feedback is defined as learners perform self-regulated and self-monitored activities to enhance their knowledge during their learning acquisition process based on the feedback they receive.

---

Askew and Lodge (Askew and Lodge, 2000) describe constructive feedback as having the following attributes:

- **Information provider.** Feedback should provide information to allow learners to make improvements.
- **Acts as a counsellor.** Feedback should invite learners to respond within a scope and context. In return, it allows learners to reflect on their own understanding by providing comments and insights.
- **Acts as a connector.** Feedback should help learners to make their own connection of their past and current experiences within the context and scope of assessment and to explore their own understandings.
- **Ping-pong.** Feedback should encourages learners and tutors to describe and discuss repetitively in two-way communication – to and fro.

Martha (Martha, 1994) describes constructive feedback to learners as a systematic process to enhance learning with distinguishable characteristics as follows:

- **Relevant.** Feedback should address learners with specific achievements, needs, interests and learning behaviours.
- **Immediate.** Feedback should provide learners with information about their performance as promptly as possible.
- **Factual.** Feedback should contain items based on actual learners’ achievements.
- **Helpful.** Feedback should provide suggestions for further improvements.
- **Confidential.** Feedback should be given directly to learners.
- **Respectful.** Feedback should respect learners’ integrity and needs.
• Tailored. Feedback should be designed to meet individual learners’ needs and circumstances.

• Encouraging. Feedback should motivate learners to continue their learning efforts.

Derived from both descriptions, this work employs constructive views of e-feedback by aiming to providing relevant, immediate, factual, helpful, confidential, respectful, personalized and encouraging feedback to learners via a systematic, independent, self-motivated and self-regulated way of learning.

Nicol and MacFarlane propose seven principles of good practice in constructive feedback. They describe two types of feedback: external and internal. Learners receive external feedback from other people involved in the feedback activities, i.e. from tutors and peers. On the other hand, learners acquire internal feedback by themselves, i.e. their own insights, understanding and motivation (Nicol and MacFarlane-Dick, 2006, Nicol and Macfarlane-Dick, 2004).

The seven principles of good feedback practice are as follows:

1. **Facilitates the development of self-assessment (reflection) in learning.** Tutors are encouraged to promote structured self-assessment and self-correction by allowing learners to self-identify their own strengths and weaknesses prior to their final work submission. This, reduces learners’ dependency on external factors for feedback and at the same time increases their gained internal feedback.
2. **Encourage tutors and peer dialogue around learning.** Tutors are encouraged to provide external feedback to learners in the form of group discussion or peer discussion.

3. **Help clarify what good performance is.** Tutors are encouraged to clarify their expected standard and criteria for learner’s work by conducting class discussions, group discussions or by giving exemplar assignments with attached feedback.

4. **Provide opportunity to close gap.** Tutors are encouraged to help learners use directly their received external feedback by allowing them multiple re-submissions. Thus, the process helps to close the gap between a tutor’s expectation and learners’ achievements.

5. **Delivers high quality information to students about their learning.** Tutors are encouraged to increase the learners’ number of practice attempts by providing instant feedback; to provide an online test which can be accessed regardless of frequency, time and place; and to focus on the weakest learners.

6. **Encourages positive motivational beliefs and self esteem.** Tutors are encouraged to boost learners’ internal feedback by providing information about progress and achievements rather than about success or failure. These can be achieved by embedding e-feedback in e-assessment and by allowing multiple resubmissions.

7. **Provides information to tutors that can help to shape the tutoring.** Learners are encouraged to inform their tutor about his/her performance, i.e. by evaluating key points in tutoring or by informing the tutor about the difficulties they have in their assignments.
Derived from this list of good practice principles, this work acknowledges the advantages of, and barriers to developing, an e-feedback system.

2.4.2 Advantages of and Barriers to E-Feedback

As seen from previous section, embedding e-feedback in an e-assessment system is a complex task. The e-feedback requires two fundamental capabilities. Firstly, the e-feedback system must ‘understand’ the task in the context of its problem domain. This is to ensure that the e-feedback system reacts intelligently upon receiving a learner’s response. Secondly, e-feedback must be humanized so that it is able to deal with learners’ insights and motivation. Out of the seven points of good feedback practices, this work aims to deal with points 1, 4, 5 and 6, which are most relevant to the objectives of the Web text-handling support tool. Inclusion of the remaining e-feedback principles would make the system very complex and successful completion of the system requirements, design and implementation would be less likely.

This research is aimed at supporting the weakest learners in the information-seeking problem domain. By automating feedback, learners receive instant feedback regardless of frequency, time and place of task submissions (point 5). Offering learners instant feedback allows an e-feedback system to accomplish two good points. Firstly, an e-feedback system allows multiple task resubmissions. This factor helps learners to clarify the standards and expectations required by their tutor (point 4). Secondly, for each submission, e-feedback encourages learners’ positive motivational beliefs and self-esteem by presenting reports on progress and achievement (point 6). By frequent resubmissions
and corrections, e-feedback helps learners to acquire structured self-assessment and self-correction skills (point 1).

### 2.5 Software Engineering

Development of an e-feedback system involves the application of a prescriptive model in the software engineering area. Pressman defines a *prescriptive model* as “a distinct set of activities, actions, tasks, milestones, and work products that are required to engineer high-quality software…. The process model is not perfect but it does provide a useful roadmap for software engineering work” (Pressman, 2005). Among the various prescriptive models available, the *interactive system development* approach has *system usability* as its primary goal (Carroll, 1997, Maguire, 2001). Braude defines *system usability* as a system which users find easy to use (Braude, 2004).

To increase system usability, Cooper introduces a *Goal-Directed Approach* in the software development process, which is a strand of interactive system design. He uses three tools to promote the usability of a system. The first tool is a fictional user to improve the representation of a system user, referred to as a *persona*. He argues that using real users to design a system is “frequently useless and often detrimental to the design process”. The second tool is identification of the persona’s goals. He argues that a system is considered good if the persona of a specific system uses it for some purpose and in return, the system helps the persona to complete related task and therefore achieve his / her purpose of using the system in the first place. The third tool is a scenario, i.e. a fictional setting in which the intended persona is in context while using the system.

One major advantage of the Goal-Directed Approach is its flexibility in how its tools are applied and how the system is implemented. The lack of a proper prescriptive model of the approach allows system developers to use their creativity in using the design tools, thus avoiding rigidity in the design and implementation of the system. Further, the aim of this approach is to start with or produces the simplest system that is usable. Later, the features of the system are added incrementally but sparingly. Hence, it can be used in combination with other software development processes, e.g. agile software process.

Generally, the agile software process is referred to in this thesis as developing software incrementally with emphasis on the following: *individuals and interaction over processes and tools, working software over comprehensive documentation, responding to change over following a plan and producing workable smaller software pieces in a shorter time over bulk software pieces in much longer time* (Hunt, 2006, Pressman, 2005, Budgen, 2003). The agile software process has a comprehensive prescriptive model. Therefore, it complements the Goal-Directed Approach and further they share some common goals, i.e. both software development approaches are highly concerned with user satisfaction and
promote a system to be developed incrementally from the simplest workable system, in a shorter software lifecycle.

The second advantage of the Goal-Directed Approach is that it is not associated with any specific programming language. Thus, any suitable programming can be used to implement the system. The third advantage of the Goal-Directed Approach is that even though it is meant for designing system purposes; its flexibility allows it to be used for different purposes, i.e. for modelling complex user behaviour in a specific setting.

In summary, both the Goal-Directed and the agile software development processes are useful to this research for two reasons: firstly, this work focuses on exploratory research where the purpose of the proposed system needs to be explored with learners. The learners are likely not to know what they want and the researcher does not know from the start what the final form of the system should be to meet the learners’ needs. By using these two incremental software processes, the requirements and functional specifications for the proposed system can be prototyped and enhanced in an exploratory way. Secondly, the support system proposed in this thesis is a new development. Therefore, the prototype system needs to be proved usable before the real system is fully developed in future research.
2.6 The Research Gap

Drawing from the literature reviewed in this chapter, this thesis addresses the problem broadly, aiming to investigate, analyze, design, implement and test an e-feedback system that provides students in higher education with:

1. A basic integrated environment for open-ended Web-based information-seeking to take place.
2. A personalized e-feedback for learners to handle Web text in a real-life setting.

The feedbacks cover three areas in their rewritten text: relevance to the given task, coherence and verbatim copying.

The e-feedback system aims to support the online learner by providing a means for self-assessment of their Web text-handling skills in a self-regulated way, which has not been attempted before.

2.7 Summary

This chapter has presented the background concepts for this thesis. Firstly, this chapter provided an overview of e-assessment and its technological advancement. Then, various mechanisms used in e-assessment system were described, specifically focusing on free-response essay assessment and Web text plagiarism detection. Next, it highlighted the usefulness of e-feedback to the research area and described a software development approach to achieve a usable e-feedback system. The chapter finally concluded by outlining the research gap.
The next chapter will describe a qualitative experiment and the use of the Goal-Directed Approach for defining and modelling the processes and products of Web text-handling. It specifically outlines the processes and products which can be automatically captured and processed by a standard computer. Chapter 4 will identify the basic requirements for integrating the model in an e-feedback system.
Chapter 3

Modelling a Learner’s Behaviour in Handling Web Text

3.1 Introduction

Learners must read ever larger volumes of information in increasingly complex forms and aggregations and this information is accessed via a growing level of interaction in a complex electronic environment (Marchionini, 1998). Many studies analyzing learners’ behaviour have reported that some learners encounter reading or navigational problems while handling Web text and are in need of various types of support (McEneaney, 2000b). McEneaney argues that competence in Web text-handling makes a significant contribution to Web text reading effectiveness. If the shortcomings in Web text-handling are left unsolved, the second-level digital divide (in which a learner has access to the Web but fails to comprehend its information) will become a serious problem (Hargittai, 2002).

Providing Web text-handling support to learners requires e-assessment with an e-feedback mechanism. However, the problem of providing e-feedback for Web text-handling is neither simple nor predictable as it involves elements of reading. Even without the complexity of Web technology, there are different views on reading definitions and different views on its assessment, along with unstable existing theories of reading and reading assessment (Sainsbury, 2006).

There is also a lack of models that represent the complexity of Web text-handling, especially ones that focus on the learner’s behaviour. This research explores the
application of the Goal-Directed Approach to model the complex behaviour of learners while handling Web text. So far, researchers in software engineering (as reviewed in section 2.5) have applied the Goal-Directed Approach (i.e. personas and scenarios) in two stages of the software process: requirement analysis and interface design (see chapter 5 in (Pressman, 2005) for further explanation on the software process). Unlike all previous research, (which involves orchestrating personas in scenarios with a futuristic setting) this chapter describes the orchestration of personas in scenarios with a present setting. This work examines the ability of the Goal-Directed Approach to model complex behaviour in a simple way and to produce a usable model.

The next section describes a qualitative experiment that gathers data about the Web text-handling behaviour of learners and the use of the Goal-Directed Approach to model the observed learners’ behaviour. As a result, seven Web text-handling processes are derived from the outcomes of the experiment and the approach, forming a model of the learners’ behaviour. Then, the model is substituted into a model of a Web text-handling scenario in section 3.2.3.4. Based on this model, section 3.3 proposes general requirements for supporting learners in handling Web text. It specifically outlines the processes and products of Web text-handling to be automatically captured and stored as real-time data in a real-life setting.

3.2 Understanding Learner’s Web Text-handling Behaviour

This section presents a preliminary study of modelling learners’ behaviour in handling Web text by using the Goal-Directed Approach. This study aims to produce a general
model of how task-driven learners (in higher education) handle Web text in a real-life setting. In this thesis, the model is treated as the basis for patterning a model of learners’ behaviour in Web text-handling, which will be referred to in later chapters.

3.2.1 Capturing Real-Life Data

Capturing real-life data (especially digitally) is important to this research for two reasons: first, it provides valuable insight about learners in a real-life situation; and second, it produces empirical data in digital form that can be automatically processed. The digital data is the primary source of interest, as they reveal potential Web text-handling information to be used and investigated within the scope of this research.

The study employs a combination of technologies to manually capture the richness of the Web text-handling process. These are:

1. Web logging – a method used to keep track of the learner’s Web data in the form of navigation trails.
2. Video camera – a method used to capture the learner’s observable behaviour.
3. Audio recorder – a method used to capture the learner’s talk-aloud and a short interview.
4. Screen recorder – a method used to capture the learner’s screen content and screen interaction activities.

3.2.1.1 Participants

Six experienced computer users, with backgrounds in the fields of linguistics, history and computer science, took part in the experiment. Although they were all regular computer
users, they were chosen in anticipation of portraying distinguishable styles of information-seeking behaviour due to their differences in background, skills and knowledge.

3.2.1.2 Task Design

The participants were given a task in handling Web text. The task was designed to enable them to demonstrate their Web text-handling skills in terms of how they seek, retrieve and select the Web resources. Furthermore, the participants were required to write a short essay as specified in the task. They were given 60 minutes to respond on the topic of Socialism. The task was presented as follows:

*The Web offers many definitions of the term Socialism. Perhaps this is because it is a term that produces strong emotions, and it is a word that is considered important by groups as well as individuals.*

*Your task is to spend an hour looking at the information available to you using whatever Web resources you are able to locate, and then to produce a document in MS-Word that presents and comments on some definitions of Socialism. The total document should be succinct, no more than 500 words in length (you can count them using the Word Count tool in the Tools menu of Word), of which up to 300 words (60%) can be quotation, so you will not be writing more than 200 of your own words. If you produce a good deal less than 500 words that will be fine, too.*
3.2.1.3 Procedure

Participants were expected to read the task written on paper, search the Web using the Web Explorer software application and write a free-response text based on selected Web resources using the Microsoft Word software application. The three activities were presented concurrently (as shown in figure 3.1) and were accessed interchangeably throughout the session.

![Figure 3.1 – The setting for Web text-handling](image)

3.2.1.4 Data

The participant saved his or her written text through the use of the text editor (Microsoft Word). Their Web navigation trails and text editing were recorded using the Camtasia software package that allowed screen and audio recordings. This combination of data from different sources, allows the data to be triangulated. Capturing audio data allows participants to think-aloud and explain why they perform specific actions at specific

---

times; this represents their implicit actions. The data captured by the video and screen recorders provide us with the learners’ explicit actions in step-by-step form. Thus, the triangulations of this data provides a fuller picture of the learners’ reasoning and interactions while Web text-handling is taking place. The recorded materials were transcribed for further analysis. Figure 3.2 depicts the data analysis process.

**Figure 3.2 – The data analysis process**

Based on the transcribed data, the participants’ actions which are in the form of verbs are extracted and then these verbs form units of verb-keyword. In order to reduce the number of keyword units, a single verb-group was used to represent keywords with similar meaning, i.e., the ‘find’ and ‘search’ keywords were grouped into the ‘search’ group. Each of these groups represents a distinct Web text-handling process, which will aid in modelling the characteristics, behaviour, motives and goals of personas.
3.2.2 Personas and their Respective Scenarios

Cooper suggests that, for every project, as few as three to twelve unique personas are useful for articulating the user population (Cooper, 1999) and Nieters et. al. suggest each persona is described as an action-figure to make them memorable and distinguishable (Nieters et al., 2007). Adler argues that the practical details about creating personas are missing in the literature (Adler, 2005). He describes practical guidelines for creating personas based on empirical data obtained qualitatively through interviews or observations. Junior and Filgueiras (Junior and Filgueiras, 2005) recommend personas are described around their behaviour patterns. The patterns may comprise of attributes such as probable usage environments, time relevant to the context, solutions, relevant relationship and goals. Therefore, to represent learners’ behaviour in a real-life setting it is important to identify the role of each persona to play a certain character.

In this modelling work, four distinguishable action-figure personas were developed based on the previously collected data. The actions were identified by observing the learners’ pattern of Web text-handling behaviour and their approach towards accomplishing the given task. These four fictional learners are intended to represent the learners in a real-life setting. In addition, they were also crafted to represent learners in higher level institutions.

Persona 1 – Open-Oliver
Open-Oliver is a first year student at the Department of Computer Science. He has good knowledge in his own subject area, but has fair background knowledge on the topic of socialism. However, he is full of enthusiasm to finish the given task and to produce a
good answer. He has always considered himself as an organized person. When he works on the computer, he likes to have all files he needs opened and ready to be used.

Scenario:
1. He reads the task quickly and he reads it only once.
2. He starts Internet Explorer.
3. He puts keywords in the search engine.
4. Then, he searches the list of URLs displayed by the search engine.
5. He selects the URL which he finds interests him most.
6. He reads the associated Web pages quickly.
7. He leaves the window open.
8. He repeats step 2-7 several times until he is sure that he has enough information.
9. He opens a Microsoft Word document and starts writing. He alternatively switches to one of the opened windows, he reads, sometimes he summarizes the Web text and sometimes he copy-and-pastes to his document straight away.
10. He does not bother to read the task again to make sure whether his answer is correct or not, or to check whether he has enough time or not. He continues his work splitting his time between writing, reading, summarizing and copying until he feels satisfied.
11. Finally, he stops his work and submits his task.

Persona 2 – Choose-Charlie
Choose-Charlie is a first year student at the Department of Engineering. She is an average student and Socialism is definitely not her favourite topic. She has a long list of things to do today, she has to finish her Math assignment, she must get something to eat and she must submit her lab work by half-past two. Therefore, she has to find a way to finish this task quickly.

Scenario:
1. She reads the given task quickly, but she has to read again and again in order to get some idea on how to answer. Finally, she plans to find good lecture notes related to the given task somewhere on the Web and to work around them.
2. She starts the Internet Explorer application.
3. She puts keywords in the search engine.
4. Then, she searches the list of URLs displayed by the search engine.
5. She selects a URL.
6. She reads the associated Web pages quickly.
7. She continues searching by repeating steps 4 and 5 until she confirms she can not find what she wanted to find.
8. She repeats steps 3 to 7 until finally she finds what she is looking for.
9. She opens a Microsoft Word document and starts writing.
10. She switches between the selected Web page and copy-and-pasting here and there in her document. She reads and revises her sentences and paragraphs.

11. While alternating between writing and reading, she keeps on looking at her watch. Every time she writes or deletes a few words, she uses the Word Count Tool to check her words limit. Time is very precious to her and achieving the word limit is the vital signal for her to stop and submit her work. Sometimes she reads again the task to check whether she is answering the given task. She is not very sure whether her answer is correct or not. But she is confident with the content on the lecture notes she found.

12. Finally, she decides that she wants to stop and submits her work. She still has a long list of ‘things to do’ in her bag.

Persona 3 – Smart-Sam
Smart-Sam is a first year student at the Department of Education. He is good in his own subject area and even more in the topic of socialism because it is among his favourite topics. He has always considered himself as a competent computer user. He is very confident he can produce a good answer for the given task. For him, the given task is easy and he starts immediately.

Scenario:

1. He skims through the task and he does it only once.
2. He starts the Internet Explorer application.
3. He puts keywords in the search engine.
4. Then, he searches the list of URLs displayed by the search engine.
5. He selects the URL which he finds interests him most.
6. He reads the associated Web pages quickly.
7. He opens a Microsoft Word document.
8. He summarizes what he found and develops sentences and paragraphs in his document.
9. He repeats step 3-8 (except 7) several times until he is sure that he has enough information. Once in a while, he checks his word count.
10. He is satisfied with his work and he decides to submit.

Persona 4 – Copy-Cathy
Copy-Cathy is a first year student at the Department of Education, specialized in History. She considers herself as fairly competent in both areas: computers and socialism. She always thinks doing a ‘Web text-handling’ task is a fairly easy job. She has nothing to worry about. The Web has always given her an answer if she needs one.

Scenario:

1. She reads the task twice to make sure she really understands what she has to do.
2. She opens a Microsoft Word document.
3. She starts the Internet Explorer application.
4. She puts keywords in the search engine.
5. Then, she searches the list of URLs displayed by the search engine.
6. She selects the URL which she finds interests her most.
7. She scans the associated Web pages quickly.
8. She copy-and-pastes a few sentences to her document.
9. She rearranges sentences, rearranges paragraphs, and adds or deletes a few words.
10. She repeats steps 4-9 several times until she is sure that she has enough information. Once in a while, she checks her word count and the time.
11. Sometimes she feels that her answer is not complete, she reads the task again.
12. She is satisfied with her work and she decides to submit. She asks how to submit her work and how to stop the application.

3.2.3 Deriving Handling Web Text Processes

The learners’ processes of Handling Web text are derived from each persona’s steps (listed in their respective scenario). In the Web text-handling model each step is turned into a node and the node is linked to other nodes via transitions. Each persona’s sequence of transitions is incorporated to form a final model. As a result, the model (as depicted in figure 3.3), consists of a combination of the Web text-handling processes for all personas, starting from when personas are given a task until they produce a written output and stop the text handling task.
3.2.3.1 Groups of Processes in Web Text-handling

Five generalized groups of processes can be derived from the Web text-handling model:

- **The first group: high-inference.** The high-inference processes include any personas’ cognitive behaviours, which can not be digitally detected or captured by a standard computer system. This is due to the computer’s lacks of ability to handle such data.
Examples of the personas’ high-inference behaviours are reading and understanding the task, timing and planning their progress.

- **The second group: seeking list of URLs.** The second group includes the personas’ ability to type keywords in the search engine; and to retrieve lists of search results supplied by the search engine.

- **The third group: reading and comprehending the Web contents.** The learners might read the content of the visited Web site in one of three ways:
  
  i. read the content of the Web page but be unsatisfied with the reading material and start the Web search again;
  
  ii. read the Web page, then copy-and-paste some of its content onto their working file;
  
  iii. read the Web content and summarize it in their working file.

- **The fourth group: writing and editing.**
  
  Learner writes or edits his or her written text.

- **The fifth group: high inference.**
  
  This group included processes that are carried out when the learner decides the task is complete. They were classified as high inference sub-processes since they are primarily related to the technical side of the software and hardware used in the experiment. For example, the learner asks a technician how to stop the software application and the learner wants to make sure all the files produced during the Web research session were saved.
3.2.3.2 Looping

There are four possible loops where the learner might repeat several processes in the Web text-handling model. The first loop might occur if the learner repeatedly selects keywords until one is identified, which seems to offer the best list of URLs. The second loop might possibly occur when the learner repeatedly selects or rejects Web sites out of the listed URLs. The third loop occurs when the learner decides to provide new keywords to the search engine and to repeat the whole process from the starting point. The fourth loop includes the looping between the processes of reading the Web content and writing. The learner decides whether to ignore, to summarize or to copy-and paste the selected content from the chosen Web page to the working text file.
3.2.3.3 The Model of Processes in Web Text-handling

The model of learners’ behaviour in task-driven Web text-handling as presented in figure 3.3 is collapsed into a simplified model of seven processes based on the groups of processes identified in the section 3.2.3.1. This is depicted in figure 3.4.

![Diagram of the model of processes in Web text-handling](image_url)

Figure 3.4 – The model of processes involved in Web text-handling
3.2.3.4 The Complexity of a Web Text-handling Scenario

In handling Web text, learners are expected to be involved in two complex activities concurrently: *information-seeking activities* and *Web text reading activities*.

**Information-seeking Activities**

Marchionini (Marchionini, 1998) defines *seeking Web information* as *an intentional activity of open-ended seeking of electronic information* and suggests that “the ability to locate and apply information is an important component of what it means to be literate”. These definitions are consistent with what was found in other reviewed literature (Broder, 2002, Comer, 2000, Marques et al., 2004).

Hendry and Harper (Hendry and Harper, 1997) describe the information seeker as “a *problem solver*, who sets goals, monitors progress, explains solutions, and optimizes for solution quality in the time available” and as “a *designer* who often produces an artefact” in task-oriented information-seeking.

Marchionini (Marchionini, 1998) describes the information seeker in a wider perspective covering pertinent factors of human-centred information-seeking in electronic environments. In the framework, he specifies six information-seeking factors as follows:

1. *Information seeker* – the person who is expected to initiate the act upon his or her information gap/problem/task.
2. *Task* – the element that triggers the information seeker to act and comprehend some document and enhance/change his or her knowledge state.
3. **Search system** – the system which is responsible for representing knowledge and regulating the information seeker on how to access and use the knowledge.

4. **Domain** – any elements of knowledge in any field.

5. **Setting** – the place or the situation where the information-seeking activities take place.

6. **Outcomes** – the products (tangible/intangible) and processes (high/low inference) produced during the information-seeking activities or after they have taken place.

Derived from the framework, in an information-seeking setting, the learner interacts with the computer search system in a task-driven manner in order to acquire domain-specific knowledge. The outcome is an artefact which consists of a trail of processes and products (e.g. in written form containing a description or summary) as described by Hendry and Harper in the previous paragraph.

**Web Text Reading Activities**

Reading is an extremely complex process (Pressley, 1997) with products (Dreyer and Nel, 2003, Snow, 2002), and there have been many attempts to define the term. A study by Harrison (2004) on reading definitions concluded that there exist two types of definition: definitions that describe the products of reading and definitions that describe the reading processes. Despite the complexity of the reading process, it consists of only three elements: the reader, the text and the activity (Snow, 2002). The activity, which is based on visual inputs, involves transactions between the published text and the reader (Goodman, 1994). Also, reading requires readers to think (Pumpfrey, 1977) and actively
to construct meaning as they read (Rose and Dalton, 2002), and also to integrate information from a variety of sources (Goodman, 1994, Harrison, 2004).

The existence of the Web presents a number of new opportunities to seek and access information. A large volume of reading materials is accessible constantly and from all over the world. As a result, reading and technology are converging in new ways in Web text reading (Schmar-Dobler, 2003, McEneaney, 2001, McEneaney, 2000a, Burniske, 2000, Eagleton, 2002).

**Refined Web Text-handling Scenario**

Combining the proposed model of Web text-handling processes (shown in figure 3.4) with previous research on information-seeking and Web text reading activities (previous two sub-sections) produces a model of Web text reading comprehension that will be used in this study (depicted in figure 3.5). In figure 3.5, Web text reading comprehension is partitioned into two groups: offline (derived from inferred data in section 3.2.1.5) and online (derived from observed data in section 3.2.1.5) Web text reading processes. The processes are partitioned into three groups: input, process and output. The offline processes (the inaccessible processes of cognition), are shown in the figure, but are not the primary focus of this research. The aim of this research is to place primary emphasis on observed rather than inferred processes.
Figure 3.5 – The model of a Web text-handling scenario (the offline materials are derived from literature, the online materials are derived from section 3.2)

Based on the model of Web text-handling shown in figure 3.5, readers are affected by various factors before and during online reading. Readers’ reading skills, vision skills,
Web skills, reading styles, goals and strategies, prior belief and the knowledge which they possess, guide them while reading and comprehending the current online text.

The pilot work with learners in a higher education setting in the discussed qualitative experiment, leads to the following reading activity cycle. Upon reading the task, learners determine the keywords. To determine the keywords, the Comprehension procedure is called where learners check their global coherence and global text structure selection. They integrate input, check input with prior knowledge and make a decision to integrate (whether to add, delete or change) the prior knowledge with the current knowledge.

Learners then determine keywords and search the Web by using a search engine. Then, learners return to the next step in the Web text Reading Procedure, which is to navigate the search list provided by the search engine. In the next step, learners select a Web page and read it. Upon reading, learners will activate the Reading Procedure, where they integrate words, phrases and sentences while at the same time checking their local reading coherence. After reading the selected Web page, learners decide whether to accept the content (copy and paste or summarize the content in the working document) or to reject the content and follow by activating the Comprehension Procedure again in order to check their current comprehension and to come up with a different keywords. The activation cycles of Web text Reading, Reading and Comprehension procedures continue until the learners are satisfied with their reading comprehension and can then produce reading comprehension activity outputs: search lists (i.e. general search URLs), selected Web
pages (i.e. specific Web pages’ URL), copy-and-pasted text and rewritten text (in the form of free-response text).

### 3.3 Supporting Learners in Handling Web Text: General Requirements

This section discusses the solution for supporting learners who encounter problems in Web text-handling in the scope of the setting where a learner is given a task which requires unconstrained Web searching but must be accomplished within a given time. The solution is discussed relative to the scope of this research, i.e. the skills which encompass the Web text reading comprehension (while evaluating search results and comprehending Web pages) and writing (while expressing Web text reading comprehension in written form).

Learners need plenty of time and space to assess and digest Web text independently of what information they need. They must learn how to find the information, evaluate how significant the found information is to their task; and how to use the information to construct their knowledge (Kuiper et al., 2005). However, it is impossible to offer immediate support manually and personally for the increasing number of learners with variable levels of skills. Alternatively, an e-feedback mechanism is capable of providing automatic assessment and personalized constructive feedback for learners (Harrison et al., 1998, Harrison et al., 2004, Harrison et al., 2006, Hawkins et al., 1993).
Derived from literature (section 2.4.1), a good practice in providing constructive feedback is to target the feedback to the weakest learners. Among the four personas described in section 3.2.2, Copy-Cathy and Choose-Charlie are regarded as the weakest learners in handling Web text, for not only did they fail to cover all requirements (as stated in the task), in their rewritten text, but also their texts are composed of much scattered word-for-word plagiarism. However, for the purpose of this work, Copy-Cathy is selected to be the primary persona (who incorporates some of the characteristics of Choose-Charlie), which the proposed support system is intended to serve.

The Copy-Cathy persona represents learners who search for various relevant Web sites to solve their Web task. However, they produce answers composed of a high percentage of sentences or paragraphs of copy-and-paste texts from the various Web resources. This raises the first requirement for the e-feedback system: the system needs to have a capability of measuring what percentage of learners’ answers are copied and from which Web sites they are copied. This will allow learners to monitor their work in order to avoid plagiarism.

The Copy-Cathy persona also represents learners who search for a single relevant Web site to solve their Web task. Consequently the answers produced are incomplete and do not cover the full requirements of the given task. This raises the second requirement for the e-feedback system: it needs to have a capability of measuring how coherent learners’ answers are and how relevant learners’ answers are against their given Web task.
The following table 3.1 (derived from the digital data described in section 3.2.1.4) shows that the computer is capable of detecting a learner’s behaviour while handling Web text. The evidence is matched against the model of Web text-handling presented in section 3.2.3.3.

Table 3.1 – Samples of evidence provided by the computer to reflect a learner’s activities in handling Web text

<table>
<thead>
<tr>
<th>Process Group</th>
<th>Group of reading sub-processes</th>
<th>Sources of evidences</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Start the reading session.</td>
<td>The task manager in the computer system will detect the selected browser is active.</td>
</tr>
<tr>
<td>2</td>
<td>High inference reading sub-processes as identified in section 3.2.2.3 (the first group).</td>
<td>Not available.</td>
</tr>
<tr>
<td>3</td>
<td>Seek list of URLs.</td>
<td>Url address with time-stamp and key words used in the search engine (listed in the Web browser’s history file)</td>
</tr>
<tr>
<td>4</td>
<td>Read the Web contents.</td>
<td>Url address with time-stamp.</td>
</tr>
<tr>
<td>5</td>
<td>Working with the writing task.</td>
<td>The task manager in the computer system detects the working file is active.</td>
</tr>
<tr>
<td>6</td>
<td>High inference reading sub-processes as identified in section 3.2.2.3 (the last group).</td>
<td>Not available.</td>
</tr>
<tr>
<td>7</td>
<td>Stop the reading session.</td>
<td>The task manager in the computer system will detect the selected browser is inactive.</td>
</tr>
</tbody>
</table>

Out of the seven processes in table 3.1, processes 1,3,4,5 and 7 have sources of evidence which can be captured digitally during the Web text-handling session. This raises the third requirement for the e-feedback system: the system needs to make use of this real-time data in a real-life setting to fulfil both requirements which were previously identified.
In conclusion, this section has identified three general requirements for developing an e-feedback system as a support tool for a task-driven learner who performs unconstrained Web search within a given time and produces a free-response text as a solution to the given task.

### 3.4 Summary

This chapter described a qualitative experiment which was conducted to obtain data on learners’ behaviour whilst handling Web text. The captured data was used to create four personas and their respective scenarios by following the Goal-Directed Approach. The personas were orchestrated in their respective scenarios to derive the processes that are involved in a Web text-handling activity. The processes were then used to develop a general model of learner’s behaviour in handling Web text. This model was combined with existing theory of Web text reading and searching to form a model of a Web text-handling that will be used in the rest of this thesis. Based on the gained understanding of learners’ behaviour, three general requirements were identified for developing e-feedback to support learners in handling Web text. The following chapter will identify the basic requirements needed for developing e-feedback according to the scope of this research.
Chapter 4

Specification of the Web Text-handling Support System

4.1 Introduction

The previous chapter identified three general requirements for developing an e-feedback system to support learners while handling Web text. This chapter focuses on identifying detailed requirements for the Web text-handling system to support the handling processes and output of Web text as described by the two models presented in previous chapter (i.e. the model of Web text-handling processes and the model of a Web text-handling scenario).

This chapter starts by providing a definition of the system in section 4.2. Section 4.3 describes the primary persona for which the proposed system will be designed along with her Web text-handling activities in a futuristic scenario. This futuristic scenario allows us to identify the main components of the proposed system. Section 4.4 then describes these components and discusses different options for their implementation. Finally, section 4.5 adds further design requirements for developing a system which follows usability and accessibility guidelines.

4.2 Definition of the Web Text-handling Support System

In this work, the Web Text-handling Support System is defined as:

*A computer application which allows learners to respond to a specific Web text-handling task on a platform where they are able to handle Web text in a real-life*
setting within a time constraint. It provides feedback to learners in real-time based on Web text processes and output data that are captured and assessed automatically.

4.3 Selected Persona and Scenario for the Web Text-handling Support System

This work aims to provide constructive e-feedback to support the weakest type of learner as represented by Copy-Cathy. She is the primary persona, which the proposed system will be designed for. This follows Cooper’s guidelines, which suggest that in order to increase system usability, the system should be developed in order to satisfy one single user, the primary persona. He argues that design for just one person has two important usability benefits (Cooper, 1999):

- “Every time you extend the functionality to include another constituency, you put another speed bump of features and control across every other user’s road. You will find that the facilities that please some users will interfere with the enjoyment and satisfaction of others. Trying to please too many different points of view can kill an otherwise good product. However, when you narrow the design target to a single persona, nothing stands between that persona and complete happiness”.

- “The broader a target you aim for, the more certainty you have of missing the bull’s eye. If you have a product satisfaction level of 50%, you can not do it by making a large population of 50% happy with your product. You can only accomplish it by singling out 50% of the people, and striving to make them 100%. It goes farther than that. You can create an even bigger success by targeting 10% of your market and
working to make them 100% ecstatic. It might seem counter-intuitive, but designing for a single user is the most effective way to satisfy a broad population" (pg. 125).

Therefore, this section uses Copy-Cathy as the primary persona and a new futuristic scenario is presented, in which Copy-Cathy uses e-feedback to improve her written essay, the product of her Web text-handling task.

**Persona Primer – Copy-Cathy**
Copy-Cathy is a first year student at the Department of Education and specializes in History. She considers herself as fairly competent in both areas: computing and socialism. She always thinks doing a ‘Web text-handling’ task is a fairly easy job. She has nothing to worry about. The Web always gives her an answer if she needs one.

**Scenario:**
1. She starts the Web Text-handling Support System.
2. She reads instructions displayed on the screen step-by-step.
3. She starts her task by clicking the *start button*, and automatically the *timer* is ticking.
4. She reads the task displayed on the active tabbed-pane twice to make sure she really understands what she has to do.
5. She starts the Web browser by clicking the *Internet Explorer button*.
6. She submits keywords to the search engine.
7. Then, she searches the list of URL addresses displayed by the search engine.
8. She selects the URL which she finds interests her most.
9. She scans the associated Web pages quickly.
10. She copy-and-pastes a few sentences to her tabbed-pane text area.
11. She rearranges sentences or rearranges paragraphs, or adds a few words or deletes a few words or just reads to make sure her text is coherent and has relevant material.
12. She repeats step 9-11 several times until she is sure that she has enough information. Once in a while, she checks her word count (by using a *word counter*) or checks her time (by looking at a *timer*) and finally goes to step 13.
13. She click the *assess* button to make sure she avoids word-for-word plagiarism and to check whether her rewritten text is coherent and relevant to the given task, then she continues her work (goes to steps 6-11 or goes to steps 7-11 or goes to step 14).
14. Sometimes she feels that her answer is not complete, she reads the task again (clicks the tabbed-pane to display the task), then she continues her work (goes to step 6-11 or goes to steps 7-11).

15. She is satisfied with her work and she decides to submit. She follows the last step in the instructions, i.e. clicks the exit button and the application stops.

4.4 The Requirements for Developing the Web Text-handling Support System

In the current setting for handling Web text (as depicted in figure 3.1), learners perform Web text-handling processes using two independent and proprietary applications: Microsoft Word (as the text editor) and Internet Explorer (as the Web browser). The present setting does not allow any e-feedback to be given to learners while they are handling Web text as both systems do not allow their real-time data to be captured, passed or processed accordingly.

In order to support learners in a real-life setting (as described in section 4.3), a feedback system must be integrated in a common platform to allow real-time Web text-handling data to be captured, passed and processed accordingly by the various components involved in the scenario of Web text-handling. Therefore, within the scope of this thesis, an e-feedback system must have the following functions (derived from the model of Web text-handling processes shown in figure 3.4 and the Web text-handling scenario model shown in figure 3.5):

- Able to provide a platform that enables learners to perform real-time Web text-handling in a real-life setting.

- Able to provide mechanisms for capturing real-time Web text-handling data in terms of its processes and output, then process the data instantly.
• Able to provide a mechanism to detect plagiarism and measure what percentage of the learner’s answer is copied and to identify from which Web sites it is copied.

• Able to provide a mechanism to measure how coherent the learner’s answer is and how relevant the learner’s answer is against their given task.

In addition, the e-feedback system must have a file system (for storing data and system information), user interfaces (for managing interactions between the system and the learner), text editing tool (to select, cut, copy, paste text, and input text), a timer and a word counter.

Figure 4.1 illustrates these components of the proposed Web Text-handling Support System.

The system will be used by three types of user: corpus builder, tutor and learner. However, the participation of the corpus builder and tutor is prior to the dissemination of the proposed system to learners. The corpus builder is responsible for building a task-specific corpus to be used by the LSA assessment technique and the tutor is responsible for preparing a Web text-handling task to be used by the system.
The file system is a place where all relevant data, information and system files are stored. The storage consists of three basic types of data and information:

1. Data produced by the learner, i.e. a file containing free-form response text (this file will be saved at specific intervals to avoid data loss in case the system breaks down).
2. Data produced by the Web browser, i.e. the browser’s history file which consists of learners’ Web browsing data.
3. Information produced by the Web i.e. text-handling mechanisms in terms of assessment results and feedback reports.

Figure 4.2 shows a screen design of the Web Text-handling Support System. Generally, the system is to be presented on a screen which is divided into three horizontal parts. The first part displays the step-by-step instructions on how to use the system. The second part
consists of three tabbed panes for displaying the task, allowing the learner to write free-
response text and providing copy-and-paste feedback in the form of highlighted text. The
third part presents the overall feedback on Web text-handling in terms of the word count
for written text (with a coloured indicator), a word count for copy-and-paste plagiarism
(with a coloured indicator), two cosine values of the LSA (which show similarity of the
free-response text to the given task and coherence) and a small tabbed-pane displaying the
addresses of the original resources where copy-and-paste plagiarism is found (if any).

Figure 4.2 – The screen of Web Text-handling Support System

The next three subsections analyse each of the software components (identified in figure
4.1) and discuss their general objectives.
4.4.1 Web Text-handling Platform Component

This section will identify and analyze the appropriate elements of the user interface (Mandel, 1997, Galitz, 1997) to accommodate the five UI functionalities needed by the learner, as presented in figure 4.1.

The five listed user interface functionalities can be grouped into the following categories based on their requirements in relation to the proposed Web Text-handling Support System:

4.4.1.1 Access to Text Based Elements

The proposed system must provide a means for the learner to display the task (i.e. a simple text editor), to enter rewritten text from Web resources (i.e. a plain text area) and to display feedback on their Web text-handling output (i.e. a plain text area). Two choices exist:

1. A simple text editor is embedded in the proposed system.
2. A separate text editor (commercial or non-commercial software developed by individuals or private organizations) is used.

Option 1 (to embed a simple text editor in the proposed system) was chosen based on the following reasons:

1. The system deals only with short-essay, unformatted, plain text so a simple text editor is sufficient for the task.
2. Providing a text box area for the Learner to rewrite selected Web resources leads to having a very flexible text format to be used by other subcomponents of the proposed system.

3. It is believed that writing in a text box is something the majority of Learners are familiar with (e.g. through use of email clients).

4. It consumes little screen space.

5. It avoids copyright difficulties that might arise if a commercial text editor were used.

6. It enables smoother transitions between the steps involved in the Web research task by grouping all the related system material in one environment. The smooth transition can be achieved by placing the simple text editor (for rewriting selected Web resources) and two text areas (for displaying the research task and feedback) side-by-side in a notebook form on a single screen.

7. Learners would not have the burden of learning new peripheral functionalities of an external text editor.

4.4.1.2 Access to a Web Browser

The proposed system must provide a means of integrating access to the Web browser. Accessing a Web browser within the proposed system can be achieved by providing a command button. By clicking the command button, the proposed system will automatically activate and bring the selected Web browser in a separate window. Thus, the learner will have concurrent access not only to the task and to the writing section, but also to the selected Web browser.
4.4.1.3 Access to the Assessment Mechanisms of Web Text-handling

The proposed system must have embedded in it a means of evaluating the Web text-handling processes and output, in order to provide feedback. Two mechanisms have been adopted for the purpose of this dissertation: assessment of the coherency and relevancy of the content, and assessment of the copy-and-paste plagiarism in the learner’s rewritten text.

A means for accessing the Web research task assessment mechanisms could be provided through another command button. This single command button tells the proposed system to retrieve the data collected by the data collection component, to convert that data into an appropriate format used within the system and to process the converted data to provide feedback. The output is displayed in the provided text area described in section 4.3.1.1.

4.4.2 Real-time Data Collection Component

This section identifies the data-collection mechanisms that are suitable for capturing the Web text processes and output. The selection process covers the following:

1. Determining what type of data is to be collected.
2. Identifying a technique to collect the data.
3. Determining a technique for processing the collected data in order to make it meaningful and useful to the different components of the proposed system.
4. Obtaining personalized data within the scope of Web text-handling.


4.4.2.1 Type of Data to be Collected

As described in section 3.2.3.4, this work places primary emphasis on observed rather than inferred Web text-handling processes and output. Thus, the proposed system deals with only two types of data:

1. Written text based on selected Web resources as the product of the Web text-handling.

2. Low inference processes and output of Web text-handling (as discussed in section 3.2.3.3).

The first type of data is captured and saved in the system’s file store and handled by the Web Text-handling Platform component through its user interface functionalities. Therefore, the Real-time Data Collection component has to deal only with the second type of data.

4.4.2.2 Identifying a Technique to Collect the Data

The aim of the Real-time Data Collection component is to obtain the group of URL addresses visited by a specific learner during the Web text-handling session along with their time stamps. There are three types of possible data collection techniques: using proxy software\(^{11}\), cookies\(^{12}\) or the browser’s history file\(^{13}\). It was decided to select the history file as the means to obtain the Web data because it has the advantages of both supporting an open-ended Web text-handling and offering personalized URL listings. An overview of all the reasons for this decision follows:


\(^{13}\) Nirsoft.net Webpage [http://www.nirsoft.net/utils/iehv.html](http://www.nirsoft.net/utils/iehv.html) <last accessed: May 11, 2007>
1. A history file which can be used for Web text-handling is automatically provided by any Web browser.

2. A list of URL addresses retrieved by a specific learner can be extracted from a history file, which is sufficient for the proposed system’s data requirements.

3. The use of a history file allows a mechanism to be embedded in the proposed system for retrieving the required data from the file’s contents. The mechanism will be automatically activated whenever the learner requests feedback.

4. It is possible to convert the history file content into a plain text format which is readable by other components in the proposed system.

4.4.2.3 Determining a Technique to Process the History File

This section discusses the techniques to be used for obtaining all the data stored in a browser’s history file. The original format of this stored data is unreadable by other software applications except the browser itself. Thus, the proposed system must provide a technique to extract the content of the history file and convert it into a plain text file format (as required by the proposed system).

However, the particular file format conversion technique is dependent on the chosen Web browser. There exists one available non-commercialized free software system, IEHistoryView, which can convert the history file and which can be integrated into the proposed system. However, it works only with the Internet Explorer browser. Hence, the limitation of browser usage is reflected in the scope of this research.
4.4.2.4 Obtaining Personalized Data

This section discusses how to extract personalized data from the browser’s history file on behalf of the learner during his/her Web text-handling session.

Generally, the data produced by the history file converter is for all users of the Internet Explorer browser. Furthermore, the life-time of data in the browser’s history file varies according to the data life-time specified in the browser. The length of its life time ranges from one day to ninety-nine days. Hence, the history file not only contains data for all its users but also that data is kept for varying lengths of time. In order to extract the personalized data of a learner within the duration of his/her online reading session, the Real-time Data Collection component must provide a data filter. Each piece of converted data extracted from the history file is in the form of a URL along with a time stamp that indicates when the Web resource was first retrieved. The role of the data filter is to extract all the selected URL addresses where the first access time falls within the specific learner’s Web text-handling duration and to store them in a specific file. There are two types of URL addresses in the filtered file:

1. URL addresses which indicate a navigation list.

   Example:
   
   http://www.google.co.uk/search?q=compression+redundancy+error+detection+correction.

   According to the model of processes in Web text-handling (as described in section 3.2.3.3), this type of URL is marked as a P3 process. The P3’s URL contains keywords used by the Learner in order to find related information using any search
engine. The P3’s URL above indicates the search engine used is Google and the key words used are compression + redundancy + error + detection + correction.

2. URL addresses which indicate a Web page.
Example: http://www.dogma.net/markn/articles/lzw/lzw.htm
According to the model of processes in Web text-handling, this type of URL is marked as a P4 process. The P4’s URL contains the following information – its resource provider, i.e. dogma; its resource file name, lzw; and the type of the resource file, i.e htm.

4.4.3 Assessment Mechanisms in Web Text-handling Platform Component

The aim of this section is to identify what type of assessment mechanisms should be adopted in the proposed system. The choices are influenced by the two types of real-time data which are made available by other components of the proposed system:

1. An essay-like written text which is produced by the Web text-handling Platform component.

2. Low inference processes in the form of a list of URL addresses which is produced by the Real-time Data Collection component.

Due to the complexities of Web text-handling as discussed in section 3.2, there are a number of possibilities for assessing its processes and products. For the purposes of this thesis, only two types of assessment mechanism will be considered (as described in section 3.3):
1. Assessment of the essay-like written form, which falls in the automatic essay assessment area.

2. Assessment of the list of URL addresses and their content, which falls in the copy-and-paste plagiarism area.

### 4.4.3.1 Assessment of the Written Essay

The essay content has various possible qualities and aspects to be assessed and could potentially have an infinite number of correct solutions. Unlike short answer marking which uses a comparison between specific strings of an answer template, the assessment of an essay is difficult and requires a more elaborate marking model (Streeter et al., 2002). Streeter et. al. describe the assessment of essays as difficult work especially in providing substantive feedback, therefore the commonly used approach of evaluating essays is primarily “content-independent writing skills such as spelling, punctuation, and grammar, rather than the more important matter of effectively conveying a message”.

Nonetheless, this work will assess content-dependent writing skills in the written essay by using LSA (as described in section 2.3.2.3). Generally, LSA offers five different semantic analyses\(^{14}\). However only two of them, specifically the sentence-to-sentence cohesion LSA (in the written essay) and the document-to-document similarity LSA (between the written essay and the given Web text-handling task) will be adopted and investigated in terms of their usability in this research.

---

4.4.3.2 Assessment of the List of URL Addresses and their Content

The process of evaluating a learner’s list of URL addresses depends greatly on the underlying concepts that the list represents. According to the model of processes in Web text-handling adopted in this research, each URL represents either a link to a Web resource (referred to as P4 in the model) or a search list which gives multiple possibilities of finding Web resources that are related to the key words presented to the chosen Web search engine (referred to as P3 in the model).

Assessment of the content of a learner’s written text has little meaning in Web text-handling if most of the text is copy-and-paste plagiarized from the retrieved Web resources. Therefore, in the proposed system, there will be a mechanism to detect copy-and-paste plagiarism in the written text. The mechanism will make use of all the P4 URL addresses of a specific learner, retrieve the content of the retrieved Web resources, and compare each of these word by word with the learner’s written text. If there is a string of more than five words that is identical to the Web resource, then the words are considered as word-for-word copied. The proposed system sets a certain limit (up to 60% as described in the task) on the word-for-word copied text in order to curb excessive direct plagiarism as “the process of deterring cheating will always be far more effective than the act of detecting the cheating, additionally the staff time and effort expended in informing students of correct academic practice is significantly lower than that needed to identify and pursue cheating students” (Jones et al., 2005). Thus, this implicitly encourages learners not only to avoid word-for-word copying but also to comprehend and summarize the retrieved Web resources before recording a summary in the given text area.
4.5 The Requirements for Designing Usable System

This thesis focuses on producing a usable system. Maguire summed up five benefits of producing a usable system: Firstly, a usable system increases productivity by allowing the user to use the system effectively. Secondly, a usable system reduces user error by minimising design inconsistencies and ambiguities. Thirdly, a usable system reinforces learning, therefore reducing the amount of money and time needed to train and support the user. Fourthly, a usable system encourages the user to accept, trust and use the system, thereby indirectly enhancing the usability of the system. Finally, a usable system promotes the good reputation of the system developer (Maguire, 2001).

Jordan describes ten principles of designing a usable system, of which nine are relevant to the scope of this work and are to be adopted as part of the requirements for the proposed system in order to increase its usability (Jordan, 1998):

- **Consistency.** The system must be consistent ensuring similar tasks are done in similar ways. Therefore, users are more adaptable to the system commands and functions.

- **Compatibility.** The system must be compatible with users’ expectations based on their knowledge of using other products in real-life activities. Therefore, users must not be confused by experiences with other previously used systems.

- **Consideration.** The system must consider users’ physical resources during interaction, i.e. their eye and hand movements. In this way, the users will be more comfortable during their interaction with the system.

- **Feedback.** The system must acknowledge the action taken by the user and provide appropriate feedback in relation to their actions. Therefore, the users know their
actions are accepted and they will wait for the system to respond accordingly without having to repeatedly issue the same command.

- **Error prevention and recovery.** The system must prevent or minimise the chance of users making errors. However, if an error occurs, the system must be quickly and easily recovered by the user. Thus, user frustration is minimised.

- **User control.** The system must allow the users to have as much control as possible over their interaction especially in terms of pacing and timing of their actions. In this way, the users are more relaxed and progress steadily at their own work pace.

- **Visual clarity.** The system must present information clearly so that the users can understand instantly. Therefore, the users are less confused about the system instructions.

- **Prioritization.** The system must be organized in terms of the priority of its functionality and information in descending order. Therefore, the users encounter and accomplish the functionality with the highest priority first.

- **Explicitness.** The system must have a clear presentation of commands so that the commands signify explicit functions to users. Hence, the users have a clear understanding of how to operate the system successfully.

Furthermore, the usability of a system is closely related to its accessibility. Galitz defines accessibility as “*providing easy access to a system for people with disabilities by minimizing all barriers that make a system difficult or impossible to use*” (Galitz, 2002). At one hand, usability is concerned about how individuals accept and use the system with effectiveness, efficiency and satisfaction in order to achieve specified goals in a specified
context of use (Jordan, 1998). On the other hand, in the broadest sense, accessibility is concerned about how many people regard the system as a usable system.

Lazar regards accessibility as a means to allow all people to have full access to information and communication tools regardless of any physical, motor, or cognitive disability (Lazar, 2002). The system should be flexible with regard to input (should not be limited to a keyboard and mouse) and output (should not be limited to a screen and ink-jet printer). The same requirement should also apply to the use of font sizes, colours, and response times. Further, Galitz suggests that the system developer should provide compatibility with installed accessibility utilities and should consider accessibility issues during system planning, design and testing. In addition, the developer should provide customizable interfaces, follow standard Windows conventions and use standard windows controls (Galitz, 2002). In other words, by improving accessibility of a system, the system is usable for a wider range of users (Theofanos and Redish, 2003).

The UK’s 2001 Special Educational Needs and Disability Act (SENDA) has made obligatory for Information technology and computing services (which include e-learning and e-assessment areas) not to discriminate against disabled persons by making related systems accessible to them (Seale, 2004). Newell and Gregor identify three trends in designing systems related to accessibility: mainstream system design, which often caters exclusively for able-bodied users; system design exclusively for people with disabilities and universal system design which caters for all users (Newell and Gregor, 2000). Vanderheiden refers to universal system design as universal usability, also known as
“design for all” or “inclusive design”. He defines universal usability as “a focus on designing products so that they are usable by the widest range of people operating in the widest range of situations as is commercially practical” (Vanderheiden, 2000).

One advantage of designing a universally usable system is that both disabled and able people will experience the same system quality regardless of having different levels of abilities. However, a major drawback of having a universally usable system is that it is very difficult to design a system to satisfy all users.

Wiles points out that “it has not been possible to find guidelines specifically for computer-based or online assessment”. Hence, the collaboration of the Association for Learning Technology (ALT), Joint Information System Committee (JISC) and TechDis (JISC funded group) aims “to be the leading educational advisory service working across UK, in the field of accessibility and inclusion”. Up to date, they provide a few suggestions, of which four are related to the scope of this thesis (Wiles, 2002):

- **Transition times**. Allow learners to appropriately set their own pace while answering questions.
- **Keyboard**. Allow learners to navigate on the screen using a keyboard rather than a mouse.
- **Assistive technology**. In case of using assistive technology, learners should be allowed ample time to learn to use new technology.
- **Active participation**. Encourage disabled learners to participate and advise in the design of any proposed e-learning and e-assessment systems.
However, only points 1 and 2 will be added as part of the requirements of the proposed support system for this thesis. As there are no disabled learners participating in the current research, points 3 and 4 will be considered in future research in order to give input for the design and to test the proposed system.

4.6 Summary

This chapter identified requirements for developing the Web Text-handling Support System. It explained the need for an integrated Web text-handling platform that allows learners to perform tasks in Web text-handling, and to have their data collected and their performance assessed in real-time. Furthermore, this chapter recognized the principles of system usability and accessibility, thus adopted the relevant principles to the scope of this thesis as part of the requirements of the Web Text-handling Support System. The next chapter will present the design and implementation of the proposed support system.
Chapter 5

The Development of the Web Text-handling Support System

5.1 Introduction

The previous chapter has set requirements for developing a system to support learners in handling Web text. This chapter presents the development of such a system. In this work, the system is referred to as the Web Text-handling Support System. It incorporates both of the previously described models: the model of processes in Web text-handling (figure 3.4) and the model of the Web text-handling scenario (figure 3.5).

The main objective for the development of the proposed Web Text-handling Support System is to allow research on providing e-assessment and e-feedback for learners who are handling Web text. Since the system is intended for providing the infrastructure that is needed to carry out the research of this thesis, it is designed to be flexible and able to accommodate changes in future system requirements. In addition, even though the system is a prototype, it has a functional design that can be implemented and tested in a realistic setting. The development of the proposed system involves two software activities which are closely related but demand different skills (derived from (Horrocks, 1999)):

i. User interface design – user interaction design which involves the look and feel and the behaviour of the system in response to a user during interaction with the system.
ii. Software construction – the software process which involves designing, coding and testing the software that realizes the user interaction.

The next section illustrates the general requirements (in the form of a UML Use Cases diagram) which must be fulfilled by the proposed Web Text-handling Support System in order to achieve the main objective of this work. Section 5.3 describes how learners’ and system activities are identified for the design specification. This design specification is illustrated as a UML activity diagram in section 5.4. Based on the activity diagram, section 5.5 describes a comprehensive design plan for the proposed system. This is followed by section 5.6 which describes a comprehensive implementation plan based on the design. The implementation of each component in the proposed system is discussed in detail in sections 5.7 to 5.10. Finally, section 5.11 provides a summary of the development of the proposed system.

5.2 General Requirements for the Web Text-handling Support System

Derived from section 4.4, the requirements can be generalized to three use cases of the Web Text-handling Support System: corpus builders, tutors and learners as shown in figure 5.1.

Corpus builders are implicit actors who are responsible for building a task-specific corpus to be used by the LSA assessment technique which checks the relevance and similarity of the learners’ Web text-handling output which is in the form of free-response texts.
Tutors are implicit actors who are responsible for preparing the Web text-handling tasks to be used by the proposed system and to be presented to learners.

Learners are the explicit and primary actors who use the system to help them in their Web text-handling task.

Figure 5.1 – Use cases diagram for Web text-handling scenario adapted from FREMA and (Millard et al., 2006)

---

5.3 Design Specification for Learners’ and System Activities

Deriving learners’ activities is a complex and repetitive process. The construction of design specifications in this work is based on both the flow of learners’ activities shown in figure 3.3 and the futuristic scenario described in section 4.3. Next, the following process was used to obtain lists of learners’ and system’s activities as illustrated in figure 5.2 (Isaacs, 2002, Mandel, 1997, Millard et al., 2005):

1. Read through both the activities in figure 3.3 and in the futuristic scenario section 4.3. A noun and verb analysis is used as a technique to identify the interactions between learner and system (i.e. indicated by the verb) and the data model (i.e. indicated by the noun).

2. Build a list of activities in sequential order by combining interactions from both figure 3.3 and the futuristic scenario (section 4.3). Any common activity (found both in figure 3.3. and in the scenario) will be grouped.

3. Turn groups of closely related interactions into one activity.

4. List the groups of activities and identify is who responsible for the activities, i.e. learner or the system.

5. Test the completeness of the design by working with the four scenarios in section 3.2.2.

6. Revisit the specification and add requirements (section 4.4) into the design.
Figure 5.2 – The factoring of the learner use case into a list of learner and system activities

<table>
<thead>
<tr>
<th>Learner’s Activities of Web Text-handling</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Start the Web Text-handling Support System.</td>
</tr>
<tr>
<td>• Read instructions displayed on the screen step-by-step.</td>
</tr>
<tr>
<td>• Start Web text task by clicking the <strong>start button</strong>, and automatically the <strong>timer</strong> is ticking.</td>
</tr>
<tr>
<td>• Read the task displayed on the active screen twice.</td>
</tr>
<tr>
<td>• Start Web browser by clicking the <strong>Internet Explorer button</strong>.</td>
</tr>
<tr>
<td>• Put keywords into search engine.</td>
</tr>
<tr>
<td>• Evaluate search engine results.</td>
</tr>
<tr>
<td>• Search the list of URLs displayed by search engine.</td>
</tr>
<tr>
<td>• Select URL which she finds interesting most.</td>
</tr>
<tr>
<td>• Read content of Web page.</td>
</tr>
<tr>
<td>• Ignore content of Web page.</td>
</tr>
<tr>
<td>• Summarize content of Web page.</td>
</tr>
<tr>
<td>• Copy-and-paste content of Web page.</td>
</tr>
<tr>
<td>• copy-and-paste a few sentences to her screen text area.</td>
</tr>
<tr>
<td>• Revise or transform task (rewrite free-response essay).</td>
</tr>
<tr>
<td>• Rearrange sentences or rearranges paragraphs, or adds a few words or deletes a few words or just reads to ensure her text has coherent and relevant materials.</td>
</tr>
<tr>
<td>• Repeats some of previous steps several times until enough information. Checks her word counts (<strong>word counter</strong>) or checks time (<strong>timer</strong>) or just continue to the next step.</td>
</tr>
<tr>
<td>• Use e-feedback system to avoid word-for-word plagiarism and to check whether the rewritten text is coherent and relevant to the given task, then continue.</td>
</tr>
<tr>
<td>• If the answer is not complete, reads the task again (click the screen for displaying task), then continue work.</td>
</tr>
<tr>
<td>• If satisfy with the work, then submit the work to the system.</td>
</tr>
<tr>
<td>• Check if task is completed.</td>
</tr>
<tr>
<td>• Stop.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Learner’s and System’s Activities of Web text-handling</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Start the Web Text-handling Support System (learner and system).</td>
</tr>
<tr>
<td>• Enter identification (learner).</td>
</tr>
<tr>
<td>• Start Web-handling session, timer is activated (learner and system).</td>
</tr>
<tr>
<td>• Read task (learner).</td>
</tr>
<tr>
<td>• Handle Web text (learner).</td>
</tr>
<tr>
<td>• Write essay (learner).</td>
</tr>
<tr>
<td>• Assess Web text-handling output (learner).</td>
</tr>
<tr>
<td>• Plagiarism detection (<strong>VQD</strong>) and relevancy and similarity check (<strong>LSA</strong>) are activated (system).</td>
</tr>
<tr>
<td>The <strong>VQD</strong> activates the Real-time Data Collection, <strong>RDC</strong>.</td>
</tr>
<tr>
<td>• Read feedback (learner).</td>
</tr>
<tr>
<td>• Stop (learner and system).</td>
</tr>
</tbody>
</table>

![Diagram](image-url)
5.4 Building a UML Activity Diagram for the Web Text-handling Task

Derived from the literature, the Web text-handling task consists of the processes of understanding the need to perform the task, selecting the material or resource, formulating the query to feed to the Web search engine, executing the search query, examining the result, extracting Web text and forming a collection of free-response texts; adding, deleting or arranging the collected free-response texts; comparing the texts with the task and, if the texts are satisfying; stopping the Web text-handling processes or if not then repeating all, or part, of the processes in a cyclic manner. In short, the processes of handling Web text are in the form of interactions between learners and the Web browser. However, the interactions are dependent on how the learners act and react in non-deterministic and opportunistic ways (Marchionini, 1998, Hendry and Harper, 1997, Marques et al., 2004, Broder, 2002, Gardner et al., 2002, Protopsaltis and Bouki, 2005, Henry, 2006).

The list of learner and system activities in figure 5.2 is converted into a UML Activity Diagram of the proposed Web Text-handling Support System (WTSS) in figure 5.3. Learners start the WTSS system by identifying themselves to the system (for data storage purposes) and by explicitly activating the system to start the Web text-handling session. Next, the learners will interact with the WTSS system via its user interface in non-deterministic and opportunistic ways by choosing process 1, 2, 3, 4 or 5 until they exit the Web text-handling session.
Figure 5.3 – UML Activity Diagram for the Web Text-handling Support System (WTSS)
5.5 A Comprehensive Design Plan for the WTSS System

A comprehensive design plan for the proposed WTSS system is derived from the UML activity diagram (figure 5.3) and is illustrated in figure 5.4. The corpus builder produces a task-specific corpus, stored in the system storage (the system’s file directory), to be used by the LSA sub-system. The tutor prepares a Web text-handling task, stored in the system storage, to be used by Web Text-handling Platform (WTP) sub-system. The learner interacts with the WTSS system via the WTP’s user interface. The learner uses WTP to perform a Web text-handling task, assess his/her task performance and read feedback of the assessment. The learner’s real-time data is captured by the Real-time Data Collection (RDC) sub-system and assessed by the Verbatim Quotient Detection (VQD) and LSA sub-systems.

Figure 5.4 – A comprehensive design plan of the WTSS system
The overall system design is decomposed into four phases of sub-system developments: WTP system, RDC system, VQD system and LSA system.

The first phase involves the development of the first system release, the WTP system. Its function is simply to create a platform where learners can interact with the WTSS system, i.e. to read a Web text-handling task, retrieve Web resources, rewrite texts which are discovered within the Web resources, assess their rewritten text and read the feedback from the assessment.

The second phase involves the integration of the RDC system in the previous system release. The function of the RDC system is to collect real-time data (processes) during the Web text-handling duration. The data is then processed in order to obtain those URL addresses which have been visited by learners.

The third phase involves the integration of the VQD system in the previous system release. It upgrades the second release by adding a function to deal with detecting the verbatim quotient based on the collected real-time data.

Lastly, the fourth phase adds a function using the LSA technique to assess the sentence-to-sentence cohesion in the rewritten text, and document-to-document similarity between the rewritten text and the Web text-handling task, in relation to the task-specific corpus.
5.6 A Comprehensive Implementation Plan for the WTSS System

The WTSS system is implemented as illustrated in figure 5.5. It consists of four system components. Three of the sub-systems, WTP, VQD and LSA, are written exclusively in Java, built upon the Java 2 Platform and were tested under the Windows XP operating system. The RDC sub-system is a freeware system obtained from http://www.nirsoft.net/utils/iehv.html. The WTP sub-system acts as the entry point to the WTSS system and integrates the other three systems through the use of their package interfaces.

![Implementation Plan of WTSS System](image)

5.6.1 Overview of Data Flow within the WTSS System

The relationship between the implemented components in the WTSS system is determined by the way Web text data is produced and used within the system. Figure 5.6 shows the data flow between the different sub-systems.
The **WTP** system allows a learner to read the Web text-handling task, access the Internet, retrieve the Web resources, digest the discovered information, select and rewrite the information from the retrieved resources, make a request for the research task assessments and retrieve the assessment feedback. As a result of the Web text-handling session, the learner produces a written text. Whilst the learner is performing the Web text-handling task, the browser is tracking the learner’s navigation trail and saving it into the browser’s history file. Upon receiving a request for assessment feedback, the **RDC** system retrieves the history file, converts its format, and submits the converted history file to the **VQD** system. The **VQD** system retrieves both the content of the history file and learner’s rewritten text, and then performs verbatim copying analysis. At the same time, the learner’s rewritten text is used by the **LSA** system to perform the sentence-to-sentence cohesion and document-to-document similarity assessment. Both results obtained from

---

**Figure 5.6 – The flow of internal and external data within WTSS system**
the VQD and LSA systems are made available to the WTP system in order to present the assessment feedback to the learner.

The WTSS system is composed of four sub-systems and implemented in two Java packages (Verbatim and LSA) and a stand alone software system, RDC. Figure 5.7 shows how the packages and the RDC software are related.

The Verbatim\textsuperscript{16} package, which houses the WTP and the VQD systems, contains 4 sub-packages which are relevant to the scope of this thesis: UI, temp, parser and searcher. UI is a package that provides the user interface for the Verbatim system. The temp package models the static information of the verbatim quotient and manages temporary information. The parser package implements parsing procedures for both a user’s free-response text and also

\begin{figure}[h]
\centering
\includegraphics[width=0.8\textwidth]{figure5_7.png}
\caption{The packages and stand alone system in the WTSS system}
\end{figure}

\textsuperscript{16} Initially, Verbatim was developed by Campo Millan at the University of Nottingham, 2005.
Web texts. Finally, the searcher package implements the comparison between text files and acts as a filter for classifying the type of Web texts, i.e. HTML, plain, pdf and etc.

The RDC system is stand-alone freeware\textsuperscript{17} that comes in the form of an executable component.

The LSA\textsuperscript{18} package contains 2 sub-packages, Jama and Mat. Jama is a package that provides classes for constructing and manipulating real, dense matrices. The Mat package contains classes which represent functions that are used to support the implementation of the LSA package.

In principle, the use of packaging separates each system component so that each implementation can work and be tested independently and therefore is easier to maintain if there is any modification to be made in the future. However, in practice, the WTP system (where all the Graphical User Interfaces for the WTSS system resides) is built within the package implementing the VOD system.

5.6.2 The Chosen Language for Implementation

The WTSS sub-systems were analysed, designed, developed, and tested independently and sometimes concurrently. The need to ensure a smooth transition along the software

\textsuperscript{17} Nirsoft Webpage \url{http://www.nirsoft.net/utils/iehv.html} <Last accessed: 11/05/2007>

\textsuperscript{18} LSA was developed by Francois Rossard at the University of Nottingham, 2005.
processes for all the sub-systems led to the need to choose a programming language which was able to give maximal support before, during and after the implementation phase.

The Java language was chosen for implementation for the following reasons:

1. Java is an object-oriented language. Since all the involved software components are analysed and designed using an object-oriented approach, Java offers a smooth transition between the design and implementation of the system.

2. Java is a simple yet powerful programming language. It offers a variety of support to cater for the complexities of complex problems.

3. Java provides a wide range of GUI components which suit the needs of the WTSS system.

4. Java offers a high level of system portability and this attribute is carried and reflected in the WTSS system.

In summary, the implementation of the WTSS system can take advantage of all of the Java attributes to enhance its qualities.

5.7 The Development of the WTP System

The WTP system is the only sub-system in the proposed WTSS system which deals with interaction with learners, thus it is designed and implemented by using a user interface development approach, which main activity deals with designing and implementing user interface screens to accommodate users’ needs and satisfaction. Derived from Galitz (2002) and from the requirements in sections 4.4.1 and 4.5, many considerations must be applied when designing and implementing the user interface screens. Rational and rules
for an effective design and implementation involve the organization of graphical screens to encourage the fastest and most accurate comprehension and execution of interface functionality. The next two sections describe consecutively the design and implementation of the WTP system through two iterations. In this work the design and implementation of WTP system is built as part of the VQD system.

5.7.1 The First Design and Implementation of the WTP System

The first attempt of designing and implementing the WTP system started with sketching its screens. Based on the requirements in section 4.4.1 and figure 5.3, there are at least eight Web text-handling steps: identify learner to the system, starts Web text-handling session, read Web text-handling task, write free-response text (essay), handle Web text, assess written essay, retrieve feedback and exit the Web text-handling session. Figure 5.8 illustrates the conceptual design and implementation of the WTP system’s screens. There are two main objectives of the design and implementation: to organize and present the screens according to the eight Web text-handling steps, thus making them easier for learners to follow; and to provide a wide text area for learners to copy-and-paste Web text and to write free-response text.
Figure 5.8 – The conceptual design and implementation of WTP system’s screens

Figure 5.9 illustrates the first screen of the WTP system which is implemented based on the conceptual design depicted in figure 5.8.
Figure 5.9 – The first screen in WTP system
The first screen presents the general instructions for the Web text-handling task (in this software development phase, Web text-handling was referred to as *online reading*). The second screen displays the Web text-handling task.

The third screen allows learners to identify themselves to the system for data storage purposes and enables learners to start and stop the Web text-handling session in their own time. Upon learners starting their Web text-handling session, the screen activates its timer. The fourth screen presents a command button for learners to launch the Internet Explorer browser.

The fifth screen presents an empty text area for the learner to copy-and-paste Web text or to write free-response text. The sixth screen presents a command button for learners to assess their free-response texts. The seventh screen presents feedback for the assessment. The eighth screen allows learners to exit the system.

The ninth screen displays the availability status of the feedback: *available* or *unavailable* (available means the assessment is completed successfully; therefore the feedback is available and is displayed on the seventh screen and unavailable means the assessment is unsuccessful due to learners failing to start their Web text session in the first place). The last screen presents the general description of the system (about the system).

Except the third (start and stop the Web text-handling session) and the eighth (exit the system) screens, the other screens can be accessed as many times as learners’ desire.
Figure 5.10a through 5.10e illustrate all the screens for the WTP system. They are arranged in the order of *left to right* according to the sequence of screens (i.e. from *top to bottom* order) in figure 5.9.
Figure 5.10c – Screen 5 and screen 6 of the implemented WTP system

Figure 5.10d – Screen 7 and screen 8 of the implemented WTP system
To evaluate the usability and accessibility of the WTP system, one faculty member and fifty-seven first-year undergraduate students who were taking a Human-Computer Communication skills module (year 2005) in the School of Computer Science at the University of Nottingham, England, were asked to use the system for their first time for one hour to complete a Web-handling task (this evaluation will be fully described in section 6.2).

Although all the participants were able to use the system accordingly and complete their Web text-handling task within the given time constraint, a few problems were identified and a number of students had complaints about the system.

The first problem is about the use of a text area for learners to collect Web texts and write their free-response texts. About five to ten students preferred to use Microsoft Word as their text editor because they were more familiar with the application than the provided...
text area. At the end of the session, they were asked to copy-and-paste their work to the text area, as otherwise the system is unable to assess their work. The second problem is about the number of the system’s screens. More than half of the students did not like how the steps are presented to them by using the sequence of screens. They said that there were too many screens to navigate within the system that made them confused with too many opened windows of Web pages. As a result they could not remember which screen they were in and which Web page they were currently referring to.

The problem regarding the students’ preference for Microsoft Word over the system’s text area is solved by allowing the students to use their preferred text editor. However, they need to remind themselves to copy-and-paste their work to the text area at the end of their Web text-handling session to allow the system to store their work in its file storage for further reference. The problem of having ‘too many system screens’ is solved by designing and implementing a new user interface for the system, which will be described in the next section.

5.7.2 The Second Design and Implementation of the WTP System

The second attempt of designing and implementing WTP system started with sketching its new screens. The main objective of the second design and implementation was to minimize the number of screens used in organizing and presenting the eight Web text-handling steps and Web text-handling activities. In this attempt, the number of screens used is reduced to three. The system retains the first and the last screens of the previous version. However, all the Web text-handling steps (presented in the form of Java GUI
command buttons, labels and text areas) and Web text-handling activities are organized on a single screen, i.e. the second screen. The screen is divided into three sections: the area for presenting the Web-handling steps; the area for displaying task, text and verbatim feedback; and the overall feedback analysis for the Web text-handling assessment. Furthermore, the system is enhanced with an automatic assessment at every twenty minute interval regardless of whether the students request for an assessment or not.

Figure 5.11 illustrates the second conceptual design and implementation of the WTP system’s screen.

Figure 5.11 – The second conceptual design and implementation of WTP system’s screen

Figure 5.12 illustrates the second screen of the WTP system which is implemented based on the conceptual design depicted in figure 5.11.
Figures 5.13 to 5.17 shows consecutive designed and implemented screens of the WTP system according to the steps learners should follow during their Web text-handling session:

The first, figure 5.13, shows the ‘Read Me’ step where the learner may find an overview of procedures to follow in order to use the WTSS system.
The second, figure 5.14 illustrates the first step of the Web text-handling procedures. In ‘Step 1’, the learner is expected to identify him/herself to the system as required on the screen. Then, in ‘Step 2’, the learner is expected to read the task displayed in the ‘Read Task’ text area.
The third, figure 5.15, presents the free-response text area on the screen after the learner proceeds with Step 3. When the learner clicks the ‘Start’ button to indicate the beginning of his or her Web text-handling session, the reading clock, which is situated on the top right of the text area, starts ticking.
Figure 5.15 – The ‘Step 3’ screen of the WTSS system

The fourth, figure 5.16, depicts the screen after the learner proceeds with ‘Step 4’ and ‘Step 5’ of the procedures shown on the screen. The learner activates the Internet Explorer browser, searches the Web resources based on the given task and writes free-response text based on the searched information in the provided ‘Write Free-Response Text’ area.
Figure 5.16 – The ‘Step 4’ and ‘Step 5’ screen of the WTSS system

The fifth, figure 5.17, shows the screen after the learner has proceeded with ‘Step 6’ of the procedures shown on the screen. The WTSS system has assessed his/her Web text-handling processes and output, and returned the feedback as shown on the screen below. The underlined words in the ‘Read Assessment Report’ text area indicate verbatim copied text. The ‘OVERALL WEB TEXT-HANDLING ANALYSIS’ area presents the feedback. The feedback is interpreted as “the free-response text consists of 267 words, of
which 265 words are considered as verbatim copied. The percentage of the overall verbatim-copied text is 99%. The LSA sentence-to-sentence cohesion value for the free-response text is 0.3003 and the LSA document-to-document similarity value between the free-response text and the given task is 0.5518 (the LSA value is in the range 0-1, where 0 indicates no coherence/similarity and 1 indicates perfect coherence/similarity). The texts are copied 59% and 39% respectively from 2 main Internet sources. The 59% is copied from the URL ‘http://en.wikipedia.org/wiki/Lossy_data_compression’.

Figure 5.17 – The ‘Step 6’ screen of the WTSS system
The learner may repeat step 2, 5 or 6 as many times as he or she chooses until he or she is satisfied with the product. Then, he or she can terminate the system execution by clicking the ‘Stop’ button in ‘Step 8’.

To evaluate the usability and accessibility of the second version of the WTP system, fifty-nine first-year undergraduate students who were taking a Human-Computer Communication skills module (year 2006) in the School of Computer Science at the University of Nottingham, England, were asked to use the system for their first time for one hour to complete a Web text-handling task (this evaluation will be fully described in section 6.3).

Although all the participants were able to use the system accordingly and complete their Web text-handling task within the given time constraint, one major problem was identified and more than half of students complained about it. This was to do with the automatic assessment at every twenty-minute interval regardless of whether the students have requested for an assessment or not. The issue is that it takes sudden control of the students’ work whereby the students have to wait until the assessments are over. As a Web text session proceeds, the time taken to assess the work becomes longer as the number of web pages visited by the student is increased. Thus the students become frustrated and restless. Consequently, the automatic assessment at every twenty-minute interval has been removed from the current WTP system.
The previous problems did not repeat in this session. Only two students had to re-do their work since they proceeded with the Web text-handling task without reading the instructions carefully, i.e. they did not start their Web text-handling session by skipping step 3. About fifteen students have asked questions however the questions were regarding their lack of understanding of what the Web text-handling task was all about or how they were expected to answer the question.

5.8 The Collection of Learner’s Real-time Data

As described in section 5.4, the processes of handling Web text are in the form of interactions between learners and the Web browser. Derived from the flow of internal and external data within the WTSS system in figure 5.6, the real-time data are produced during learners’ interactions with the system and the browser.

When learners interact with the WTSS system via its user interface (managed by the WTP system), they produce real-time data in the form of free-response texts i.e. the texts in the text area, which are collected by learners from Web pages or are written by learners using their own words. Capturing the free-response texts in real-time is implemented in the WTSS system by putting a segment of Java code in the WTP system to automatically save the texts in the “Write Free-Response Text” text area (refer to the screen in figure 5.15) in the WTSS system’s directory (as text files which are named as ‘minute-the-texts-are-saved.txt’) at every one-minute interval (to prevent data loss in case the system is halted accidentally due to the system or power failure).
When learners interact with the Web browser (by typing a URL in the Web browser’s address bar or click on a link in the browser), they produce real-time data in the form of a list of URL addresses. These addresses are automatically added to the history index file and saved on the computer which is currently used by the learner. However, the Web browser does not allow viewing or editing of the content of the history file. Further, the browser saves the list of URL addresses for all of its users in the same history file. Capturing the list of URL addresses for specific learners during each of their Web text-handling session in the WTSS system is achieved via the use of the IEHistoryView system. This system is developed and distributed as a freeware utility system by Nir Sofer\textsuperscript{19}. The system developer describes the IEHistoryView system as a system that reads all information from the history file on a computer and allows one or more URL addresses to be selected and saved into a text file. Therefore, the list of URL addresses in the saved text file can be used by other systems. The list is then filtered by the URL Filter (part of WTP system) in order to select URL addresses which are visited by the specific learner only within their Web text-handling session. This uses the timer, part of the WTP system, which records the duration of Web text-handling sessions. The filtered URL addresses are stored as a text file in the WTSS system’s directory and used by other sub-system components (VQD specifically).

\textsuperscript{19} NirSoft Web page - \url{http://www.nirsoft.net/} (Last accessed: 06/03/2008)
Figure 5.18 provided an overview of this process, showing how real-time data are captured by both the WTP and RDC systems and are made available to other components in the WTSS system.

![Diagram of the process of collecting learners' real-time data captured by the WTP and RDC sub-systems in the WTSS system]

**Figure 5.18 – The process of collecting learners’ real-time data which are captured by the WTP and RDC sub-systems in the WTSS system**

### 5.9 The Integration of the VQD System

This section discusses how the VQD system is integrated as part of the WTSS system. The VQD system is developed based on the Verbatim Engine system. Millan has developed the Verbatim Engine system in-house, which detects free-text plagiarism and it is operated on a standard personal computer (Millan, 2005). Although the Verbatim Engine system is a stand-alone system with its own user interface, it allows other systems to access its core engine via its Java package and class interfaces. This work adopts part of the Verbatim Engine system in developing the VQD system by both using directly and modifying its Java
classes for detecting verbatim copying while learners are handling Web texts via the WTSS system.

The Verbatim Engine system offers two methods of accessing its verbatim copying core engine:

• **Local.** This method detects verbatim copying by comparing a text file with other text files located on the same computer in a directory specified by the user.

• **Internet.** This method detects verbatim copying by comparing a text file with Web texts. The system offers two detection mechanisms: by manually providing a URL address via its user interface, then the system compares the text file with the Web text retrieved from the URL; and by using a search engine to repetitively try-and-locate as many as possible plagiarized sources (based on a string of words extracted from the text file) and compare with the text file.

The VQD system adopts the method of detecting Internet resources, specifically the one using specific a URL address because it is relevant to the scope of this work. However, this work has modified the mechanism so that it uses the URL addresses which are automatically produced by the RDC system.

The advantages of integrating the Verbatim Engine system in the VQD system are as follows: firstly, the Verbatim Engine system is built in-house as an open source system, thus the system is modifiable to suit the need of this work at no cost. Secondly, unlike most of plagiarism detection systems (as reviewed in chapter 2) which are hosted on servers, the Verbatim Engine system is built as stand-alone software, which is portable and usable on
most standard computer systems. Therefore, the WTSS system which integrates the VQD system is generally accessible to wider groups of users who do not have access to servers. Thirdly, the Verbatim Engine system is a stable and reliable system that performs according to what is needed in the WTSS system. Therefore, adopting this system in the WTSS system has saved development time and effort.

Nevertheless, adopting the Verbatim Engine system in the VQD system has disadvantages because the development of VQD system is influenced by the original design and implementation of the Verbatim Engine system. Further, there exists a conflict in how the WTP system counts the number of words in a learner’s free-response text and how the VQD system counts the number of words in retrieved Web texts. A few modifications have been made on the implementation of parsers (in both systems) to improve their synchronization by using the word counter in Microsoft Word as the benchmark. Figure 5.20 describes how the VQD system processes a learner’s overall verbatim detection quotients in the WTSS system.
Upon a learner’s request for the WTSS system to assess their Web text via the WTP system, the WTP system retrieves the learner’s real-time data which is stored in the WTSS system’s directory. For each learner’s URL addresses, the WTP system passes both the URL and the learner’s free-response text to the VQD system. Next, the VQD system retrieves a Web page (which is specified by the URL), parses its texts, then compares with the given free-response text. Then, the VQD system passes verbatim detection quotient results in the form of how many words and percentage of verbatim copying there are for the specific Web page to the WTP system for recording. After all the learner’s URL addresses have been processed, the overall result for verbatim copying will be presented to the learner via the WTP system and at the same time the results are stored in the WTSS system’s directory for each learner.
The next two sections will describe generally how the Verbatim Engine system works and how it is implemented. The Verbatim Engine system is built as a Java package which is referred to as Verbatim package. The discussion starts with the overall functions of the package, followed by how these functions are distributed among Java sub-packages and how the sub-packages are implemented.

5.9.1 The Distribution of the Functions

There are two essential functions of the Verbatim package. Firstly, its function is to provide a platform for learners to produce a free-response text while handling Web text, referred to as WTP system. Secondly, its function is to evaluate learners’ free-response text in terms of verbatim copying against the retrieved Web resources, referred to as VQD system. The two functions are further divided into four supporting sub-functions. One of the functions acts as the WTP system and the rest of the functions represent the VQD system. The functions are as follows:

1. **UI** – The UI functions as an entry point for the learner to use the WTSS system. It contains all graphical user interfaces needed by the system and acts as WTP system.

2. **Parser** – The Parser is responsible for extracting text from Web resources at locations specified by any URL addresses collected by the RDC system and putting the associated text in a temporary file.

3. **Temp** – Temp handles temporary files in terms of creating a temporary file and saving the extracted Web resources into it, recording similarity indexes against the learner’s free-response text along with URL addresses, and removing the temporary file when it is not needed.
4. **Searcher** – Searcher performs several functions encompassing the following:

   i. Initiating the search process for all URL addresses collected by the RDC system one at a time.

   ii. Loading the learner’s free-response text and then breaking the text in the file into individual words. Each word has its own record in terms of whether it is a verbatim word or not. Each word has a list of information of *verbatim coincidence* which records the location of the first verbatim copied word, the number of words copied, the location of the URL’s temporal file in the WTSS system’s directory and the URL address of the Web-page where the percentage of the verbatim copying is recorded.

   iii. Loading each of the Web resources from a temporal file and then breaking the text in the file into individual words.

   iv. Determining the verbatim copying between the list of the learner’s words and the list of words in the Web text (determined by the selected URL address). This function is expected to compare the learner’s free-response text with each of the texts from the Web, which the learner retrieved in their Web text-handling session.

   Figure 5.21 illustrates how the comparison process is done. The comparison makes use of the content of both files which are converted into individual words. Each word is individually stored in two vectors. For each comparison, the percentage of verbatim copying is recorded in relation to each of the words in the free-response text in the form of a vector.
5.9.2 The Implementation of the Functions

All the functions of the `Verbatim` package are implemented in sub-packages. This section will describe how all the functions are implemented by using the Java language. Figure 5.21 illustrates the overview of the implementation.
The UI sub-package contains all the classes for representing the WTSP system. It is implemented in 14 Java concrete classes and has 12 class interfaces. There are approximately 86 methods, 24 attributes and 7824 lines of code.

The Parser sub-package contains only 1 class for extracting text from a specific URL and stores it in a temporary file. It is implemented in 1 Java concrete classes and has 1 class interface. There is 1 method, 0 attributes and 38 lines of code.

The Temp sub-package contains all the classes for handling temporary files. It is implemented in 5 Java concrete classes and has 9 class interfaces. There are approximately 14 methods, 10 attributes and 336 lines of code.

The Searcher sub-package contains all the classes for detecting and recording verbatim coincidence. It is implemented in 7 Java concrete classes and has 32 class interfaces. There are approximately 22 methods, 23 attributes and 1044 lines of code.

5.10 The Integration of LSA System

This section discusses how the LSA system is integrated as part of the WTSS system. The LSA system is developed based on the Latent Semantic Analysis system. Francois has developed the Latent Semantic Analysis system in-house for reproducing the LSA tools available on the Colorado Web site and it is operated on a standard personal computer.

---

Like the Verbatim Engine system, the Latent Semantic Analysis system is a stand-alone system with its own user interface. Further, it allows other systems to access its core engine via its Java package and class interfaces. This work adopts the Latent Semantic Analysis system in developing the LSA system by both using directly or modifying its Java classes for detecting sentence to sentence cohesion and similarity to task while learners are handling Web texts via WTSS system.

The advantages of integrating the Latent Semantic Analysis system in the LSA system are as follows: firstly, the Latent Semantic Analysis system is built in-house as an open source system, thus the system is modifiable to suit the need of this work at no cost. Secondly, unlike most of Latent Semantic Analysis applications which are hosted on servers, the Latent Semantic Analysis system is built as stand-alone software which is portable and usable on most standard computer systems. Therefore, the WTSS system which integrates the LSA system is generally accessible to wider groups of users do not have access to servers.

However, adopting the Latent Semantic Analysis system in the LSA system has several disadvantages: Firstly, the development of the Latent Semantic Analysis system is not as stable as the Verbatim Engine system. It has been tested only on its accuracy of replicating results produced by Landauer’s example, which describes the principle of LSA and how it works. It uses a small corpus which consist of a set of nine documents and twelve unique words (see (Landauer et al., 1998a) for further description). However, the Latent Semantic Analysis system has not been tested on dealing with a bigger corpus. Secondly, there exists a conflict in the Latent Semantic Analysis system’s rigid parser which makes it unable to
develop a corpus from longer texts or texts which include special characters. Thirdly, the Latent Semantic Analysis system is not capable of manipulating large matrices as needed in this work. Therefore, in dealing with these problems, a few modifications have been made to the implementation of the Latent Semantic Analysis system in order to improve and synchronize its performance within the WTSS system before adopting it.

Figure 5.22 describes how the LSA system processes a learner’s free-response text in terms of its sentence coherence and its similarity to the given Web text-handling task in WTSS system.

Figure 5.22 – The process of measuring LSA coherence and similarity in learner’s free-response text by LSA system in WTSS system

Upon a learner’s request for the WTSS system to assess their Web text via the WTP system, the WTP system retrieves the learner’s free-response text which is stored in WTSS system’s directory. The WTP system passes the learner’s free-response text to the LSA system.
Next, the LSA system parses the text, cleans the text from common words (i.e. English common words such as pronouns, when, how, etc.) and converts the text into a vector of single words. Then, LSA coherence of the vector and LSA similarity of the vector against the Web text task are measured in relation to the LSA corpus used in the WTSS system. Lastly, the LSA system passes results of both measurements to the WTP system for presentation to the learner. At the same time the WTP system passes the results to the WTSS system to be stored in the system’s directory for the learner.

The LSA technique can be used to measure several aspects of free-response written text. In this thesis, the LSA technique is used to measure two aspects of learners’ free-response text to give an indication of the quality of the written text i.e. the text coherence and text similarity with the given Web-task. The Latent Semantic Analysis system is built as the Java package called LSA. The next section discusses how the LSA package works and how it is implemented. The discussion starts with the preliminary requirements for the LSA package, and is then followed by how its functions are distributed among its sub-packages and how these sub-packages are implemented.

5.10.1 The Distribution of the Functions

The essential function of the LSA package is to enable the WTSS system to make use of its sentence-to-sentence cohesion and document-to-document similarity detection techniques in evaluating learners’ written text. LSA is a technique which makes use of a language corpus. The result it produces is not only corpus but also subject dependent. There are three major steps involved in order to make LSA work in the WTSS system:
Chapter 5 – The Development of the WTSS System

1. Firstly, build a raw matrix containing a language corpus for a selected subject;
2. Secondly, transform the raw matrix into a Single Value Decomposition matrix with each cell containing a cohesion value in relation to the specific semantic space;
3. Thirdly, incorporate the LSA sentence-to-sentence cohesion and document-to-document similarity functions in the WTSS system.

The transformation of the raw text in a corpus into the knowledge representation in LSA requires a text analysis process which comprises the following steps (Landauer et al., 1998a, Landauer et al., 1998b):

1. The raw text in a corpus is parsed into individual words. Each word is defined as a group of characters terminated by a space character. Only unique words are kept in a data file.
2. The raw text in the corpus is parsed again into an individual ‘document’. The document is defined as a sentence, paragraph or passage terminated by a blank line. All these new documents are kept in another data file.
3. All unique words produced in step 1 are checked against all documents produced in step 2. Each unique word that occurs in at least two of the documents is referred to as a ‘term’ and is kept in another data file.
4. The matrix $M$ is composed by making each row stand for a term in step 3, and each column stand for a document in step 2. Each cell in the matrix $M$ contains the frequency of a term (denoted by its row) occurring in each given document (denoted by its column).
5. The rectangular matrix $M$ is transformed into a Singular Value Decomposition (SVD) Matrix where $M$ is decomposed into three other matrices such that:

$$M_{ij} = U_{ik} \cdot S_{kk} \cdot V_{jk}'$$

where $U$ and $V$ have orthonormal columns, $S$ is a diagonal matrix of singular values, and $k \leq \max(i,j)$.

6. Choose a dimension $d$. Set all values in $S$ to zeros except the $d$ largest singular values and reduce the number of columns in $U$ to $d$. The power of a SVD matrix lies in the choice of the $d$ value due to the fact that pre-multiplication of the decomposed matrices where the number of dimensions $d$ is retained, produces a ‘least-squares best approximation’ to matrix $M$. The produced matrix, in LSA’s theory induces human-like relationship among terms and documents based on the corpus used (see Landauer, Laham et al. 1998a for further explanation).

The resulting reduced matrices $U$ and $S$ for the given corpus are saved for later use in any LSA application as a semantic representation of the corpus.

Thus, the LSA system must anticipate the three steps, which were previously described, by dividing the essential function into 3 sub-functions. One of the sub-functions acts as the corpus builder, another one acts as the Single Value Decomposition matrix builder and the third sub-function represents the LSA tools. The sub-functions are described as follows:

1. **CorpusBuilder** – The CorpusBuilder offers facilities for inputting different written materials in the form of text files into the system, cleaning the gathered text file
from common words and building a raw matrix of the input texts (rows represent
the unique words and columns represents text elements).

2. **SVDBuilder** – The SVDBuilder is responsible for producing a Single Value
Decomposition matrix out of the raw matrix produced by the CorpusBuilder.
However, in the current WTSS implementation, SVDBuilder is replaced by an
independent commercial software O-Matrix\(^{21}\). This is due to the incapability of
the SVDBuilder to deal with large corpus and matrices.

3. **LSAtools** – The LSAtools provide a range of tools that offer access to the LSA
techniques to evaluate written text, along with a variety of supporting functions
needed by the techniques. The LSA techniques encompass the following:

   i. **Matrix comparison** - This function allows a submission of any text
      elements and returns an \(n \times n\) matrix of the cosine values resulting from
      comparisons between the text elements. Text element referred to is in the
      form of word(s), sentence(s), paragraph(s), or passage(s).

   ii. **Sentence Comparison** - This function allows a submission of \(n\) sentences
      and returns \(n-1\) lines of the cosine comparisons between the sentences.

   iii. **Pairwise Comparison** - This function allows a submission of any text
       element in even numbers and returns a similarity comparison for each pair.

   iv. **One-To-Many comparison** - This function allows a submission of one
       primary text element and \(n\) other texts to compare to it. The function will
give a \(1 \times n\) matrix of the cosine comparisons.

v. **Near Neighbours comparison** - This function allows a submission of a term or a short text and returns a list of terms nearest the submission (depending on the LSA corpus used).

### 5.10.2 The Implementation of the Functions

All the sub-functions of package LSA are implemented in 2 sub-packages. This section will describe how all the sub-packages are implemented by using the Java language. Figure 5.23 illustrates the overview of the implementation.

![Figure 5.23](image)

Figure 5.23 – The overview of the implemented sub-packages in the LSA system.

The *Jama* sub-package contains all the classes for representing the Single Value Decomposition sub-function and its utility classes. This sub-function is used by the SVDBuilder to produce a Single Value Decomposition matrix. It is implemented in 2 Java concrete classes and has 4 class interfaces. There are approximately 5 methods, 5 attributes and 559 lines of code.
The Mat sub-package contains all the classes for representing the CorpusBuilder and the LSAtools sub-functions including their utility classes. It is implemented in 3 Java concrete classes and has 12 class interfaces. There are approximately 101 methods, 13 attributes and 2861 lines of code.

5.10.3 The Development of Corpus for the LSA System

Building an LSA corpus is dependent on text in the specific area in which the application is intended to be used. Haley et. al. highlight that the corpus developer “must choose many options that are intrinsic to the success of any LSA-based marking system” (Haley et al., 2007). However, they reveal that most reports on LSA-based works are incomplete, which makes it difficult for other developers to replicate their successes or to avoid their failures especially regarding which intrinsic options they have chosen (Haley et al., 2005). Further, Haley et. al. argue that none of the previously built corpora involved the domain of computer science, thus, how to build kinds of corpora to be used in the computer science domain is still an open question (Haley et al., 2003).

Wiermer-Hastings et. al. argue that building a smaller corpus has several advantages: firstly it takes less time to train the corpus. Secondly, the corpus occupies less storage space. Thirdly, an application that uses the corpus requires less processing time for its tools to measure LSA cosines values. Thus, they suggest that building larger corpora should be avoided if there is no significant performance benefit. In the case of choosing texts for a corpus in a specific domain, they found that “an approximately equal balance or one slightly favoring the specific texts is advantageous to LSA” (Wiermer-Hastings et
Since the LSA system is intended to be operated on standard computer system and to be used in assessing domain-specific (i.e. computer science subject) texts, as an initial step, this work builds a computer-science corpus out of an equal balance of domain-specific and general texts. The corpus is built based on 500 passages and has 8006 unique total words. After cleaning common English words, the total unique words are reduced to 7457 words. Finally, there are 4280 terms (a term is a unique word which co-occurs in two or more passages). Therefore, the matrix M has 4280 rows and 500 columns. However, the current SVDBuilder package in the LSA system is not capable of processing and producing a SVD matrix out of this matrix M. Therefore, the process of building matrix SVD and other SVD related matrices is done by using commercial software that specifically is intended for processing large and sparse matrices, i.e. O-Matrix system. Then, the matrices produced by the software are stored in the WTSS system’s directory as text files to be used by the LSA system for assessing learners’ free-response texts while they are handling Web text.

5.11 Summary

This chapter described the development of a prototype of the WTSS system. The development of the system has involved a range of different processes which include developing a user interface (WTP system), making use of a freeware system (RDC system), integrating two in-house developed systems (VQD and LSA systems) and making use of a commercial product (O-Matrix system) to support the development process. The nature of the adopted techniques offers the ability for the development of the system to respond to changing requirements in the design and implementation of the WTSS system and allows
the system to evolve in the future. Even though, the WTSS system is developed as a prototype, it can be tested against the real-world data used in the scope of this research and in a real-life setting. Furthermore, features can be flexibly added to it in the future, to become a production standard software system for the purpose of assessing learners’ free-response texts produced while handling Web text tasks. The next chapter will present the evaluation of the WTSS system.
Chapter 6

Evaluation

6.1 Introduction

This chapter presents two evaluations of the Web Text-handling Support System which embeds the proposed model of Web text-handling processes. The system has been evaluated twice due to the fact that it was developed in stages. The first evaluation is intended to confirm the usability of the WTSS system to support learners in handling Web text along two dimensions, i.e. assessing the authenticity of learners’ performance and assessing verbatim copying in learners’ free-response text. Whilst, in the second evaluation, the system is evaluated along three dimensions to confirm the results in the first evaluation and to confirm the usability of the WTSS system in assessing sentence-to-sentence coherence in learners’ free-response text along with similarity between learners’ free-response text with the given Web text-handling task.


- **Effectiveness: the extent to which a goal, or a task, is achieved.** Effectiveness is measured in two aspects: task completion and quality of output. The system is considered effective if users can complete their task when using the system and can produce output with an acceptable quality.
• **Efficiency: the amount of effort required to accomplish a goal.** Efficiency is measured in two aspects: deviations from the critical path and error rate. The system is considered efficient if users can avoid deviating from the critical path (i.e. users take the least steps or the shortest time in completing their task) and users can avoid making errors. There are two types of error: slip and mistake. Normally, users make a slip error because they accidentally perform wrongly due to the system having an inadequate user interface design – for example because control buttons are placed too closely together. On the other hand, users make a mistake because they have an erroneous model of how the system works – for example they use shortcuts for other system (of which they are familiar with) but the current system does not support those shortcuts.

The results of errors can be categorized into four: minor (where users can notice and correct instantly), major (where users can spot the error but they are annoyed and take some time to rectify the problem), fatal (where users are unable to complete their task) and catastrophic (where users are unable to complete their task and also cause other problems).

• **Satisfaction: the level of comfort that the user feels when using a product and how acceptable the product is to users as a vehicle for achieving their goals.** Satisfaction is measured by asking users to comment on how they feel about the product i.e. was the product easy to use, did they enjoy using the product or did they feel frustrated or annoyed using the product.
Two usability evaluations of the WTSS system have been carried out. The main objective of each evaluation was to answer the following research questions which cover usability issues, i.e. effectiveness, efficiency and satisfaction, as derived from (Jordan, 1998):

1. Can learners perform the Web text-handling task with the current WTSS system within acceptable time limits and without making an unacceptably high number of errors?
2. Are there any usability faults in the product? If so, what are they?
3. How can these faults be solved?
4. What are the learners’ attitudes towards the use of WTSS system?
5. What design changes are required to make learners’ attitude more positive?

Section 6.2 describes the evaluation of the first version of the WTSS system. Next, section 6.3 describes the second evaluation of the new version of the system. Finally, section 6.4 summarizes the discussions arising from the two evaluations.

### 6.2 The First System Evaluation

This section describes the first evaluation of the WTSS system after the implementation of its WTP, RDC and VDQ sub-systems. In this evaluation, the WTP system had the first user interface design as described in section 5.7.1.

#### 6.2.1 Participants

One faculty member and fifty-seven first-year undergraduate students participated. The students were taking a Human-Computer Communication skills module (year 2005) in the School of Computer Science at the University of Nottingham, England.
6.2.2 Task Design

This experiment was designed to enable the participants to demonstrate their Web text-handling skills. The participants were given 60 minutes to respond on a specific topic of “lossy vs. lossless” compression of information. All participants were expected to complete this research task, using Web resources, in not more than 500 words, of which up to 300 (or equivalent to 60%) could be verbatim quotation. The work they undertook in the task was to be integrated into a subsequent piece of coursework for the module, and the mark they got for the Web text-handling task contributed 10% to their overall module mark. The search task was defined as follows:

*Shannon's first theorem in Information Theory enables us to calculate the entropy of a system of events if we know the probability of each individual event occurring. We can then use the Huffman algorithm to construct a compact code for the entire system. Recall that in general any coding of a system will not achieve the entropy limit. That is to say it will not generally be 100% efficient -- there will almost certainly be some redundancy.*

*Information can be compressed if a new encoding can be found that reduces redundancy. If the original information can be totally recovered by undoing this encoding (i.e. by *decoding*) then the compression is called *lossless*; if so much redundancy has been removed that the original information cannot be fully recovered on decoding, then the compression is called "lossy".*
Making careful use of sources on the Internet, write an essay about lossless and lossy encoding of information, with particular respect to LZW encoding of text, JPEG encoding for still pictures and MPEG encoding for DVDs. The essay should be no more than 500 words long of which at least 40% should be your own words.

6.2.3 Procedures

Participants were expected to read the task displayed on a screen, research the Internet by clicking a button to access the Internet Explorer software application and to write a free-response text based on selected Web resources in a text area provided by the system. By using the WTSS system, learners could access these three activities (as shown in figure 6.1) concurrently and interchangeably throughout the Web text-handling task session.

Figure 6.1 – Web text-handling activities performed by learners in the first evaluation

By activating the WTSS system, each participant had immediate access to an Internet Explorer browser to perform Web text activities and access to the text areas for reading
the Web-task and writing free-response text or copying Web text. The system recorded the start and end times. The system automatically saved learner’s free-response text and his or her Web navigation trails throughout the session.

At any time within the session, the learner was able to activate the VDQ system in order to generate a report on verbatim copying. The VDQ system converts the Internet Explorer history file into an appropriate format, which is then be used by the RDC system to check the time stamps. Only those URL addresses visited within the time parameters are considered by the verbatim detector. The verbatim detector then accesses the Web pages and compares the content with the words in the written text. If there are five or more consecutive words copied, then the words in the written text are flagged as copied. At the end of the process the verbatim detector is able to indicate the learner’s percentage of written words that are copied. However, in this evaluation, the verbatim detection results were not made available to the participants during the experiments because the e-feedback system was not fully implemented at that time.

6.2.4 Data

All the data was captured using the real-time data capture embedded in the system. The captured data for each participant was in the form of a text file consisting of all the visited URL addresses and a text file consisting of an essay. In this evaluation, the WTSS system produced three types of files for all participants:

1. A file containing their free-response text.
2. A file containing a list of their visited URL addresses.
3. A file containing the feedback report on their Web text-handling task in terms of the number of their rewritten words, the percentage of verbatim copying found and the URL addresses for their copied resources. In addition the participants’ behavior and comments were noted by the researchers during the Web text-handling session.

6.2.5 Results

All the results were transferred to a Microsoft Excel file. The results were then analyzed in two categories.

1. Participants’ Web text-handling strategies.

2. Verbatim copying found in the free-response text (based on the Web text-handling processes and output which are captured during the session).

6.2.5.1 Participants’ Web text-handling Strategies

Figures 6.2 and 6.3 show that even on a specified task, participants’ Web text-handling strategies showed a massive degree of individuality. First, 78.5% of the total of 600 URL addresses visited by the participants were visited only once; secondly, 76.6% of the search terms used in navigating to these sites were unique, and based on word combinations that were not used by others. These results were interpreted as indicating that Web text-handling strategies are much more varied than is generally supposed.
6.2.5.2 Verbatim Quotient Detection

Table 6.1 shows the highest, lowest, mean and standard deviation percentages of material copied. The range of percentage of verbatim text copied was great, ranging from 0% copied to 95% copied. The 0% copied might result from the rephrasing of the sentences so that the number of running words copied was no more than five (though participants were not made aware of how the VDQ system operated).
Table 6.1 – The highest, lowest, average and standard deviation Verbatim Quotient Detection scores for all the participants in experiment one (n= 58)

<table>
<thead>
<tr>
<th>Highest Percentage</th>
<th>Lowest Percentage</th>
<th>Average Percentage</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>95.00</td>
<td>0.00</td>
<td>33.34</td>
<td>24.90</td>
</tr>
</tbody>
</table>

Although verbatim quotient feedback was not made available to the participants during the experiment, the participants were aware of the existence of the verbatim quotient detection mechanism. Figure 6.4 indicates only 10 out of 58 (17%) of participants copied more than 60% from the Web, the maximum verbatim copying that they were allowed.

Figure 6.4 – Graph shows participants with the percentages of words copied in their rewritten text in the first system evaluation
6.2.6 Findings

This section discusses usability issues of the WTSS system from two perspectives: the users of the system (learners) and the developers of the system (researchers).

6.2.6.1 From Learners’ perspective

Derived from the results outlined in the previous section and from participants’ (learners’) comments during the evaluation session, the usability of the WTSS system can be assessed as follows:

• **Effectiveness.** The WTSS system is effective because all participants were able to complete their task within their time limit. Fifty-six participants were able to complete the given Web text-handling task within the given time limit without making unacceptable number of slips and minor errors. Even though two participants had to restart their task due to misunderstanding of how the system worked, and thus making a major error, at the end of the session they were able to submit their work.

• **Efficiency.** Here, the WTSS system is less efficient given that some fifty percent of the participants deviated from their critical path. Based on their comments during the Web text-handling session, the participants who deviated were confused by having the Web text-handling steps presented in different windows. At the same time, the majority of them were working with multiple windows of Internet Explorer. Therefore, they suggested that there were too many windows to deal with concurrently, which made it more difficult to concentrate on solving the Web text-handling task. This problem was solved by designing a new interface with fewer windows for the system. Learners are
expected to have a clearer overview of the steps in handling Web text with this new interface and can therefore focus more on completing their Web text-handling task.

- **Satisfaction.** The WTSS system users were not well satisfied, in general. Based on observation of the participants and their comments and questions during the evaluation session, it can be concluded that the majority of them were dissatisfied for two reasons: the user interface of the system with “lots of windows” and the need to write a free-response text in the text area provided by the system. Therefore, to encourage a more positive attitude among learners, the design of the system user interface needed to be changed to show fewer screens and the text area, where learners write their free-response text, should be enhanced to imitate simple text editing as provided in Microsoft Word, with which most learners are familiar.

### 6.2.6.2 From Research Perspective

The aim of the first system evaluation was to establish the extent to which Web text-handling processes and output can be assessed. In this evaluation, the WTSS system consists of the WTP, RDC and VDQ sub-systems which cover the user interface, real-time data collection (which automatically captures data in ways recommended by McEneaney (McEneaney, 1999)) and the verbatim text detection tool. Even though the participants were aware of the existence of the verbatim detector, the feedback was not made available during the experiment session because the e-feedback in the WTSS system was not fully developed at that stage.
The results have outlined some aspects of learners’ search and composition behavior from which this research can assess their performance based on their Web text-handling processes and output. The results confirm the findings produced by other related research in which learners’ processes and output were highly individualized (Hendry and Harper, 1997, Henry, 2005, Henry, 2006, Juvina et al., 2002, McEneaney, 2001), i.e. none of the learners’ processes and output is the same even though they are given the same task. In addition, the learners controlled the number of verbatim copied words in their free-response text when they were informed of the existence of the verbatim copying detection mechanism.

Two important outcomes derived from this evaluation of the WTSS system are: firstly, the WTP and RDC sub-systems are usable for producing learner’s real-time data; and secondly the VDQ system is usable for handling verbatim copying based-on the produced data. Therefore, both systems will be formally adopted as parts of the revised WTSS system.

### 6.3 The Second System Evaluation

This section discusses the second evaluation of the WTSS system which was done after the new version was implemented. In the new version, the system was embedded with an LSA sub-system along with an enhanced user-interface consisting of the instructions, task and formative feedback on one screen (to replace the eight screens used to present similar functions in the previous version of the WTSS system). In addition, the formative feedback (the number of free-response words, the number of verbatim words and their percentage, sentence-to-sentence cohesion in the free-response text, and document-to-
document similarity between the free-response text and the given task) was presented graphically. The formative feedback was made available to the participants during the experiment. The data were obtained in real-time, periodically (at one minute intervals) during the experiment, and processed automatically for each participant at every twenty-minute time interval. However, learners could also request that their work be assessed at any time they liked by clicking the ‘Assess’ button on the screen. The second evaluation replicates the same task design and procedures as used for the first evaluation. Data was captured as described in section 6.2.4.

### 6.3.1 Participants

Fifty-nine first-year undergraduate students participated and performed the same task as described in section 6.2.2. These students took a communication skills module (year 2006) in the School of Computer Science at the University of Nottingham, England.

### 6.3.2 Results

All the results were transferred to a Microsoft Excel file. The results were then analyzed in three categories.

1. Participants’ Web text-handling strategies.

2. Verbatim copying found in the free-response text based on the Web text-handling processes and output which are captured during the research session.

3. The results produced by the LSA system to provide an indication of two factors:
   
i. The degree of match between the participant’s free-response texts with the given Web text-handling task.

   ii. The overall cohesion of the participant’s free-response texts.
6.3.2.1 Participants’ Web Text-handling Strategies

Figures 6.5 and 6.6 show similar findings to those found in the first system evaluation. That is, even on a specified task, participants’ research strategies exhibit a very large degree of individuality.

![Visited URLs vs Frequency](image)

Figure 6.5 – Shows the number of unique URL addresses visited by the participants in the study, with their frequency in the second system evaluation

In this experiment, 72.9% of 483 URL addresses were visited only once by the participants and 83.69% of the search phrases used in their navigation were unique. Therefore, these results confirm those found in the first system evaluation that the Web text-handling strategies used by the participants are indeed much more varied and individualistic than is generally supposed when learners are given the same task.
### Figure 6.6 – Shows the number and frequency of different keywords used in navigation (i.e. most words were used once only; fewer than 20 were used by two participants, etc.) in the second system evaluation.

#### 6.3.2.2 Verbatim Quotient Detection

Table 6.2 shows the highest, lowest, mean, and standard deviation of material copied. The range of percentage of verbatim copied text was nearly as great as found in experiment one, ranging from 0% copied to 81% copied. As in the first evaluation (section 6.2.5.1), the 0% copied text might have resulted from the rephrasing of the sentences so that the number of running words copied was no more than five (though the participants in this experiment were also not made aware of how the VDO system operated).

<table>
<thead>
<tr>
<th>Highest Percentage</th>
<th>Lowest Percentage</th>
<th>Average Percentage</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>81.00</td>
<td>0.00</td>
<td>34.20</td>
<td>22.30</td>
</tr>
</tbody>
</table>
Figure 6.7 shows the percentage of words copied in all the participants’ rewritten text. In this experiment, where the percentage of verbatim words copied was made available to the participants in the Web text-handling session, the percentage of participants that exceeded the maximum verbatim copying allowed is reduced to 8% (5 out of 59 participants) which is about a 50% reduction of the figure found in the first evaluation.

![Graph showing percentage of words copied for all participants](image)

**Figure 6.7 – Graph shows participants with the percentages of words copied in their rewritten text in the second system evaluation**

### 6.3.2.3 LSA

In this experiment, LSA has been performed on the collected data based on two LSA corpora: the Colorado Corpus (referring to the corpus found at [http://LSA.colorado.edu/](http://LSA.colorado.edu/) which was built by using texts for general-reading-up-to-1st-year-college22); and the Nottingham Corpus (task-related reading). Unlike the Colorado Corpus which is accessed

---

through its Website http://LSA.colorado.edu/, the Nottingham Corpus is embedded within the WTSS system. The Nottingham Corpus was first trained on a large sample of text (about 200,000 words) from the same domain as the task given in both of the system evaluations. The training helps the WTSS system to acquire the knowledge of words and concepts in relation to the Web text-handling task.

In this analysis, the participants’ data were tested by using 300\(^{23}\) and 20\(^{24}\) of the LSA space dimensions for each corpus. For each participant, two LSA analyses were performed: sentence-to-sentence cohesion in the free-response text; and document-to-document similarity between the free-response text and the given Web text task. These figures were produced by the WTSS system and relayed in each participant’s final formative feedback.

Then, the collected data were processed by using four different formulae (referred to as Formula A, Formula B, Formula C, and Formula D) to produce four different ‘projected marks’. A projected mark is referred to as a mark given to each participant’s free-response text based on calculation by the specific formula (Formula A, B, C or D); and considered as a computer based mark as oppose to a human marker’s mark. However, the marks produced by Formula A are of most interest to this work since the marks are calculated without any use of pre-graded essays, which is in the scope of this thesis.

\(^{23}\) The number of dimensions reported to perform well by (Foltz et. al, 1998)

\(^{24}\) The number of dimensions reported to perform well by an experienced user of the Single Value Decomposition matrix application and O-Matrix software - LARMUSEAU, P. (2007).
The different formulae are used for comparison purposes in order to determine the formula that will give the best projected result (the nearest to human marker’s marks). In the formulae, two LSA Cosine values are used: the Corpus Colorado sentence-to-sentence cohesion value (space dimension = 20) and the Nottingham Corpus document-to-document similarity value (space dimension = 300). Several pre-assessments on the experimental results indicated that these two LSA cosine values are the best for producing the projected marks (the nearest to human marker’s marks). The formulae are formulated solely based on the experimental pre-assessments and are described as follows:

**Formula A:**

This formula applies both LSA Cosine values directly and takes the average of the two values as the projected mark.

\[
\text{Projected Mark} = \frac{\text{(the Colorado Corpus sentence-to-sentence cohesion value)} + \text{(the Nottingham Corpus document-to-document similarity value)}}{2}.
\]

**Formula B:**

In this case, six model answers (the top six marked by human markers) are used to produce the Nottingham Corpus document-to-document similarity values between each participant’s free-response text and each model answer.

\[
\text{Projected Mark} = \text{(the human mark which has the highest Nottingham Corpus document-to-document similarity value among the six model answers)} \times \text{(the participant’s Nottingham Corpus document-to-document similarity value)}.
\]
**Formula C:**

In this case, the same six model answers as chosen in Formula B (the top six marked by human marker) are used to produce the Nottingham Corpus document-to-document similarity values between each participant’s free-response text and each model answer.

\[
\text{Projected Mark} = \frac{\text{the sum of } ((\text{Each model’s answer mark}) \times (\text{its relative Nottingham Corpus document-to-document similarity value of the participant’s rewritten text}))}{6}.
\]

**Formula D:**

In this case, the same six model answers as chosen in Formula B and C are used to produce the Nottingham Corpus document-to-document similarity values between each participant’s free-response text and each model answer. However, in this formula, the Projected Mark is obtained statistically by using multiple regressions.

\[
\text{Projected Mark} = \text{Multiple Regression Constant} + \text{The sum of } ((\text{Each model’s Multiple Regression coefficient value}) \times (\text{its relative Nottingham Corpus document-to-document similarity value of the participant’s rewritten text})).
\]

Where \( \text{Multiple Regression Constant} = 0.356, \) and \( \text{Each model’s Multiple Regression coefficient value} = 0.410, 0.240, 0.156, -0.416, -0.081, \) and \(-0.065 \) respectively.

In this section, a graphical analysis was done. The analysis encompassing the LSA sentence-to-sentence cohesion value for the participant’s free-response text and the LSA document-to-document similarity value between the participant’s free-response text and
the Web text-handling task (as produced by the WTSS system in each participant’s final formative feedback) have been collected. The collected data is processed using the four different formulae (referred to as Formula A, Formula B, Formula C, and Formula D) to produce four different projected marks.

Figures 6.8 and 6.9 show a graphical comparison in terms of the number of participants versus the range of differences between the human marker’s marks and projected marks for each formula in this evaluation.

In this experiment the 0 to 5% range of differences between the human marker’s mark and its relative projected mark was considered as an acceptable range of differences in the free-response formative type of Web text-handling assessment. The LSA analysis shows a 41% agreement between the human markers’ and the projected marks (refer to Formula A) where the projected marks are solely produced by LSA without the use of pre-graded essays. The results signify a significant usability of e-assessment using the LSA techniques to assess Web text-handling processes and product without any contribution of human markers effort.
Nevertheless, figure 6.9 shows that among the four projected mark formulae, Formula D (with multiple regression constant and coefficient values) which is based on selected human-graded models, produces the most consistent projected marks.
The current version of the WTSS system has been further evaluated with data captured during the first system evaluation. Figures 6.10 and 6.11 depict the graphical comparison in terms of the percentage of participants versus the range of percentage differences between the human marker’s marks and projected marks for each formula.

Consider that if the 0 to 5% range of differences between the human marker’s mark and its relative projected mark were considered as an acceptable range of differences in the Web text-handling assessment, 40% (Formula A) of participants would fall into the acceptable range. The result, which is relatively similar to the one in the second system evaluation.
evaluation, reflects a significant usability of LSA values in projecting computer based marks for Web text-handling tasks.

Figure 6.10 – Bar graph showing the percentage of participants in different ranges of percentage differences between human marker’s marks and their projected marks after application of four different formulae in the first system evaluation
This analysis is based on a human marker’s marks. It would be more interesting to have several sets of human marker’s marks from many experts and to see how they correlated. However, this idea is too resource intensive to be implemented in this research.

### 6.3.3 Findings

This section discusses usability issues of the WTSS system from two perspectives: users of the system (learners) and developers of the system (researchers), which is similar to what has been discussed in section 6.2.6.
6.3.3.1 From Learners’ perspective

Based on the results in section 6.3.2 and participants’ (learners’) comments during the evaluation session, the usability of the WTSS system can be assessed as follows:

- **Effectiveness.** The WTSS system is effective in the all participants were able to complete their task within the time limit. Fifty-six participants were able to complete the given Web text-handling task within the given time limit without making an unacceptable number of slips and minor errors. Three participants had to restart their task due to misunderstanding how the system worked and their errors were considered as major. They were still able to submit their work at the end of the session. However, all the participants were interrupted by the automatic evaluation at twenty-minute intervals.

- **Efficiency.** The WTSS system is not efficient as measured by the fact that the majority of the participants deviated from their critical path due to problems with e-assessment which is embedded in the WTSS system. Derived from their comments, the deviated participants were impatient with the automatic invocation of assessment at twenty-minute intervals. They prefer the e-assessment to be invoked only if they requested it. This problem was solved by deleting the automatic invocation of e-assessment in the system.

- **Satisfaction.** The WTSS system users were not fully satisfied with the system. In this case, based on observation of the participants and their comments and questions during the evaluation session, the majority of them were not satisfied for one main reason, namely, the automatic evaluation.
6.3.3.2 From Research Perspective

The aim of the second system evaluation was to establish the extent to which the LSA system can assess Web text-handling output. In this evaluation, the WTSS system consisted of the WTP, RDC, VDQ and LSA sub-systems which cover the user interface, real-time data collection, verbatim text detection tool and LSA tools. In this evaluation, the participants were aware of the existence of the verbatim detector and LSA. In addition, the feedback was made available during the evaluation session.

The results have revealed the same aspects of the search and composition behaviour of the learners as in first evaluation. The results confirm the findings produced in the first system evaluation, i.e. none of learners’ processes and output is the same even though they are given the same task. In addition, learners were more controlled about the number of verbatim copied words in their free-response text when they were informed of the existence of the verbatim copying detection mechanism and presented with e-feedback.

One important outcome derived from this evaluation of the WTSS system is that the LSA system is usable for handling assessment of learner’s free-response text as described in point 3, section 6.3.2. Therefore, the LSA system will be formally adopted as a part of the final WTSS system.

6.4 Conclusions

The results of both system evaluations show that all components of the WTSS system (WTP, RDC, VDQ and LSA sub-systems) had significant usability when the system was
accessed by two groups of able learners in higher education. In general, these evaluations confirm the usability and accessibility of the WTSS system specifically in making use of real-time data (which are produced by learners in a real-life setting) and providing immediate constructive feedback as a support tool in Web text handling.

In addition, the LSA technique in the WTSS system is usable for determining the quality of learner’s free-response texts in terms of its sentence-to-sentence coherence and its similarity with the given Web text-handling task. However, results produced by the LSA system are dependent on the corpus used. The 40% projected marks (by using Formula A, in the first evaluation) and 41% projected marks (by using Formula A, in the second evaluation) in the range of 0-5% different from human-marker’s marks could be further improved by enhancing the corpus used. Both evaluations show that LSA techniques are able to produce significant indicators of how well learners perform Web text-handling task based solely on the techniques and the corpus without the need for the system to be trained with a large amount of pre-graded learners’ answers prior to the use of the system.

6.5 Summary

This chapter has described how the implemented WTSS system based on the proposed model of Web text-handling processes and scenario, is useful. It has explained the usability issues by presenting two system evaluations. Generally, the first evaluation tested the usability of the WTSS system which consisted of the WTP, RDC and VDQ sub-systems. The second evaluation tested the usability of the WTSS system which consisted of the WTP, RDC, VDQ and LSA sub-systems. Both evaluations have shown that the proposed system produces significant usability results in terms of three issues: the
authenticity of the Web text-handling processes and output; verbatim quotient detection; and the LSA sentence-to-sentence cohesion values and document-to-document similarity values. It can be concluded that these have significant roles in supporting learners in handling Web text. In summary, the requirements for the development of the WTSS system and its objectives have been met. The next chapter will present some conclusions and future research ideas.
Chapter 7

Conclusions and Future Research

7.1 Introduction

This chapter discusses how this research has met its objectives as specified in chapter 1. Next, it presents three major contributions of this work to areas as set in section 1.4, i.e. World Wide Web, Learning Technology and Goal-Directed Approach software process. Finally, this chapter discusses areas for future work.

7.2 Meeting the Objectives

This thesis has presented two models for conceptualizing learners’ behaviour in handling Web-text within a Web-based information-seeking domain: a model of Web text-handling processes and a model of a Web text-handling scenario. Based on both models, this work has produced a prototype Web Text-handling Support System where learners (in higher education) are expected to produce a free-response text via handling Web-texts. In return, the Web Text-handling Support System assesses and provides feedback in three areas: verbatim copying from Web resources, sentence-to-sentence coherence of the free-response texts and similarity between the free-response texts and the given Web text-handling task. This section revisits the objectives for this research which have been set out in section 1.3. It demonstrates that the objectives were accomplished as follows:

1. Two models were developed to conceptualize Web text-handling activities (which emphasize processes and output) and scenario in the domain of Web-based
information-seeking domain by using a Goal-Directed Approach towards software design.

2. Web-text processes and output were identified which can be captured in real-time and in a real-life setting to be used as input for the proposed Web Text-handling Support System.

3. Two mechanisms, LSA technique and cut-and-paste plagiarism detection, were used as tools to analyze the real-time data in order to generate items in the e-feedback.

4. The proposed Web Text-handling Support System (embedded with e-feedback) was developed by using the Goal-Directed Approach software design.

5. The proposed support system was evaluated.

Modelling of learners’ processes and scenario in handling Web-texts prior to developing a system to support them is a necessary step towards conceptualizing their Web text-handling processes and output. The resulting models greatly support experimentation and research in the new domain of e-assessment, i.e. Web-based information-seeking. Both models have been successful in presenting complex learners’ Web-text behaviour to researchers in a conceptual and simplified form.

The models lay out a range of potential learners’ Web text-handling elements (in the form of learners’ processes and output) which learners produce while handling Web-text in a real-life task-driven setting to be automatically captured and processed in real-time via a standard computer system. For the purpose of this work, only two elements have been
identified for capture (via two real-time data collection methods embedded in the WTP and RDC sub-systems) and assessment: the learners’ navigation trails and free-response texts.

In addition, the models expose a wide range of potential mechanisms to analyze the selected real-time data. For the purpose of this research, two mechanisms are used to assess the data in real-time: verbatim copying detection and LSA techniques. Based on the models, the Web Text-handling Support System was developed and both assessment mechanisms have been successfully incorporated into the Web Text-handling Support System.

The use of verbatim copying detection was explored to analyze learners’ free-response text and it was shown to be successful in measuring what percentage of learners’ answers is copied and from which Web sites it is copied. Similarly, the applicability of LSA techniques to assessing learners’ free-response text was tested and it was shown that the techniques have significant capabilities of measuring two factors: how coherent learners’ free-response texts are and how relevant learners’ free-response texts are against their given Web text-handling task.

The proposed Web text-handling Support System has been evaluated twice. The results of the system evaluations indicate that assessing learners’ real-time data in a real-life setting is both successful and useful. Incorporating a verbatim copying detection mechanism in the support system positively discourages learners from excessively verbatim copying of Web-text resources. This research found that the percentage of verbatim text, specifically the ‘copy-and-paste’ type, could be reduced when immediate formative feedback was
made available during the Web text-handling session. This finding supports previous research (as described in section 2.5.2), specifically work by McGowan (McGowan, 2003) and Barret and Malcolm (Barrett and Malcolm, 2005) who emphasize that detecting plagiarism in written output produced via handling Web-text is beneficial.

The incorporation of LSA techniques has produced relatively significant results when compared to the human marker’s results (about 40% agreements if the range of difference between computer’s marks and human marker’s marks is 0%-5%). That a higher percentage of agreement was not achieved possibly due to three reasons. Firstly, the Nottingham Corpus (the domain-specific corpus used by the LSA techniques in the Web Text-handling Support System) is still in its infancy and is not expected to be fully capable of representing learners’ domain-specific knowledge. Secondly, the formula which has been used to produce the computer’s marks (which are based solely on values produced by the LSA techniques) lacks accuracy to match the marks produced by the human marker because the formula has only two parameters (similarity and coherence) and it does not include other relevant parameters normally used by human marker such as organization of ideas and syntax usage in the text. Thirdly, the human marker\textsuperscript{25} has the ability (which the computer does not possess) to form an overview of the performance of all learners which has an effect on the allocated marks. This can be seen from his comments about the way in which he marked learners’ free-response text: “there were a lot of answers that went into too much technical detail (which had obviously been copied from some online

\textsuperscript{25} James Ollis, a researcher at the Department of Computer Science, University of Nottingham, acted as the human marker for both system evaluations.
source) and these received lower marks since the idea of the exercise was to produce a summary rather than a detailed essay. Also, sections of the text that had clearly been copied from online sources (Wikipedia seems to be a favourite) were not awarded many marks since the work was not that of the student in these cases”. However, the findings of the system evaluation, specifically focusing on the results produced by the LSA techniques, are interpreted as having a relatively positive agreement with what is reported in the literature (Henry, 2006, Leu et al., 2004), i.e. Web text-handling is an important attribute in Web-based knowledge acquisition.

7.3 Contributions

The main contributions of this research are in the areas of World Wide Web and Learning Technology, but advances can also be demonstrated in the area of Software Design.

7.3.1 World Wide Web

The primary contribution of this thesis is in the area of the World Wide Web. The initial work of modelling learners’ complex behaviour while handling Web-texts contributes to the conceptual presentation of the Web text-handling setting, processes and output. Moreover, providing e-feedback in real-time while a learner is handling Web-text in a real-life setting is new to the Web-based information-seeking field. Other contributions to this area include the application of LSA and the cut-and-paste plagiarism detection techniques to a new domain, i.e. Web-based information-seeking. In this work, both techniques which use real-time data obtained in a real-life setting (in the forms of Web text-handling processes and output) are evaluated as usable within the scope of this research.
7.3.2 Learning Technology

This work contributes to the Learning Technology area in two aspects: the development of a new type of e-assessment and the application of e-assessment to a new domain, i.e. Web text-handling.

The first aspect of the contribution involves the development of a new type of e-assessment to assess real-time data and provide real-time constructive feedback in a real-life setting. This e-assessment addresses Web text-handling activities, whereas current assessments of the Web-based information-seeking skills of students do not address this matter. Moreover, current assessment of the skills uses Multiple Choice Questions which is based solely on learners’ output. This research provides an alternative method of assessing Web-based information-seeking skills which focuses on Web text-handling activities and dealing with real-time data, obtained in a real-life setting (the real-time data includes both learners’ Web text-handling processes and output).

The second aspect of the contribution involves the fact that e-assessment of Web-based information-seeking activities, generally, and Web text-handling activities, specifically, using real-time data in a real-life setting has not been attempted prior to this work.

7.3.3 Software Design

The Goal-Directed Approach is normally used at the system analysis and design stage in a software development process. This work contributes to the area of software design via the application of the Goal-Directed Approach prior to the start of the software
development process, i.e. to conceptualize and model the complexity of learners’ behaviour in handling Web-text. In this research, the approach has been used to conceptualize and to model learners’ Web text-handling processes and learners’ Web text-handling scenario. The models have been used as the basis for the development of the proposed Web Text-handling Support System. This has shown that it is possible to use the Goal-Directed Approach not only to analyse and design a system but also to conceptualize and model a complex human behaviour for a specific application.

7.4 Dissemination

This section describes how the work reported in this thesis has been disseminated by listing the publications, which have been derived from it.

The following papers present the complexity of Web text-handling behaviour in relation to online reading comprehension and online reading assessment:


This paper presents the model of Web text-handling processes. In this paper, the model is referred to as *Computer-Based Low Inference Model*.


The following papers describe how the WTSS system captures real-time data in a real-life setting and processes the captured in terms of verbatim quotient detection. In these papers, the WTSS system is referred to as the Internet Research Support System.


7.5 Future Research

Amongst the main contributions of this research has been the conception a substantial basis for future research directions. This section considers two areas for potential extensions: learning technology and World Wide Web.

7.5.1 Learning Technology

An important aspect for future research is to transform the current prototype Web Text-handling Support System into a software standard system so that the system can be used to study learners’ performance longitudinally and therefore, the effectiveness of the e-assessment can be statistically measured. The development of the prototype system opens a wide range of further research as follows:

- **Developing a domain-specific corpus.** Developing a suitable and reliable corpus for domain-specific application of LSA techniques especially in computer science studies is still in its infancy. Further research is much needed to answer the question of how many texts (general and domain-specific texts) are needed in order for the corpus to sufficiently represent learners’ knowledge for a domain-specific Web text-handling task and therefore, for the corpus to be able to assess both coherency and relevancy of learners’ free-response texts. In addition, further research is much needed in investigating efficient ways of developing the corpus via a standard computer system.

- **Refining the marks formula.** Further research could also aim to further augment the formula used to obtain the computer’s marks for learners’ Web text-handling processes.
and output. In the current prototype system, only values produced by the LSA system are used to represent the computer’s marks. Finding an appropriate formula to assess learners’ Web text-handling processes and output without using human pre-graded samples whilst taking account of verbatim copying is very much needed.

- **Evaluating other factors.** The current prototype of the proposed Web Text-handling Support System evaluates plagiarism and the semantic aspects of learners’ free-response texts. An interesting direction for future research would be to investigate other possible assessment factors, for example, whether it is possible to assess learners’ free-response texts syntactically or whether it is possible to assess the organization of learners’ ideas in their free-response texts.

- **Presenting constructive feedback.** Further research could seek to investigate different ways to construct and present constructive feedback to learners in order to obtain the best possible benefits for learners while using the system.

### 7.5.2 World Wide Web

An interesting idea for further research is to allow the mixing of assessment criteria within the Web Text-handling Support System. The e-assessment system could use a combination of techniques, i.e. by adding support for assessing different aspects of the Web-based information-seeking. This idea necessitates revisiting both Web text-handling models (as discussed in chapter 3) and investigating potential Web text-handling processes and output to be included in the assessment.
One of the important aspects of Web text-handling processes is searching the Web resources. Only searched material and resources on the Web can be viewed by learners. Therefore, search is the key process and it plays a crucially important role in the success of any Web-based information-seeking task (Leu et al., 2007, McEneaney, 2000b, Henry, 2006, McEneaney, 2000a). In addition, searching includes both systematic and opportunistic Web-text processes (Marchionini, 1998, Hendry and Harper, 1997, Marques et al., 2004, Broder, 2002, Gardner et al., 2002, Protopsaltis and Bouki, 2005, Henry, 2006). Further, it is reported that different Web users carry out searching in different styles and the searching strategy styles display distinctive characteristics and reflect a personalized search trail (Juvina and van Oostendorp, 2004). McEneaney argues that the personalized search trails correlate with the level of Web users’ online reading comprehension (McEneaney, 1999a, McEneaney, 2000a, McEneaney, 2001, McEneaney, 2000b, McEneaney, 1999b, McEneaney, 2004). Therefore, further research could be conducted on how to assess learners’ search style and strategies, and then constructive feedback could be produced as an added support to learners while handling Web-text.

### 7.6 Closing Remarks

This research has demonstrated that it is possible to assess Web text-handling processes and output as a means of supporting learners in handling Web texts. The evaluations of the proposed Web Text-handling Support System confirm the usability of assessing the processes and products of learners’ Web text-handling activity that are left out of existing e-assessment systems. However, the proposed models and the implemented system are not complete, especially in covering all aspects of Web text-handling activities. Indeed,
the system needs further improvements to accommodate the emergence of new theories and practice in its related areas. It is felt that the models and the system have the potential to increase support for learners by providing a wider range of tools than the currently available verbatim detection, text cohesion and document similarity checking.

Finally, with the ever increasing amount of available Web materials and resources, this research has produced a product that is crucial in relation to Web-based information-seeking, thus significant for further investigation.


MCENEANEY, J. E. (1999) New Approaches to Data Collection and Analysis in Online Reading Studies. *Annual Meeting of the National Reading Conference*. Orlando, FL USA.


