TOWARDS ELECTRONIC ASSESSMENT
OF WEB-BASED TEXTUAL RESPONSES

by

MARTHA MARIA CONRADIE

submitted in part fulfilment of the requirements
for the degree of

MASTER OF SCIENCE

in the subject

INFORMATION SYSTEMS

at the

UNIVERSITY OF SOUTH AFRICA

SUPERVISOR: PROF M R DE VILLIERS

JUNE 2003
Abstract

Web-based learning should move away from static transmission of instruction to dynamic pages for effective interactive learning. Furthermore, automated assessment of learning should move beyond rigid quizzes or multiple-choice questions.

This study describes the design, development, implementation, testing and evaluation of two prototypes of an electronic assessment tool to enhance the effectiveness of automated assessment. The tool was developed in the context of a distance-learning organisation and was built according to a development research model entailing a cyclic design-intervention-outcomes process.

The first variant, E-Grader, was developed to test an algorithm for assigning marks to open-ended textual responses. The second variant, Web-Grader, was an interactive web-based extension of E-Grader. It provided immediate interactive support to students as they responded textually to content-based questions.

This multi-disciplinary study incorporates principles and techniques from software engineering, formal computer science, database development and instructional design in the quest towards electronic assessment of web-based textual inputs.

Key terms (in alphabetic order): database-driven system, development research, dynamic content, e-learning, electronic assessment, end-user involvement, feedback, free text, interaction, intervention, open-ended assessment, rapid prototyping, study support tool, virtual classroom, web-based learning.
Acknowledgements

I would like to express my gratitude and appreciation to the people who assisted me in many ways:

My supervisor -
  Prof. Ruth de Villiers

My husband Pieter

Computer services -
  Magdaleen Bekker

  Bertie Koch

My colleagues in the
School of Computing at
Unisa

My two sons -
  Danie
  Johan

COS321-6 students of
2001 and 2002

I really do appreciate it!

Martjie Conradie
June 2003

My sincere appreciation to you Ruth for your special way of giving advice, guidance, encouragement and assistance.

Thanks Pieter for your advice, support, patience, encouragement and domestic assistance

For going out of your way in supplying electronic data for E-Grader.

For technical support on Web-Grader.

Thank you for your continuous encouragement and support.

For your love and valuable practical and technical support.

For participating in the study.
Table of contents

Chapter 1: Introduction

1.1 Problem statement ............................................. 1
1.2 Research question ........................................... 2
  1.2.1 Primary research question ................................. 2
  1.2.2 Subquestions ............................................ 3
1.3 Value of the study ............................................ 3
1.4 Scope of the study ............................................ 4
  1.4.1 Domain of the study .................................... 4
  1.4.2 Limitations and delimiters ............................... 5
1.5 Resources consulted .......................................... 6
1.6 Research design and methodology ............................ 6
  1.6.1 Development research .................................. 6
  1.6.2 Theoretical research ................................... 7
  1.6.3 Intervention ........................................... 8
1.7 Structure of the study ........................................ 9

Chapter 2: Learning theories and instructional design

2.1 Introduction ................................................. 14
2.2 Theoretical and conceptual aspects .......................... 14
  2.2.1 General perspectives on learning ....................... 14
  2.2.2 Learning theories ...................................... 16
2.3 From theory to practical implementation .................... 22
  2.3.1 Introduction - the role of technology ................. 22
  2.3.2 Theoretical foundations of instruction and technology-based instructional design.. 24
2.4 Conclusion .................................................. 28

Chapter 3: Web-based learning

3.1 Introduction ................................................. 32
3.2 Role of the WWW in implementing and supplementing learning ................................. 32
3.2.1 Main features of web-based learning ........................................... 34
3.2.2 Internet technology used in web-based learning .............................. 38
3.3 Problems related to assessment and interactive learning in web-based instruction ................................................................. 66
  3.3.1 Introduction .............................................................................. 67
  3.3.2 Problems related to assessment .................................................. 67
  3.3.3 Problems related to interactive web-based learning ......................... 68
3.4 Possible solutions for some of the problems ...................................... 74
  3.4.1 Solutions to assessment problems .............................................. 74
  3.4.2 Interactive web-based learning environments as a solution ............... 77
3.5 Conclusion ..................................................................................... 86

Chapter 4: Automated assessment

4.1 Introduction .................................................................................. 94
4.2 Automated assessment systems .................................................... 94
  4.2.1 Free-text assessment based on surface features .............................. 97
  4.2.2 Free-text assessment based on content ........................................ 98
  4.2.3 Automatic assessment of computer programming assignments ......... 111
4.3 Bennet’s speculations on the future of large-scale assessment ............... 114
  4.3.1 Three generations of large-scale assessment ................................ 115
4.4 Conclusion ..................................................................................... 117

Chapter 5: Methods, models and tools used for developing E-Grader and Web-Grader

5.1 Introduction .................................................................................. 123
5.2 Software engineering methodologies and principles .......................... 124
  5.2.1 Software process models with a linear sequential approach ............ 125
  5.2.2 Software process models with an evolutionary approach ................ 127
5.3 Instructional systems development models ...................................... 129
  5.3.1 Evolution of ISD models ........................................................ 130
  5.3.2 The Dick & Carey model ...................................................... 131
  5.3.3 Rapid prototyping for educational applications ............................ 133
  5.3.4 Assessment of performance and instructional design ...................... 133
  5.3.5 Similarities between ISD models and SE models .......................... 134
5.4 The use of rapid prototyping for instructional software development ............... 135
5.4.1 Advantages of prototyping ....................................................... 136
5.4.2 Disadvantage of prototyping .................................................. 137
5.5 Pressman's generic view of software engineering ........................................... 138
5.6 Tools used to develop and implement E-Grader and Web-Grader .................... 139
5.6.1 Regular expressions .............................................................. 139
5.6.2 Apache web server .............................................................. 146
5.6.3 Linux operating system ......................................................... 146
5.6.4 Data validation using PHP: Hypertext Preprocessor .................................. 154
5.7 Methodology and tools used to extend E-Grader to Web-Grader ...................... 155
5.7.1 Methodology used to produce Web-Grader ....................................... 156
5.7.2 Tools and methods used to produce Web-Grader .................................... 157
5.8 Conclusion ................................................................................. 165

Chapter 6: E-Grader: development, implementation and evaluation

6.1 Introduction .............................................................................. 168
6.2 Design and development process of the E-Grader prototype .......................... 170
6.2.1 Definition phase: What was built? .............................................. 170
6.2.2 Development phase: How was the system built? ................................. 177
6.3 Implementation of the E-Grader prototype in a real-world situation ............. 181
6.3.1 The first operational E-Grader - implementation on assignment questions ... 182
6.3.2 The second implementation of E-Grader on assignment questions ............ 189
6.4 Discussion and summary .................................................................. 193

Chapter 7: Web-Grader: development, implementation and evaluation

7.1 Introduction .............................................................................. 196
7.2 Design and development process of the Web-Grader prototype .................... 197
7.2.1 Definition phase: What was built? .............................................. 198
7.2.2 Development phase: How was the system built? ................................. 198
7.3 Implementation of the Web-Grader prototype in a real-world situation .......... 213
7.3.1 Implementation of Web-Grader as a cyber classroom ............................ 215
7.3.2 End-user involvement in the cyclical development process of Web-Grader .... 215
List of figures:

Figure 1.1: Development research model (Plomp, 2002) ........................................... 7
Figure 1.2: Structure and interrelationships of the dissertation .................................. 11
Figure 4.1: Marked-up text of a course to be processed by Apex ................................. 109
Figure 5.1: The waterfall model (Schach, 1996:54) ....................................................... 126
Figure 5.2: The rapid prototyping model (Schach, 1996:59) ....................................... 128
Figure 5.3: The Dick and Carey model (Dick & Carey, 1996:2,3) ............................... 132
Figure 6.1: Development research model (Plomp, 2002) ........................................... 168
Figure 6.2: How E-Grader works ................................................................................. 169
Figure 6.3: The current assessment and recording process at Unisa .............................. 172
Figure 6.4: An electronic grading system included into the current system to conduct assessment of open-ended textual input ................................................................. 175
Figure 6.5: The basic algorithm of E-Grader ............................................................... 181
Figure 6.6: Correlations between grades obtained by E-Grader and a control data set .... 188
Figure 6.7: Correlations between grades obtained by E-Grader and two control data sets 192
Figure 7.1: Interactive Web-Grader ............................................................................. 196
Figure 7.2: Database-driven Web-Grader .................................................................... 198
Figure 7.3: Basic algorithm of Web-Grader ............................................................... 200
Figure 7.4: Web-Grader's dialogue chart .................................................................... 201
Figure 7.5: Icons used in Web-Grader ....................................................................... 202
Figure 7.6: A combination of different types of menus ................................................. 202
Figure 7.7: Pop-up windows used in Web-Grader ...................................................... 203
Figure 7.8: Database structure used in Web-Grader .................................................... 205
Figure 7.9: Basic Web-Grader algorithm with different feedback based on input ......... 207
Figure 7.10: Feedback for receiving less than full marks but more than zero ............... 209
Figure 7.11: Feedback for receiving full marks ............................................................. 209
Figure 7.12: Screen shots of the COS321-6 CyberClassroom2002 ............................. 210
Figure 7.13: Home page of the cyber classroom .......................................................... 216

List of tables:

Table 5.1: Differences between E-Grader and Web-Grader ....................................... 157
Table 6.1.1: Question 1.2 ......................................................................................... 183
Table 6.1.2: Question 2.1 ......................................................................................... 183
Table 6.1.3: Question 6.1 ......................................................................................... 184
Table 6.2.1: Question 1.4 ........................................................................................................ 190
Table 6.2.2: Question 5a ....................................................................................................... 190
Table 6.2.3: Question 5b ....................................................................................................... 191
Table 7.2: Black and William’s elements applied in Web-Grader ........................................... 206
Table 7.3: Overall performance of students in the COS321-6 examination .......................... 222
Table 7.4: Performance of cyber classroom users in the COS321-6 examination ................. 223
Table 7.5: Performance of students using the cyber classroom to obtain admission to the COS321-6 examination ........................................................................................................ 224
Table 7.6: Educational design principles of Web-Grader addressed in the study ................. 225
Table 7.7: Software engineering principles of Web-Grader addressed in the study ............... 226

List of example boxes:
Example box 5.1 ...................................................................................................................... 141
Example box 5.2 ...................................................................................................................... 142
Example box 5.3 ...................................................................................................................... 143
Example box 5.4 ...................................................................................................................... 143
Example box 5.5 ...................................................................................................................... 144
Example box 5.6 ...................................................................................................................... 144
Example box 5.7 ...................................................................................................................... 144
Example box 5.8 ...................................................................................................................... 145
Example box 5.9 ...................................................................................................................... 145
Example box 5.10 .................................................................................................................... 146
Example box 5.11: File named “result” .................................................................................. 149
Example box 5.12: File named “temp” .................................................................................. 149
Example box 5.13: File named “temp2” ................................................................................ 149
Example box 5.14: File named “temp3” ................................................................................ 150
Example box 5.15: File named “temp4” ................................................................................ 150
Example box 5.16: File named “result3_5b” .......................................................................... 150

List of text boxes:
Text box 6.1: How E-Grader marked different responses to a question ................................. 185
Text box 7.1: A typical student interaction with the system ..................................................... 214
Text box 7.2: How Web-Grader marked two different responses to one question ................. 216
List of code fragments:

Code fragment 5.1 ................................................................. 141
Code fragment 5.2 ................................................................. 147
Code fragment 5.3 ................................................................. 147
Code fragment 5.4 ................................................................. 148
Code fragment 5.5 ................................................................. 151
Code fragment 5.6 ................................................................. 153
Code fragment 6.1 ................................................................. 177
Chapter 1: Introduction

1.1 Problem statement

Usage of the Internet and World Wide Web (WWW) for instruction and learning is proliferating. As an instructional technology, the WWW is unique in its ability not only to transmit a variety of media, but to do so independently of time and place, providing asynchronous access on a global basis. The WWW permits sharing of information, and has the potential to become the most comprehensive communication system ever developed.

However, web-based learning environments are in danger of becoming passive learning technologies, due to the common practice of uploading conventional educational material, with little optimisation of the unique features of the WWW. Most course-based or learning sites simply serve as channels for course materials and information. This type of utilisation ignores the WWW's potential as a technology for powerful interactive learning experiences. For web-based learning to be more effective, it needs to be more interactive.

A reason why web-based learning is not optimally effective, is that most web pages are static. This is because the majority of web pages are mere HTML or text files, which are downloaded to the browser and are instantly displayed. However, web pages need not be static, because programs such as Java, JavaScript, PHP and CGI programs, which generate HTML as their output and then return that HTML to the browser, can be used. This results in dynamic pages that return different results based on the arguments supplied when requesting a page. For more effective interactive web-based learning, developers must move towards dynamically-generated web sites. There is therefore a need for research on how new technological developments such as XML and PHP can be used to enhance HTML in order to produce better interactive web-based learning.

For optimal learning, interactive web sites should not only support interactivity in the form of learners selecting options and manipulating information. They should also support learners in making their own input in the form of communication and contributions to the site, but particularly in presenting projects, assignments and other deliverables electronically. Automated assessment of such input is therefore a desirable feature for effective electronic learning (e-learning). Most of the assessment currently done on the WWW entails multiple-choice testing. There is, however, a need for software tools that can conduct accurate assessment of open-ended input.
"Current web-based learning environments are better at presenting the content than assessing the learning. Indeed, these environments provide sophisticated access to learning content by means of images, videos, sounds, hypertexts, glossaries, etc. However, their main weakness lies in the assessment part. Either the tests are based on rigid technologies such as quiz or multiple-choice questions, or a teacher is required to manage forums or asynchronous e-mail exchanges. Vigilante (1999) measured the amount of time teachers spend to manage typical on-line courses with intensive discussion/collaboration areas. He showed that 25% of on-line time is spent in grading written assignments. This time-consuming task could be reduced by the development of automated grading systems" (Dessus, 2000: http://www.upmf-grenoble.fr/sciedu/pdessus/caps00.PDF).

This MSc half-dissertation focuses on the development of one such tool that might increase the effectiveness of automated electronic assessment (e-assessment) and hence the utility of web-based learning. The tool was developed in the context of the University of South Africa (Unisa), a distance-learning organisation.

"At a time where increased costs and resource constraints are experienced by many tertiary institutions globally, the pressure to find alternative methods to deliver teaching and assessment increases" (Gardner, Sheridan & White, 2002:125).

1.2 Research question

1.2.1 Primary research question

One of the greatest challenges in creating interactive web-based learning, is the issue of electronic assessment (e-assessment). Current e-assessment consists mainly of multiple-choice answers, quizzes, true-false statements, and dialogue boxes for short answers, which present little opportunity for interactive communication. The highly interactive nature of the WWW is a motivating factor for adopting new methods of assessing users' responses and open-ended inputs, such as textual and graphical products. E-assessment and marking (grading) of open-ended inputs is a complex task, requiring considerable further research, both on a principle level
as well as on the technical implementation of sensitive judgement and feedback facilities. **Scope exists for research into automated means of assessment that go beyond the mere judging of multiple-choice and true-false questions.** The implementation of such assessment requires complex computer programming skills.

In this study, an e-assessment tool was designed, developed, implemented, tested, and analysed. The functionality explored entailed firstly, assigning basic grades to learners' products and secondly, the provision of interactive feedback to learners' efforts.

The main research question that guides this study therefore is:

**To what extent can e-assessment of textual inputs enhance web-based learning?**

### 1.2.2 Subquestions

The main research question gives rise to sub-questions:

- How do the marks (grades) obtained using the prototype e-assessment tool correlate with marks assigned by human assessment?
- What is the value of interactive feedback to learners?

These questions will be answered in the course of the study.

### 1.3 Value of the study

This study on e-assessment has potential utility for both academics and learners - for academics by reducing their marking overload, and for learners by providing them with more than mere grading.

If assessment can offer meaningful feedback in addition to grading, it could provide learners with more control over their learning process resulting in greater learner-centricity. E-assessment of assignments could also enable learners to receive immediate feedback, bypassing the tedious
process of submitting assignments by hand or through the postal service and waiting for them to be returned (as in the case of Unisa).

Academic institutions could benefit by using the tool to reduce the labour-intensive, time consuming and expensive process of collecting assignments, marking them manually, storing results, returning marked assignments, providing results, and handling administrative enquiries.

1.4 Scope of the study

1.4.1 Domain of the study

Research literature and existing software programs (precedents) for e-assessment of textual inputs have been investigated as the foundation of this study. Background information is presented on the progression and features of web-based learning. This is done to set the context and to establish a common frame of reference. The purpose is not to give a complete overview of the topic of web-based learning.

Furthermore, markup languages (such as HTML and XML) and CGI scripting languages (such as Perl, Python, JavaScript and PHP) were studied to investigate their utility and to determine which languages are most appropriate for implementing the tool.

Two variants of the prototype software system are described in this study:

(1) The first variant, E-Grader, which assigned marks to learners’ work products, was developed to initially test the researcher’s algorithm. This prototype was built to conduct e-assessment in the Unisa context where assignments can be submitted electronically via Students On-line (SOL). This variant was not a truly interactive system, since input first had to be converted from the form in which it was received at Unisa. The textual data used in the study was taken from electronic versions of assignments (in MSWORD format), which had to be individually converted to files in a text format (.txt), and had to be divided into separate files for each question before they could be batch-processed by E-Grader.

Students On-line (SOL) is a service for registered Unisa students to get access to administrative and academic services via the Internet. The system includes most of the services that are normally done via mail or by visiting the campus.
The second variant called Web-Grader, was an interactive web-based extension of the first variant. It provided interactive support to students in answering questions as they respond textually to content-based questions as part of their learning process. This variant was designed to be implemented as a study support tool rather than as part of a formal assessment system. Model answers to the questions were not made available for reading or downloading, but the answers keyed in by students were assessed, and guidelines based on the correct answers were provided.

The two systems mentioned above investigated free-text assessment based on keyword matching, and took into account a limited amount of semantic content such as:

1. Testing text input for "... not..." to prevent learners obtaining marks for texts containing negation of correct information, while including correct or key terms. For example in assessment of a discussion on the rainbow and its colours, marks would not be awarded for "not red", "not blue" etc.).

2. A mark for a specific aspect could only be awarded once. This feature would ensure, for example, that learners who enter "red" seven times do not receive seven marks when listing the colours of the rainbow.

1.4.2 Limitations and delimiters

Although search engines are web-based services that evaluate input text to help users in finding files on the WWW, a study of the operation of open-source search engines was excluded. The central problem when building adequate search engines is creating and maintaining an inverted index (most search engines use B-trees for indexing), whereas the prototype software system developed for this study does not make use of indexing.

The system developed for this study was not a complete system, but only a prototype to demonstrate e-assessment and how it could be extended to be used for more than mere grading. The intention of this study is to test the argument, and not to prove the argument.
Text data was validated by using regular expressions or pattern matching; no artificial intelligence techniques were used.

1.5 Resources consulted

The following resources were consulted:

- In order to investigate web-based learning, a survey was conducted of journal articles, articles published on the WWW, and conference proceedings.
- Furthermore, existing software for creating web-based learning (see Section 1.6.2) was investigated.

1.6 Research design and methodology

The study undertaken by this half-dissertation can primarily be classified as research with a development goal (Reeves, 2000). It aims for the dual objectives of developing a creative approach to solving a real-world problem in the field of educational technology, while simultaneously constructing software engineering design approaches/principles for future development.

1.6.1 Development research

Plomp (2002) describes the starting point in development research as the process of designing and/or developing an intervention (or product) as a solution for a need. Plomp’s development research model is shown in Figure 1.1.
The researcher conducted a theoretical study to overview the field and to identify a need amongst practitioners of web-based learning. The development effort comprised the design, development and implementation of two prototype software systems, generated as possible solutions to the needs identified. The prototype was then used in interventions followed by empirical investigation, which entailed the evaluation of these prototypes and their consequent outcomes. This study entails a cyclical process similar to Plomp’s development model, but without an explicit process hypothesis, intervention hypothesis and impact hypothesis.

![Diagram of Development Research Model](image)

**Figure 1.1 Development research model (Plomp, 2002).**

### 1.6.2 Theoretical research

1. A survey was undertaken of learning theories and instructional design. This forms the foundation for further study on the progression and features of web-based learning.

**Methods:**
- Study research literature on web-related learning and concepts obtained from journals, articles published on the WWW, and conference proceedings.
(2) Research was conducted on e-assessment techniques currently used for web-based textual inputs.

**Methods:**
- Study research literature and existing programs on e-assessment of textual inputs.
- Investigate existing software products.
- Study markup languages (such as HTML and XML) and CGI scripting languages (such as Perl, Python, JavaScript and PHP).

### 1.6.3 Intervention

The design process and intervention (see Figure 1.1) undertaken for this study comprised the design, development, implementation and evaluation of two variants (*E-Grader* and *Web-Grader*) of a prototype assessment system that enabled automatic marking of open-ended textual responses. The domain was a module in the content area of Operating Systems and Architecture. The design and development of these systems consisted of iterative cycles of design and development activities, and was done according to software engineering principles. Implementation and evaluation were conducted among a target group of third-level Computer Science students at Unisa.

(1) The first variant of the prototype system was called *E-Grader*. This was a non-interactive system developed mainly for testing the algorithm and the feasibility of the concept of e-assessment of open-ended textual responses.

**Methods:**
- Investigation of how text data can be validated by using regular expressions or pattern matching in PHP.
- Development of algorithms.
- Prototyping - Software development using Linux scripts and PHP on the Linux operating system.
- Testing *E-Grader* by applying it to some questions from Assignments 1 and 3 of COS321-6 (a 3rd-level Computer Science module on Operating Systems and Architecture) and comparing the results to students’ scores obtained by hand-marked assignments and on hand-crafted test data. Findings were compared by using *Microsoft Excel*. 
The second variant of the prototype system was called Web-Grader. This was an extension of E-Grader, which enabled automatic marking of open-ended textual responses, and also provided users with instant learning support in the form of detailed feedback without human involvement.

**Methods:**

- The software of the first variant (E-Grader) was extended to Web-Grader by:
  - adding a graphical user interface (GUI) using HTML, PHP, and JavaScript,
  - connecting the system to the Apache web-server,
  - converting the E-Grader system to a database-driven web site using MySQL relational database,
  - implementing Web-Grader as a cyber classroom,
  - involving end-users in the development of the system, and
  - evaluating the outcomes of the system both qualitatively and quantitatively.

- Adhering to appropriate ethical standards of informed consent, participants were informed beforehand of the implications of their involvement (Cohen, Manion & Morrison, 2001).

### 1.7 Structure of the study

The study is divided into two parts:

1. Part 1 describes the progression of web-based learning, and identifies the need for electronic assessment of web-based textual inputs.

2. Part 2 involves the development, implementation, and testing of two variants of an intervention in the form of prototype software systems called E-Grader and Web-Grader respectively, that can be used for electronic assessment of web-based textual inputs.

Part 1, comprising Chapters 1 through 4, provides background information on:

- The features of web-based learning, and how constructivism introduced a move towards more interactive learning.

- Research on e-assessment techniques currently used for web-based textual inputs.
The purpose of the background information presented in Part 1 is to establish a common frame of reference. The structure and interrelationships of the dissertation are set out in Figure 1.2, and a description of the general layout follows.

A brief introduction overviewing the research problem, as well as the value and scope of the study, an outline of the methodology used, and the structure of the report, are presented in Chapter 1.

Chapter 2 provides an overview of some of the main learning theories that have developed in different fields of scholarly endeavour during the past century. The chapter starts by highlighting some of the characteristics of two of the mainstream learning theories, namely behaviourism and constructivism. It then focuses on early perspectives on learning in different parts of the world, and the move towards constructivism. Theoretical concepts of behaviourism and constructivism are discussed, and their practical implications for education and technology-based instruction and learning are examined.

The role of the WWW as an electronic learning environment is discussed in Chapter 3. Its potential for implementing or supplementing distance education is described, as well as the main features of web-based learning. Problems related to assessment and interactive learning in web-based learning are also discussed in Chapter 3, and some possible solutions are proposed.

In Chapter 4 on-line assessment is discussed. This chapter focuses on currently available automated grading systems, and three types of e-assessment are investigated and discussed:

- Free-text assessment based on surface features (semantic content of the text is not taken into account).
- Free-text assessment based on course content.
- Automatic assessment of computer programming assignments.
Part 2, comprising Chapters 5 through 7, involves the design, development and evaluation of two variants of the prototype system, *E-Grader* and *Web-Grader*.

Chapter 5 describes the methodology and tools used to produce these prototype systems.

Chapter 6 sets out the development process, implementation and evaluation of the first variant called *E-Grader*, which conducted automated assessment of textual responses. Principles and techniques from the areas of software engineering, formal computer science, and database development were incorporated in the design and development of the prototype software system. The implementation was validated by comparing the results to hand-marked assignments, and on hand-crafted test data. Findings of the implementation and lessons learnt from the implementation exercise were recorded.

The development process and implementation of the second variant called *Web-Grader* is described in Chapter 7. *Web-Grader* operated on the web server, Osprey, of Unisa’s School of Computing, as the COS321-6 *CyberClassroom2002*. This interactive system was used by COS321-6 (Operating Systems and Architecture) students to answer learning support questions from a collection of previous years’ assignments. The answers keyed in by students were graded, and feedback was provided in the form of guidelines based on the response. An electronic notice board, which could be read by all the other students, was available where students could communicate with each other. Findings of the implementation, and lessons learnt from the outcomes of the implementation exercise were recorded.

Summary, conclusions drawn and recommendations are presented in Chapter 8.
References - Chapter 1


Plomp, T. 2002. *Some reflections on development research (DR).* Notes for a lecture presented at the University of Pretoria. Enschede, the Netherlands: University of Twente.

Chapter 2: Learning theories and instructional design

2.1 Introduction

The practice of instructional design should be based on sound learning theories. The aim of this chapter is to give an overview of some of the main learning theories that have developed in different fields of scholarly endeavour during the past century, prior to focusing on web-based learning in Chapter 3.

The chapter starts by introducing general perspectives on learning, and then highlights some of the characteristics of two of the mainstream learning theories, namely behaviourism and constructivism.

Practical implications of the theoretical concepts of behaviourism and constructivism for technology-based instruction and learning are then examined in Section 2.3.

2.2. Theoretical and conceptual aspects

2.2.1 General perspectives on learning

During the first half of the twentieth century the scholarly study of learning in the United States was heavily influenced by the behaviourism that dominated American psychology during this period, and especially by the work of Thorndike and Skinner. Learning was conceptualised as a phenomenon that occurs from the outside-inwards: environmental stimuli impinge on an individual who then makes a response (Skinner, 1938). It was believed that the consequences of this response (i.e. reinforcement) determine the probability of that response re-occurring when the same or a similar situation is encountered. In the United States research on learning consisted largely of laboratory studies (often with animals) using simplistic tasks for the purpose of identifying universal laws of learning (Marx & Hillix, 1963).
In contrast to the functional orientation towards learning in the United States research, European literature and research on learning during this period focused more on the structural characteristics of mental functioning. The principal aim of this approach was an understanding of the mental processes responsible for thinking and cognition. This focus was evident in the studies of perception by Gestalt psychologists and the studies of cognitive development by Piagetian researchers. These studies were based on a philosophical concern for understanding the human mind. In contrast to the interest in the USA approach of identifying factors responsible for educational change, there was little European concern for investigating how mental processes could be altered through instructional intervention (Inhelder & Piaget, 1958; Marx & Hillix, 1963).

The behavioural explanation of learning that emerged from research in the United States was considered too mechanistic by European psychologists, while European approaches were criticised by American psychologists for ignoring the ‘change’ they considered central to learning and for producing studies that lacked methodological rigour. Following the Second World War, however, a variety of social and philosophical changes, along with certain technological advances, set the stage for new conceptions of the human mind. Theories of human information processing and the impact of technology, particularly computers, began to influence educational psychology in the USA, as did changing beliefs about the individual’s role and responsibility in society. The spirit of this age - reflecting the individual’s ability to influence the course of events both historically and personally - became most pronounced during the 1960s, when a strong egalitarianism philosophy began to influence both United States psychology and social action. These factors made it possible to reconcile some of the differences between American and European approaches to learning (Marx & Hillix, 1963).

Cognitive scientists, influenced by German Gestalt psychology and the Wurzburg school in Europe, began to view the learner as an active human information processor analogous to computers, and conceptions of learning were proposed from the perspective of cognitive psychology (Newell & Simon, 1972). Consistent with European interest in mental processes, these early cognitive psychologists focused on the internal mediation that occurs between the originating stimulus and the learner’s response, for example, how the individual interprets the stimulus and processes it before making a response. This focus on the internal and active processes of the

---

1Piaget is well-known for his formulation of different stages of cognitive development. He developed his cognitive theory by observing children while they were learning (see Section 2.2.2.2).
individual generated new conceptions of what constitutes learning. It should be noted, however, that while prevailing theories focus primarily on complex, meaningful forms of learning, other types of learning also exist. There are several different types of learning, and a comprehensive understanding of learning must include both behavioural and cognitive theories (Marx & Hillix, 1963).

2.2.2 Learning theories

Behaviourism and constructivism are two perspectives on learning and instruction that entail different perspectives regarding how meaning is created, how learning occurs, and how knowledge should be presented.

2.2.2.1 Behaviourism as a learning theory

Behaviourism started as a school of thought within psychology. It was based on the work of John B. Watson, an American psychologist who reasoned that psychology should be concerned only with behaviour, and not with the mind or with human consciousness. The implication of this view was that people could be studied objectively, like animals (Black, 1995; DeMar, 1988; Marx & Hillix, 1963).

Watson's views were based on the research findings of Ivan Petrovich Pavlov, a Russian physiologist who investigated animals' responses to certain stimuli. In a study of the salivary responses of dogs, a bell was rung as the dogs were fed. As soon the dogs heard the bell that signalled an upcoming meal, they would begin to salivate. When the bell rang, even unrelated to the bringing of food, the dogs still salivated, for they were conditioned to salivate upon this stimulus. This salivating was called a conditioned reflex (or conditioned response), and the stimulus, the bell ringing, was called a conditioned stimulus. Watson, like Pavlov, believed that humans react to stimuli in the same way (Skinner, 1938; Marx & Hillix, 1963; DeMar, 1988; Black, 1995).

Another prominent behaviourist was B. F. Skinner, whose contribution lay mainly in testing Watson's theories in the laboratory. Skinner's research findings led him to conclude that
Watson was wrong to emphasise only reflexes and conditioning. Skinner found that although people do respond to their environment, they also operate on the environment to produce certain consequences (Skinner, 1938; Marx & Hillix, 1963; DeMar, 1988).

Another major contribution of Skinner to behaviourism was his development of the theory of *operant conditioning*. According to this theory, people may behave in a certain manner, because that kind of behaviour has had certain consequences for them in the past. For example, if an educator rewards a learner with a prize for achieving the top mark in a test, the learner will be likely to attempt this again, hoping to receive another reward. Skinner, like Watson, believed that human behaviour is shaped by experience of reinforcements, rather than being determined by mental processes or by feelings.

The basic belief of behaviourism is that human behaviour is a product of the stimulus-response interaction, and that behaviour is modifiable. The study of animals is beneficial in that it helps to understand and analyse human behaviour (Skinner, 1938; Marx & Hillix, 1963).

Although behaviourism stems from psychology, behaviouristic assumptions about mankind have had an impact on many other fields. In the field of education, many courses, including those at tertiary-level, are based on concepts and methods rooted in behaviourism. The behaviouristic belief that moral values are rooted in biology forms the basis of sociobiology, and this has been incorporated into sociology. DeMar (1988) distinguishes four assumptions underlying behaviourism:

- Behaviourism is naturalistic. Everything humans do or experience can be explained in terms of natural laws functioning in a material world. There is no other reality than this material world. Man has a brain that responds to external stimuli, but not a soul or a mind on any deeper level.

- According to behaviourism, man is fundamentally a machine that responds to conditioning. The extreme behaviourist conception of man as a biological machine without a mind that can influence his actions, is in contrast with the religious view of man being created in the image of a creative, planning, and thinking God. For
Skinner, man as a spiritual being does not exist. In his opinion, the mind and mental processes are metaphors and fictions, and behaviour is simply part of the biology of the organism.

- Behaviourism teaches that humans are not responsible for their actions. If humans were mere machines without minds or souls, reacting to stimuli and operating on the environment to attain certain ends, then anything man does is inevitable. Sociobiology, a type of behaviourism, compares man to a computer: 'Garbage in, garbage out'.

- Behaviourism is manipulative. It attempts not only to understand human behaviour, but also to predict and control it. Skinner developed the idea of behaviour shaping from his theories. He believed that by controlling rewards and punishments, one can shape the behaviour of another individual i.e. positive and negative reinforcement.

As a therapist, one of Skinner's goals was to shape his patients' behaviour though the use of positive reinforcement so that they would react in more socially acceptable ways. He proposed behaviourism as the basis for manipulating patients, learners, and even societies. Skinner believed that a person is not truly free to choose, because what individuals do in a given situation depends almost entirely on what happened to them in the past. The ethical consequences of behaviourism are severe, in that humans are stripped of their responsibility, freedom, and dignity, and reduced to purely biological beings, to be shaped by those who are able to use behaviourist tools of effectively (Skinner, 1938; Marx & Hillix, 1963; DeMar, 1988; Black, 1995).

Critique on behaviourism is that it oversimplifies human behaviour, viewing the human being as an automaton instead of a person of will and purpose (Black, 1995). However, the behaviourist approach has had a strong influence on psychology.
2.2.2.2 Constructivism

An alternative perspective on teaching and learning is called constructivism. Constructivist viewpoints on how knowledge is represented, how meaning is created, and therefore how learning occurs, differ radically from behaviourist viewpoints. Constructivists believe that individual learners build an internal and personal representation of knowledge which is influenced by their own unique experiences (Bannan & Milheim, 1997).

Jean Piaget has been labelled an interactionist as well as a constructivist, therefore a brief discussion of his research in cognitive development is presented before the principles of constructivism are discussed.

(1) Jean Piaget's views

Piaget's interest in cognitive development came from his training in the natural sciences and his interest in epistemology, a theory of knowledge. Using a standard question, or set of questions, as a starting point, he followed a child's chain of thought and spontaneously asked additional questions so that the interviews became more structured. Piaget believed that children's spontaneous comments provided valuable clues to understanding their thinking. He was not interested in a right or wrong answer, but rather in what forms of logic and reasoning the child used. After many years of observation, Piaget concluded that intellectual development is the result of the interaction of hereditary and environmental factors. As the child develops and constantly interacts with the world around him, knowledge is invented and reinvented. His theory of intellectual development is strongly grounded in the biological sciences. He saw cognitive growth as an extension of biological growth and as being governed by the same laws and principles. He argued that intellectual development controlled every other aspect of development - emotional, social, and moral (Ginn, 1995).

According to Piaget, knowledge cannot merely be transmitted verbally, it must be constructed and reconstructed by the learner. In order to learn and construct knowledge of the world, the learner must act on objects. It is this action which provides knowledge of those objects, because the mind organises reality and acts upon it. The learner must be active and should not be approached as 'a container to be filled with facts'. Piaget's approach to learning is a readiness approach which emphasises that children cannot learn
something until they have matured enough to meet certain requirements. The ability to learn any cognitive content is always related to their stage of intellectual development, therefore, children at a particular stage cannot be taught the concepts of a higher stage (Ginn, 1995).

(2) Constructivist principles

Constructivism is an epistemology (a theory of knowledge) that claims that 'we know what we know' through our senses. This means that an individual should see, hear, touch, smell and taste in order to learn. A picture of the world is then constructed with these messages from the senses. Constructivism claims that knowledge resides in individuals and that objective knowledge cannot be transferred intact from an educator to learners. Each learner attempts to make sense of what is being taught by interpreting it according to his/her own experience (Lorsbach & Tobin, 1997).

It is often difficult for learners in an objectivist learning environment to make an appropriate connection between what they learn in the classroom (e.g. the content of subjects such as biology, chemistry, history, etc.) and an understanding of the nature of problems and their solutions in the real world. Everyday real-world problems are different from the well-structured, single-solution textbook problems that learners encounter in a objectivist learning environment (Siegel & Kirley, 1997).

Constructivists believe that the complex nature of knowledge should be reflected by learning, and the development of multiple approaches in perspective should be encouraged. The goal of a constructivist learning environment is not the direct and accurate transmission of content from the instructor to the learner, but instead, learners are given tasks and opportunities, information resources and support, and are encouraged to construct their own version of the content. Many alternative paths through the lesson are allowed, and collaboration with other learners is encouraged as preferable to isolated individual learning (Dodge, 1996).

"In general, a constructivist approach is more learner-focussed, and less teacher-focussed. The emphasis is on making a set of tasks and resources available to learners, and creating an environment in which the learners can
actively create their own meaning in that context, rather than passively absorb knowledge structures created by the instructor. In this approach, the instructor's role moves toward being a coach and orchestrator of resources, and moves away from being the sole source of information. The emphasis is on case studies, problem solving, and the creation of meaning" (Dodge, 1996: http://edweb.sdsu.edu/EdWeb_Folder/People/BDodge/CTPTG/ctptg.html).

The constructivist approach requires that the educator be more than a mere presenter of knowledge, but rather a mentor and guide i.e. a facilitator. It means that the educator must shift attention more from the whole class toward the individual, resulting in greater awareness by the educator of individual needs and differences (Murphy, 1997).

Compared to traditional views of learning, constructivism is more concerned with how learners construct knowledge, not on the object of knowing. The constructivist paradigm posits that learners do not receive units of information and then store them mentally, but rather that learners take information from the real world and then construct their own view of that knowledge domain. Learners have to discover concepts without being directly given the information they need. The more the problem-solving learning situation represents the real-world, the more likely it is that the learner will transfer the skills to other problem-solving situations.

Constructivism is an approach to teaching and learning based on the assumption that learning is the result of mental construction. Learners learn by integrating new information with prior knowledge. Constructivists believe that learning is affected by the context in which an idea is taught as well as by learners' beliefs and attitudes. Constructivist teaching is based on recent research about the human brain and what is known about how learning occurs (Dodge, 1996; Greening, 1998).

With regard to assessment of learning, constructivists believe that it should be embedded in the context of learning, and that learning gain should be measured, rather than whether the learner has achieved a certain standard or attained a
particular skill (Dick, 1991). The focus of assessment lies on what has been constructed by the learner during the learning process.

2.3 From theory to practical implementation

2.3.1 Introduction - the role of technology

This section takes the theoretical concepts of behaviourism and constructivism introduced in Section 2.2 and examines their practical implications for technology-based instruction and learning. Various views exist on the role of technology in education. Some of these are mentioned in this introductory subsection.

Greening (1998) believes that one of the effects of technological change on society is that the educational system has lagged behind, resulting in a chasm between the learning that takes place in educational settings and that which occurs in wider society. He warns that the temptation to address the problem of rapidly changing educational expectations by over-eagerly adopting any new technology may lead to little more than "outwardly appealing Band-Aid solutions". Greening points out that technology should be integrated into educational settings with discernment, and the process has to rest upon a sound theoretical foundation.

For many years the proponents of technology in teaching have been predicting that technology (programmed instruction, television, video, computers) would revolutionise the way people learn. Kearsley (1998) argues that 'technology has become the great siren song of education'. The amount of attention and resources devoted to the use of technology in education distracts educators and learners alike from the really important problems and issues that need to be addressed. He feels that educational technology is primarily a distraction from what matters most, namely effective learning and good teaching. Kearsley argues that until some of the major problems of education (such as ineffectual educators, incompetent administrators, irrelevant curricula, weak leadership and political interventions) are addressed, there is no reason to expect that educational
technology will make much difference. Kearsley believes that a new conceptual basis for applying technology should be developed, and in the meantime the use of technology in education should be de-emphasised and the focus placed on addressing the genuine needs of learners, communities and societies.

Bartasis and Palumbo (1995) state that rapid advances in computer hardware and software continue to lay the foundation for new and imaginative designs in technology-based instruction. Easy-to-use authoring systems have opened up the possibility of development of instructional systems by non-programmers, especially educators. Nevertheless, an important aspect of education, namely learning theory, should not be overlooked in the excitement of creating useful, engaging instruction in a brand new medium. Despite the fact that great care has been taken in the development of design strategies, needs assessment, and evaluations, instructional designers often fail to focus on learning theory.

"It is important to remember that a theory of learning is an outgrowth of a philosophy of education and further, that a learning theory which is not a subset of a philosophy of education, is not strong enough to allow for confident implementation and expected resultant changes in learning outcomes" (Bartasis & Palumbo, 1995: http://129.7.160.115/inst5931/discovery_learning.html).

Bartasis and Palumbo (1995) believe that a lack of understanding at the theoretical level results in various self-defeating, ineffectual, and superficial practices within educational technology. Effective teaching requires that one possess a theory of learning, i.e. a body of conceptual tools concerning the definition of learning and beliefs about how learning takes place, about the significance of motivation, and the meaning of forgetting or non-learning.

Bartasis and Palumbo warn that, in order for technology and learning theory to complement or enhance each other's performance, designers should not 'borrow' pieces of prominent learning theories from behaviourists and cognitive psychologists in the same way that educators use bits and pieces of learning theory in their classroom instruction. They also warn against the use of a variety of methods, each based on a different theory of learning, because it results in a complex and ineffective foundation for prediction of
results, performance, and evaluation. They further warn that it would be unacceptable to invent a theory to meet the needs of a particular instructional design, simply because a new medium is used to develop the instruction. They point out that the introduction of video recordings into the classroom did not change the mechanisms of human cognition. Instead, educators relied on their knowledge of how learning occurs to incorporate the use of videos in the classroom to enhance the process of learning (Bartasis & Palumbo, 1995).

"What would seem to be most effective, however, would be to find a well-suited match between a particular type of technology-based instruction and a learning theory. The ideal candidate for the learning theory component of this equation would be one that has stood the test of time, that has proven merit, that would allow for predictable results and replication. The next step would be to select a technological framework that shares the same principles as the theory, and design instruction highlighting the best of both" (Bartasis & Palumbo, 1995: http://129.7.160.115/inst5931/discovery_learning.html).

2.3.2 Theoretical foundations of instruction and technology-based instructional design

One of the first learning theories to positively influence the design and effective use of technology-based instruction was that of the behaviourists, most notably B.F. Skinner (discussed in Section 2.2.2.1). The impact of behaviourism on instruction and instructional technology is discussed in Section 2.3.2.1.

Learning theorists and instructional designers have lately abandoned this external, objectivist philosophy for an internal, constructivist philosophy of learning. Implications of constructivism for education and for instructional technology are discussed in Section 2.3.2.2.
2.3.2.1 Behaviourism - impact of behaviourism on instruction and instructional technology

Skinner's introduction of *programmed instruction* was one of the first attempts to implement behavioural principles in an instructional situation. Subject matter is broken into small, understandable steps, chunks or *frames*, each followed by a question which the learner can almost always answer correctly. The object of this technique is reinforcement of the learning process through an immediate response, and the reward of getting it right (Bartasis & Palumbo, 1995; Black, 1995).

Bartasis and Palumbo (1995) state that the following principles of behaviourism which contribute to effective instruction, have been incorporated into the design of technology-based instruction, and have become standard design considerations:

- Positive reinforcement.
- Feedback.
- Errorless learning.
- Use of complex schedules of reinforcement.

Black (1995) states that behaviourist principles have been proven useful for managing both classroom behaviours and the delivery of instruction, and that behaviourism has influenced the development and design of several technologies such as the following:

- In the 1960s *teaching machines* were introduced that provided self-paced delivery of instruction based upon the principles of programmed learning. In 1968 Fred Keller proposed the use of the Personalized System of Instruction (PSI) for college instruction. Two programming designs for this technology exist, namely linear and branch design. Linear design lays out a single sequence of frames for all learners to follow; while, in branch design, there are various optional paths through the instruction, and the
system places learners on an appropriate path(s) due to each individual response to key questions.

- **Computer-Assisted Instruction (CAI).** Since the mid 1980s the rapid growth of personal computers in society has been accompanied by an explosion of educational (instructional) software packages. The first generation of these software packages was generally designed in a linear manner. Black lists the key behaviour modification principles used to program these applications:
  - Stating the purpose of the software.
  - Application of the appropriate reinforcer in the form of text, visual or audio.
  - The use of shaping, chaining, modelling, punishment, and award principles depending on the application.
  - A scoring or monitoring system.
  - Providing the status of progress in various forms: drill and practice activities, simulations, and tutorials.
  - The use of Virtual Reality (VR) creating a new learning environment to explore.

Black suggests that VR can be particularly useful in the discipline of Science, because it provides a 3D view of the world. She also believes that behaviourist principles, like other learning theories, will continue to play a major role in the building of this environment. In her view, the human-computer interface - as well as the lack of appropriate forms of reward or reinforcement - are the weaknesses of CAI. In her opinion VR can be used to address the interface issue by providing a 'real' teacher to give instruction, and even more so, if the learners can create their own teacher image to teach them, the reward system would be more effective than a plain 'good job' message.
2.3.2.2 Constructivism

(1) Implications of constructivism for learning

Ginn (1995) believes that the teaching methods commonly used in the USA namely lectures, demonstrations and audio-visual presentations by teachers/instructors, teaching machines, and programmed instruction, do not correspond with Piaget's ideas on the acquisition of knowledge. Piaget advocated active discovery-learning. He believed that learners should explore, manipulate, experiment, question, and search out answers for themselves. In short, activity is essential. This does not mean that learners should be allowed to do whatever they want. Educators should be able to assess learners' current cognitive levels, as well as their strengths and weaknesses. Instruction should be individualised as far as possible and learners should have opportunities to communicate with one another, to discuss and debate issues. Constructivists view educators as facilitators of knowledge whose purpose is to guide and stimulate the learners as they play an active role in constructing their own knowledge. Learners must be allowed to make errors and to learn from them. Learning is much more meaningful if learners are allowed to explore on their own, rather than passively receiving information. The educator should present learners with materials, situations, and occasions that allow them to discover new learning. In active learning, the educator must have confidence in the learners' ability to learn on their own.

(2) Implications of constructivism for instructional technology

Ginn (1995) states that technologies that encourage interactivity - such as multimedia, hypermedia, and virtual reality - implement Piagetian thought. She also believes that computer software that is strictly drill-and-practice, does not fit in with an active discovery environment, since it does not encourage creativity or discovery. Learners can create their own multimedia learning environments by using authoring tools, thus using multimedia to learn, as well as to communicate their understanding of the subject to others within the context of collaborative learning. Peer-teaching is used as the learners jointly tackle tasks and develop products. This way, learners become active participants instead of passive recipients (Greening, 1998).
While behaviourist instruction is designed to proceed in a linear fashion (as controlled by the application), with regular progress checks based on predetermined learning objectives for feedback, constructivist use of technology tends to support non-linear navigation of information and learning experiences, rather than the 'walking in the footsteps of experts' approach. Non-linear navigation of information gives learners a choice between multiple paths of learning (Greening, 1998).

(3) Hexa-C Metamodel
The Hexa-C Metamodel (De Villiers, 1999; 2002) is a framework that integrates concepts from contemporary instructional and learning theory. It proposes six inter-related elements to be considered in the design of e-learning systems and in evaluating them from the perspective of learning theory, namely: constructivism, cognitive learning theory, componential instruction, collaborative learning, creativity, and customisation.

2.4 Conclusion

This chapter explored the changing world of teaching and learning, and the implications of learning theories such as behaviourism and constructivism on technology and learning. The purpose of this investigation is to ensure that subsequent development efforts are founded on a sound theoretical base, and that an appropriate match is achieved between technology and learning theory (see Section 2.3.1).

The chapter started by concentrating on the theoretical and conceptual aspects of teaching and learning. Learning theories have changed dramatically since the 1960s. Traditional theories (based largely on stimulus-response views of behaviour) have been replaced by newer theories based on cognitive psychology and a concern for social, cultural and developmental factors.

Traditionally, learning has been defined as a change in behaviour or performance resulting from experience and practice. Current instructional and learning theories,
however, are tending towards a constructivist bias. Although a concern for change is still evident, the emphasis in the 1990’s shifted to the restructuring of knowledge and changes in understanding rather than changes in behaviour. Learning is not viewed merely as the acquisition of behaviours or isolated facts. The current emphasis is on problem-solving rather than recollection. The restructuring of a learner’s prior knowledge as a foundation for new knowledge, is acknowledged as a more appropriate way of attaining meaningful learning.

Constructivism emphasises learning and not teaching. In the constructivist theory, the emphasis is placed on the learner rather than on the instructor, who becomes less of an instructor and more of a facilitator. The learner interacts with the real world, and thereby gains an understanding of real-world concepts. Learners construct their own conceptualisations and solutions to problems.

Section 2.3 showed that behaviourism and constructivism offer divergent viewpoints on how knowledge is presented, how meaning is created, and therefore how learning occurs. When instructional design is based on behaviourism, the content of instruction is organised by the instructor and delivered to the learner. Such instruction involves shaping desirable behaviours through the arrangement of stimuli, responses, feedback and reinforcement. On the other hand, constructivist design of the same course would offer a contextualised environment including opportunities for learners to synthesise, organise, and restructure information, as well as to create and contribute their own resources.

It is important that developers of instructional material, educators, and researchers bear in mind how learning occurs, the various types of learning, and the ways in which learning can be influenced by design. These factors have a major impact on the nature of technology-based learning environments and resources.
References - Chapter 2


Chapter 3: Web-based learning

3.1 Introduction

As indicated in Chapter 1, the emergence of the WWW has led to fundamental changes in industrial training, formal education and self-directed learning. The WWW holds the capability to foster active participation and higher-order thinking skills within learners, i.e. the type of interactive constructivist learning described in the previous chapter. In order to achieve full potential, however, it is essential that e-learning (electronic learning) environments are well designed and managed.

The main features of web-based learning, as well as aspects of the Internet used in web-based learning, are discussed in this chapter, that also serves as a reference manual of Internet technology for Chapters 5, 6 and 7 where the tools and development process of the software systems developed for this study are discussed. Furthermore, problems related to assessment and interactive learning in web-based learning are discussed here, and some possible solutions are proposed.

3.2 Role of the WWW in implementing and supplementing learning

The WWW is, in principle, accessible to all learners worldwide who have an Internet connection. It is a user-friendly, fast and efficient, global, multi-national and multi-lingual, time-unrestricted, and cost-effective information distribution mechanism. The WWW is a new way of organising and presenting instruction; moreover, it is delivery medium, content provider, and subject matter all in one. Information on the WWW is organised in an ever-expanding network of nodes and links that represent the more traditional domains of knowledge. The versatility and interconnectedness of the WWW make it one of the most effective ways to reach learners who are geographically widespread.
Web-based learning

The WWW is changing the way in which humans communicate and the way they use computers (Campbell, 1998). Due to technological innovation, physically remote individuals and groups can communicate with each other in an instant. Friends, family and colleagues stay in close contact through e-mail; medical experts can care for patients in remote areas using video-conferencing software; learners can study on-line courses or attend lectures using distance-learning tools, and day-to-day business operations are transformed by on-line data transmission, e-commerce and web sites.

In particular, the WWW offers many possibilities for implementing or supplementing learning including (McLellan, 1998):

- on-line tutorials;
- virtual classrooms where classes are taught entirely on-line;
- correspondence courses where instructional materials are sent out to individual learners and returned via the WWW rather than by mail;
- project-based instruction centred around individual or group projects developed and implemented on-line;
- event-based instruction where activities are centred around an event in which learners participate via a real-time linkup of some kind (text, audio, video), and
- project-based, problem-based and case-based learning where learners conduct searches to access web-based information.

In addition, learning management systems which supplement academic content with learners’ records, curriculum information, etc. can be added to McLellan’s list.

The Internet serves as a two-way tool for communication and a source of information in general. WWW resources provide both learners and educators with access to the most current information and professional contacts in their subject areas. Because of learning benefits such as these, many institutions are using the WWW within their instructional and learning programmes. Its potential as an educational technology is due to its ability to transmit varied platform-independent educational contents over space and time. Time-independence and space-independence allow learners to adjust the education to their lifestyle. Networking and data-sharing across different platforms practically extend the power of individual computers. This internationally-used medium narrows the distance between nations and brings cultures together (Dimitrova, 1997).
3.2.1 Main features of web-based learning

Web-based instruction entails teaching and learning supported by the attributes and resources of the Internet (Khan, 1997). Many existing web-based learning sites have been developed to provide learners with access to instructional resources from a distance. These sites range from those that merely post lecture notes and course materials on the WWW with links to other digital resources, to sites that involve learners in collaborative activities with the aim of promoting interaction, involvement, and engagement.

This section introduces various features commonly found in web-based learning.

3.2.1.1 On-line registration and enquiries

Learners can enroll in web-based courses via interactive on-line registration, enable on-line credit card payments, and make administrative enquiries from anywhere in the world (distance-independent) by using any computer platform (device-independent) at any time of the day or night (time-independent) (Khan, 1997).

3.2.1.2 Access control

With the openness of the WWW any user can access a page unless it is password-protected. Because only enrolled learners, and not general web-surfers, should be allowed to participate in or view classroom discussions, many instructors choose to password-protect their home pages, so that only registered learners may access them. Proper security measures are also required to ensure confidentiality and integrity of records stored in database files containing grades, scores and feedback on evaluations (Malikowski, 1997; Rasmussen, Northrup & Lee, 1997).

3.2.1.3 On-line communication

E-mail, bulletin boards, list-servs, news groups, chat groups and desk-top video-conferencing provide learners with a choice of asynchronous or synchronous communication media that can be used in accordance with their needs (see Section 3.2.2.1). These media play valuable roles in exposing learners to real-life contexts. By
subscribing to specific list-servs, and news groups, learners can have access to leading experts in various fields of expertise, and can learn about current, topical real-life problems and solutions that are not available in textbooks.

3.2.1.4 On-line assessment

Assessment, in general, can be *summative* or *formative*.

- Summative assessment is the attempt to summarise a learner’s learning at some point in time, such as at the end of a course. Summative assessment is formally graded, and counts towards the final mark for the course.
- By contrast, formative assessment provides feedback to learners on their progress during a course, in ways that enable them to improve their learning. This can take many forms, ranging from graded tests to informal advice (Nicoletto, 2001).

On-line assessment, also called electronic assessment (or *e-assessment*), can be integrated into courseware in various ways. Current electronic assessment consists mainly of multiple-choice answers, quizzes, true-false statements, and dialogue boxes for short answers. Automatic scoring of such tests writes scores directly to a database, enabling on-line record keeping. Most tests occurring in web-based learning are summative. They are not designed to provide immediate, contextualised feedback for helping the educator and learner during the learning process. Automatic marking of answers to open-ended questions is relatively scarce in web-based learning. Some of the most prominent approaches to on-line essay scoring and automated grading of computer programming assignments are discussed in Chapter 4.

Educators have used a form of assessment known as portfolio assessment for many years. This involves learners collecting all of their best work to present for grading at the end of a term. Portfolios offer a simple and fairly effective way of assessing learners’ work without the typical multiple-choice end-of-term tests. Some web-based learning environments serve as channels for electronic portfolio assessment. Goals are posted on the web site, and resources for learning are made readily available. Learners make use of video, audio, text and data to reflect, present, and submit their work. Their progress can be captured on-line in various stages of the learning process (Serim & Koch, 1996). The assessment of such is, however, not automated but is conducted by the human educator.
On-line assessment has the advantage of saving valuable time for educators in terms of the management of learning. Once the courseware, tests or exercises have been designed, automatic record keeping significantly reduces time traditionally spent on marking and processing results. The on-line interface also enables peer-assessment, because learners have automatic access to other learners' projects and assignments and they can also co-operate by doing tasks collectively.

The issue of on-line assessment is addressed in more depth in Sections 3.3.2 and 3.4.1.

3.2.1.5 On-line feedback

Feedback generally refers to educators' response to learners' efforts (discussed in the previous section on formative assessment). However, reverse feedback, in the form of learners' reactions, also has a major role to play. Many web-based learning sites provide a forum where learners can present input to educators about subject-specific problems, or about the assistance they receive from educators. Services include dedicated, subject-specific mailing lists or bulletin boards to which only the particular educator, tutor/s and learners have access. On-line questionnaires are often included at the end of courses. These are completed interactively and submitted automatically to the educator or academic development unit. This feedback from the learners can be used to ensure quality of courseware and general service rendered. Databases are compiled for different subjects offered by an institution, and are used later for developmental purposes, i.e. to change certain aspects of a subject, based on the learner-feedback, or the data can be used for compiling some of the indicators in performance appraisal.

3.2.1.6 On-line virtual communities

Web-based learning encourages community building and networking. It can serve as an environment for learners to engage in a range of discussion topics with their peers and authorities in their field of specialisation (Kahn, 1997). This section mentions some of the facilities that support this sense of 'presence' as well as various features that offer simulated experiences.
(1) **Virtual classrooms**

A virtual classroom creates an environment where on-line resources are used to facilitate collaborative learning among learners, between learners and instructors, and between a group of learners and a wider academic and non-academic community (Kahn, 1997).

(2) **Virtual libraries**

Various web sites exist for the purpose of providing learning resources. One such web site is the [Web Virtual Library](http://www.w3.org/), which is hosted on:

http://www.w3.org/

This web site represents an extensive collaborative effort to gather and present information on a wide range of subjects to be used by learners (Kahn, 1997).

(3) **Virtual outings**

Learners can 'visit' and explore existing places such as a zoo or a museum which would otherwise have been inaccessible. For example, a web page that provides virtual outings is:

http://www.smartweb.fr/louvre/

by means of which users can visit one of the most famous museums in the world, the Louvre in Paris, and view each of its 350 rooms on-line by clicking on the rooms. Users can also discover Paris, with its shopping streets, monuments, parks, and activities on-line at this web site which provides thousands of pictures, haute couture, Parisian activities, as well as Carrousel's Stores, a luxurious mall next to the Louvre Museum. Sixty shops and twelve restaurants are presented with more than three hundred pictures and restaurant menus. It claims to be the world's largest guide about Paris (Musee du Louvre, 2000).

(4) **Virtual laboratories**

A virtual laboratory in the field of Natural Sciences, for example, can be a web site that learners can visit to perform experiments. Virtual laboratories offer learners the opportunity to experiment through simulations. Such experiments are integrated into assignments in a move away from text-based, printed assignments. A laboratory can be constructed
virtually on web sites, or by engaging in real-life projects, such as conducting research by using the WWW, or by doing work-related projects for companies (Khan, 1997).

An example of a virtual laboratory where a virtual frog can be interactively dissected can be found at:

http://george.lbl.gov/ITG.hm.pg/docs/dissect/info.html

This interactive web site enables a user to dissect a digitised frog named "Fluffy" and can generate views of the frog from many different directions, under various stages of dissection. Dissection is done at the major organ level, and an organ's name and function can be displayed by clicking in the image. It can also generate movies which are MPEG compressed (see Section 3.2.2.7(4)), and a player such as mpeg_play is needed to view the movies. Visitors can also play a game called "The Virtual Frog Builder game". This site is available in a number of languages (Lawrence Berkeley National Laboratory, 2002).

### 3.2.2 Internet technology used in web-based learning

Information on the WWW is presented as a series of documents referred to as web pages. Each web page includes references or links to other documents containing related information, so that a giant web* is created of easily-accessible information located on computer systems around the world. Many varied Internet technologies and software tools are involved in providing web-based learning, for example:

- Communication tools that enable learners to interact with each other.
- Remote access tools needed to connect to other computers.
- Connections and service providers.
- Operating systems.
- Web servers.
- Navigation and search tools.
- Publishing tools used by educators to produce web-pages.
- Programming and scripting languages.
- Presentation and management tools.
These technologies are discussed in this section. Since the terms server and client-server occur frequently, it is necessary to clarify these concepts. The term client-server refers to the traditional model of interaction between a host computer (or server) that provides files or services to client computers. Although the terms are sometimes used to refer to hardware, in fact they actually refer to the software running on these computers. On the Internet, the relationship between one computer and another follows the client-server model (Mercer, 2002).

3.2.2.1 Communication tools in web-based learning

Communication tools contribute to the creation of an interactive virtual learning community on the WWW. Web-based learners should be able to interact with each other, with instructors, as well as with on-line resources. Instructors and experts may act as facilitators who provide support, feedback, and guidance via both synchronous (time-dependent) and asynchronous (time-independent) communications:

- **Asynchronous communications** such as e-mail, news groups, list groups and bulletin boards, allow for time-independent interaction at times convenient for each learner.
- **Synchronous communications** such as conferencing tools and chat facilities enable live interaction.

(1) Asynchronous communication

*Electronic mail, news groups, list-servs and bulletin boards*

*Electronic mail* (e-mail) can be very useful in keeping educators and learners in contact with one another, and it enables them to communicate asynchronously at times that are convenient for each. It can be used for one-to-one exchanges, for presenting information to learners, conducting class discussions and for the assessment of learner performance, but groups or lists can move beyond that by allowing any member to address the group as a whole.

Unlike e-mail, each news group message is sent to a worldwide audience. Readers can respond to messages by sending e-mail directly to the person who posted the message, or by posting a message to the news group, allowing the entire group to see the response.
List servs (mailing lists) are like an e-mail club, in that everyone on the mailing list can send messages to the list. Each person on the list receives a copy of every message, so that the mailing list can provide a discussion group on the Internet. Anyone with an e-mail account can participate as a member of a mailing list. GroupWise, Pegasus Mail, Eudora and Outlook are well known e-mail software packages.

A bulletin board system (BBS) is a computer or an application dedicated to the sharing or exchange of messages on a network. BBSs date back to the early days of telecommunications, when users accessed announcement and conference facilities through a local telephone number and modem. Some BBSs allowed the user only to read the information posted, while on other BBSs, a user could read and input information (Serim & Koch, 1996).

Currently, many BBSs can be found on the WWW. Most of them are devoted to a particular subject, such as religion, a profession, hobbies and sports, multi-player games, medical practices, information for special interest groups such as the disabled, etc. (Whatis?com, 2002).

Using the asynchronous environment of the WWW for instruction allows learners as well as educators more time to assimilate and reflect on the information (Romiszowski, 1997).

(2) Synchronous communication

Computer conferencing

Web-based conferencing (WBC) is like an e-mail list or a news group in that it is easy to add information to a discussion and to view information that others have added. Computer conferencing incorporates features such as a shared whiteboard, moderated and open chat, group web browsing, slide markup and floor control. According to Malikowski (1997) its advantage over e-mail lists is that discussions tend to be more focussed because of the structure implemented in WBC.

CUseeMe is a low-cost product for web-based conferencing. Users need to install the CUseeMe software and, assuming they have an Internet connection, they can then have person-to-person or group discussions. A virtual whiteboard enables users to exchange
messages and to collaborate. Users with a small attached video camera can transmit their picture to other users in a conference (Whatis?com, 2002).

**Chat facilities**

Chatting on the Internet entails one Internet user conversing electronically with one or more other Internet user(s). Usually, this involves the exchange of typed-in messages requiring one site as the repository for the messages (or chat site) and a group of users who take part from anywhere on the Internet. Transcripts of a chat session can be archived for later reference. Chats are conducted via on-line services (especially America Online), by bulletin board services, and by web sites. Several web sites, such as Talk City, exist solely for the purpose of conducting chats (Whatis?com, 2002).

In a web-based learning environment, an educator and a number of learners can all access the chat forum (sometimes termed a chat room) via the course web site at a predetermined time. They can then present and discuss topics by typing text on the page. Learners can also pose questions in this forum. The text of the entire chat session is automatically saved for future access by the learners, and can then become a resource for learners that can be accessed via hyperlinks (Welsh, 1997).

**I seek you**

ICQ (pronounced "I seek you") is a program that will not only let a user know when friends and contacts are also on-line on the Internet, but will also page them, and make it possible to chat with them. In order to get maximum benefit from ICQ, both parties must have downloaded the ICQ program and have received a user identification number (UIN). The download and registration procedures are simple, and enable a user to send messages and files (single, multiple or whole directories) directly to other users' desktops (Whatis?com, 2002).

**Internet Relay Chat**

*Talk City* and many other chat sites use a protocol¹ called *Internet Relay Chat* (IRC). IRC is a system for chatting that involves a set of rules and conventions as well as client-server

¹ A *protocol* is a specification for communicating information (Mercer, 2002).
software. On the WWW, certain sites such as Talk City, or IRC networks such as the Undernet, provide a server and also provide help for downloading an IRC client to a personal computer on-line (Whatis?com, 2002).

3.2.2.2 Remote access tools used in web-based learning

(1) TCP/IP
TCP/IP stands for Transmission Control Protocol / Internet Protocol and is the communication protocol used to allow computers to exchange digital information over the Internet (Whatis?com, 2002).

TCP/IP is a multi-layered protocol. Transmission Control Protocol (TCP) is responsible for guaranteeing that the 'packets' transmitted between two hosts on the Internet reach their destination, and it manages the reassembling of packets into the original message. Because of network peculiarities, all of the Internet packets might arrive at the target computer totally out of sequence. TCP puts the packets in their correct order and determines whether or not they have all arrived. When a packet gets 'lost' somewhere along the path between two hosts, TCP makes sure the lost packet is retransmitted (Eckel & Hare, 1995; Lepage & Iarrera, 1998).

Internet Protocol (IP), divides an Internet message (data stream) into packages (usually around 200 bytes each), 'wraps each packet up' to make it easy to transmit across the Internet, and then labels its contents and destination. An IP address uniquely identifies a system on an Internet network. IP handles the address part of each packet so that it reaches the right destination. Each gateway computer on the network checks this address and forwards the message to its destination (Eckel & Hare, 1995; Lepage & Iarrera, 1998).

(2) FTP and HTTP
FTP is the acronym for File Transfer Protocol, which is a method for transferring files from one computer to another. FTP can be used to connect to local systems or to systems anywhere on the Internet. When a user runs the FTP program, it connects to ftpd (the FTP
server daemon\textsuperscript{2} on the remote system) and the user will be prompted for a username and password before he/she is permitted to gain access to files on the server. FTP servers that are available for anyone in the world to use are called anonymous servers. Typically a user will log in using ‘anonymous’ or ‘guest’ as a username and his/her e-mail address as password to gain access to files on the server (Eckel & Hare, 1995; Husain, \textit{et al}, 1996).

FTP servers can be set up for access by learners to download resources from a library of files on another computer. FTP functions can also be built into web pages to allow easy delivery in the opposite direction, for example files that currently reside on the learner’s disk can be copied to the server. This function provides the learner with the convenience of working off-line, and sending the completed work at a later time.

HTTP is the acronym for \textit{Hypertext Transfer Protocol}. When people refer to a web server, they are actually referring to an HTTP server program that is running on a computer that is connected to the Internet. The WWW is the larger phenomenon created by all the HTTP servers running throughout the world (Eckel & Hare, 1995).

\textbf{(3) Telnet}

Telnet is an Internet service (with an underlying TCP/IP protocol) that enables users to log on to remote computers over telephone lines. The WWW or HTTP protocol and the FTP protocol can be used in conjunction with Telnet if a user wishes to request specific files from remote computers (Eckel & Hare, 1995).

\textbf{(4) URL}

Every document on the WWW has an address called a URL or \textit{Uniform Resource Locator}. A typical URL looks like this:

\textit{http://osprey.unisa.ac.za/index.html}

\textsuperscript{2} A daemon is a program that runs continuously and exists for the purpose of handling periodic service requests that a computer system expects to receive. The daemon program forwards the requests to other programs (or processes) as appropriate. Each server of pages on the WWW has an \textit{httpd} or Hypertext Transfer Protocol daemon that continually waits for requests to come in from web clients and their users (Whatis?com, 2002).
3.2.2.3 Connections and service providers in web-based learning

(1) Modems
A modem allows a computer to communicate with other computers through regular phone lines. A modem modulates outgoing digital signals from a computer or other digital device, converting them to analog signals for a conventional copper twisted-pair telephone line. Conversely it demodulates the incoming analog signal and converts it to a digital signal for the digital device.

(2) Dial-up
There are two kinds of dial-up accounts: one that offers a shell service, and another that offers SLIP (Serial Line Internet Protocol) or PPP (Point-to-Point Protocol). Personal computer users usually connect to the Internet through the SLIP or the PPP protocol (Eckel & Hare, 1995).

The shell account provides a Unix command line interface, and it is up to the user to enter the commands to accomplish tasks. SLIP or PPP connections make use of a graphical interface (Eckel & Hare, 1995).

SLIP is a protocol for communication between two computers using a serial interface, typically a personal computer connected by phone line to a server. PPP is a full-duplex protocol that can be used on various physical media, including twisted pair or fibre optic
lines or satellite transmission. It uses a variation of High Speed Data Link Control (HDLC)
for packet encapsulations. PPP is usually preferred over the earlier de facto standard SLIP
because it can handle synchronous as well as asynchronous communication. PPP can
share a line with other users and it has error detection that SLIP lacks. Whereas SLIP only
encapsulates TCP/IP traffic, PPP can handle many protocols, including IP, IPX (Novell),

(3) Gateways and proxies
A gateway or proxy is used to protect the users of a network from the Internet, and vice
versa by acting as a traffic monitor between the Internet and the internal network. The
gateway converts between different data formats. Everything must pass through the
gateway before it is usable by the network on the other side (Lepage & Iarrera, 1998).

A proxy server, which can be a program running on a Unix server, or a separate computer,
is a special type of gateway. A proxy server acts as a monitor of network traffic like a
gateway, but it doesn't convert between data formats as a gateway does. Instead, a proxy
server receives requests from the users on the network, then forwards the requests to their
destination as if they originated from the proxy server instead of from within the network.
Thus, the servers on the Internet interact with the proxy server, but never with the systems
located on the internal network. This protects the internal systems from being tampered
with by users located on the Internet (Lepage & Iarrera, 1998).

(4) Internet Service Providers
Because the Internet is a collection of networks, it has no central point to connect to, and
no true authority that controls it, though many are involved in administering it. To connect
to the Internet, users connect to a provider that already has an Internet connection. An
Internet service Provider (ISP) is a company that has made a business of selling
connections to the Internet via their own connection. The ISP pays for a fast connection
and provides the service and support for those who buy the connections from them
(Lepage & Iarrera, 1998). Well known service providers in South Africa are mweb, iafrika,
Telkom and Uninet.
3.2.2.4 Operating systems - the Linux operating system

An operating system acts as a communication service between the hardware (physical equipment of a computer) and the software (applications which use the hardware) of a computer system. Harris (2002:1) describes an operating systems as "the software that takes the raw capabilities of the hardware and builds a more practical platform for the execution of programs". Examples of well-known operating systems are Unix, Linux, Apple Macintosh, DOS, MS Windows\(^3\), OS2, NetWare, VMS, OS2, OS/400, AIX, and z/OS. Only the Linux operating system is discussed in this section.

(1) What is Linux?

Linux is a Unix-like operating system that was designed to provide personal computer users a free or very low-cost operating system comparable to traditional and usually more expensive Unix systems. It was initially created as a hobby by Linus Torvalds (a student at the University of Helsinki in Finland). He was interested in a system called Minix (a small Unix system developed by Andy Tanenbaum), and decided to develop a system that could exceed the Minix standards. He began his work in 1991 when he released the first version of Linux. It is currently used on machines ranging from handheld computers, to embedded instrumentation, to personal workstations, as well as to the largest mainframes available from IBM (Kroll, 1999; Linux International, 2001; Brown, 2002; Whatis?com, 2002).

Linux is developed under the GNU\(^4\) General Public License, and its source code is freely available. Nearly all development software for Linux is free, and covered under the GNU Public License, which guarantees that it will always remain free. Linux is used for a wide variety of purposes including networking (see point (3)) and software development (see point (4)). A wide range of application software (see point (5)) has been developed for or ported to the Linux platform. It is considered to be an excellent, low-cost alternative to

---

\(^3\) Throughout this document, the term MS Windows refers collectively to Windows 98, 2000, ME and XP as well as Windows NT.

\(^4\) The GNU project provides Unix-like software freely to anyone, with the provision that it remains free if distributed to others. Many GNU utilities are improvements of existing Unix counterparts and include many new implementations of shells, the C compiler, and a code debugger. Most Linux distributions make extensive use of GNU utilities. Other types of GNU software include games, text editors, calculators, and communication utilities (Husain et al, 1996).
other more expensive operating systems. It has become extremely popular worldwide, and a large number of software programmers have adapted Linux’s source code to meet their individual needs. Linux machines are also known to be extremely fast, because the operating system is very efficient at managing resources such as memory, CPU power, and disk space (Linux International, 2001; Linux Home Page, 2002).

(2) How does Linux work?
The Linux kernel (the core of the operating system code that runs the whole computer) contains all of the features that could normally be expected in any operating system. Some of the features included are (Linux International, 2001):

- **Multitasking**, a technique for sharing a single processor between several independent jobs.
- **Virtual memory**, which allows repetitive, extended use of the computer’s RAM for performance enhancement.
- **Fast TCP/IP drivers** for speedy communication.
- **Shared libraries and programming utilities**, which enable applications to share common code.
- **Multi-user capability**, which enables many people to use the computer simultaneously.
- **Protected mode**, allowing programs to access physical memory, while still protecting the stability of the system.

(3) Networking
Kroll (1999) states that Linux is ideal for networking because it is based on Unix, which was originally designed as a multi-user system, i.e. to serve many users simultaneously. Setting up a network on a Linux machine is relatively simple, because Linux handles most of the work. Users only have to provide the correct addresses. Linux is an excellent choice for an Internet server, often performing better than *Windows NT*, *Novell* and most Unix systems on the same hardware. Linux is used by leading businesses for superior server

---

5 Protected mode is a mode of program operation (used in computers with an Intel-based microprocessor) in which the program is restricted to addressing a specific contiguous area of 640 kilobytes, in order to ensure that essential data is not unintentionally overwritten.
and network performance, and offers a variety of commercial productivity packages and office suites which can import and export files from other platforms. Linux is compatible with Local Area Networks (LANs), and provides full support for Macintosh, DOS, MS Windows, Novell, and OS/2, using their own native communication protocols. Linux supports all of the most common Internet protocols, including Electronic Mail, Usenet News, Gopher, Telnet, Web, FTP, Talk, POP, NTP, IRC, NFS, DNS, NIS, SNMP, Kerberos, WAIVES and many more. Linux can operate as a client or as a server for all of the above, and has been widely used and tested (Linux International, 2001).

(4) Software development
Kroll (1999) further believes that Linux is an ideal development platform, since C, C++ compilers, and an assembler are included by default on all distributions. Many other programming languages are also available for use on Linux, including Ada, APL, BASIC, Dylan, Eiffel, Euler, Forth, Fortran, GOMscript, INTERCAL, LISP, Logo, Mercury, Modula, Oberon, Objective C, Pascal, Perl, Prolog, Python, PHP, Rexx, Sather, SIMULA, Tcl/Tk, Java and several more, which are all available free of charge. Programmers have a choice between using gcc (the GNU C Compiler) and egcs (the Experimental GNU Compiler System), the latter of which has become quite popular due to its features, flexibility, and functionality. The source code for most Linux programs is freely available and is often included by default. This not only means that bugs are discovered and corrected almost immediately, but also that development of software proceeds at a much faster pace than at some of the successful commercial software houses.

(5) Software applications
Many applications ranging from web servers to database engines and programming tools, to word processors to games, are available for use on Linux. Software availability has exploded in recent years, and commercial developers have been producing excellent software for the Linux platform. Netscape Navigator and Communicator are freely available (with some licensing restrictions) as well as many spreadsheets, databases, and word processors (such as Word Star and Word Perfect 9). These are often included on Linux distributions.
(6) Graphical Interface

The earliest Unix systems were text-based systems, and did not have windowing systems. The X Window system (also called X Windows) was developed at the Massachusetts Institute of Technology (MIT), and because they distributed X Windows freely, it soon became the standard window system for the Unix/Linux operating system. Currently, almost all Unix/Linux systems use a graphical user interface based on X Windows. Many X-specific applications have been developed, including games, graphics and programming utilities, and documentation tools. Linux has at least a dozen different, highly configurable graphical interfaces (known as window managers) which run on top of XFree86, which is a free implementation of the X Window system. KDE (the K Desktop Environment) and GNOME (the GNU Network Object Model Environment) are currently the most popular window managers. These offer the point-and-click, drag-and-drop functionality associated with other user-friendly environments (for example, MS Windows and Apple Macintosh), and are extremely flexible. A Linux machine running KDE, can be customised to look like an Apple Macintosh, MS Windows, BeOS, or NextStep machine. Complex tasks such as system administration, package installation, upgrading, and network configuration can be done easily through graphical programs. Programs that work with one window manager usually work with all the others (Husain, et al, 1996; Kroll, 1999).

3.2.2.5 Web servers - the Apache web server

A web server is a program that serves web pages to users on the WWW, using the client-server model and the Hypertext Transfer Protocol (HTTP) (see Section 3.2.2.2). Every computer that contains a web site and is connected to the WWW must have a web server program. Two leading web servers are Apache and Microsoft’s Internet Information Server (IIS). Other web servers include Novell’s Web Server for users of its NetWare operating system, and IBM’s family of Lotus Domino servers, primarily for users of IBM’s OS/390 and AS/400 (Whatis?com, 2002).

Web servers are oftentimes distributed as part of a larger package of network-related programs for serving e-mail, downloading requests for FTP files, and for building and publishing web pages. Considerations in choosing a web server include (Whatis?com, 2002):
How well does it work with the operating system and other servers?
Can it handle server-side programming?
Security characteristics.

Apache web server
A large part of the WWW runs on Linux machines using the Apache Web Server. Apache is a freely available web server that is distributed under an open source license. It runs on most Unix-based operating systems (such as Linux, Solaris, Digital Unix, and AIX), on other Unix/POSIX-derived systems (such as Rhapsody, BeOS, and BS2000/OSD), on AmigaOS, and on Windows 2000 and Windows XP. According to the Netcraft Web server survey in February 2001, 60% of all web sites on the Internet are using Apache (62% including Apache derivatives). This means that it is more widely used than all the other web servers together (Whatis?com, 2002).

Apache can be downloaded from the official Apache web server site at:

http://www.apache.org/

3.2.2.6 Navigation and search tools in web-based learning

(1) Browsers
A browser is the software that resides on a computer and enables users to access and display web documents. It is required to access services on the WWW. It is a web client program that uses the Hypertext Transfer Protocol (HTTP) to request web pages from web servers throughout the Internet on behalf of the browser user (Descy, 1997; Whatis?com, 2002).

One of the simplest browser is Lynx which is a text-only browser that was developed at the University of Kansas primarily for students who used Unix workstations. It has also been rewritten to run on VMS operating systems for users of VT100 terminals. The vast majority of users have a graphic browser that can display graphics as well as text. Mosaic was the first graphic browser in 1993, and currently there are many web-browsers, of which Netscape, Internet Explorer and HotJava are the most commonly known and used (Whatis?com, 2002).
(2) Search engines

A search engine is a web-based service that helps users find documents on the WWW by means of keyword searches. A search engine is a coordinated set of programs that consists of (Whatis?com, 2002):

- A spider (also called a crawler or a bot) that searches every web site that wants to be searchable and reads it, using hypertext links on each page to discover and read a site's other pages.
- A program that creates an extensive index (sometimes called a catalog) from the pages that have been read.
- A program that receives the search request, compares it to the entries in the index, and returns the results found.

Google, Yahoo, AltaVista, Lycos, Infoseek, Excite, HotBot, and Ask Jeeves are well-known search engines. Yahoo is the most widely-used search engine on the WWW (Whatis?com, 2002).

3.2.2.7 Publishing tools in web-based learning

(1) Page description languages

A page description language (PDL) is a defined format for specifying the appearance of a document when displayed or printed. Adobe's Acrobat and Postscript, Novell's Envoy, and Hewlett Packard's Printer Control Language (PCL), are examples of PDLs. Page description languages can be contrasted with content description languages, such as the eXtensible Markup Language (XML), which describe the content rather than the arrangement of information (Whatis?com, 2002).

(2) Markup languages

**HTML**

HTML (Hypertext Markup Language) is the basic tool for displaying images and text on the WWW. HTML is not a page description language (like Acrobat or Envoy) or a programming language (like C++ and Java), but a hypertext markup language. It consists of a set of markup symbols or codes inserted in a file intended for display on a web browser. The markup tells the browser how to display a web page's words and images i.e.
it indicates how the page should appear to the user, and also implements links to other web sites or web pages. The individual markup codes are referred to as elements or tags. While the simplicity and flexibility of HTML contributes to the success of the WWW, it does have limitations. Two major efforts, Dynamic HTML and XML address those limitations (Smith & Bebak, 1998).

**Dynamic HTML**

Dynamic HTML is used for displaying information only when it is needed. It is an extension to HTML that allows multiple layers of information to be sent to the user during a server connection. The user only sees part of the information at first. Additional information can then be unveiled as time passes, or as the user undertakes different actions, i.e. certain information is hidden until the user requests it. With dynamic HTML, a web page can change once it has been loaded into the browser without having to re-connect to the server. For example, a piece of text can change from one size or colour to another, or a graphic can move from one location to another, in response to some kind of user action, such as clicking a button (Needleman, 1997).

**XML**

HTML tags describe how something should render, but they do not contain any information about what the data is. The tags only describe how the data should look. XML (eXtended Markup Language) is a superset of HTML, but a subset of the overall SGML (Standard Generalized Markup Language) standard on which HTML is based. SGML is based on the idea that documents have structural and other semantic elements that can be described without reference to how such elements should be displayed. The actual display of such a document may vary, depending on the output medium and style preferences. XML allows complex data structures to be built into a web page, giving authors the ability to build data-driven applications and deliver them across the WWW (Tidwell, 1999).

The meaning of the tags is built into the XML code, and can be understood by the computer. For example, `<zipcode>34829</zipcode>` is the description of a zip code. When using XML, users can receive and send XML-tagged data from and to other systems without having to know anything about that system. By following the document rules
defined in the document type definition (DTD)\(^6\), they do not have to write any code to exchange data between different systems (Tidwell, 1999).

**VRML**

VRML (Virtual Reality Modelling Language) was developed by Silicon Graphics as a language for describing three-dimensional (3D) image sequences and possible user interactions with them. VRML creates 3D worlds on web pages that can be virtually maneuvered through as one would walk around in real space within a room or building. By using VRML, a sequence of visual images can be built into web settings with which a user can interact by viewing, moving, rotating, and otherwise interacting with an apparently 3D scene (Descy, 1997; Whatis?com, 2002).

VRML comprises a set of tools and conventions for representing three-dimensional space on the WWW. Instead of seeing a static picture, the user is able to hold the mouse over a graphic and view it from multiple perspectives, zooming in for detail and turning it in any direction. This makes a more intuitive representation of content possible, which enhances web-based learning. For example, VRML can be used to create a simulated learning environment with a metaphorical reference room, laboratory, and help desk, etc.

A VRML viewer or browser is required to view a VRML file. Viewers that can be downloaded for the MS Windows platforms are *Blaxxun's CC Pro, Platinum's Cosmo Player, WebFX, WorldView, and Fountain*. *Whirlwind* and *Voyager* are two VRML viewers for the Macintosh (Whatis?com, 2002).

(3) **Web page editing and authoring tools**

A web page authoring tool can be used to create and edit the HTML-tagged text files that, when viewed from a web browser, appear as web pages. Setting up a web page can be done by using:

\(^6\) A DTD is a specification that accompanies a document and identifies what the markup codes are that separate paragraphs, identify topic headings, and so forth and how each is to be processed. By mailing a DTD with a document, any location that has a DTD reader (or SGML compiler) will be able to process the document and display or print it as intended. This means that a single standard SGML compiler can serve many different kinds of documents that use a range of different markup codes and related meanings. The compiler looks at the DTD and then prints or displays the document accordingly (Tidwell, 1999).
an ASCII editor such as Notepad or WordPad (MS Windows), and vi (Unix and Linux),

- the <Save as HTML> option of a word processor such as WordPerfect or Microsoft Word, or
- a web page authoring tool such as Microsoft FrontPage 98, Adobe PageMill, HomeSite, HotDog Pro, HotMetal, Web Wizard, Netscape Composer, FrontPage Express, and DreamWeaver.

Some authoring programs are shareware (copyrighted computer programs available on trial basis, with payment expected after a specific period of use) or freeware (copyrighted programs available without any charge) such as Microsoft’s FrontPage Express and Netscape Composer. Large software companies such as Microsoft and Adobe are now major players in web page publishing with their retail software products such as Microsoft FrontPage 98 and Adobe PageMill (Smith & Bebak, 1998).

Multiple, independently controllable sections or frames on a web page can be achieved by building each section as a separate HTML file, and having one ‘master’ HTML file that calls the sections. When a user requests a web page that uses frames, the address requested is actually that of the ‘master’ file that defines the frames. Multiple HTML files are returned as the result of the request - one for each visual section. Links in one frame can request another file that will appear in another frame, or the same frame (Whatis?com, 2002).

Netscape originally created frames as an extension to HTML, and it is now part of HTML 4.0. The specification of frames are defined with HTML FRAMESET and FRAME tags. Sites that use frames need to create an alternative scheme of pages for requests from browsers that do not support them, and possibly for users that prefer a non-frames version (Whatis?com, 2002).

(4) Text, images, audio and video

The WWW is a multimedia environment comprising text, images, audio and video. Each of these different file types requires different application software to display, hear or view.
**Multimedia and hypermedia**

The WWW is the multimedia part of the Internet, and is currently the most widely used part of the Internet. Multimedia is a combination of video, sound, text, animation and graphic images in a computer-based environment. Hypermedia are Internet-based applications for consulting multimedia information resources. Its outstanding feature is hypertext, a method of instant cross-referencing to related web pages by using links (words or phrases that appear in a different colour from the rest of the text, and are usually underlined). By clicking on a link, a user is transported to another site or another page that is relevant to the word or phrase in the link. The WWW provides access to millions of pages of information (Whatis?com, 2002).

Multimedia and hypermedia are used very successfully to create rich interactive learning materials. Digital video and sound are increasingly being incorporated into web pages. Previous limitations such as large file sizes and high bandwidth (amount of data transferred) required by digital video are being overcome by means of file compression and higher speed modems.

**Video**

There are presently three popular digital video formats: AVI from Microsoft, QuickTime from Apple, and MPEG from Motion Picture Experts Group. QuickTime and MPEG are cross platform data formats and AVI is supported by Microsoft.

**Graphics**

One of the main factors contributing to the success of the WWW is the blending of graphics and text in web pages. Graphical User Interface (GUI) software makes it possible to interact with a computer by using a pointing device like a mouse to manipulate icons, windows, images and menus on the screen instead of using a purely character-mode interface. MS Windows, X Windows and Macintosh user interfaces are examples of GUIs (Smith & Bebak, 1998). The term WIMP, which stands for windows, icons, and mouse pointing, is often used for referring to a GUI interface.

MS Windows includes a free graphics program, Paint, that can be used to create graphics. Popular commercial software are Autodesk Animator, Adobe PhotoShop, and Adobe
Web-based learning

Illustrator. Each graphics program saves files in its own graphics format with extensions such as .jpg (known as jpeg), .gif (known as GIFs), .bmp (known as bitmaps), .png, .tiff and .pcx.

Only certain types of images are supported by browsers, and these types are:

- Compuserve Graphics Interchange Format (GIF).
- Joint Photographic Experts Group (JPEG or JPG).
- Portable Network Graphics (PNG).

Compress for delivering graphics

Image file types suitable for the WWW are commonly compressed, meaning the file type takes up less space than the full-resolution original. The compression for GIF and JPEG is called lossy compression because some data is lost when the image is converted to the file type. Unlike most file types (such as executable files), where the loss of bits can render the file unusable, these image file types often appear much the same to the human eye, and therefore the data loss is acceptable. Lossy compression is desirable because smaller file sizes mean faster downloads, and for many users communication on the WWW is still constrained by file size (Mercer, 2002).

PNG is a relatively new file format used for saving images to be used on the WWW. The PNG file format uses lossless compression, and therefore image quality does not degrade during the compression process. PNG supports interlacing and transparency, but not animation (Mercer, 2002). The PNG format is unfortunately not supported by older browsers.

Compress for delivering sound

Digital sound has been presented via web pages since the development of the first Mosaic browser. Many current software applications enable Internet users to play sound files on their computers. Typically these applications require the whole sound file to be downloaded and decompressed before it can be played, which can be time consuming when files are large and modems slow. A three-minute file can take as long as 30 minutes to download (Descy, 1997).
Web-based learning

On some web sites, after clicking on a picture to access an audio greeting or introduction, the user has to wait for the sound file to download before hearing anything. One of the newer forms of WWW technology, which addresses this problem, is called Audio Streaming. Audio Streaming applications start to decompress sound files and play them immediately, even while they are being received. This expedites the process dramatically and allows for such things as live concerts and radio broadcasts (Smith & Bebak, 1998; Whatis?com, 2002).

RealAudio

RealAudio is a plug-in for real-time playback of audio files. It is a continuous, or streaming, sound technology from Progressive Networks' RealAudio. A RealAudio player or client program may come included with a web browser, or can be downloaded from various web sites. A RealAudio server is required to deliver RealAudio sound from a web site, and files can be recognised by their file name extensions of .ra or .ram. A number of web sites now offer programs with RealAudio, one of the most popular being Tom King's AudioNet (Whatis?com, 2002).

RealAudio holds utility for learning on the WWW, in that the spoken communication used a traditional class can be digitised and made available as needed. RealAudio could become a standard feature of distance learning on the WWW.

MP3

MP3 (MPEG-1 Audio Layer-3) is a standard technology and format for compressing a sound sequence into a very small file (about one-twelfth the size of the original file) while preserving the level of sound quality. MP3 files (identified with the file name suffix of .mp3) are available for downloading from a number of web sites. Many operating system packages have a player built into their operating system. Otherwise, the player can be downloaded from one of several MP3 sites (Whatis?com, 2002).

3.2.2.8 Programming and scripting languages

In order to process information, either within the client (the browser) or at the server, some programming language has to be employed (Herrmann, 1996). To maintain broad platform compatibility, the language used must run on most browsers. A variety of programming
languages, scripting languages, and application-development technologies are used on the WWW:

(1) **Common gateway interface**

Common gateway interface (CGI) programming involves designing and writing programs that receive their starting commands from a web page. CGI programming does not involve merely a single language or application, it has to work closely together with HTML and HTTP (Herrmann, 1996).

- HTML code (see Section 3.2.2.7(2)) defines the way the user sees the program interface, and is responsible for collecting user input. It is the window through which the user and the program interact. Forms are text boxes and pull-down menus that are part of HTML and enable users to enter the requested information.

- HTTP (Section 3.2.2.2) is the mechanism (or protocol) for transmitting data between the CGI program and the user; it translates and sends information between the web-client and the CGI program. The CGI program is responsible for understanding both the HTTP directions and the user requests. The CGI code takes the requests from the user and sends back valid and useful responses to the web client.

Because the interface is consistent, a programmer can write a CGI application in a number of different languages and on various platforms. The most popular languages for CGI applications are: C, C++, Java, and *Practical Extraction and Reporting Language* (PERL).

An alternative to a CGI application is *Microsoft’s Active Server Page* (ASP), in which a script embedded in a web page is executed at the server before the page is sent (Whatis?com, 2002).

CGI scripts can be used to compare information that learners input on-line into text fields, buttons, or check boxes with preset answers in a database or text file. In a web-based learning environment, feedback can provide individual learners with a deeper explanation of the consequences of their choices, and can also provide active links which guide them to additional information. CGI scripts can also be written to capture variables from
learners, store them, and access this information at a later date. On-line tests can be constructed with CGI scripts in a similar way, and grades can be assigned, and/or other feedback can be given. This process can be automated for objective tests, or saved in files for educator critique if more open-ended questions are used (Ritchie & Hoffman, 1997).

(2) Programming languages

Java

Java is not a markup language like HTML, nor is it a page description language like Adobe Acrobat or Envoy from Novell. Rather it is an object-oriented general purpose programming language used for creating platform-independent programs that can be used in the distributed environment of the Internet. Java was originally designed to facilitate communication between computers and devices, and was developed by Sun Microsystems for hand-held electronic devices. It was engineered to be small and platform-independent, making it well suited to the WWW (Descy, 1997; Dragan & Seltzer, 1997).

Most programming languages (such as C++) compile source code into machine code, or native code, optimised for a specific platform such as Intel x86 or PowerPC. In contrast, a Java compiler translates source code into Java byte code (.class), a set of instructions that is not optimised for any particular platform (Dragan & Seltzer, 1997). A program written in Java can run on almost any computer on most operating systems, including MS Windows, OS/2, Macintosh, AIX, MVS, OS/400, Sun Solaris, OSF, HP-UX, DEC Alpha, Linux, NextStep, Amega and BeOS. Java programs can be run on any system that has a Java virtual machine (VM). Web browsers such as Microsoft Internet Explorer, Netscape Navigator, and Sun's own HotJava have Java VMs built-in. Like an actual CPU, the Java VM interprets the Java byte code at runtime for the specific hardware and software platform. The advantage of this is that Java executables can be run on any system that has a Java byte-code interpreter. The disadvantage is that Java programs run considerably slower than those optimised for particular platforms. Some Java VMs also include just-in-time compilers (JITs), which significantly increase performance (Dragan & Seltzer, 1997).
Java provides connectivity to databases via the JDCB (Java Database Connectivity). The JDBC allows a connection to be made to a database in the same manner as connecting to a resource or image, and also provides for SQL (Standard Query Language) statements to be sent, and the results of queries to be interpreted on return (Bishop, 1998).

Java applications are standalone programs that can be executed independent of a web browser. A Java applet, by contrast, is run from inside a web browser. It is a program that can be downloaded from the WWW by a web browser, and is executed inside a web page. Applets can create scrolling banners, animations, games, interactive programs, and other multimedia effects on web pages. A reference to the applet is embedded in the web page’s HTML code between special tags (<APPLET> code </APPLET>) (Lemay & Perkins, 1997; Lepage & Iarrera, 1998). Applets make it possible for a web page user to interact with the page.

According to Bishop (1998) the advantages of having applets in a web page are:

- The work is done on the computer where the results are needed, rather than sent there. Consequently there is less traffic on the network.
- Because the user’s computer can be dedicated to the applet, it can run much faster there than on the server machine where the web page resides.
- The full facilities of the Java programming language are available, unlike some specially designed web-languages that have restricted calculation and structuring powers.

Java can also be used to create complete applications that may run on a single computer or be distributed among servers and clients in a network. To write a Java application, a development environment such as Sun’s Java Development Kit (JDK) or a graphical Integrated Development Environment such as Symantec’s Café or Microsoft Visual J++ is needed (Dragan & Seltzer, 1997).

Like Java applets, JavaBeans components (or ‘Beans’) can be used to give web pages (or other applications) interactive capabilities for purposes of doing calculations or varying page content based on user or browser characteristics. Reuseable, platform-independent components can be created with the JavaBeans application programming interface. A component is a piece of code that plugs into larger programs and can be used over and
again. Buttons, check boxes, list boxes and scrollbars are examples of components. Most components are visible on the screen, but some, such as links to a database, are not (Burd, 1997; Lemay & Perkins, 1997).

From a user's viewpoint, a component can be an interactive button or a small calculating program that is initiated once the button is pressed. From a developer's viewpoint, the button component and the calculator component are created separately and can then be used together or in different combinations with other components in different applications or situations. Sun's Bean Development Kit (BDK) is a development environment used for creating and testing JavaBeans. It runs on any major operating system platform inside a number of application environments including browsers, word processors, and other applications (Whatis?com, 2002).

JavaScript should not be confused with Java. JavaScript (discussed next), which originated at Netscape, is interpreted at a higher level and is easier to learn than Java, but lacks some of its portability, and the speed of bytecode. Because Java applets will run on almost any operating system without requiring recompilation, and because Java has no operating system-unique extensions or variations, Java is generally regarded as the most strategic language in which to develop applications for the WWW. However, JavaScript can be useful for very small applications that run on the web client or web server (Whatis?com, 2002).

(3) Scripting languages

Programs written in a scripting language (such as JavaScript, VBScript, ASP, PHP, Python and Perl) are interpreted within the browser at run-time, and need not be compiled.

JavaScript

JavaScript programs, or scripts, are usually embedded directly in HTML files. The script executes when the user's browser opens the HTML file. JavaScript is an interpreted language - meaning that scripts execute without preliminary compilation, i.e. without conversion of the script text into a system-dependent machine code. The user's browser interprets the script, that is, analyses and immediately executes it. To maintain broad platform compatibility, the language must run on most browsers. The language of choice
Web-based learning

is currently JavaScript. JavaScript commands are inserted directly into the HTML, and are interpreted as the browser reads through the code, rather than being compiled as executable files (Mercer, 2002).

JavaScript can be used to perform functions in web-pages that cannot be done with ordinary HTML, and when this is done, these functions are referred to as client-side scripts. HTML is not considered a programming language because it does not process information. It is primarily used for formatting and displaying text, images, multimedia, hypertext links, and HTML forms (sometimes with tables and frames included to provide added display capability). In order to process information, either within the client (the browser) or at the server, some programming language must be employed. JavaScript scripts can also be used to run programs on a server, and these programs are referred to as server-sided scripts. Since the platform is known in advance on the server, portability is not a major issue, and many scripting languages (such as VBScript, Perl, PHP), programming languages (such as Visual Basic and Visual C++, and Java), and application-development technologies (such as Microsoft's ASP) are in use (Mercer, 2002).

**PHP: Hypertext Preprocessor**

PHP: Hypertext Preprocessor (PHP) is a free scripting language and interpreter similar to JavaScript and Microsoft's VBScript. It is used primarily on Linux web servers for web programming because most modern Linux distributions come bundled with the Apache server and PHP Apache module. PHP is a cross-platform alternative to Microsoft's Active Server Page (ASP) technology which runs only on MS Windows NT servers (Heng, 2000).

PHP originally stood for 'Personal Home Page tools' and was created by Rasmus Lerdorf in 1994 by using a number of Perl scripts to track visitors to his resume on the WWW. Gradually more people started to get interested in the scripts, and they were later released as a package called Personal Home Page tools. In view of the interest, he wrote a scripting engine in 1995, and incorporated another tool to parse input from HTML forms: Form Interpreter, thereby creating what was called PHP/FI or PHP2. These tools were increasingly used for more complex tasks, and the development process migrated from development by a single individual to a group of core developers. This was the beginning of PHP3. The scripting engine was improved and extended and a simple application
programming interface (API) was added that allows other programmers to add more functionality to the language by writing modules for it (Castagnetto et al, 1999; Medinets, 2000; Ullman, 2001).

The definition PHP: Hypertext Preprocessor means that PHP handles data before it becomes HTML. PHP script is processed by the web server, and then returns plain HTML to the browser. Before the page is sent to a user who has requested it, the web server calls PHP to interpret and perform the operations called for in the PHP script. An HTML page that includes a PHP script is typically given a file name suffix of .php, .php3, or .phtml (Serrano, 2000; Ullman, 2001).

PHP is a hybrid between C, C++, Perl, Microsoft's VBScript, and Java. It incorporates certain of the most prominent features of each into one complete language. Some of the burdens of C (e.g. string handling techniques) have been eliminated through the Perl-like approach of PHP, and it has also acquired some of the object-based nature of Java.

PHP is a combination of programming language and application server. It is tightly integrated into the web server, and therefore executes on the web server, whereas Java applets, ActiveX, or JavaScript are executed by the client's web browser. All these languages can be combined with PHP applications. PHP generates whatever HTML is needed to activate Java applets or ActiveX controls, or to generate JavaScript statements dynamically. Like JavaScript, PHP code can be called and processed, and the results of executing the statements included in special start (<?php) and end (?>) tags can then be returned back to the HTML as text.

The main reason for using a server-side scripting language is to be able to provide dynamic content to a web site's users. This means that the content on the web site changes according to a user's needs or inputs.
Why is PHP used for software development?

**Reusability of code:**
Several methods (such as creating separate source files and object-oriented classes) and techniques (encapsulation and information hiding) can be used to implement reusable code with PHP. Because PHP is written in C and C++, any script code (such as Perl scripts) or any code that can be compiled into an executable file, can be integrated directly into PHP. Java support is not included in PHP by default, but PHP can be installed with the “with java” configuration as a *cgi executable*, or as a *dynamically loaded Apache module*, to include Java.

**Database connection:**
PHP has the ability to connect with relational databases such as MySQL, Informix, Oracle, Sybase, Solid, DB2, MS SQL, PostgreSQL, Adabase, mSQL and dBase, and any database that supports the open database connection (ODBC) standard.

**Server-sided:**
Because PHP runs on the web server, PHP programs can be large and complex without slowing down clients.

**Cross-platform:**
PHP can be used on MS Windows, or on any Unix-like platform such as Linux, BSD, and Solaris.

**Affordability:**
PHP is free and can be downloaded from: [http://www.php.net](http://www.php.net)

What makes PHP different from CGI scripts written in Perl or C, or programs written in Java?

PHP script is embedded within a web page along with its HTML. Instead of writing an executable file (as in the case of Perl or C), or a Java program with many commands to
output results in an HTML format, an HTML script is written with some embedded code to be executed. As already stated, PHP code is enclosed in special start (<?php) and end tags (?>) that allow jumping into and out of PHP-mode.

3.2.2.9 Presentation and management tools

Creating a web-based course is not a simple task. In the past, creating multimedia presentations and instructional programs required the services of a professional programmer, or the use of platform-specific authoring systems such as Plato, TenCORE, Toolbox and HyperCard. However, the current generation of authoring software makes it possible for non-programmers to create presentations with sound and animations, and button-click links to different parts of the presentation. Similar projects can now be undertaken to create multimedia environments for web-based learning, with the major advantages being that courseware is no longer platform-dependent, and is accessible through the Internet (Hansen & Frick, 1997).

Plato (now Novanet) was one of the earliest and most successful authoring systems. TenCORE, Toolbox, HyperCard, Authorware and Director are examples of course authoring tools that emerged as the personal computer market grew, with courseware being distributed on computer disks and CD-ROMS. Many new authoring programs are on the horizon that will make development on and for the WWW a simpler process (Hansen & Frick, 1997).

The Department of Computer Science at the University of British Columbia (UBC) developed a web educational resource for their third-year course on the fundamentals of operating system design, CPSC 315. This resource was intended to be used either as a complement to the existing lecture-based course, or as a self-contained distance alternative to a lecture-based offering. It was, however, very expensive and time consuming to create. The interactivity in the resource was based on CGI scripts written in Perl. Approximately twelve months of full-time programming and web-development were involved for one experienced programmer. After completing this resource the development team was presented with the opportunity to do the same for a further set of science courses. Instead of working on those science courses directly, they decided to actually
create a web-based course authoring environment that could be used by other academics. This is how the course management system called WebCT originated (Goldberg, 1997).

**WebCT**

WebCT (*Web Course Tool*) is a tool that simplifies the creation, delivery and management of sophisticated web-based educational environments. It can be used to create entire online courses, or simply to publish materials that supplement existing courses. WebCT requires minimal technical expertise on the part of the developer of the educational material, as well as on the part of the learner. The content of the course is provided by the course developer or designer, while interactivity, structure, and educational tools are provided by WebCT (Anderson & Ashley, 1998; Goldberg, 1997).

WebCT is entirely web-based. A single web server running the WebCT software is used both for course creation and delivery. Learners can interact with course material, and designers can create new courses or modify existing courses by using a web browser to connect to the WebCT server (Anderson & Ashley, 1998).

Depending on the class of the user connecting to a course, a different view will be presented by WebCT. For example, a course designer will be presented with a view that allows for both viewing and modification of the course when signing on. A learner is presented with a view that permits only the viewing of a course when signing on, except in the case of learner presentation areas where the learner is allowed to interactively update particular areas (Goldberg, 1997).

### 3.3 Problems related to assessment and interactive learning in web-based instruction

With the foregoing comprehensive set of technical aspects as background, the remainder of this chapter moves on to discuss problematic issues and potential pitfalls peculiar to web-based learning. The problems are set out in Section 3.3 and are followed by proposed solutions in Section 3.4. There is frequent reference to some of the technical aspects of Section 3.2.
3.3.1 Introduction

Although the WWW is a highly useful learning support medium, it has been criticised in certain literature, where it is suggested that it is unsuitable for presenting courses or material which should be navigated primarily in sequence.

Kearsley (1998) believes that the promise of providing individualised instruction via computers has been achieved only in the most trivial ways. In his opinion many of the drill-and-practice and tutorial programs that have been developed are used with little impact. Such programs allow learners to work at their own pace and to control the sequence of material. However, this is not the ideal kind of personal learning experience that individual instruction should entail, in that it is not based on the background and interests of the learner at any particular moment.

Conradie (2001) identifies other problems related to web-based learning, such as information overload, communication issues, assessment, interactive learning, and copyright. Problems related to assessment and interactive learning are discussed in this section, while some solutions to these problems are proposed in Section 3.4.

3.3.2 Problems related to assessment

Assessment of web-based learning can be done either on-line or off-line, through objective tests, or through development of products and artifacts of learning. Most current web-assessment involves multiple-choice questions, quizzes, true-false statements, and dialogue boxes for short answers that write scores directly to a database. These tend to focus on atomistic knowledge - i.e., knowledge portrayed in parts, out of context, and unrelated to the body of knowledge as a whole (Fetherston, 2001).

The next sub-section overviews various problems encountered.
3.3.2.1 Problems regarding constructivism and assessment

Learning in a constructivist environment is assessed through performance-based projects rather than through traditional paper and pencil testing. Because constructivist learning is based on learners' active participation in problem-solving and critical thinking regarding a learning activity (Bruner, 1999; Jonassen, 1992; Jonassen, 1999), the assessment of constructivist learning is a much more complicated task than assessing traditional learning. Constructivists object to decontextualised evaluation and the separation of evaluation environment from the learning environment (Jonassen, 1992).

Wilson (1996) believes that traditional assessment has been a major force in retarding educational reform, and that it could have the same negative effect on the development and implementation of constructivist learning environments. The task of the instructor in traditional assessment usually involves activities such as designing task scenarios and test questions, communicating performance standards, observing performance of learners, evaluating or marking performance, and then communicating results in a comprehensible fashion. This requires considerable time and effort.

3.3.3 Problems related to interactive web-based learning

Before these problems are discussed, a description is required of what is meant by the term interactive in web-based learning.

Interactivity in a computer environment is the dialogue that occurs between a user and a computer program. Programs that run without immediate user involvement are not interactive, and are usually called batch or background programs. The graphical user interface (GUI) of the WWW promotes interactivity because it offers the user more interaction options. Users interact not only with the browser (the WWW application program), but also with the pages of text and/or graphics that the browser brings to his/her computer. In addition to hypertext, the WWW offers other possibilities for interactivity. Any kind of user input, including typing commands, clicking the mouse, and responding to questions, is a form of interaction. Displayed images and text,
printouts, motion video sequences, and sounds are output forms of interactivity (Whatis?com, 2002).

3.3.3.1 Problems regarding passive versus active learning on the WWW

The phenomenon of web-based learning is continually on the increase. However, Doherty (1998) warns that web-learning is in danger of becoming a passive learning technology because instructors tend to present traditional educational material with little consideration of the unique features of the WWW. He believes that this type of web publication ignores the potential of using technology to create an interactive, participative learning experience for the user.

Hill (1997) addresses the variation of purpose within web sites which have been developed to provide learners with access to instructional resources from a distance. These sites range from those that post course materials such as syllabi, class notes, and review materials, to sites that immerse learners in activities with the aim of promoting interaction, involvement, and engagement. She believes that sites that portray high-end use of technological and educational techniques are currently the exception rather than the rule. Most course-based or learning sites simply serve as a channel for course materials. She states that use of a web-based learning site merely as an 'electronic book' falls far short of the potential offered by the medium, and that the WWW has the potential for moving instruction away from a repository model to one where active learning can occur.

According to Perkins (1991), current education is failing to prepare learners for success in society, and he proposes that instruction should focus on retention, understanding, and active use of what is learned.

By contrast, the Internet and WWW can offer learners a learner-centric, interactive, and highly participative learning experience. WWW environments are highly appropriate for implementation of the constructivist ethos. Hedberg, Brown and Arrighi (1997) believe that the complete freedom of the web environment suggests that the user is operating entirely in a constructivist framework.
Constructivist learning is based on learners' active participation in problem-solving and critical thinking. Constructivists believe that learners construct their unique understanding of the world by experiencing, evaluating and incorporating their interpretations into their personal view of the world.

3.3.3.2 Problems regarding constructivism and active learning

Learners in constructivist environments (discussed in Section 2.2) have to discover concepts without being directly given all the information they need. The more the problem-solving learning situation represents the real-world, the more likely it is that the learner will transfer the skills to other problem-solving situations.

Compared to traditional views of learning, constructivism is primarily concerned with how learners construct knowledge, while the emphasis in objectivism is on the object of knowing. The constructivist paradigm posits that learners do not receive bits of information and then store them mentally, but rather that learners take information from the real-world and then construct their own view of that knowledge domain.

In spite of strong support of constructivism in the contemporary literature, there are also authors who express their concern with regard to constructivism.

According to Carr et al (1998) constructivist learning environments make many new demands on participants. In traditional learning environments learners approach a learning experience expecting to passively listen to an instructor who controls both the flow and the content of the class. In constructivist situations, however, learners have to take active roles, and the transition may not be easy. Many learners are successful (success being defined by grades) within the current system and have no wish to learn in unfamiliar ways which are more demanding. Learners may also experience discomfort with the increased levels of ambiguity and frustration that necessarily accompany real-world and complex problems. It is expected of learners not only to construct their own meaning, but also to adopt a new style of learning, all at once (Carr et al, 1998).

Constructivist learning activities also make many new demands on educators as they change their role from that of an authority figure who presents knowledge to learners, to
that of a senior partner and facilitator. It is expected of them to coach and instruct not only in the content area, but also in the area of new and probably unfamiliar learning techniques. They have to engage learners in real-world, complex and authentic problems. They have to create learning situations where learners are able to learn through the experimentation, collaboration and sometimes even failure. These are often characteristics of constructivist learning environments where learners are expected to construct knowledge and make meaning from a model where a problem or project drives the learning (Jonassen, 1999). They must adjust the curriculum so that constructivist activities, which may take longer, can be balanced with “covering the material” required at certain levels (Carr et al, 1998).

Most educators were not trained to adopt the constructivist paradigm.

“.....can we expect overworked teachers, who are unfamiliar with the constructivist paradigm, to be able to create constructivist learning experiences, or to help prepare learners to be open and successful to these experiences? Just as the nature of the task for the learner changes significantly in a constructivist setting, so does the task of the teacher” (Carr et al, 1998:10).

Educators and learners do not operate in a vacuum but rather in a community. Just as most teachers were educated in an objectivist tradition, so were most parents, administrators, evaluators, curriculum designers and other key participants. If they do not accept the validity of constructivism, they can undermine the success of constructivist learning environments. For example, the constructivist approach requires alternative assessment procedures, which can be very difficult for some institutions and regulatory bodies to accept (Carr et al, 1998). For successful implementation of constructivist programmes, all stakeholders must support the process.

Landa (1998) states that instructional media alone are not able to produce better learning environments. What makes instructional technology effective, are the content and methodology of instruction that are used in conjunction with the instructional media.
In the next section a number of reasons are discussed why web-based learning environments sometimes fail to promote active learning. These include drawbacks of current educational software, and problems related to the use of HTML and CGI.

3.3.3.3 Drawbacks of current educational software

Landa's (1998) analysis of American educational software reveals the following drawbacks:

- Educational software products teach how to solve problems in a specific domain, but do not teach general methods of thinking which would allow learners to solve any problem of a given class or type.
- They predominantly teach solutions to problems, rather than processes of independently arriving at solutions.
- In cases of attempting to teach processes, the problems and processes are broken into sub-problems and sub-processes by the computer instead of teaching learners how to do this on their own.
- When learners experience difficulties and make errors, the instructional programs, in many cases, ask them to 'try again', which develops and strengthens the bad habit of guessing instead of teaching learners how to think correctly from the start.
- If learners fail to produce the correct response to a question posed by the computer, the instructional programs simply supply them with the right answer. This is done frequently without explaining why this answer is correct, and the process of thinking that leads to the correct solution is seldom demonstrated.
- If learners experience difficulties and turn to the \(<\text{Help}>\) function, the help, in most cases, consists of providing learners with prompts, clues, and suggestions which, in fact, represent ready-made solutions. Thus, instead of teaching learners how to solve problems, many instructional programs simply eliminate problems by solving them for the learners.
- Software products that attempt to teach thinking, as a rule, do not focus on the dynamics of mental operations involving creative processes. Instead, by placing learners in situations requiring creativity, the programs simply confront them with creative problems.
- Some instructional programs conduct diagnostics of learners' knowledge and skills with the purpose of adapting the level of instruction to them. Though necessary,
such diagnoses are insufficient. For deeper adaption and flexibility of instruction, it is necessary to diagnose not only to the level of the learner’s knowledge and skills, but also the psychological cause of learners’ difficulties and errors. In cases of error, typical instructional programs react to the nature of the errors, rather than diagnosing what caused them.

3.3.3.4 Drawbacks of HTML

One major problem with the WWW is that search engines cannot process HTML (Section 3.2.2.6) intelligently. For example, if a search is conducted on the word ‘Chip’ when a user is searching for someone named Chip, he/she might get pages on chocolate chips, computer chips, as well as persons named Chip. But if there were a document type definition (DTD)\(^7\) for name and address records, searching for a person named Chip could generate more accurate and useful search results (Tidwell, 1999).

3.3.3.5 Drawbacks of CGI

CGI scripts (Section 3.2.8) are responsible for collecting user input, and can be described as the ‘window’ through which the user and the program interact.

Unfortunately there are some problems related to the use of CGI, such as:

- CGI scripts execute relatively slowly.
- Writing CGI scripts is a complicated task that requires advanced programming skills.
- Debugging CGI code can be a complex, time-consuming process, because error messages do not always spell out the exact cause and location of the error. Even for those experienced in debugging programs, the process of debugging CGI programs poses a different kind of challenge (Herrmann, 1996).

\(^7\) A DTD is a specification that accompanies a document and identifies what the codes (or markups) are that separate paragraphs, identify topic headings, and so forth and how each is to be processed. By mailing a DTD with a document, any location that has a DTD reader will be able to process the document and display or print it as intended. This means that a single standard compiler can serve many different kinds of documents that use a range of different markup codes and related meanings. The compiler looks at the DTD and then prints or displays the document accordingly (Whatis?com, 2002).
3.4 Possible solutions for some of the problems

3.4.1 Solutions to assessment problems

Effective, efficient means of assessing learner skills and learning processes need to be identified. There are various different ways of assessing learning in the 21st century. One method entails observing learners as they focus on problems, gather information, identify key variables, develop research questions, etc. individually and within cooperative groups. This study, however, is focused on automated assessment. On-line testing can be constructed with CGI scripts, and then used for providing guidance and feedback, gathering information from learners, comparing established criteria in text or database files, and for assigning grades. It is not difficult to implement interactive multiple-choice assessment, but the complex issue remains the extent to which open-ended unconstrained responses can be judged by software programs.

This section briefly outlines the constructivist approach to assessing learning and then addresses the role of technology in assessment.

3.4.1.1 Solutions regarding constructivism and assessment

Constructivist learning environments (Section 2.2.2.2), whether open, structured, or virtual, place learners in positions where they explore, experiment, and actively solve problems. It is obvious that alternative means of assessment are required for constructivist learning (Wilson, 1996).

Wilson claims that alternative assessment cannot be realized unless technology is employed in changing the forms of assessment, establishing procedures for judging performance, and sharing criteria and standards for assessment. He describes several functions of technology in alternative assessment, including:

- supporting extended, authentic learning activities,
- making work portable and accessible,
- making performances repayable,
- providing libraries of examples and interpretive tools,
expanding the community of participants in assessment, and
> publishing and sharing learners' work.

According to Wilson (1996), assessment in constructivist learning environments needs to be as varied and broad as the environments themselves. For example, an assessment task in teaching literature might require learners to write an original sonnet, if the goal of the learning environment is to help learners develop such a talent. Alternatively, the assessment might consist of self-reports or journals in cases where the learning is aimed at unique learning outcomes, individually selected by the learners.

Assessment, in these situations, is clearly a task carried out by human teachers, although the learners' work may be generated and presented on the WWW, rather than paper-based.

In discussing constructivist evaluation criteria, Jonassen (1992) reviews the purpose of evaluation. Evaluation implies an appraisal or value judgement regarding the learner's performance. If learning is viewed as the process of constructing knowledge (as well as the development by learners of products) then that itself is an appropriate goal, and the constructors (learners) themselves can play a role by self-evaluating. In this case evaluation is a means of self-analysis and metacognition.

3.4.1.2 Solutions regarding the role of technology in assessment

Hansen & Frick (1997) distinguish between computer-based assessment and computer-mediated assessment. Computer-based assessment entails grading by the computer, whereas computer-mediated assessment occurs when the computer is used as the conduit between the learner and the educator. Web-based instruction, however, goes beyond conventional computer-mediated assessment because the assessment can occur at a distance, with the Internet as the conduit.

The creation of web-based courses is a complex task. In this light, some schools of thought suggest that all the elements of information, interaction, and course management should be addressed by a web-based instruction authoring tool. Hansen & Frick (1997) state that a good web-based instruction authoring tool should support a variety of
assessment methods. It should have the capability to create tests, quizzes, and guided practice with feedback. It should:

1. Enable the educator to create the assessment interface consisting of:
   - a form for multiple-choice tests,
   - a web page with a dialogue box for short answers,
   - a response area,
   - a form that will be submitted to the educator for grading,
   - a page that streamlines electronic file transfer of papers or other documents from learner to educator and back again,
   - a self-assessment response form, and
   - an adaptive test with feedback for both learners and educator.

2. Generate Common Gateway Interface (CGI) script that can be used to grade, examine, and provide feedback without involvement of the educator.

3. Create database files to store results from tests or other interactions that the educator would like to examine. These files should also be automatically forwarded to the appropriate location, such as the educator's e-mail account.

Hansen & Frick (1997) warn that, before a decision can be made about which authoring tool to use for developing a web-based instruction course, the way the course will be conducted must be considered carefully. The use or non-use of such a tool depends on the context and content of the course.

3.4.1.3 Assessment challenges

The integration of assessment into constructivist learning is not an easy task. Wilson (1996) states that more research on assessment is needed, especially on technical issues such as reliability, theoretical issues such as validity, and practical issues such as feasibility.
It is important that the development of policies, methods, and technologies for alternative assessment should keep pace with the development of constructivist learning environments.

Fetherston (2001) states that one of the main functions of assessment in a constructivist environment is to assist learners in finding out what they know or do not know. Thus assessment practices in a constructivist environment should change from the traditional assignment/exam system, to one based on giving learners greater opportunity to portray what they know. He challenges educators to construct assessment systems that support this.

Fetherston therefore proposes that assessment needs to be changed. Traditional methods that focus on recognition, recall, congruence with course objectives, and the measurement of surface learning should be replaced by holistic assessment focused on learners’ own conceptions, the measurement of deep learning, long lasting knowledge, and allowing learners to express interpretations.

Fetherston believes that the WWW has the potential rise to this challenge and play a meaningful role in the provision of good assessment. **Use of its interactive and multimedia capabilities can lead to many creative assessment strategies.** Fetherston further proposes this as a fruitful research and development area.

### 3.4.2 Interactive web-based learning environments as a solution

Ritchie & Hoffman (1997) define instruction as a *purposeful interaction* aimed at increasing learners’ knowledge and skills. Within this context, simply publishing web pages with links to other digital resources does not institute instruction.

Rather than merely designing instructional sequences, the emphasis should be on designing better learning environments based on certain constructivist principles.
3.4.2.1 Solutions regarding constructivism and interactive learning

The Internet and WWW can offer learners a learner-centric, interactive, and highly participative learning experience. WWW environments are highly appropriate for implementation of the constructivist ethos. Hedberg, Brown and Arrighi (1997) believe that the complete freedom of the web environment suggests that the user is operating entirely in a constructivist framework.

Constructivist learning is based on learners' active participation in problem-solving and critical thinking. Constructivists believe that learners construct their unique understanding of the world by experiencing, evaluating and incorporating their interpretations into their personal view of the world.

Constructivists object to formal testing, preferring integrated assessment. They believe that the emphasis should be on learner control and the ability to manipulate information. Learners must learn how to think effectively, reason, solve problems and how to develop learning skills. Linear chains of instruction no longer apply. When the learning is done, assessment should focus on the transfer of newly-learned skills to other contexts (discussed in Section 2.2.2.2).

The role and accessibility of the WWW have increased considerably during the past few years, aspects which lend themselves well to active constructivist learning. Web-learning enables learners to produce work and reflections in real time in collaboration with others, particularly peers who are working towards similar goals. Instead of occasional progress reports to each other in class meetings or sharing one-to-one, all the learners can post their progress on a regular basis on web pages or on an electronic bulletin board. This dynamic, highly visible learning increases collaboration and builds a community of learners. Public postings provide learners with feedback on their work, knowledge of what others are doing, and the chance to comment on their own and others' work (Serim & Koch, 1996).

With advances in visual and interactive technologies such as 3D interactive computer environments, the process of knowledge construction and the derivation of meaning from visual, auditory and other sensory perspectives can be more fully explored. By creating their own environments, learners can develop their own set of objects, relationships and
behaviours that are meaningful to them, and that can be shared and experienced through interaction (Osberg, 2000).

"An understanding of interactive technology and how it can support education and learning is essential, as our social infrastructure increasingly relies on that technology for communication and information transfer" (Simms (1997) cited in Doherty, 1998:62).

Doherty states that dynamic interaction, which he believes is the most under-utilised feature of the WWW, has the potential to be the most important feature. "It is unique to the WWW as no other medium or technology has identical capabilities" (Doherty, 1998:63). He presents the following dynamic interaction features of the WWW:

- **Dynamic navigation** - a user, using a search engine to conduct a search, could uncover different search results from those accessed in the same search conducted by him/her the previous day, since the content and structure of the WWW changes dynamically all the time.
- **Dynamic inputs** - a search for a particular author at an on-line bookstore, returns a web page created dynamically to match the input search criteria.
- **Dynamic human feedback** takes on many forms such as feedback on questions posted in a on-line conference, or unrequested comments. It is described by him as "a random or chaotic event" where one could receive feedback from an unknown, unrelated person, living far away.

Doherty suggests that "It is now time to find out how best to utilize the Internet as an (inter)active learning technology" (Doherty, 1998:63).

Plowman (2002) warns that although interactivity is the strong point of interactive media, it can also cause problems for users. Designers of interactive media are in a difficult position, as there is limited theoretical basis for design decisions. Until recently, design processes for interactive media have been interpreted mainly in terms of system design, requirements analysis or instructional design. Plowman believes that this approach is misconceived, because interactive media are radically different from the original computer-based training technologies on which these approaches were partly based, and
which are more concerned with training than education. They often embody behaviorist theories of learning, which are manifested by the software (rather than the user) controlling the route taken, and they generally contain instructional segments that are usually highly structured and sequenced. Their design is based on an assumption that there is a direct relationship between instruction and learning outcomes. More recently, there has been a move towards using constructivist theories of learning, although these are currently associated more with the development of materials for research, than for the commercial market. Constructivist environments are more learner-centred, and take account of the ways in which learners bring prior knowledge to a learning situation, and actively construct their own understanding in a social context. Experimental, laboratory-based evaluation methods are unsuitable for understanding these processes.

According to Plowman, many educators who had held high expectations of interactive media, were largely disappointed. The interactivity associated with interactive media is often seen as concerned solely with technological capability, such as control over pace, choice of activity and sequence. Yet there is not necessarily a correlation between the sophistication of the platform and the level of interactivity. She argues that the design of technological interactivity has much to learn from human interaction. For example, an older friend, or family member, reading a picture book with a young child, can be highly interactive, as the more experienced reader offers guidance and elicits both tacit and explicit knowledge. In such a case, the interaction is not technological, but is based on the relationship between the two people and the book, and the relationship between the book's use of text and image.

Plowman believes that this scenario can provide a model for the guided interaction required of interactive software, which should be able to elicit and encourage existing competences, should be adaptive, and should offer 'scaffolding' for extending learning. The active role in teaching and learning within the classroom is distributed between the teacher, the learners and other artifacts such as textbooks and worksheets. Discussions between learners and teachers promote the guided construction of knowledge. At this stage in the development of interactive media, however, it is not usual to find these teaching and learning processes imbodied in the design. Instead, many examples can be found of needless interactivity in which learners respond to opportunities for interaction that have
been designed into the software with little clear purpose. These systems tend to produce little learning gain, and the interactions are empty and passive rather than active. Measuring learning gain in any meaningful way is very difficult, as the variables are so complex. However, Plowman believes that qualitative research can be used to illuminate understanding of the experience of using interactive technology, and to identify a number of design issues for consideration.

The next section discusses new technological developments that can be used to enhance HTML so as to produce better interactive learning environments.

### 3.4.2.2 Technological developments that enhance interactive web-based learning

It is not easy to predict tendencies related to the rapid development of the WWW. It is certain, however, that new sources of information and tools to enhance learning are rapidly on the increase.

The WWW enables learners to fill in forms, watch videos, complete multiple-choice tests, have it marked, watch live videos, listen to live broadcasts, watch and interact with animations, submit written assignments, send e-mail other learners, talk to other learners realtime, and even to see other learners. These capabilities are resourceful technical achievements, and Fetherston (2001) believes that development in this area is unlikely to slow down.

Web pages used for web-based learning need not be static, because programs (such as XML, *JavaScript*, and PHP) that generate HTML as their output and then return that HTML to the browser, can be used. This results in dynamic pages that return different results based on the arguments supplied when requesting a page.

The enhancement of HTML by using XML, *JavaScript*, PHP and databases is now addressed. Proposals are made regarding how these means can be used towards solutions of some of the problems mentioned in Section 3.3.3.
(1) Java as a solution

Java (see Section 3.2.2.8(2)) is increasingly being used to create web pages with dynamic and interactive content, to develop large-scale enterprise applications, to enhance the functionality of web browsers, and to provide application devices for consumer devices such as cell phones, pagers and personal digital assistants (Deitel & Deitel, 1999).

(2) HTML and XML as a solution

XML (see Section 3.2.2.7(2)) is a new technology for web applications that provides powerful new features to web programmers and users. XML's strongest point is its ability to handle data interchange. Because different organisations, or even different parts of the same organisation, rarely standardise on a single set of tools, it takes a significant amount of work for two groups to communicate. XML makes it easy to send structured data across the WWW so that nothing gets lost in translation. With XML, the computer can understand the meaning of tags, because the programmer can build meaning into the tags. The computer can understand that the tag `<first-name>` Chip `</first-name>` means that 'Chip' is a person's first name (see Section 3.3.3.4) (Tidwell, 1999; Whatis?com, 2002).

Programs save files in various formats - some use binary, while others use text. Many formats do not work across platforms, for example, MS Word files for the Macintosh platform differ from those for MS Windows, and Framemaker creates an intermediate format called MIF to deal with platform and version differences. XML solves this problem by providing a standard way to define file formats. XML is not a format itself, but a way to create formats. XML will not replace HTML because XML is a way to present data formats, and is not a format itself. The main difference between HTML and XML is that the developer defines the tags in XML, while HTML ignores all unknown tags (Asbury, 2000).

(3) HTML and JavaScript as a solution

JavaScript is a scripting language developed by Netscape to add interactivity and power to web documents (see Section 3.2.2.8(3)). JavaScript differs from most other programming languages in that it is relatively easy to master. It should be noted that JavaScript is completely different from the Java programming language developed at Sun Microsystems. However, the two languages can interoperate well. Java is much more powerful, more complex, and harder to master, belonging in the same league as C, C++,
and other more complex languages. A Java program needs to be compiled before it can run, whereas with JavaScript, no compilation is needed. The user simply opens up a text editor, types it, saves it, and the browser is ready to run it.

Most WWW users have browsers that support JavaScript and that is why JavaScript is one of the most popular tools for adding interactive features to web pages.

(4) HTML and PHP as a solution

Writing and debugging CGI scripts (described in Section 3.2.2.8) is a complicated task that requires advanced programming skills. Unlike Perl, C++ and Java, which are general purpose scripting and programming languages that can be used for a wide variety of purposes, PHP (discussed in Section 3.2.2.8(3)) was custom designed to be used specifically for scripting web pages. As a result, many facilities that web designers typically use in a scripting language (such as sending and receiving mail, sending and retrieving documents via HTTP or FTP) are built into PHP. These facilities have to be explicitly programmed by web developers when using Perl, C++ or Java (Heng, 2000).

PHP-enabled web pages are treated like regular HTML pages, and can be created and edited in the same way that regular HTML pages are created and edited. However, like ASP, PHP can be used as a tool for creating dynamic web pages, because the content will vary based on the results of interpreting the script. The goal of the language is to allow web developers to write dynamically-generated pages easily and quickly (Serrano, 2000; Whatis?com, 2002).

The PHP script language and interpreter can be thought of as a hybrid between C, Perl, Microsoft's VBScript and JavaScript, incorporating many of the most prominent features of each and integrating them into one complete language. Therefore, a user with experience in any of these languages will quickly master the basic syntax of PHP. What makes it different from a CGI script written in other languages like Perl or C, is that instead of writing a program with many commands to output HTML, an HTML script is written with some embedded PHP code to execute a task (e.g. to output some text). The PHP code (enclosed in special start and end tags) is not visible when viewing the source code of a page, because the server processes the code and returns only the output. As with ASP,
PHP script is embedded within a web page along with its HTML. This means that, like JavaScript, PHP code can be called and processed, and then return back to HTML. What distinguishes PHP from client-sided scripting languages such as JavaScript, is that the code is executed on the server, and then returns plain HTML to the browser (Gilmore, 2000; Heng, 2000; Serrano, 2000).

PHP can also be run as an external CGI process, a stand-alone script interpreter, or an embedded Apache module. At the most basic level, PHP can do anything any other CGI program can do, such as collect form-data, generate dynamic page content, or send and receive cookies. Since the HTML form, and related code, are all in one page, and PHP displays all errors on the browser, PHP code is much easier to code and debug than writing CGI scripts in Perl or C (Heng, 2000; Serrano, 2000).

PHP has the ability to connect with relational databases (discussed next) such as MySQL, Informix, Oracle, Sybase, Solid, DB2, mSQL and PostgreSQL, and also supports ODBC (open database connection) (Heng, 2000; Serrano, 2000).

(5) Relational databases
A database is a collection of interrelated files that store information in a structured format. Databases enable users to store, search, sort, and retrieve data. The difference between data and information is that data is disorganised, while information is organised. The relational data model represents the database as a collection of relations. Data is implemented as a series of two-dimensional tables that are related to one another. Each table (also called a relation) consist of named columns (which are fields or attributes) and any number of unnamed rows (which correspond to records). Each relation resembles a table of values, or a file of records. When a relation is thought of as a table of values, each row in the table represents a collection of related data values. Data models that preceded the relational model include the hierarchical and network models (Elmasri & Navathe, 2000; Medinets, 2000; Whitten, Bentley & Dittman, 2001).

———

8 A cookie is information for future use that is stored by the server on the client side of a client/server communication. Typically, a cookie records a user's preferences when using a particular site (WhatIs.com, 2002).
SQL is a language used to define database tables and columns, query databases for information, and update information, which collectively make application development possible (Medinets, 2000).

(6) MySQL
MySQL is a relational database management system (RDBMS) that is freely available under the Open Source license. It supports standard SQL (Structured Query Language), which is the standard database query language worldwide. It compiles on a number of platforms, such as Unix, Linux and MS Windows, which means that web applications developed in MS Windows, can operate on a Unix or Linux platform (Serrano, 2000; Welling & Thomson, 2001).

A MySQL server controls access to stored data to ensure that multiple users can work with it concurrently to provide fast access to it, and to ensure that only authorised users obtain access. Therefore, MySQL is a multi-user, multi-threaded server. It has been publicly available since 1996, but has a development history going back to 1979, and has won the Linux Journal Readers' Choice Award three years running (Welling & Thomson, 2001).

(7) Combination of PHP and MySQL as a solution
Since Serrano (2000) believes that the combination of PHP and MySQL is the best choice to use for e-commerce and business applications on data-driven web sites, it should be an excellent choice for presenting web-learning applications. When support for PHP is activated, and MySQL is enabled on a web server, PHP supports several features that include authentication, XML, dynamic image creation, WDDX (Web Distributed Data Exchange)⁹, shared memory support, and dynamic PDF document creation.

Support is freely available from PHP and MySQL web sites in the form of documentation, manuals and mailing lists. Bugs are fixed rapidly, and requests for features are always attended to, evaluated, and if feasible, implemented (Serrano, 2000).

---

⁹ WDDX (Web Distributed Data Exchange) is a free, XML-based technology that allows web applications created with any platform to exchange data without users having to write special interfaces (Hovind, 1999).
3.5 Conclusion

Despite the criticisms, problems and issues mentioned in this chapter, many institutions are promoting the practice of instruction and learning on the WWW. As an instructional technology, the WWW is unique in its ability not only to transmit a variety of media, but to do so from virtually anywhere, providing asynchronous access on a global basis. It permits the sharing of information, and has the potential to become the most comprehensive communication system ever developed. Furthermore, the WWW holds the potential to technologically implement the ethos of constructivist learning. Constructivism is moving beyond the realm of research and is increasingly being implemented in authentic situations of teaching and training.

Problems related to assessment and interactive learning with special reference to constructivist learning environments were discussed. It was noted that the integration of assessment into constructivist learning environments is a complex task. Although the learners' work may be on the WWW and not paper-based, assessment is still a task frequently undertaken by carried out by human educators. Because constructivism focuses on learning rather than on instruction, educators are forced to move away from the mere provision of instructional content on the web site, and must also provide opportunities for communication and interaction. It is proposed that effective, efficient means of assessing learner skills and learning processes need to be investigated.

Certain factors underlying the failure of web-based learning to promote active learning were discussed. These include drawbacks of current educational software and problems related to the use of HTML and CGI. The need to move away from static educational web pages to dynamic web pages in order to produce more effective interactive web-based learning was identified. The use of a combination of PHP and MySQL was proposed to create more dynamically generated web pages.

The various media that can be used, and the many kinds of possible interactions, pose the challenge of finding good instructional designs and methods that will enhance the interactive nature of the WWW to promote learning.
Web-based learning

From the literature studied it is clear that a large gap exists between traditional learning environments and the real-world. Some authors recognise that a constructivist approach has learning benefits, and believe that constructivism might be the foundation for a bridge to close this gap. However, when compared to traditional instruction, the nature of constructivist learning imposes increased demands on educators as well as on learners.

Although proponents of constructivism are generally opposed to formal testing, this does not mean that all aspects of constructivism are irrelevant for assessment. In this study the view is taken that some principles of constructivism can usefully be implemented in order to design more interactive learning products that:

- focus on learning as a process;
- provide opportunities for communication and interaction;
- integrate assessment with instruction, thereby enabling performance to be sampled repeatedly over time, and
- are designed according to cognitive principles.

Using the WWW for instruction means that educators should move beyond the practice of merely publishing their printed course material on web sites (i.e. using it mainly as a convenient delivery medium). Appropriate instructional design strategies should be employed in the development of new material for the WWW that capitalise on its unique capabilities. Moreover, educators should recognise the learning benefits of a constructivist approach.

By focusing on the learning process, and by providing useful feedback, assessment can be used for much more than mere grading. Use of the interactive and multimedia capabilities of the WWW to produce creative assessment strategies was proposed, therefore on-line assessment and means of implementing these, are discussed in detail in the next chapter.
References - Chapter 3


Herrmann, E. 1996. Teach yourself CGI programming with Perl 5 in a week. Indianapolis, USA: Sams Publishing.


Chapter 4: Automated assessment

4.1 Introduction

It was concluded in Chapter 3 that most current web-based learning environments are more successful in presenting content than in assessing the learning process. Automated assessment is based mainly on rigid technologies such as quizzes or multiple-choice questions, as well as on educator-managed communication forums or asynchronous e-mail exchanges. Nevertheless, good teaching should be characterised by instructional practices that include effective assessment of learners' work including textual products. Furthermore, meaningful feedback by educators on learners' efforts can provide scaffolding to the learning process.

This chapter focuses on automated assessment by investigating the most prominent approaches to essay scoring and automated grading of computer programming assignments. The chapter concludes with speculation on the future of large-scale electronic assessment.

4.2 Automated assessment systems

El-Tigi and Branch (1997) state that all learning sessions, whether on-site, at a distance, delivered electronically, or conducted in any other way, should provide opportunities for learners to:

- interact with the educator,
- control the information process, and
- give and receive feedback about the knowledge being constructed.

Milton (2001) states that on-line assessment offers few new challenges for educators because the issues of assessment (the essential questions of why, what and how to assess) remain much the same, no matter the medium. He, however, believes that on-line assessment offers new possibilities for technological enhancement of assessment and learning, and that the potential of on-line assessment is primarily formative (see Section 3.2.1.4) in the sense that it can help:

- learners to get a clearer idea of what is expected;
- facilitate constructive feedback, and
- engage learners in learning from their peers.
Serim and Koch (1996) believe that the typical perception of assessment views it as comprising multiple-choice standardised tests, quizzes, and term papers (i.e. written assignments and projects). However, researchers investigating the way learning occurs query the worth of the regurgitation skills required to achieve in these kinds of tests.

"Even the Princeton Review\(^1\), which makes a very healthy income from helping people increase their test scores, is forthright in their assertion that performance on such tests is an accurate predictor of absolutely nothing, relative to later success in college or at work" (Serim & Koch, 1996:136).

Assessment should not merely involve tests taken at the end of a course, but should rather entail learners reviewing their own comprehension, making sure that they are building on a sound foundation. Assessment must become an ongoing, integral part of instruction, and evaluation should be a meaningful determination of progress towards a goal (Nicoletto, 200; Serim & Koch, 1996).

Vigilante (1999) measured the amount of time educators spend managing typical on-line courses with intensive discussion and collaboration areas, and found that 25% of their time is spent in manually grading written assignments. Dessus, Lemaire and Verner (2000) propose that this time-consuming task be reduced by the development of automated assessment systems.

\(^1\) The Princeton Review started with fifteen standardized achievement tests (SAT) learners in 1981. Currently The Princeton Review courses help more than 75,000 learners a year prepare for various standardized tests at the high school, college, and professional levels. The Princeton Review is a leader in test preparation, offering courses in cities across the United States and abroad (Serim & Koch, 1996).
criticizing written products is important not only as an assessment method, but also as a feedback device to help students better learn both content and the skills of thinking and writing. Nevertheless, essays have been neglected in many computer-based assessment applications since there exist few techniques to score essays directly by computer" (Foltz, Laham & Landauer, 1999: http://www-psych.nmsu.edu/~pfoltz/reprints/Edmedia99.html).

Extended-response items, typically short essays, form an integral part of most large-scale assessments. Such items allow learners to demonstrate a wide range of skills and knowledge, including higher order thinking skills such as synthesis and analysis. However, assessing learners' writing is one of the most expensive and time-consuming activities for human educators - prompts need to be designed, rubrics (model solutions) created, multiple raters (human markers) need to be trained, and then the extended responses need to be scored (marked), typically by multiple human markers. With different people evaluating different essays, inter-rater reliability becomes an additional concern in the assessment process. Differences in background, training, and experience of raters can lead to subtle but important differences in grading (Rudner & Gagne, 2001).

Computers and artificial intelligence have been proposed as tools to assist educators in the evaluation of learner essays. In theory, computer scoring can be faster, reduce costs, increase accuracy and eliminate concerns about rater consistency and fatigue. Furthermore, the computer can quickly re-score materials should the scoring rubric be redefined (Rudner & Gagne, 2001).

The most prominent approaches to essay scoring can be subdivided according to two basic techniques namely free-text assessment based on surface features and free-text assessment based on course content (Educational Resources Information Center, 2002). Various precedent studies on existing artifacts used for e-assessment of textual products such as essays were investigated:

- Project Essay Grade (PEG) (free-text assessment based on surface features), introduced by Ellis Page in 1966.
- E-rater (free-text assessment based on course content), used by Educational Testing Service (ETS) and developed by Jill Burstein.
Latent Semantic Analysis (LSA), first introduced for essay grading in 1997 by Thomas Landauer and Peter Foltz, and which is also an implementation of free-text, *course content* assessment.

The discussion of automated grading systems in this section is subdivided according to these two techniques. Free-text assessment based on surface features (PEG) is discussed in Section 4.2.1, and then free-text assessment based on course content (*e-rater* and LSA) is discussed in Section 4.2.2, while Section 4.2.3 focuses on automatic grading of computer programming assignments. All these systems are commercial products, and exact details of their internal operation are not generally available.

### 4.2.1 Free-text assessment based on surface features

#### 4.2.1.1 Project Essay Grade

The earliest effort in grading free texts was done by Page in 1966 with the *Project Essay Grade (PEG)*. After two years of research, he found that computers could grade essays as effectively as a human grader by checking for easily detectable attributes such as sentence length, essay length and the presence or absence of particular words. *PEG* grades essays on the basis of writing quality by using computer variables that approximates the intrinsic qualities of a person's writing style (called *trins*) that need to be measured (such as aptness of word choice). Approximations of these variables (called *proxes*) are used to measure these underlying traits. Specific attributes of writing style such as average word length, number of semicolons, and word rarity are examples of proxes that can be measured directly by *PEG* to generate a grade. More than 30 years of research has consistently shown exceptionally high correlations with essays marked by human raters (Wresh, 1993; Rudner & Gagne, 2001).

*PEG* focuses on measurement of writing style and does not take into account the *semantic content* of the essays. Even an irrelevant essay could score a high grade, which is a serious problem. Two tools based on assessment of free-text based on *content* (*e-rater* and LSA) are discussed in the next section.
4.2.2 Free-text assessment based on content

A great deal of research has been conducted on information retrieval techniques. Although natural language processing (NLP) techniques and artificial intelligence (AI) systems are capable of extracting information that can be used for grading, they require huge efforts of painstaking knowledge engineering to develop, and are relatively unpredictable when dealing with unexpected input. Some of the e-assessment tools discussed in this section make partial use of certain AI approaches, but are not pure artificial intelligence systems.

4.2.2.1 E-rater

This section describes the e-rater system, and research conducted on the use of e-rater, and is based on Burstein et al (1996); Burstein et al (1998); Burstein & Marcu (2000a); Burstein & Marcu (2000b); Burstein et al (2001); Dessus, Lemaire & Verner, (2000); Educational Testing Service (2001); Rudner & Gagne (2001).

_E-rater_ is an automated essay scoring system developed at the Educational Testing Educational Network (ETS)](2) by the Natural Language Processing Group, and is based on a decade of research in natural language processing (NLP). Their goal was to create an electronic system that can score free-responses (such as short-answer free-responses to test questions and essays) using the same criteria used in hand-scoring.

_E-rater_ 'learns' to score essays on a particular topic by processing a number of assays on the topic. The system was designed to automatically analyse essay features based on writing characteristics specified in the holistic scoring guide used by expert readers for manual scoring of the analytical writing assessments of the Graduate Management Admissions Test (GMAT) essays. This scoring guide has a six-point scoring scale. One

---

2 Educational Testing Service (ETS) is the world's largest private educational testing and measurement organisation, and a leader in educational research. It is a nonprofit company dedicated to serving the needs of individuals, educational institutions and agencies, and governmental bodies in 181 countries. ETS develops and annually administers more than 11 million tests worldwide on behalf of clients in education, government and business (Educational Testing Service, 2001).
of their goals was to design a system that could score essays based on features of writing that expert readers normally use to evaluate learners' writing. Natural language processing (NLP) and information retrieval (IR) techniques were used to extract and measure the following three features:

- **Syntactic variety** (or structure) is an important feature in evaluating essays. *E-rater* implements a syntactic parser included in the Microsoft Natural Language Processing tool (MSNLP) to parse each of the sentences in an essay. Other *e-rater* programs then quantify the features based on the arrangement of phrases and clauses in these parses.

- **Organisation of ideas** is another important feature. Characteristics associated with the orderly presentation of ideas, such as rhetorical features and linguistic cues (logical connections between sentences and clauses) are then identified by a second program. It relies on words or sentences that contain rhetorical arguments such as: "in conclusion…", "in summary…", "perhaps", etc.

- **Topical content**: A content score is computed by translating the word pattern of the essay (based on vocabulary related to the topic, such as relevant information and precise or specialised vocabulary) into a representative vector, which is then compared with the vectors of manually graded essays.

Research by Burstein *et al* (1998) showed that the topical analysis component of *e-rater* is one of the strongest indicators of the score of an essay. They found that essays with similar scores tend to use the same vocabulary and discuss the same topics. Therefore, test essays can be scored automatically on the basis of their similarity with essays in a training sample of essays that have been manually scored by human graders. They showed that scores that yield exact or adjacent agreement with human graders in about 83% of the cases can be assigned by using only the topical analysis of the features of *e-rater*, and not the syntactic and organisational components of the assessment system.

*E-rater* was proven to be highly effective and accurate, and has provided one of the two scores for over 730 000 examinee responses to essay questions administered as part of
the Graduate Management Admissions Test (GMAT). The correspondence rate between the scores of e-rater and those of expert faculty readers on GMAT essays consistently exceeds 97 percent.

Burstein & Marcu (2000b) believe that it is insufficient merely to provide learners with a grade (or score). In order to help learners improve their writing skills, essay grading systems need to provide feedback that is specific to each individual's writing, and that is applicable to essay revision. They investigated the use of automatically generated summaries in order to provide instructional feedback, and found a significant indication of improvement in learners' performance.

An automated scoring technology called c-rater (concept-rater) is currently under development at ETS Technologies. They are investigating the feasibility of automating the scoring of short-answer content-based responses to questions. C-rater uses natural language processing technology. Initial results are promising, since the system achieved over 80 percent agreement with scores assigned by human graders (C-rater, 2003).

Two other e-rater-based free-text assessors, Criterion and Questionmark Perception, are discussed in the following subsection.

4.2.2.2  E-rater-based free-text assessors based on course content

(1)  Criterion

E-rater was further developed to produce a web-based essay evaluation system that instantly provides holistic evaluation of learners' essay writing skills. This system, called Criterion, provides diagnostic feedback that learners can use to improve their writing skills (Burstein & Marcu, 2000b). It can be used with any language arts program, any writing text, or any other curriculum presented in the English medium and enables earners to:

- Write and revise on more than 80 topics.
- Submit an unlimited number of essays.
- File work in an on-line portfolio.
- Gain access from any Internet-connected computer.
- Receive help, including sample essays and advice for writers.
- Receive immediate feedback.
(2) Questionmark

This section expands on Section 4.2.2.1 by introducing an e-rater based system named Questionmark. Questionmark is a registered trade mark of Questionmark Computing Ltd., that provides products and support services for education and training. The company also sells and supports software used for testing and assessment, accordingly enabling users to write, deliver and mark surveys, tests, and questionnaires. In the fourteen years of its existence, Questionmark has been used by thousands of corporate human resources professionals, trainers and educators in more than 40 countries. Applications include academic tests, product knowledge tests, attitude surveys, personnel evaluations, self paced study guides, and many others. Clients are from a wide range of industries (banking, manufacturing, insurance, accounting, health care) and education (primary and secondary education institutions, colleges and universities). Questionmark has been translated into numerous languages and is also available from local distributors in South Africa (Questionmark, 2002).

Earlier products were; Questionmark for DOS, Questionmark for Windows, and Questionmark for Macintosh. Current products are; Perception for Windows, and Perception for Web. All these products are focused on providing software to create, deliver and report on assessment used for placement, formative assessment, pre-course assessment, and needs analysis.

By using Questionmark Perception's authoring tool, questions can be created and designed in a number of different formats including multiple choice, multiple response, fill-in-the-blank, word response, numeric questions, and essay questions (Questionmark Perception, 2002).

For assessment of essay questions, the educator can define what is right or wrong by entering an advance list of acceptable answers. Learners answer by typing up to 30 000 characters of text, and scoring is based on the presence, or absence, of keywords or key phrases.
The *Perception Server* automatically deploys the test (or survey) across the WWW or an intranet to any standard browser. Participants answer the questions and, if desired, can receive instant feedback. *Perception Server* automatically scores, tabulates and stores the answers in a secure database (Questionmark Perception, 2002).

**Essay grading with Questionmark Perception**

*Questionmark* presents four ways to grade an essay (Questionmark Perception, 2002):

- A grade can be assigned automatically by performing a ‘text match’ and determining the presence or absence of key words and/or key phrases.
- A subject expert determines a grade by reading the text in a browser and then assigns a grade manually.
- At least two subject experts grade the same essay manually and the resulting score is recorded.
- If sufficient samples are available, then an essay can be graded automatically using Questionmark.

### 4.2.2.3 Latent Semantic Analysis

*Latent Semantic Analysis* (LSA) is a statistical technique originally designed for indexing documents for the purpose of information retrieval. It is a computational model of human knowledge representation, and a method for extracting words with similar meaning from text. The meaning of a word is determined from all of the contexts in which that word occurs. LSA can also be used to grade essays. To assess the quality of essays, LSA is first trained on domain-representative text, and then, based on this training, derives a representation of the meaning of these words as they are contained in the domain. It uses *singular value decomposition* (SVD), a general form of factor analysis, to condense a very large matrix of word-by-context data into a much smaller, but still large (typically 100 - 500) dimensional-representation. Essays are characterised by vectors based on the

---

3 SVD is closely related to a number of mathematical and statistical techniques in a wide variety of fields, including eigenvector decomposition, spectral analysis, and factor analysis (deerwester et al., 1990). SVD works as follows; all vectors are reduced by a method close to eigenvector decomposition to, for instance, 100 dimensions. The matrix $X$ is decomposed as a unique product of three matrices: $X = T0 \cdot S0 \cdot D0$ such that $T0$ and $D0$ have orthonormal columns and $S0$ is diagonal. Only the 100 columns of $T0$ and $D0$ corresponding to the 100 largest values of $S0$ are then kept, to obtain $T$, $S$, and $D$. The reduced matrix $X$ such that: $X = TSD^T$ permits all words and paragraphs to be represented as 100-dimensional vectors. It is this reduction which is the heart of the method because it allows the representation of the meaning of words, by means of the context in which they occur. If the number of dimensions is too small, too much information is lost. If number of dimensions is too big, not enough dependencies are drawn between vectors. It was empirically shown that a size of 100 to 300 produces the best results in the domain of language (Dessus, Lemaire & Verner, 2000).
combination of all their words⁴. These vectors are then compared with vectors for essays of known quality. The angle between the two vectors represents the degree to which these two essays discuss information in a similar way. For example, when an ungraded essay is compared to an essay that has already been graded, and the angle between these two essays is small, then they are considered to be similar in content. In this way, a score can be derived based on the similarity of the contents of two essays. A word can be considered semantically close to another word although they do not co-occur in texts. In the same way, two documents could be considered similar although they do not share any words. The semantic information is derived only from the co-occurrence of words in a large collection of texts. There is no need to code semantic knowledge by means of a semantic network or logic formulas. The power of LSA, lies in reducing the dimensions, and thereby inducing semantic similarities between words or paragraphs (Foltz, Latham & Landauer, 1999; Dessus, Lemaire & Verner, 2000).

Wiemer-Hastings (1999) examined the claim that the power lies in the compression step by comparing several approaches which assess the quality of learner contributions in an intelligent tutoring situation. Human judgements of quality were used as a baseline, and were compared to three different models:

> The full LSA model.
> A version of LSA without SVD.
> A simple keyword matching mechanism.

This research showed that the performance of the non-SVD algorithm, as well as the simple keyword-based approach, were very close to that of the full LSA implementation. The maximum performance of the full LSA model compared to the human raters achieved a correlation of $r = 0.48$, while that of the non-SVD model was $r = 0.43$, and the simple keyword-based model was $r = 0.40$. He concluded that if a proper set of texts is available as a basis of comparison, a text evaluation mechanism based on comparison can be produced by using any of these three models. The simple keyword match performed

---

⁴ Dessus, Lemaire and Verner (2000) illustrates this principle with the following example: If the word bike occurs generally in the context of handlebars, pedal, ride, etc., and a word such as bicycle occurs in a similar context, the two words are considered semantically close to each. Their corresponding vectors in the semantic space will be also close to each other. This semantic space is built by considering the number of occurrences of each word in each piece of text (paragraphs). For instance, 300 paragraphs and a total of 2,000 words produce a $300 \times 2,000$ matrix. Each word is then represented by a 300-dimensional vector and each paragraph by a 2,000-dimensional vector.
surprisingly well, and was relatively inexpensive computationally. A mechanism like the non-SVD model did not produce better maximum performance than the keyword model on the relatively short texts used in the investigation, but it did produce good performance across a range of thresholds, indicating a robustness to be able to handle a variety of inputs. The full LSA model exceeded the performance of both these models, achieving results comparable to those of humans with intermediate domain knowledge (Wiemer-Hastings, 1999).

Several systems currently being used for automatic text evaluation, are based on the full LSA model.

### 4.2.2.4 LSA-based free-text assessors based on course content

Various LSA-based systems are instances of free-text assessment based on course content.

#### (1) Intelligent Essay Assessor

The *Intelligent Essay Assessor* (IEA) is a set of LSA-based software tools used for scoring the quality of essay content. IEA is first ‘trained’ on several texts related to the domain. Essays are then compared with pre-graded essays by means of two kinds of scores (Foltz, Laham & Landauer, 1999):

- the *holistic score*, which returns the score of the closest pre-graded essay, and
- the *gold standard*, which returns the LSA proximity between the learner essay and a standard essay.

An experiment using the holistic score was performed on 188 essays in the domain of Biology. A correlation of 0.80 was shown between IEA grades and human graders. A notable feature of IEA is that the learner can re-submit the essay over and again in order to improve the score. Foltz, Laham and Landauer (1999) note that the average grade increased from 85/100 (first submission) to 92/100. However, one drawback is that the educator needs to manually grade standard essays in advance (Dessus, Lemaire, & Verner, 2000).
(2) **Summary Street**

Summarising is a strategy that can be used to determine whether learners understand what they have read, and whether or not they have learned from it.

"Summarization not only reveals the existence of comprehension breakdowns during reading, but also helps to pinpoint the location and cause of the breakdown. Writing a summary also involves active meaning construction, despite its focus on the textbase. To a much greater degree than notetaking or outlining it requires the construction of a mental representation that joins elements of text information with each other and with elements of prior knowledge. To the degree that students do this, they will acquire new knowledge that is useful and long lasting; a well elaborated knowledge representation will support knowledge application as well as recall. Finally, summaries are a communication tool, a means of sharing one's knowledge with others" (LSA Website Executive Summary, 2002: http://lsa.colorado.edu/exec.html).

**Summary Street** is an educational software system that uses LSA to support the reading and writing activities by which learners develop and expand their knowledge in new topic areas. **Summary Street** determines the degree to which a learner’s summary covers important source content, and conforms to requirements, such as length. It provides critical feedback on missing information, as well as comments on redundancy, and on insignificant content. The feedback enables learners to engage in extensive, independent practice in writing and revising without placing excessive demands on educators for manual feedback (Kintsch et al, 2001).

After typing summaries into a textbox, learners receive the following types of feedback on request (Kintsch et al, 2001):

- Misspelled words are highlighted, and can be corrected in the textbox. The summary is automatically saved by this operation.
- The request for feedback provides a graphical display indicating the length of the summary, and how well the content of each section of the original text has been covered.
Further help for revision is available at this point. For example, should the summary exceed the prescribed length, a redundancy check and a relevance check are provided. These tools help learners to locate sentences that have overlapping content, or that are not directly related to the topic being summarised, and that could well be deleted.

Finally, a <Format for Printing> button enables the learner to obtain a double-spaced version to print out, review and submit to the educator (Kintsch et al, 2001).

(3) Select-a-Kibitzer

Select-a-Kibitzer is another commercial LSA based free text assessor based on content. The Select-a-Kibitzer project was funded by the Cognitive Studies for Educational Practice Program of the James S. McDonnell Foundation. The program was sponsored by the Psychology Department at the University of Memphis (Dessus, Lemaire, & Verner, 2000).

Select-a-Kibitzer is a computerised tool that provides feedback to learners on their compositions in a unique way. After entering a composition into the tool, the learner requests feedback. Various natural language processing techniques are used to analyse the text. Select-a-Kibitzer differs greatly from standard ‘style checker’ mechanisms which focus on surface features of the text. It is an agent-based computer tool, and each agent is responsible for providing specialised advice in the form of feedback which comments on a different aspect of the text, for example, spelling, grammar, coherence, and relevance to the assignment. These assessments are quite similar to those provided by Summary Street. However, this software emphasises the negotiated construction of the text by associating each agent to a character. Learners can decide whether or not they want feedback from any particular agent. The goal of the system is to improve learners’ writing skills by making them aware of the different types of constraints that they have to satisfy when creating a composition (Dessus, Lemaire, & Verner, 2000; Wiemer-Hastings, 2001; Wiemer-Hastings & Graesser, 2000).
(4) Apex

Apex differs from IEA (see point (1) in this section) in that it does not depend on pre-graded essays, but rather on various semantic comparisons with the course. It also differs from Summary Street and Select-a-Kibitzer, because it takes into account the structure of the course to grade an essay (Dessus, Lemaire & Verner, 2000).

Apex is a web-based learning environment which can be used to assess learners' knowledge based on the content of free textual responses. It manages learners' productions, assessments, and courses. Assessment depends on LSA for comparing an essay to the text of a given course in order to measure how well the essay matches the course. Learners connect to the system and then select a topic (part of a course or a question) that they wish to work on. They use a text editor to type a text about the chosen topic, and they can receive a three-part evaluation of the essay at any time. After reading any of these three assessments, the learner can go back to the essay to modify the text and re-submit. The educator, who also connects to the system, can either add a course, view learners' assessments, or create exam questions (Dessus, Lemaire & Verner, 2000).

All courses, learners' texts and exam questions are stored in a database, which is managed by an administrator. The system runs on the Linux operating system (discussed in Section 3.2.2.4) and uses the following three tools:

- Apache server 1.3.12 (discussed in Section 3.2.2.5),
- PHP 4.0.0 (discussed in Sections 3.2.2.8 and 3.4.2.2), and
- MySQL 3.22.32 (discussed in Section 3.4.2.2).

The Apache server runs PHP scripts in order to produce dynamic pages. PHP is a server-side HTML-embedded scripting language, and PHP scripts enable communication with Apex assessor, which in turn, runs LSA routines. MySQL is a relational database that is freely available, and supports standard Structured Query Language (SQL).

**Marking up the course**

In order to implement a course on Apex, the course has to be marked up by the educator to give it a two-level structure. This task is much less time-consuming than preparing a
multiple-choice test. The text is divided into topics, and each topic divided into notions. Basically, the structure of the file corresponds to an outline view of a word processor, therefore it is quite straightforward to do such a marking-up on a document. Under the Apex system first level titles should begin with #T (for Topic) whereas second level titles should begin with #N (for Notion). Only second level titles are followed by a paragraph. A notion can belong to several topics. However, to avoid redundancies, the text of such a notion is written only once. To specify such a cross-reference, the notion title should begin with #S. The educator can specify questions by using the same mechanism to indicate the relevant parts of the course (Dessus, Lemaire & Verner, 2000).

Figure 4.1 presents an example of such a marking-up: the notion title Introduction to the solar system is defined in the topic, The solar system, and is referred to under the topic, Eclipses. Learners’ responses to the question "Describe the different types of eclipses" should cover two notions of the course, namely: Sun eclipses and Moon eclipses.

Discussion of LSA-based assessment tools

Dessus, Lemaire and Verner (2000) claim that the holistic format of the assessments provided by Apex enables more authentic feedback than the marking of multiple-choice questions.

All the systems discussed in this sub-section use LSA, a tool that represents the meaning of words as vectors in a high-dimensional space. Dessus, Lemaire and Verner state that these systems are production-centred rather than learning-centred, and that the authors of these systems pay more attention to providing adequate prompts to the users than to presenting rich environments for active learning. They believe that learners should not only be receivers of assessment, but also active planners and producers. Learners involved in distance learning programmes often write textual inputs in the form of contributions to discussion forums and e-mails, but these inputs are seldom assessed by the educator. Furthermore, self-assessment (such as writing portfolios) is being used more and more as a means to promote the learning process.
The main criticism of all these approaches is that the computer could be fooled by a learner writing an essay that contains only keywords. However, Dessus, Lemaire and Verner believe that a learner would have to have a good knowledge of a domain in order to be able to provide the right keywords, and this is exactly what they want to measure. Further criticism is that spelling and syntax are not taken into account. They suggest that these systems should be supplemented with adequate third-party software.

Figure 4.1 Marked-up text of a course to be processed by Apex (Dessus, Lemaire & Verner, 2000).
4.2.2.5 Using classification as a lexical semantic technique to grade free responses

Burstein, Kaplan and Lu (1996) view grading as a classification problem, and make use of lexical semantic techniques to implement a prototype scoring system for short-answers and free-responses to test questions. Responses are automatically scored by assigning appropriate classifications to words. Their ultimate goal is to develop a scoring system that can reliably analyse response content.

This technique differs from natural language understanding systems, which are designed to use large collections of full text sources, by using lexical semantic techniques to build scoring systems based on small data sets. A concept-based lexicon, as well as a concept grammar were built to represent a response set. They believe that lexicons restricted to dictionary knowledge of words are not sufficient for interpreting the meaning of responses for unique items, and that concept knowledge bases, built from an individual data set of examinee responses, should be used for representing a domain-specific language (Burstein, Kaplan & Lu, 1996).

4.2.2.6 Comparison of PEG, e-rater and LSA

Both PEG (introduced in Section 4.2.1.1) and E-rater (Section 4.2.2.1) use a regression model. PEG uses a regression model with surface features of the text (document length, word length, and punctuation) as the independent variables and the essay score as the dependent variable, and does not take into account the semantic content of the essays. E-rater uses a regression model with content features as the independent variables. LSA (Sections 4.2.2.3 and 4.2.2.4) uses a factor-analytic model of word co-occurrences which emphasises essay content (Rudner & Gagne, 2001).

Although high correlations have been reported between scores produced by grading essays with these systems and essays scored by humans, the use of automated essay scoring is debatable. For example, a well-written essay on "How to fly a kite" could receive a high score in a case where PEG was used to grade essays about the causes of a particular war. On the other hand, LSA could be tricked into giving a high score to an essay that contains a string of relevant words with no grammatical sentence structure.
Automated assessment

\textit{E-rater} appears to overcome some of these criticisms at the expense of being fairly complicated. Compared to LSA and \textit{e-rater}, \textit{PEG} has the advantage of being conceptually simpler and less expensive on computer resources. \textit{PEG} is also the better choice for evaluating writing style, since it relies on writing quality to determine grades, whereas \textit{LSA} returns grades that have literally nothing to do with writing style. \textit{LSA} and \textit{e-rater}, however, appear to be the superior choice for grading content in domains where questions have clearly defined answers (Rudner & Gagne, 2001).

4.2.3 Automatic assessment of computer programming assignments

In this section, one commercial automated assessment system namely the \textit{Cellidh} system is described.

4.2.3.1 The \textit{Cellidh} system

(1) Overview

The \textit{Cellidh} system is a courseware application used for administration of academic courses and the automatic assessment of learners' work. The system was developed by the \textit{Learning Technology Research} (LTR) group of the Department of Computer Science at Nottingham University in the UK in 1988 to support the automatic assessment of computer programs written in C. Since then \textit{Cellidh} has evolved into a system that supports a continuously changing curriculum, and a large number of learners. It has been used internationally, and has aided the running of a variety of university courses, providing automatic assessment of computer programs in many languages, multiple choice questionnaires, question-answer exercises, essays/reports. It also provides helpful feedback to learners, tutors and teachers (Higgins \textit{et al}, 1999; Foxley, Higgins & Burke, 2001).

The system has so far been adopted by 300 institutions in 30 countries, and can be used to run a variety of programming courses. Each course includes many exercises complete
with marking schemes and course notes. Institutions using Ceilidh are encouraged to write their own new exercises and marking tools (Foxley, Higgins & Burke, 2001).

Ceilidh was replaced in 1998 by LTR's newest courseware system CourseMaster. The CourseMaster system is a complete re-implementation of the Ceilidh assessment and administration system, and is based on 10 years of experience of using the Ceilidh system in many institutions world-wide. CourseMaster provides learners with instant detailed feedback on their submitted coursework, while enabling lecturers to monitor the learners, to mark their work automatically, and to generate reports about possible plagiarism. The system is client/server based with the possibility of using distributed servers, and is written in Java, which provides portability to a wide range of platforms. It is more flexible than the previous Ceilidh system, uses a more user-friendly interface and provides more detailed learner feedback. It further enhances the learning experience by providing more sophisticated assessment tools, improved security and web integration. CourseMaster assesses and monitors learners' work through the use of an on-line submission system. Moreover, it archives learners' work and manages the presentation of information to its users (Higgins et al, 1999; Foxley, Higgins & Burke, 2001).

The system's assessing mechanism evaluates the quality of the work submitted by using a combination of scripts, properties and metric tools that perform in-depth analysis, inspection and testing. Metric tools have been built for the areas of imperative, logical and object-oriented programming languages (Higgins et al, 1999).

The LTR group is in the process of further enhancing CourseMaster by including the assessment of diagrammatic inputs such as flow charts, circuit schemas for electronics and UML diagrams for OO design. Research is also undertaken on the use of object oriented (OO) metrics for assessment of OO analysis, design and architecture of Java programs; a course in formal specification using Z and TOAD (an OO tool which makes use of OO heuristics) to facilitate the analysis and design of C++ programs (Higgins et al, 1999).

(2) **Method used by Ceilidh and CourseMaster**

A program called an oracle is used to recognise whether a given piece of text contains a particular meaning. It is used to check that the output from a program represents the
correct output, and whether the program source contains particular features (Zin & Foxley, 2001).

The Unix concept of a regular expression is used. Regular expressions are involved in many Unix commands, text editors, the sed⁵ stream editor, the grep family of commands, and awk (Zin & Foxley, 2001). Regular expressions, grep, awk and sed are discussed in more detail in Sections 5.6.1. and 5.6.3. To check the program output, the educator provides a number of regular expressions which are searched for in the text. A regular expression (described in more detailed in Section 5.6.1) is a pattern comprising a set of strings, with each string in turn being a set of characters. For example, the regular expression "(ferrous|iron) *oxide" can be used to search for "ferrous oxide" or "iron oxide". All possible alternatives have to be provided when looking for a certain keyword. These regular expressions are provided one-per-line in a file of which the name is supplied as the first argument in a call to the oracle program. The program uses awk⁶ to detect the regular expressions, and the output of the program is of the form "$score 93" which is a percentage of the mark awarded out of the maximum possible mark, depending in which regular expressions were found. The program is normally SUID⁷ (set user ID) to enable the file containing the regular expressions (which is opened from within the program) to be kept hidden from the user (i.e. the learner) by having 'no public read' permission (Zin & Foxley, 2001).

To inspect computer programs, Ceilidh has a two-fold approach. First it investigates the output of the program, looking for certain patterns and expressions to see if the programmer has solved the given problem correctly. Second, it examines the

---

⁵ sed is a stream editor used for editing streams of text that might be too large to edit as a single file, or that might be generated on the fly as part of a larger data processing step. The most common operation done with sed is substitution; replacing one block of text with another.

⁶ awk is a powerful utility that can be used to find lines containing specific string of characters, to manipulate data and to print selected portions.

⁷ Normally Unix and Linux scripts and programs run with the same permission as the user who executes the file. However, SUID programs execute with the permission settings of the file's owner, rather than those of the user executing the command (Raikow, 2000; Akin, 2001).
programming code for particular warning messages such as "undeclared variable" or "undefined symbol".

The electronic assessment programs discussed in this section vary in their purposes, techniques, and scope. Since many educational institutions globally are experiencing problems in dealing with increasing numbers, there is an increased need for alternative e-assessment methods that could address these increases. Large-scale assessment is therefore under great pressure to change. Speculation on its future is discussed in the next section.

4.3 Bennet’s speculations on the future of large-scale assessment

Large-scale assessment consists of tests administered to large numbers of people for the purposes of placement, course credits, graduation, educational admissions, and school accountability (Bennet, 1998).

Bennet points out that large-scale assessment has changed very little over the last 20 years, and that this cannot continue. Most such tests still serve only institutional purposes, are administered to big groups in single sittings on a few dates per year, make little use of technology, and are based on a psychological model that is derived more from the behaviourism (discussed in Chapter 2) of the first half of this century than from the cognitive science of the second half. He believes that this situation is about to change because of different demands, such as changing demographics, the emergence of a competitive global economy, market forces within the testing enterprise, and calls for change from the educational community. The same competitive forces driving industry will force test makers to satisfy new market needs through continuous innovation, improved productivity, enhanced customer service and attention to population diversity. Bennet predicts that, in response, large-scale assessment will re-invent itself, and will facilitate both summative and formative purposes. It should benefit high-stakes decision-making as well as instructional purposes, become curriculum-embedded and performance-based, be able to occur
at a distance, and measure skills newly valued by society. This re-invention will be assisted by advances in cognitive and measurement science.

Bennet proposes a scenario of the future of large-scale testing to speculate on how assessment might change in the future. He suggests three generations through which testing might progress, each being distinguished by evolution in test purpose, format, content, and delivery location, as well as the extent to which the tests utilise new technology.

4.3.1 Three generations of large-scale assessment

4.3.1.1 First-generation computer-based tests

Depending on the testing program, individuals register by phone or e-mail, pay by credit card, test by appointment in a dedicated test centre, and receive scores at the conclusion of the session. Advances in psychometrics and technology are combined in order to deliver large-scale tests adaptively. These tests are administered as 'one-time' measurements. Existing processes are computerised and automated by these tests. They resemble paper-based tests because they measure the same skills, use the same behavioural designs, and depend primarily on the same types of tasks. In addition, costs would be increased because item pools must be continuously renewed (Bennet, 1998).

4.3.1.2 Next-generation computer-based tests

The typical test in this generation would be qualitatively different from those in the first generation. Change occurs first in the nature of test questions, secondly in the formats for response, and finally in the possibilities these provide for measuring new skills.

Permanent computer-based test centres with equipment capable of delivering high quality multimedia would enable test questions to include sound, video and animation more effectively, changing the nature of test questions.
Response formats would change from a single correct solution to more open formats, given certain constraints. Some answers would be completely open-ended, for instance essay-type answers. Tasks calling for other types of performance such as oral presentation, sign language, and the display of an artifact, could also be administered routinely. Responses may be captured digitally through microphones and video cameras.

Test development, scoring and administrative processes would also be re-engineered. New tools that allow developers to construct question templates, or to select them from large libraries, should emerge. Bennet predicts that tools could become sufficiently sophisticated to generate items without human intervention (Bennet, 1998).

4.3.1.3 Generation “R” (re-invention)

In this third generation, Bennet predicts that testing will re-invent itself by breaking drastically with tradition in several ways.

The rapidly escalating cost of traditional higher education, combined with the convenience of electronic networks will establish distance learning as a dominant force in higher education. Distance learning will traverse international boundaries. Bennet speculates that this approach will spread to secondary school, enabling even school learners to benefit from world-class educators and curricular materials.

Large-scale, one-time, test-centre-delivery examinations would be replaced by large-scale distance examinations. Computerised assessment will be integrated with instruction so that performance can be sampled repeatedly over time. Assessment will be designed according to cognitive principles, and new technologies will be used. As changes occur in the skills valued by society, new competencies will be measured.

Decisions such as certification of course mastery, graduation eligibility, and school effectiveness will incorporate information from a series of measurements, and will no longer be based largely on one examination given at a single time.
There will be a shift from the accuracy of assessment to the adequacy of instruction.

Tasks associated with electronic learning and assessment tools will be very different from those of prior eras. Simple multimedia exercises will be replaced by virtual reality simulations that model complex environments such as science laboratories and field experiences, thus enabling learners to learn and be assessed under conditions similar to those encountered by practitioners.

Bennet further suggest that generation "R"s computers will support more natural forms of interaction, for example, by responding directly to the learner's physical actions in virtual reality simulations, and by using speech recognition. He predicts the use of complex computer-based simulations in assessment, including virtual reality in a natural way.

4.4 Conclusion

There is a need to assess work products of large numbers of learners effectively and cost-efficiently. It is important to recognise that multiple-choice-type tests are not inherently more reliable than other type of questions. Furthermore, the reason why such questions have been used extensively and over a lengthy time span is the ease with which they can be automatically scored. The scoring of tests by technology means continues to be a major research domain.

Several studies have reported favourably on assessment via PEG, e-rater, and LSA. These programs have been found to return grades that correlate significantly with those assigned by human raters. E-rater was deemed so impressive that it is now operational and used to score the General Management Aptitude Test (GMAT) in the USA. A major drawback of these systems is that essay-type answers still have to be manually graded (Select-a Kibitzer and Summary Street) or tagged beforehand (Apex).

The Ceilidh system (now replaced by CourseMaster) used for automatic marking of learners' work (mainly computer programming) on computer-related courses, was also investigated. This system is also used for administering the resulting marks, solutions and course material, programming
exercises, and for administration of academic courses. It provides helpful information and feedback to learners and lecturers.

Since much manual pre-processing of textual data has to be done in preparation for the assessment of textual responses, the question arises as to whether computer-based methods would indeed be efficient with regard to resources such as educators' time and the cost of grading. It is the researcher's strong belief that as long as grading remains a problematic and costly issue in large-scale testing, researchers should pursue any solution that automates the process.

As with human judgments, automated scoring can be unstable. Total accuracy should not be expected from any automated assessment or grading approach. Despite the limitations, the time has come to consider the use of automated scoring and to develop validation tools for formative evaluation of textual responses.
References - Chapter 4


Chapter 5: Methods, models and tools used for developing E-Grader and Web-Grader.

5.1 Introduction

The study of the progression and features of web-based learning as described in Chapter 3, resulted in a number of proposals. First, it was suggested that the practice of using the WWW mainly as a delivery medium for instructional course material should be replaced by the development of new material explicitly designed for the WWW, and that this should be done by employing appropriate instructional design strategies. Use of the interactive and multimedia capabilities of the WWW to produce creative assessment strategies was also proposed. This led to the question of how open-ended textual inputs can be validated electronically, which is a major focus of this investigation. The first step taken to address this issue, was a theoretical study (see Chapter 4) of existing software programs involving electronic assessment of textual inputs. The need was identified for effective and cost-efficient assessment of large numbers of learners. In addition, research was conducted on electronic assessment techniques currently used for textual inputs. Particular attention was paid to the various techniques used for textual processing, essay scoring and automated grading of computer programming assignments.

The second step of this research consists of the development of a prototypical software system for electronic assessment of open-ended textual inputs, as well as its practical implementation and on-going validation and upgrading of the system. This chapter focuses on the methods and tools used in this second step. The various software engineering process models and instructional design models which were investigated in order to choose the most suitable means for development of the new software system, are summarized here. The development of this system was done in two separate phases, producing two different variants of the system called E-Grader and Web-Grader respectively. The development process of E-Grader, which was developed to initially test the concept of assigning marks to learners' work products electronically, is described in Chapter 6, while Web-Grader, an interactive web-based extension of the first variant, is described in Chapter 7. The tools used to build these systems are described in Section 6.5.
5.2 Software engineering methodologies and principles

The development of instructional (educational) software should be based on software engineering (SE) methodologies and principles. Software engineering is a discipline of which the aim is to produce fault-free software, delivered on time, and within budget, that satisfies the user's needs (Schach, 1996).

Pressman (1997:22) states that in order to engineer software adequately, a software development process must be defined. The software process is the way the software is produced, and he defines it as "a framework for the tasks that are required to build high-quality software". This process defines the approach that is taken as software is engineered. Software engineering is the discipline that integrates process, methods, and tools for the development of software. Software engineering methods provide the technical support for building the system. Methods encompass a broad array of tasks that include requirements, analysis, design, program construction, testing, and maintenance. Software engineering tools provide automated or semi-automated support for the process and methods.

The words methodology (meaning the science of methods) and paradigm (meaning a model or pattern) are widely used within the context of software engineering. Both these words can be used to indicate a collection of techniques that cover the life cycle of a system (Schach, 1996), also termed the software development life cycle (SDLC). The series of steps that software undergoes from concept exploration through to final retirement, is termed the life cycle.

"A systems development methodology is a very formal and precise system development process that defines a set of activities, methods, best practices, deliverables, and automated tools for system developers and managers to use to develop and maintain most or all information systems and software" (Whitten, Bentley & Dittman, 2001:78).

A model is intended to serve as a framework to guide the thinking and practices of designers. Some of the development models used in software engineering are discussed next.
During the life cycle of a system the product goes through a series of phases:

- Analysis (requirements specification),
- planning,
- design,
- implementation,
- integration,
- maintenance, and
- retirement.

A variety of different process models are used for software engineering, each exhibiting strengths and weaknesses, and most of them are built around these or similar phases. Software process models can broadly be divided into two categories according to the paradigm or approach used to structure the phases of the software process, the most well known being the linear approach and the evolutionary (or iterative) paradigm.

### 5.2.1 Software process models with a linear sequential approach

The classic Systems Development Life Cycle (SDLC) is the oldest paradigm used for software engineering, and deals with system development in a sequential and linear fashion. This approach assumes that a complete system will be delivered after a linear sequence has been completed. The waterfall model is discussed here as an example of this type of approach.

#### 5.2.1.1 The waterfall model

During the 1960s and 1970s, software development projects were characterized by large-scale cost overruns and schedule delays, and the focus was on planning and control. Until the end of the 1970s most organizations were producing software using as their life-cycle model what is now termed the waterfall model (see Figure 5.1). This model was developed in order to tackle the growing complexity of large software systems. There are many variations of the model, but generally the product goes through the phases mentioned in the introduction of this section, with explicit feedback loops to the previous stage (Schach, 1996).
Considerable emphasis is placed on careful analysis before the system is actually built. The user's requirements are therefore identified as early as possible, and well documented in the requirements specification.

One of the problems with this model is that the client sees the system for the first time only after the entire system has been coded. Real-world projects rarely follow the sequential flow that this model proposes. Although the linear model can accommodate iteration, it does so indirectly. As a result, changes in the requirements can be time consuming and expensive.
The waterfall model has a definite and important place in software engineering. It provides a template into which methods for analysis, design, coding, testing, and maintenance can be placed. It remains one of the most widely used process models for software engineering. While it does have weaknesses, it is significantly better than a haphazard approach to software development (Pressman, 1997).

5.2.2 Software process models with an evolutionary approach

Evolutionary models are incremental and iterative. They enable software engineers to develop increasingly more complete versions of the software. Examples of evolutionary models are:

- the prototyping model,
- the incremental model,
- the spiral model,
- the rapid application (RAD) model,
- the component assembly model,
- the concurrent development model, and
- the rational unified process model (RUP).

The basic idea shared by these models is that users' needs for different types of software are so complex to understand, and changes in hardware and software technologies are so continuous and rapid, that it is unwise to attempt a complete design of a software system in advance.

"Instead, projects may need to iterate while concurrently managing many design, build, and testing cycles to move forward toward completing a product" (Snyder, 2002: http://faculty.winthrop.edu/SNYDERS/SE.EP/se1-06.asp).

The rapid prototyping model was used in this study for the development of the prototype software system (discussed in Chapters 6 and 7) to assess short-answer free response items. Consequently, this model is discussed next.
5.2.2.1 The prototyping model

A *prototype* is a miniature version of a final product. It is an executable version of an actual product, incorporating key elements of the final version, but it is incomplete in terms of functionality and robustness (Wong, 1993).

*Rapid prototyping* (see Figure 5.2) is a technique used for quickly building a functioning, but incomplete, model of the information system by using rapid application tools (Whitten, Bentley & Dittman, 2001).

![Diagram of the prototyping model](image)

*Figure 5.2 Rapid prototyping model (Schach, 1996:59).*
It enables the developer to create a working model of the intended software, and to test and modify it. A prototype typically evolves into the final version of the system or application through an iterative process (Chao, 1998).

Schach (1996) defines a rapid prototype as a piece of software hurriedly assembled, that incorporates much of the functionality of the target product, but omits those aspects generally invisible to the client, such as file updating or error handling. The client and users then experiment with the prototype to determine whether it meets their requirements. The rapid prototype can be changed until the client and users are both satisfied that it encapsulates the required functionality.

The first step in the rapid prototyping life-cycle model is to build a rapid prototype and to experiment with it. The software process continues from there as shown in Figure 5.2. A major aspect of this model is that the prototype must be built for change. If the first version of the rapid prototype is not what the client needs, then the prototype must be transformed rapidly into a second version that might better satisfy the client’s requirements (Schach, 1996).

5.3 Instructional systems development models

"Educational software developers have been criticized for the lack of software engineering methodologies and principles in their analysis and design and therefore have led to low quality products" (Wong, 1993:155).

Instructional design is defined by Hoffman and Margerum-Leys (2002) as a systematic approach to course development which ensures that specific learning goals are accomplished. It is an iterative process that requires ongoing evaluation and feedback. Hoffman and Margerum-Leys warn that although numerous instructional design theories and models provide guidance in this domain, only a few have been empirically validated and tested in a classroom environment. The lack of validation should not, however, be viewed as a shortcoming, because the learning process is unpredictable and subject to numerous extraneous variables.
5.3.1 Evolution of instructional systems development models

Several innovations in the 20th century contributed to the development of the Instructional Systems Development (ISD) field in the United States of America and the models currently used. Self-paced instruction and learning were emphasized in the 1920s, and procedures for writing behavioural objectives were refined in the 1930s. Military training focussed on the use of varied media developed by instructional design teams during the 1940s. Programmed instruction, based on B. F. Skinner's research (described in Section 2.3.2.1) and task analysis procedures used by the US Air Force, played a major role in the 1950s. The U.S.A. Department of Education encouraged the implementation of ISD procedures in the 1960s. A proliferation of ISD models emerged during the 1970s, and needs assessment became an important feature of the process. The growth of computer-based instruction in the 1980s, and the need for cost effectiveness in the 1990s also contributed to the evolution of the ISD field (Barry, 2002).

The most popular ISD models are the Dick & Carey (1985), Kemp (1985), Gerlach & Ely (1980), and Seels & Glasgow models which are based on a behaviourist foundation (discussed in Section 2.3.2.1) where the focus is on aspects such as learning objectives and operant conditioning through reinforcement of the desired behaviour. In contrast to these traditional models, are the Second Generation Instructional Design (ID₂) model which is a derivative of component display and transaction theories (Merrill, Zhongmin & Jones, 1990), and the Prototyping models. These are more progressive models, moving towards constructivism and the constructivist approach (discussed in Section 2.3.2.2). They are less constrained to a sequential process of design, and also tend to incorporate more collaborative learning activities and place less emphasize on the educator's role (Hoffman & Margerum-Ley, 2002; Preстera, 2002).

The Dick & Carey model, and the prototyping models are discussed next.
5.3.2 The Dick & Carey model

Dick & Carey (1996) propose a comprehensive and systematic model for developing instruction in a wide variety of contexts (corporate, education, military, etc.) and at a variety of levels (curriculum, program, course, unit). The model (see Figure 5.3) follows the basic stages of analysis, design, development, implementation, and evaluation, and emphasizes the role of needs analysis, implementation, and evaluation (Prestera, 2002):

- Front-end analysis is a critical component in this model. If an initial needs assessment warrants the development of instruction, then goals, tasks, learners, and context are analysed. A hierarchical list of learning objectives is a critical outcome of this stage.

- In the design phase, two sets of products are designed: the assessment items and the instructional content itself. The assessment items are derived and should correspond directly with the criteria (learning objectives) written in the analysis stage. Once the assessment items are written, the instructional content can be designed.

- In the development phase, specific artifacts such as tests, handouts, instructor guides, presentation slides, and instructional media are prepared.

- In the implementation phase, the instruction is implemented in the actual learning setting with real learners.

- This model puts much emphasis on evaluation and quality control, distinguishing between formative and summative evaluation. The formative evaluation is an ongoing process, evaluating the program during the development and revision stages, while the summative evaluation is conducted at the end of the program.
Methods, models and tools used for developing E-Grader and Web-Grader

Figure 5.3: The Dick and Carey model (Dick & Carey, 1996:2,3).

The Dick & Carey approach has been through several iterations, and the version shown in Figure 5.3 has evolved from an initial base that was strongly behaviourist; such overtones are still evident.

As a result of its comprehensiveness, the Dick & Carey model has gained widespread acceptance and is seen as representing classic ISD. It has, however, been criticised for its assumption that what will work and not work (with regard to client needs, subject matter experts, and learners as well as the learning context) can be predicted with accuracy at the beginning of a development process. The model does not sufficiently allow for mistakes that commonly occur, or for human uncertainty.

Prestera (2002) suggests the rapid prototyping model as an alternative to the Dick & Carey model since it implements analysis, design, development and evaluation as an ongoing recursive process. The use of rapid prototyping for educational applications is discussed next.
5.3.3 Rapid prototyping for educational applications

Prototyping and rapid prototyping were introduced in Section 5.2.2 in the context of software engineering development models. In this section they are addressed with respect to the development of educational applications.

Rapid prototyping improves on the basic elements of the Dick & Carey model. In prototyping models analysis, design, development, and evaluation are integrated into an ongoing, recursive process that involves all the stakeholders. Rapid prototyping is both a systems model and a product development model, and it is gaining popularity in software development contexts due to its perceived benefits of improving quality, while simultaneously reducing development costs. Generally, rapid prototyping models involve end-users (learners in this context) and subject matter experts (SMEs) interacting with prototypes and instructional designers, in a continuous cycle of revision. Developing a prototype is the first step, while front-end analysis is generally reduced or converted into an on-going, interactive process between subject-matter, objectives, and materials. Evaluation is a recurring event that moves from global issues (navigation, structure, design, motif, colour scheme, etc.) to fine tuning of instructional content (e.g. semantics). This model makes course development and analysis simultaneous processes with evaluation loops recurring frequently throughout the life of the project (Prestera, 2002).

The merits of using rapid prototyping are discussed in Section 5.4, where it is proposed as an appropriate approach for E-Grader and Web-Grader.

5.3.4 Assessment of performance and instructional design

Means of assessment typically used in commuter-based learning systems are often inadequate. Prestera (2002) warns against instructional developers using exclusively multiple-choice, true-false, matching, etc. as a way of fitting the learning objectives to the test, rather than fitting the test to the learning objective. Although Prestera views rapid prototyping as the best model for instructional design, he advocates the Dick & Carey approach of writing test items before designing instruction. He believes that the design process should commence with a decision on how the system will assess performance as an effective way of ensuring alignment between the instruction and real-life performance goals.
5.3.5 Similarities between ISD models and software engineering models

Software design and instructional design are fields that have similar methodologies and purposes such as (Tripp & Bichelmeyer, 1990):

- In both fields, designers attempt to be systematic in approaching large, complex problems, and they attempt to bring orderly and replicable practices to disciplines which are dominated by individual practitioners.

- The waterfall model of software design (Section 5.2.1.1), and the Dick & Carey model (Section 5.3.1), represent two well-known models from the respective fields. The steps of the waterfall model (analyse, design, implement, test and maintain) are similar to the steps of the Dick & Carey model (analyse, design, develop, implement and evaluate).

- Rapid prototyping is a design methodology which is used successfully in both software design and instructional design.

- Both these disciplines have advocated the use of formative evaluation procedures in the development of systems, and they have to deal with similar constraints in planning, budgeting, and scheduling.

Tripp believes that the most fundamental difference between the two fields is the degree of rigour. Software designers deal with systems that are based on mathematical logic. Instructional designers deal in part with computer software, but primarily with systems based on human cognition, which entail more uncertainty and accept more ambiguity. According to Mayes & Fowler (1999) a purpose of instructional software is to make the learner think.

Based on the large number of similarities and the minor differences that exist, practitioners in the two fields often use similar models to create effective systems. As already stated, a method of software design that has been widely supported in the development of instructional design is rapid prototyping, which is proposed as an appropriate model for instructional design by Tripp and Bichelmeyer (1990), Wong (1993); De Villiers (1995); Chao(1998), Hoffman and Margerum-Ley (2002), and by Prestera (2002). It is discussed in depth in the next section.
5.4 The use of rapid prototyping for instructional software development

Rapid prototyping has been described in the context of software engineering development models (Section 5.2.2) and instructional systems development models (Section 5.3.3). In this section it is proposed as the most appropriate method for developing educational applications, and thus as the development method for the automated assessment tools of this study.

Rapid prototyping is proposed by Chao (1998) as an efficient and effective model for developing and testing educational materials. He states that rapid prototyping has become a common practice in the world of interactive multi-media design, where its popularity stems from the utility of testing prototypes on end-users to receive authentic feedback. This ensures better design and reduces the risk of abandoning a finished product due to its failure to meet user needs.

"Its strength is in the integral process of needs assessment, content analysis, objective construction, design and evaluation. It is also a combination of design and research, wherein the utilization of the prototype provides prescription to a particular instructional situation. The most valuable feature of this model is its inclusion of the end-users in the design process. In brief, if the instructional project is complex and interactivity is a key element, rapid prototyping yields better results because it is more pragmatic, efficient and cost effective" (Chao, 1998: http://www.arts.ualberta.ca/~tchao/rpwebsite/rpwhole.html).

Chao believes that in terms of cost associated with development and production, it is less expensive to build a prototype than to use a traditional design model that aims at an end product right from the beginning. As long as the strengths of rapid prototyping are understood, and its weaknesses can be overcome, it is possible to use this method to produce quality design in a shorter period and at a lower cost.
Chao (1998) puts forward the following as characteristics of rapid prototyping:

- **Integral process from needs assessment to evaluation:**
  Although there is a sequence of events in the rapid prototyping model, all these events can occur at the same time. In other words, needs assessment and objective construction can be conducted while creating a prototype, and design, evaluation and implementation can be carried out simultaneously.

- **Concurrent formative evaluation:**
  This model enables design and evaluation to merge and interact as the results of evaluation are formatively used to refine the design.

- **Pragmatic and prescriptive for real world projects:**
  Rapid prototyping enables the designer to consolidate design ideas quickly, communicate them directly to the users, and test them to determine the effectiveness of the instruction.

- **Efficient and economic production:**
  The development cycle can be much quicker with this model, and the risks and costs due to abandoning a final product can be avoided by testing and modifying a prototype.

- **User-centred methodology:**
  Chao views the inclusion of end-users in the development of a system as the most noteworthy characteristic of this model. Research on formative evaluation has proven that incorporating users’ feedback to revise materials has the most impact on improving learning outcomes (Chao, 1998; Black & Wiliam, 1998).

### 5.4.1 Advantages of prototyping

Hoffman and Margerum-Leys (2002) took the conclusions of Whitten, Bentley & Dittman (2001) and adapted them specifically to customise the advantages of prototyping in a systems engineering context to the instructional design environment. They present the following advantages of rapid prototyping:
Prototyping encourages and requires active user/learner participation in the design process.

Iteration and change are natural consequences of instructional systems development. Clients tend to change their minds.

Clients frequently do not know their requirements until they see them implemented.

An approved prototype is the equivalent of a specification on paper with one exception - errors can be detected earlier.

Prototyping can increase activity through faster user feedback.

Prototyping accelerates the development cycle.

This set of advantages clearly indicates the user-centred nature of development by prototyping.

5.4.2 Disadvantage of prototyping

The main potential disadvantage of prototyping is that it can encourage informal design methods which may introduce more problems. Hoffman and Margerum-Leys (2002) advocate caution, suggesting that this failure can be avoided if the following issues are borne in mind:

- Prototyping can lead to a design-by-repair philosophy, which is only an excuse for lack of discipline.
- Prototyping does not eliminate the need for front-end analysis, nor can it replace a paper-based analysis.
- There may be instructional design problems which cannot be addressed by prototyping.
- Prototyping may lead to premature commitment to a design. It should be remembered that a design is only a hypothesis.
- When prototyping an instructional package, creeping featurism (the adding of bells and whistles) may lead to designs that can get out of control.

Bearing the advantages and disadvantages of rapid prototyping in mind, this model was chosen for the design and development of the electronic grading systems (E-Grader and Web-Grader) that form part of this study. The development of the two variants is described in detail in the next two chapters.
5.5 Pressman’s generic view of software engineering

Pressman (1997) proposes a generic view of software engineering. He describes engineering as the analysis, design, construction, verification, and management of technical (or social) entities. Regardless of the entity to be engineered, the following questions must be asked and answered:

- What is the problem to be solved?
- What are the characteristics of the entity that is used to solve the problem?
- How will the entity (and the solution) be realised?
- How will the entity be constructed?
- What approach will be used to uncover errors that were made in the design and construction of the entity?
- How will the entity be supported over the long term when corrections, adaptations, and enhancements are requested by the users of the entity?

He believes that the work associated with software engineering, whatever the application domain, can be categorized into three generic phases, regardless of application area, project size, or complexity. Each phase addresses one or more of the questions above (Pressman, 1997):

1. **The definition phase focuses on what:**
   Key requirements of the system and software are identified. Methods applied during this phase will vary according to the software engineering model that is applied.

2. **The development phase focuses on how:**
   The way the data must be structured, the software architecture, procedural details, interfaces, how the design will be translated into a programming language, and how testing will be performed, are defined in this phase. Methods applied will vary, but three technical tasks should always occur: software design, code generation, and software testing.

3. **The maintenance phase focuses on change:**
   This phase focuses on change that is associated with error correction, adjustments required as the software's environment evolves, and changes due to enhancements brought about by changing customer requirements. The steps of the definition and development phases are re-applied in this phase, in context of existing software.
These generic phases proposed by Pressman are followed in the design and development of the *E-Grader* and *Web-Grader* software systems. Tools used for these developments and implementations are described next.

### 5.6 Tools used to develop and implement *E-Grader* and *Web-Grader*

*E-Grader* is a prototype system developed by the researcher to investigate how textual data can be assessed electronically. The first implementation of this system is a non-interactive system to test the feasibility of the researcher’s algorithm. The development and implementation of this first variant, called *E-Grader*, is described in Chapter 6. The ultimate intention is to implement it as an interactive system on the web server of Unisa’s School of Computing. The development and implementation of a prototype of this second variant, called *Web-Grader*, is described in Chapter 7.

Scripting tools such as PHP, and Linux text processing and editing tools, as well as shell scripts were used to investigate how textual data can be inspected by using regular expressions, or pattern matching. The next sections in this chapter provide general descriptions of these types of tools, while their specific application in *E-Grader* and *Web-Grader* will be discussed in Chapters 6 and 7 respectively.

#### 5.6.1 Regular expressions

The Ceilidh system used for automatic marking of programming assignments makes use of regular expressions\(^1\) (REs) to check that the output from a program represents the correct output, and whether the program source contains particular features (see Section 4.2.3.1).

---

\(^1\) Regular expressions were defined by Kleene in 1956. Well known names such as McCulloch and Pitts, McNaughton, Chomsky, Rabin and Scott made significant mathematical contributions to this field in the 1950s (Pretorius & Viljoen, 2001).
The software system developed for this study also uses the principle of regular expressions as a mechanism for recognising keyword combinations (i.e. pattern matching) in textual inputs. It is however, beyond the scope of this study to examine regular expressions in depth. General regular expressions are discussed briefly in this section to illustrate how they can be used for pattern matching. All the regular expressions used as examples are displayed in bold in this section.

5.6.1.1 What is a regular expression?

A regular expression is a pattern that describes a set of strings; a string being a collection of characters. Regular expressions can be used as a powerful tool for manipulating text and data or to test whether a string fits into a specific syntactic shape. In the present study they are used to search a textual input-string for substrings that fit a pattern (e.g. one or more keywords).

Regular expressions are constructed (analogously to arithmetic expressions) by using various operators to combine smaller expressions. Regular expressions can be found in:

- Operating system utilities (such as awk and grep).
- Scripting languages (including Perl, Tcl, awk, PHP and Python).
- Text editors (including Emacs, vi, and TextPad).
- Programming environments (including Delphi and Visual C++).

Several functions are built into PHP (see Section 5.6.3) for the purpose of matching a pattern. Before these functions can be used, a pattern which can be used by the function for matching purposes, must be defined. The following discussion on how patterns are defined is based on Ullman (2001).

5.6.1.2 Defining a simple pattern

Two types of characters can be used for defining simple patterns that can be used in textual searches - literals and metacharacters. This sub-section gives examples of literals and introduces the defined metacharacters, illustrating how they are used.
(1) Literals

A literal (or constant) is a value that never changes, and is interpreted exactly as it is written. For example, the literal (or pattern) `s` will only match the letter "s", and the literal `as` will only match "as", and so forth. Literals can be used to match exact combinations, such as to search for certain keywords in a text. However, the same can be accomplished by using a string-search function in a program (see Code fragment 5.1).

```
IF (((word = "utilise") OR (word = "utilize") OR (word = "utilisation") OR (word = "utilization")
    OR (word = "utilising") OR (word = "utilizing")))
```

Code fragment 5.1

Regular expressions, however, can be used for much more than mere literal matching. Instead of using a string-search function to search for a particular word and its explicitly-defined variations, a regular expression could rather be used to search for the range of variations (i.e. different ways of spelling or conjugating) by seeking occurrences of what the required terms have in common, along with acceptable options. For example, when searching for the variations of the keyword "utilise" in a text-string, the string-search function in Code fragment 5.1 containing the if-statement, could be replaced by the much shorter regular expression `utili(s|z)` to search for those words.

(2) Metacharacters

The matching capabilities of patterns are based on the concept of metacharacters. Metacharacters are special characters or symbols (such as *, +, ^, ., $ and ?) that have meaning beyond their literal value (see Example box 5.1).

```
t. will match any single character (a, b, c, d, etc.)
that is preceded by a "t", such as to, ta, tm, etc.
```

Example box 5.1
For example, if the metacharacter period ( . ) is placed after the literal t (e.g. t.), this regular expression will match any single character (a, b, c, d, etc.) that is preceded by a "t", such as to, ta, tm, etc.

Regular expressions make use of the pipe ( | ) as an *alternation operator* (the equivalent of or). Therefore a|b matches the character a or b, and gr(e|a)y matches both potential spellings of the colour gray (grey and gray) (see Example box 5.2).

```
utili(s|z)e matches utilise and utilize
optimi(s|z)e matches optimise and optimize
generali(s|z)e matches generalise and generalize
```

Example box 5.2

The following metacharacters:

- * (also known as the *Kleen star operator* or an *striks*),
- ? (questionmark), and
- + (plus)

are called *repetition characters*, and are used for multiple occurrences of a literal (see Example box 5.3).
a* matches zero or more a's (a, aa, aaa, etc.).
ha*t matches ht, hat and haat, etc.

a? matches up to one a (a or no a).
ha?t matches ht and hat, but not haat, etc.

a+ matches one or more a's (a, aa, aaa, etc., i.e. there should be at least one a).
ha+* matches hat and haat, etc. but not ht

Example box 5.3

The metacharacter s is used to match a space, and S is used to match any character.
A metacharacter should be preceded by a backslash (\) to indicate that it is not the literal "s", but the metacharacter s (see Example box 5.4).

time \s*\s*\s* share can be used to search for the word "time-share", "time share" or "timeshare" to accommodate different ways the word might be spelled

Example box 5.4

See Tables 6.1.1, 6.1.2, 6.1.3, 6.2.1, 6.2.2, and 6.2.3, and Text box 7.2 for examples of how such features were implemented in E-Grader and Web-Grader.

To match a certain quantity of a letter, the quantity is placed between curly brackets (\{\}), starting with either a specific number, a minimum, or a maximum (see Example box 5.5).
\( a(3) \) matches \( aaa, \ aaaa, \ \) etc. (three or more a’s).
\( a(3,5) \) matches only \( aaa, \ aaaa, \ aaaaa \) etc. (between three and five a’s).

**Example box 5.5**

The caret (\(^\wedge\)) will match a string that begins with the letter following the caret (see Example box 5.6).

\(^\wedge a \) will match any string beginning with an \( a \), succeeded by any other character, such as \( at, \ as \) and \( all \).

**Example box 5.6**

The dollar sign (\( $ \)) is used for matching any string that ends with the preceding letter in a regular expression (see Example box 5.7).

\( a$ \) will match any string ending with an \( a \), such as \( data, \ era, \ zebra, \) and \( arena \).

**Example box 5.7**

### 5.6.1.3 Defining more complicated patterns

Characters can be grouped together, or can be defined into classes to form more complicated patterns.

(1) **Character groupings**

Using the basic symbols established so far, parentheses can be incorporated in order to
group characters into more complicated patterns, and to restrict and control the pattern (see Example box 5.8).

Example box 5.8

\[
\begin{align*}
\text{a\{3\}} & \text{ matches } aaa \\
\text{(abc)\{3\}} & \text{ matches } abcabcacb \\
\text{bon+} & \text{ matches bonnet, bond and bonfire} \\
\text{(bon)+} & \text{ matches bonbon}
\end{align*}
\]

See Code fragment 6.1 and Tables 6.1.1, 6.1.2, 6.1.3, 6.2.1, 6.2.2, 6.2.3, for examples of how this was implemented in E-Grader and Web-Grader.

(2) Classes
Regardless of how letters are combined into various groups, they will only be useful for matching specific words. In order to be able to match any four-letter lowercase word, or any number sequence, character classes have to be defined and utilised.

Classes are created by placing characters within square brackets ( [ ] ) (see Example box 5.9).

Example box 5.9

- Any vowel can be matched with [aeiou].
- The hyphen ( - ) can be used to indicate a range of characters, i.e.
  
  \[
  \begin{align*}
  \text{[a-z]} & \text{ matches any single lowercase letter,} \\
  \text{[A-Z]} & \text{ matches any upper-case letter,} \\
  \text{[A-Za-z]} & \text{ matches any letter in general, and} \\
  \text{[0-9]} & \text{ matches any digit.} \\
  \text{[A-Za-z]}\{3\} & \text{ matches ABC, abc, COW, cow, etc.}
  \end{align*}
  \]
Within the square brackets, the caret symbol (\(^\wedge\)) , which is usually used to indicate an accepted beginning of a string, is used to exclude a character (see Example box 5.10).

\(^\wedge a\) will match any single character that is not an a.

Example box 5.10

5.6.2 Apache web server

The web server at Unisa's School of Computing is known as Osprey. It is an Apache server running on the Red Hat Linux operating system. E-Grader and Web-Grader were developed on Mandrake Linux 7.1 (which is fully compatible with Red Hat Linux), using the Apache HTTP Server Version 1.3. Mandrake Linux distributions are packaged with PHP, MySQL, and the Apache web server. Web-Grader was ported to the Osprey server after it was completely developed on a PC running on Mandrake Linux.

5.6.3 Linux operating system

Linux was discussed in detail in Section 3.2.2.4. Unix/Linux shells and shell scripts, text processing and editing tools (used for the development and implementation of E-Grader and Web-Grader) are discussed in this section, based on Welsh (1995), and Husain & Parker (1996).

5.6.3.1 Unix/Linux tools used for text processing

One of the most important features of Unix/Linux is the wide variety of utility programs. Various text processing utilities, such as grep, sed and awk (briefly mentioned in Section 4.2.3.1) enable searches on patterns (or regular expressions) instead of on fixed strings. These utilities are backbones of the Unix/Linux programmer's toolbox, and were used in the development of the prototype software systems for compiling keyword combinations, and manipulating files.
(1) grep

grep stands for *global regular expression and print*. The *grep* utility can be used to search a set of characters in a file (without first loading the file into memory) or in a directory, and to display all the lines that contain those characters. For example, the *grep* command in Code fragment 5.2 can be keyed in on the command line to investigate if the keyword (literal) "time share" can be found. The flag "-I" is used to search for lower case as well as upper case occurrences of the word in the files containing students responses to a certain question (represented in Code fragment 5.2 by "_3_5b" to indicate Question 5b of Assignment 3). This command will display the names of all the files in the current directory that contain the words "time share, timeshare, time-share, time shared, time-sharing, time-sharing", etc., as well as the sentences in which these words occur in files stored with a "txt" extension, and of which the filenames begin with "st" and end with "_3_5b".

```
grep -l "time.share" st*_3_5b.txt
```

**Code fragment 5.2**

The *grep* command including the "-c" flag of Code fragment 5.3 can be used to investigate how many occurrences of the word "time share" can be found in these files. Only the number of occurrences of the word will be displayed.

```
grep -l -c "time.share" st*_3_5b.txt
```

**Code fragment 5.3**

(2) sed

The *sed* utility is a *stream editor* for editing streams of text that might be too large to edit as a single file, or that might be generated on the fly as part of a larger data processing step. Most text editors load a file into a buffer before the text in the file can be edited. If the file is too large for the buffer, the editor cannot be used to change the text. This limitation is not a problem for the *sed* utility, because it does not load the entire file into
memory, but reads one line at a time from the file. The most common operation done with sed is substitution (i.e. replacing one block of text with another).

The sed command can be used in combination with the awk command (discussed in the next sub-section) to manipulate a file. The sed commands in Code fragment 5.4 can be used to convert an output-file produced by E-Grader from a text format file containing redundant data to a database file containing only the student number and the mark obtained. This is done in the following way:

- **E-Grader**'s output is stored in a file named result (see Example box 5.11) after batch-processing all the answers.
- All occurrences of "st" in result are then replaced by " " (space) and is piped (using " > ") to a new file named temp (see Example box 5.12).
- All occurrences of "_3_5b" in temp are then replaced by " " (space) and is piped to a new file named temp2 (see Example box 5.13).
- All first occurrences of "," (commas) in each line of temp2 are then replaced by "." (space) in temp3 (see Example box 5.14).
- Next, all occurrences of "," in temp3 are replaced by "," (a space and a comma) and piped to a file called temp4 (see Example box 5.15).
- Only the second and fifth fields from the database file temp4 are then piped to a new file named result3_5b in order to produce a comma-delimited file (see Example box 5.16) containing only the student number and mark obtained by E-Grader, separated by a comma.

```bash
sed 's/st/ /' result > temp
sed 's/_3_5b/ /' temp > temp2
sed 's/,/ /' temp2 > temp3
sed 's/,/ /' temp3 > temp4
awk '{print $2,$5}' temp4 > result3_5b
```

Code fragment 5.4
Example box 5.11: File named “result”

Example box 5.12: File named “temp”

Example box 5.13: File named “temp2”
Example box 5.14: File named "temp3"

Example box 5.15: File named "temp4"

Example box 5.16: File named "result3_5b"
(3) awk
The awk utility is a pattern scanning and processing language developed by Aho, Weinberger and Kernighan, hence the name “awk” using the first letters of each name. Unlike many conventional languages, awk is data driven. By specifying the kind of data, and the operations to be performed, awk can be used to open and close data files automatically, read records, break-up records into fields, and to count records. While awk provides the features of most conventional programming languages, it also includes some unconventional features, such as extended regular expression matching and associative arrays.

The awk utility is one of the most popular tools that programmers use to search a database file. It consists of two parts - a search pattern, and the data that needs to be displayed. The awk command in Code fragment 5.4 can be used to search the database file called temp4 and print only the second and fifth fields (containing the student number and mark obtained) to a comma-delimited file called result3_5b.

Alternatively, instead of using the code in Code fragment 5.4, awk and sed can be combined into one single instruction using Linux/Unix’s pipe (|) mechanism (see Code fragment 5.5), thereby eliminating the use of four intermediary files (temp, temp2, temp3 and temp4) to produce the file result3_5b.

```bash
sed 's/a/a/ //s//3,5b/ //s// /s// /s// result|awk '{print $2,$5}'>result3_5b
```

Code fragment 5.5

5.6.3.2 Unix/Linux shells and shell scripts
Unix/Linux has various different commandline-based user interfaces called shells. A shell is a program that reads and executes commands from the user. It is a command language interpreter, and to many users, it is the most important utility provided by the operating system. Most shells provide the following features:
Methods, models and tools used for developing E-Grader and Web-Grader

- Job control (enabling the user to manage several running processes at once);
- Input and output redirection, and
- A command language for writing shell scripts, where a shell script is a file containing a program in the shell command language (analogous to a batch file under MS-DOS).

Unix/Linux shells can be text-based, as well as window-based (see Section 3.2.2.4). There are many types of text-based shells available for Unix and Linux, and all the popular shells fall into two groups - the Bourne shell and the C shell. The Bourne Again Shell (bash) called bash was used in the development and implementation of E-Grader and Web-Grader. This is a Bourne shell variant which includes many advanced features, such as job control, command history, command and filename completion, and an Emacs-like interface for editing the command line. It is a powerful extensions to the standard Bourne shell language.

The choice of a shell is often based on the command language that it provides. The shell provides many mechanisms to customise the work environment and is more than a command interpreter - it is also a powerful programming language. When a series of commands is frequently used, and a user would like to shorten the amount of required typing, the series can be grouped into a single command. For example, the shell script in Code fragment 5.6 can be used to run all the commands in Code fragment 5.4, and additionally print the data, and save the data on the Windows partition of the PC, from where it could later be imported into a spread sheet. This script can be saved in an executable file called makedata.
Shell scripts are plain text files that are created with an editor such as **emacs** or **vi**.

Text editing tools are discussed next.

### 5.6.3.3 Unix/Linux tools used for text editing

A text editor is a program used to edit files which contain text, such as documents, programs, or system configuration files. While there are many such editors available for Unix/Linux (such as **ed**, **vi**, **ex**, **pico**, **jove**, **joe** as well as GNU **Emacs** and variants such as **Lucid Emacs**, which incorporate extensions for use under **X Windows**), the only editor which is guaranteed to be found on any Unix or system is **vi**, the "visual editor". This is not a user-friendly editor, and due to its age, it has many limitations, therefore more modern editors such as **joe** and **Emacs** are gaining popularity. Although the **vi** editor has some limitations, it was used to create shell scripts, and to edit the programs written in PHP for the development of **E-Grader** and **Web-Grader**.
5.6.4 Data validation using PHP: Hypertext Preprocessor

PHP version 3.0.16 was used for the development of E-Grader and Web-Grader. PHP was discussed in detail in Sections 3.2.2.8 and 3.4.2.2. The way PHP uses regular expressions to validate data is discussed here.

The use of regular expression functions in PHP is the most powerful way to validate data (Schwendiman, 2001). PHP supports two types of regular expressions - Perl style and POSIX style.

5.6.4.1 Perl-style functions

According to Medinets (2000), Perl appears to have the most flexible set of pattern matching functions. Many of these features have been duplicated in PHP. The following PHP functions are Perl-style function, and all of them were used in the development of E-Grader and Web-Grader:

- `preg_match` searches a piece of text for all non-overlapping instances of a pattern, and returns the number of matches found, otherwise, `false` is returned. This function was used in E-Grader to search for keywords.
- `preg_split` breaks up a piece of text based on a delimiter, and returns an array. This function was used in E-Grader to convert strings into arrays.
- `preg_replace` searches a piece of text for all instances of a pattern, and replaces any matching text with another pattern. This function was used in Web-Grader to display keywords in the feedback - regular expressions were replaced with spaces in the display.

5.6.4.2 POSIX style functions

POSIX is an acronym that means Portable Operating System Interface for Unix. It is a series of IEEE and ISO standards that defines interfaces between programs and operating systems. Using these standards helps programmers to create programs that can be ported to most varieties of Unix, as well as some versions of Windows. POSIX standards are developed by the Portable Application Standards Committee (PASC). The following PHP functions are POSIX-style functions:
• `ereg` returns a value of "true" if its pattern parameter is matched, and "false" otherwise.
• `eregi` is identical to `ereg` except that case distinctions are ignored.
• `ereg_replace` which replaces text matching a pattern.
• `eregi_replace` which is identical to `ereg_replace` except that case distinctions are ignored.
• `split` separates a string into parts according to its delimiter parameter.
• `sql_regcase` creates bracket expressions from its parameter so that case-insensitive SQL statements can be generated. For example, `sql_regcase ("ab")` returns "[aA][bB]".

5.7 Methodology and tools used to extend E-Grader to Web-Grader

The goal of this study was to generate a tool that could conduct automated assessment of learners' textual inputs. Two variants were developed, which differ in two respects - namely the way in which textual data is input, and also in their functionality, i.e. their outputs after processing the learners' answers.

The first variant, E-Grader, is a prototype system developed to grade answers to short open-ended questions. E-Grader simply assigns a mark (numerical score) out of a given maximum. The maximum scores vary between 3 and 7. By contrast, the second variant, Web-Grader, is an interactive web-based extension of E-Grader which provides interactive student support. It reads in learners' responses to open-ended questions, assigns a mark (as done with E-Grader), and furthermore offers feedback on inadequacies in the learners' answers.

The methodology and tools used to extend E-Grader to Web-Grader are discussed in this section.
5.7.1 Methodology used to produce Web-Grader

The two major paradigms of instructional design, namely behaviourism and constructivism, were introduced in Chapter 2. Models used by these two major schools of theory were discussed in Section 5.3. None of these ISD design models deal specifically with the design of web-based instruction (WBI). McManus (1996) believes that web-based instruction (WBI) can be designed under either paradigm, and that many aspects can be extrapolated from these models in creating instructional systems for the WWW. This section briefly describes the methodology used to produce Web-Grader, while the full design process is set out in Chapter 6.

Bartas and Palumbo (1995) believe that hypermedia (discussed in Section 3.2.2.7(4)) have much to offer learners in terms of providing a constructivist environment that engages the learner, while allowing construction of knowledge in a meaningful way through the association or linking of pieces of information. They believe that hypermedia, when structured into learning environments that motivate guided inquiry, have the potential to develop better user metacognition than linear media. Although transfer of learning has been a debatable issue over the years, it appears that experience with hypermedia systems could improve learners’ ability to effectively construct knowledge that, as a process, could be transferred across learning domains.

Using the rapid prototyping model for developing hypermedia systems

Similarities between ISD models and software engineering models were discussed in Section 5.3.5, and it was found that both these disciplines have advocated the use of formative evaluation (the ongoing process of evaluating the system during the development and revision stages) as the procedure to enhance the development of systems.

According to Chao (1998) rapid prototyping has become a common practice in the design of interactive multi-media and hypermedia systems. Its popularity is due to the testing of the prototype on end-users in order to receive authentic feedback. This ensures a better design, and eliminates the risk of abandoning a finished product due to its failure to meet the users’ needs. Therefore, in terms of cost associated with development and production, it is less expensive to build a prototype than to use a traditional instructional design model that aims right from the beginning at producing a complete product.
Chao states that building a web site is a complex task. Information provided on a web site should be useful, and the structure should make sense to the users. She believes that users' interaction with the site is crucial, because the requirements and goals may not have been clear initially. Therefore, the web site should be tested by users, and be modified according to their requirements. She proposes rapid prototyping as the appropriate model for multi-media development, because more than one kind of media (text, graphics, sound, and video) are involved, which could be an expensive process to revise.

Design and development of Web-Grader were merged by using a user-centred methodology (see Section 5.4). This prototype system was implemented on the WWW, and students were encouraged to use it as a study support tool (discussed in Chapter 7). Web-Grader was produced by extending E-Grader with a graphical user interface, providing interaction and feedback, and by adding a database connection. Tools used for producing Web-Grader are discussed next.

### 5.7.2 Tools and methods used to produce Web-Grader

Various tools and components of web design are overviewed here, while the full design process of Web-Grader is described in Chapter 7. The main differences between E-Grader and Web-Grader are summarised in Table 5.1:

<table>
<thead>
<tr>
<th>E-Grader</th>
<th>Web-Grader</th>
</tr>
</thead>
<tbody>
<tr>
<td>Commandline interface.</td>
<td>Graphical user interface (GUI).</td>
</tr>
<tr>
<td>A non-interactive system that executes the answers in a batch and stores grades collectively in a file.</td>
<td>An interactive system that provides immediate feedback in the form of grades obtained and clues on how to improve the grade. On-line communication was included by providing an e-mail facility, as well as a public notice board.</td>
</tr>
<tr>
<td>Six separate programs were used (one for each question) and keywords were hard-coded in each program.</td>
<td>Only one program was used for a keyword search. Learning content and keywords were provided in a relational database.</td>
</tr>
</tbody>
</table>

**Table 5.1: Differences between E-Grader and Web-Grader**
Tools used for adding a graphical user interface (GUI) and components of a GUI are discussed in Section 5.7.2.1. Interaction, feedback and communication are discussed in Section 5.7.2.2. The way in which learning content is provided in a relational database is briefly explained in Section 5.7.2.3.

5.7.2.1 Tools and components of a graphical user interface

The primary focus of website design is the front end, or user-interface, i.e. the pages and functions users see on their screens. Graphical user interfaces (GUIs) were discussed in Section 3.2.2.7(4). A website can be seen as a series of screens that the user sees, and the overall flow of screens and messages is called a dialogue. This dialogue can be modelled by using a dialogue chart (see Section 7.2.2.2 for the dialogue chart used for designing Web-Grader).

1. GUI components considered for use in Web-Grader

Graphical user interface controls

A GUI usually includes user interface controls such as text boxes, buttons, radio buttons, check boxes, list boxes, drop-down lists, etc. These controls normally contribute to the selection of or input of data.

- **Text boxes**

  A text box is the most common control used for user-input of data. It consists of a rectangular box into which the user enters data via the keyboard. Single or multiple lines of data characters can be entered into a text box. In Web-Grader, text boxes are appropriate for entering usernames and passwords, as well as for entering answers to questions.

- **Buttons**

  A text-input box is useless without a <Submit> button that enables a user to commit data to be processed. Other buttons offer the facility to cancel a transaction, or to get help. Web-Grader was not designed to provide a help function. Submit buttons are required to submit usernames, passwords, and other responses entered into text boxes.
Radio buttons, check boxes, list boxes and drop-down lists were not used in Web-Grader. These controls normally require a user to select a data item's value from a list of possible choices. Since Web-Grader was designed to assess textual inputs, such controls are not required.

**Menus**

The menu is an important component of the user interface. Menus provide an easy-to-use visual interface that enables the user to browse and select an item from a list of choices or commands that the application provides, rather than having to recall the commands, options or data from memory. Menu selection is the oldest and most commonly used dialogue strategy (Whitten, Bentley & Dittman, 2001; Marcus, Smilonich & Thompson, 1995).

Various types of menus can be found on GUIs, such as pull-down and cascading menus, tear-off and pop-up menus, iconic menus and toolbars, as well as hypertext and hyperlink menus. Menu systems considered for use in the design of Web-Grader were:

- **Iconic menus and toolbars**

  Iconic menus use pictures to represent menu options. Toolbars consist of a row of icons (small clickable pictures) that present menu shortcuts for actions and commands.

- **Windows and frames**

  A window is a basic construct of a GUI. It is a rectangular, bordered area, which can be smaller or larger than the display monitor's viewable area. It usually includes standardized controls in the upper right-hand corner to maximize itself to the display screen's size, minimize itself to an icon (at the bottom of the screen), and an exit (or close) option (see Section 7.2.1.2 for how windows are implemented in Web-Grader).

A window may be divided into zones called frames. Each frame can act independently of the other frames in the same window, using features such as paging, scrolling, display attributes, and colour. Each frame can be defined to serve a different purpose. Frames are commonly used in windowing systems and in web-browsers (Marcus, Smilonich & Thompson, 1995).
Dialogue boxes

Dialogue refers to the communication between the user and the computer. A dialogue box is one of the many ways in which the user communicates with the computer. "It is an observable two-way exchange of symbols and actions" (Marcus, Smilonich & Thompson, 1995:147). It is a type of a window that presents choices to the user, and provides a graphical means to input or display information. It supplements the interaction in the primary (main) window, and it is considered by the GUI as a secondary (supplementary) window. A dialogue box usually suspends the application until the user either provides the required information, or cancels the operation (Marcus, Smilonich & Thompson, 1995).

Hypertext and hyperlink menus

Hypertext and hyperlinks (see Section 3.2.2.7) were originally created to navigate within web pages and sites. This technology can easily be extended to implement menus in web-based applications. Each menu-item is a hypertext phrase (or hyperlinked icon or button) that invokes actions or forms on other web pages (see Section 7.2.1.2 (2) for how this is implemented in Web-Grader).

(2) Publishing tools for graphical images

Graphical images for a web site can be produced from:

- existing digital photographs or videos,
- scans of hard copy photographs or artwork,
- free images and animations found on the WWW or on compact discs (CDs), or
- newly created by using a graphics program (see Section 3.2.2.7(4)).

The use of graphics on the WWW was discussed in Section 3.2.2.7. Adobe PhotoShop 6, the graphics tool used to produce images for Web-Grader, is an image-editing application with various tools and utilities for working on digital images or bitmaps. A bitmap image consists of a rectangular grid (or raster) of pixels\(^2\), similar to a mosaic. When editing a bitmap, the colour values of individual pixels or groups of pixels are being edited.

\(^2\) A pixel is short for "picture element", and is the smallest element in a bitmap image.
Image-editing applications differ fundamentally from vector-based applications (such as Adobe Illustrator and Macromedia FreeHand) which work with objects that can be moved, scaled, transformed and deleted as individual or grouped objects, while each object exists as a complete, separate object. Each object is defined by a mathematical formula, and therefore they are resolution-independent, i.e. vector drawings can be scaled up or down and will still display smoothly and crisply. In contrast, bitmaps are created at a set resolution (a fixed number of pixels per inch). If an image is created at a specific resolution, and is then scaled up to double the original size, its resolution is effectively halved. This is likely to produce a block, jagged image, because the size of the individual pixels that make up the bitmap image, has been increased (Shufflebotham, 2001).

Clip art pictures obtained from various CD collections and the WWW were used for Web-Grader, and had to be converted into a web-format (see Section 3.2.2.7(4)) by using Adobe Photoshop 6.

JPEG, GIF and PNG are compression formats widely used for saving images for the WWW (see Section 3.2.2.7(4)). Because the PNG format cannot be used by the older browsers, it was decided to use only GIF and JPG formats. Most of Web-Grader's clipart images used were in a windows metafile (WMF) format, and required conversion to GIF and JPG formats. Adobe Photoshop cannot import files in a WMF format, so Corel Presentations was used as an intermediary system to display the images, that were then "copied and pasted" into Adobe Photoshop, where they were optimised and saved into a web-format.

5.7.2.2 Interaction and feedback

(1) Designing interaction and feedback for assessment of learning

Interaction means the collaboration between a user and the computer in order to perform a certain task (Marcus, Smilonich & Thompson, 1995). On-line assessment was discussed in Section 3.2.1.4, with special reference to the distinction between summative and
formative assessment. The use of more interactive web-based learning environments was proposed in Section 3.4.2 as a solution to some of the problems (such as problems with constructivism and assessment) discussed in Section 3.3.3.

The terms formative assessment and feedback have occurred frequently in this study. In this section, where the design and development of feedback and interaction with Web-Grader is discussed, it is necessary to further explore and clarify these concepts. Formative assessment of learners occurs when information is fed back to them in ways that enable them to improve the learning process, or when learners can engage in a similar, self-reflective process. Research on formative assessment (Weston, 1997 as cited Chao, 1998) in has proven that incorporating users' feedback to revise materials has the most impact on improving learning outcomes.

Black and Wiliam (1998) believe that assessment is part of the learning experience, and that learners should be active in their own assessment to picture their learning in the light of what it means to improve. The purpose of learning is to close the gap between achievement and desirable goals.

Formative assessment should provide feedback to learners on their progress during a course, in ways that enable them to improve their learning. Black & Wiliam believe that the core of formative assessment lies in two actions:

- the perception by the learner of a gap between desired goals and his/her present state (of knowledge, understanding, and/or skill), and
- the action taken by the learner to close that gap in order to gain the desired goal.

Black and Wiliam believe that for assessment to be formative, feedback information is essential. The quality of the feedback provided is the key feature. The instructional effect of feedback was reviewed by Bangert-Drowns et al (1991) as cited in Black and Wiliam (1998). They found that the quality of feedback has the greatest impact on performance, and that feedback was most effective when it was designed to stimulate correction of errors.
The term feedback was originally used to describe an arrangement in electrical and electronic circuits whereby information about the level of an output signal (specifically the gap between the actual level of the output signal and some defined reference level) was fed back into one of the system’s inputs. Where the effect of this was to reduce the gap, it was called negative feedback, and where the effect of the feedback was to increase the gap, it was called positive feedback. In applying this model to the behavioural sciences, Black and Wiliam (1998) identify four elements making up the feedback system:

1. data on the actual level of some measurable attribute,
2. data on the reference level of that attribute,
3. a mechanism for comparing the two levels, and generating information about the gap between them, and
4. a mechanism by which the information can be used to alter the gap.

Ramaprasad (1983) as cited in Black and Wiliam (1998) defines feedback as information about the gap between the actual reference levels that are used to alter the gap in some way. If the information is not actually used to alter the gap, then it is not feedback. Black and Wiliam believe that instructional feedback should be designed to encourage, to be private, and to link to opportunities for improvement. Feedback should support the view that mistakes are part of learning.

This view of formative assessment corresponds with Plowman’s model for guided interaction (a requirement for interactive software), and her theory that the interaction process should be able to elicit and encourage existing competences, that it should also be adaptive, and should offer “scaffolding” for extending learning (discussed in Section 3.4.2.1).

The principles introduced in this sub-section were followed in designing interaction and feedback in Web-Grader. Feedback was designed, aiming to encourage the learner to redo an answer in order to obtain a higher mark, i.e. to reduce the gap between present achievement and a desired goal. The researcher also aimed to incorporate the four elements identified by Black and Wiliam in the feedback.
(2) Communication tools
Asynchronous communication (electronic mail, news groups, listgroups and bulletin boards) and synchronous communication (computer conferencing, chat, ICQ, and IRC) tools used as part of web-based learning were discussed in Section 3.2.2.1 Asynchronous communication facilities such as e-mail, and a bulletin board were included in Web-Grader to provide for time-independent interaction between students, and between students and the lecturer.

5.7.2.3 Publishing learning content

The use of technology to enhance interactive web-based learning was discussed in Section 3.4.2. It was stated that the main reason for using a server-side scripting language (such as PHP) is to provide dynamic content to a web site's users, i.e. content that changes according to a user's needs or inputs.

(1) Database management system used for Web-Grader
Web-Grader was designed to display learning content that is stored in the MySQL relational database (discussed in Section 3.4.2.2), in contrast to E-Grader where separate programs were written, one for each set of question-responses to be graded, and keywords were hard-coded into each program. This feature enables the system to serve as a shell and use only one program for all the different questions. The system can therefore be ported context-free to any other database using similar tables with similar structures, containing other questions, model answers, keywords, and total marks.

(2) Remote access tools used for publishing and maintaining Web-Grader
Web pages can be created on-line, or on an off-line computer from where they are uploaded on to a web server. This process of uploading web pages can be performed in several different ways, by using remote access tools such as TCP/IP, HTTP, FTP, and Telnet (discussed in Section 3.2.2.2). Web-Grader was developed off-line on a personal computer (PC), and FTP was used to upload the programs and images from the PC where it was developed to the Osprey server on which the web site was implemented. Telnet was used to maintain and update the programs on the Osprey server.
5.8 Conclusion

A variety of different process models used for software engineering and for instructional design were discussed in this chapter, each exhibiting strengths and weaknesses. Most of them are built around the generic phases: analysis, planning, design, implementation, integration, maintenance and retirement. Some of the models execute these phases in a sequential and linear fashion to deliver the system only after the linear sequence is completed. Others are iterative and produce increasingly more complete versions of the system, their underlying assumption being that it is not wise to attempt to design a complete software system in one attempt.

It was noted that the disciplines of software design and instructional design have certain similar methodologies and purposes. Rapid prototyping is a design methodology used in both software design and instructional design, which allows the designer to quickly build a working prototype of a system, test and modify it, and complete the final product through an iterative and evolutionary process. This was the approach chosen in this study for the development of a software system to assess short-answer free response items.

Systems and tools (such as Linux, the Apache server, regular expressions for pattern matching, scripting tools, and text editors) selected for the software development of E-Grader and Web-Grader were discussed in this chapter. The chapter thus moved away from the general background (set out in Chapters 2 to 4) towards the specific environment, methods and tools used for the development of two prototypical variants. The scene is set for Chapters 6 and 7 that focus directly on E-Grader and Web-Grader. In these chapters, the design and development process, implementation and evaluation of these systems are described according to Pressman's generic view of software engineering (as set out in Section 5.5) and Plomp’s development research model (introduced in Section 1.6.1).
References - Chapter 5


Chapter 6:

E-Grader: development, implementation and evaluation

6.1 Introduction

Plomp’s (2002) development research model, introduced in Chapter 1 as Figure 1.1, and reproduced here as Figure 6.1, represents a cyclical process that incorporates:

- a design process,
- an intervention, and
- consequent outcomes.

![Diagram](image)

Figure 6.1: Development and research model (Plomp, 2002).

The research undertaken in this study culminated in development and application in line with Plomp’s model, the phases of which are used to structure both this chapter and the next. It includes only immediate outcomes, i.e. distant outcomes as indicated in Plomp’s model are not included.

Section 6.2 describes the design and development process of the first variant of the software system, namely E-Grader. The discussion of this process, in turn, is structured under the first two
phases of Pressman's (1997) generic phases of software development described in Section 5.5 namely:

- Definition phase.
- Development phase.
- Maintenance phase.

In other words, the content of this chapter is structured according to Plomp's model, and then Pressman's phases within Plomp's processes.

Pressman's third phase, the maintenance phase, which focuses on change, is not discussed here, but is included in Chapter 7, where the extension of E-Grader to produce Web-Grader is described.

In Section 6.3 the first operational implementation of E-Grader and the outcomes of this intervention are discussed. In contrast to the traditional manual marking of assignments at Unisa, this system received input from text-files containing electronic versions of students' textual answers, and assessed them electronically by E-Grader. The results were stored in a comma-delimited text-file (see Section 5.6.3.1(2) and Example box 5.16). The process is shown in Figure 6.2.

![Figure 6.2: How E-Grader works](Image)

The prime purpose of E-Grader is thus to facilitate the task of the lecturer by automating and reducing work load.
6.2 Design and development process of the E-Grader prototype

The development of E-Grader was conducted in line with Pressman’s view of software engineering introduced in Section 5.5. The definition phase that focuses on what was built, and the development phase that focuses on how it was built, are described in this section.

6.2.1 Definition phase: What was built?

Key requirements of the system and software are identified in the first phase (see Section 5.5). This phase addresses the questions (Pressman, 1997):

- What is the problem to be solved?
- What are the characteristics of the entity that is used to solve the problem?

Step 1: Analysis - the real-world problem

In the previous chapter (Section 5.4.2) caution was expressed that prototyping does not eliminate the need for front-end analysis, and that a prototype is not a substitute for a paper analysis. With this in mind, the first step in the definition phase of the new software system was an analysis of the current assessment system used at Unisa in order to determine the need for alternative mechanisms.

Identifying the nature of the problem to be solved

Many tertiary institutions worldwide are experiencing problems in dealing not only with increasing student numbers, but also increasing costs in the form of additional teaching resources required per student. This has resulted in a growing need among teaching institutions for alternative teaching and assessment methods to address these increases.

To date Unisa’s response to this issue has not been very effective. The Students On-line (SOL) system (see Section 1.4.1) entails mainly a continuation of Unisa’s traditional approach of acquiring paper copies of assignments for manual assessment. The main
difference between traditional correspondence procedures and transmission procedures via SOL is the factor of time. Under the traditional system students post assignments in the mail, whereas with SOL, students submit assignments electronically. These assignments are then printed at Unisa and physically distributed to lecturers for manual marking along with the conventionally mailed assignments. This approach does not address Unisa’s requirement for alternative means of assessment itself, so as to reduce the load on lecturers.

Students studying at Unisa experience problems that are unique to distance education environments. These include:

- Studying in isolation.
- A lack of immediate feedback on material submitted to the university.
- Coping simultaneously with demands from different and competing environments (job-related, personal and study-related).
- A need to study at their own pace according to their own time schedule.

**Investigating the current assignment grading system at Unisa**

The functioning of the present Unisa system used for the assessment of assignments was investigated in order to identify the nature of the problem, and to outline the proposed new development. This section should be read while referring to Figure 6.3 which graphically represents the assessment and recording process.

Unisa is a distance education university with over 127 000 students. Tuition is handled mainly by correspondence. Study material is mailed to students in a paper-based format, and can also be downloaded electronically from the web site of the School of Computing (called Osprey) as well as from the SOL system. As already stated, students can submit their assignments by mail or electronically via SOL.

Assignments are graded in two ways at Unisa - *hand marked* and *marked on-line*. 
Figure 6.3: The current assessment and recording process at Unisa
(1) **Hand-marked**

Manual assessment is done both by professional lecturers and by so-called "external markers", contract consultants who are trained by lecturers to undertake some of the marking. Assignments submitted electronically via SOL do not receive immediate feedback. They are printed at Unisa's Assignment Section and are hand-marked by lecturers and/or external markers together with the assignments submitted by mail. Turnaround time frequently exceeds four weeks. Figure 6.3 illustrates how assignments received via SOL are integrated with those received in the post. After assignments have been graded, the covering docket, which also serve as mark sheets, are sent to the Assignment Section where the marks/grades are entered into the student database. The actual assignments are taken to Unisa's Dispatch Section, where they are posted back to students via mail. Marks are also available on-line.

(2) **On-line**

Two on-line marking systems are available at Unisa - one for electronic marking of assignments with multiple-choice questions, and another system (in limited use) where assignments containing open-ended answers are marked on the computer by the lecturer.

(2a) The current on-line assessment environment used for automated marking of multiple-choice assessments, marks students' responses electronically by comparing them with templates prepared by lecturers, and stores grades directly in the student database. Students do not receive immediate feedback, and usually wait three or more weeks for their results which are posted by mail. However, they are also published on the SOL web site for faster access, as indicated in Figure 6.3.

(2b) The other system enables the lecturer to mark written assignments on-line, and is currently in an experimental phase. These assignments are not graded automatically, but are edited by the lecturer who edits the student's work and inserts comments via the computer. The lecturer enters the student's grades on the electronic cover page of the assignment, and this information is saved on the student database.
These on-line systems eliminate the handling of assignments in paper format, which speeds up turnaround time. However, students do not receive immediate diagnostic feedback on their work and the nature of their errors, nor do they have the opportunity of working on-line.

**Step 2: Requirement - a new system for grading assignments**

This study proposes extension of the current system by including an electronic grading system into the current system to conduct accurate assessment of open-ended textual input. Figure 6.4 indicates the *E-Grader* extension on the upper right hand side. The technology should update databases and ideally provide students with instant, automated feedback which is both detailed and diagnostic (see Chapter 7). Such a system would bypass the tedious, time consuming and expensive manual processes described above, and would eliminate the need for human external markers.

Section 4.2.2.3 described research on three different models used for assessment of essays (two models based on LSA, and a third model using a simple keyword matching mechanism). Results indicated that the performance of these three models were very similar, and that a simple keyword match, excluding artificial intelligence techniques, performed surprisingly well. The researcher set out to design and build a prototype system using a simple keyword search in order to test the algorithm and the feasibility of the concept of e-assessment of open-ended textual responses.

**Step 3: Identifying subject matter and a target group for a prototype**

Appropriate content and an associated group of students had to be identified prior to developing a prototype. COS321-6 is a 3rd-level Computer Science module on Operating Systems and Architecture. This module presents an overview of, and introduction to, operating systems, process management, storage (memory) management, distributed systems, and protection and security. Students have to submit three assignments during the year, each of which is scheduled by a specified due date. Each assignment carries a specified number of credits. To gain admission to the examination, students have to accumulate a total of at least 100 credits for the three assignments together. During 2001, 192 students were enrolled for COS321-6. The researcher, one of the lecturers on the
Figure 6.4: An electronic grading system included into the current system to conduct assessment of open-ended textual input
COS321 team, identified this module as an appropriate and convenient domain within which to conduct research on e-assessment. The target group for the study comprised the students who submitted their assignments (written in English) electronically via SOL during 2001, a total of 45 students, i.e. approximately 25% of the student body. The textual responses selected for electronic analysis and validation came from six questions taken from two assignments.

6.2.2 Development phase: How was the system built?

In the development phase the following decisions are made (see Section 5.5):

- Structure of the data.
- The software architecture, procedural details.
- Interfaces and interaction style.
- How to translate the design into a programming language.
- Testing mechanisms.

According to Pressman (1997), the methods applied in the course of this phase will vary, but three technical tasks should always occur, namely, software design, code generation, and software testing.

The development phase addresses the questions (Pressman, 1997):

- How will the entity (and the solution) be realised?
- How will the entity be constructed?

Rapid prototyping (see Section 5.2.1.2) was used as the development methodology for the E-Grader prototype system. The tools used to built this prototype (Linux, Apache web server, shell scripts, text editors, PHP and regular expressions) were discussed in Section 5.6.

Step 4: Feasibility testing of the algorithm

The next step undertaken by the researcher, was to design a basic algorithm, translate it into a programming language, and test the feasibility of the concept of electronically grading open-ended textual data.
The idea of writing test items before designing instruction (Dick & Carey's 1996 model) and also proposed by Prestera (2002) (see Section 5.3.4) was applied. The very first version of the prototype was constructed to test design basic concepts independent of the context of Unisa assignments. Concepts were tested by assessing answers to the simple question: "Describe what a rainbow is", worth 16 marks. This prototype conducted a simple keyword search, being designed to award a mark for each specific aspect once only, so as to ensure that a keyword (for instance "red") could not be entered seven times to gain seven marks when listing the colours of the rainbow. The system was tested by using hand-crafted test data. A search was conducted on the keywords: reflect, spectrum, seven, colour, yellow, indigo, orange, red, green, blue, violet, 40 (or forty) degrees, 42 (or forty two) degrees, drop or water, refract or bend or bent, and sunlight (see code fragment 6.1), counting one mark each.

The search mechanism used was PHP's Perl-style function preg_match ("/$pattern/l", $target) (see Section 5.6.4.1) where $target was a file containing the input-answer as a response to the question. Perl-style functions require that parameters be enclosed by pattern delimiters which mark the beginning and end of a given pattern. The standard delimiter for Perl-style patterns is the forward slash ("/" ) character. However, alternative pattern delimiters could also be used.

```php
$pattern = array("reflect", "spectrum", "seven", "colour", "yellow", "indigo", "orange", "red", "green", "blue", "violet", "40\{forty\} degrees", "\{42\{forty two\} degrees\}", "drop\{water\}", "refract\{bend\{bent\}\}", "sunlight");
```

Code fragment 6.1

A counter was programmed to count all occurrences of the keywords in order to produce a total score.

The initial prototype was expanded to a subsequent version to ensure that marks were not awarded for using "... not..." before a keyword (for example "not red", "not blue" etc.), i.e.
limited contextualisation was introduced to ensure that a negation of a concept would not be viewed as an occurrence of that concept.

In the continued evolution of E-Grader, a prototype web-based version of this system was developed, which displayed on the screen:

- the question to be answered,
- maximum marks,
- an input form for entering the answer
- a facility to submit the answer
- keywords found, and keywords not found, after submission;
- feedback regarding keywords found and not found, and
- the total score.

Feedback consisted of displaying the answer which was entered by the "student" into the input box, using a blue coloured font for displaying the keywords found, and black for the other words.

In the next evolutionary step, the program was expanded by grouping related keywords (such as the colours) together in order to provide more specific feedback such as:

- "Well done, you mentioned all the colours of the rainbow!",
- "You mentioned none of the colours of the rainbow", and
- "You mentioned only two of the colours of the rainbow, what about the other colours?".

Four evaluators interacted with and tested this version - two academics and two Computer Science students (one postgraduate student and one undergraduate) - whose suggestions were used in refining the system. Following the use (and abuse) of this system by the evaluators, it became evident that the concepts of:

- searching and counting defined keywords in a piece of text, and
- providing feedback on those found and not found,

were feasible in a web-based environment.

It was decided to adapt this concept to the Unisa assignment environment in order to produce a prototype of an electronic grading system. The first variant of this system was
called E-Grader, and was a non-interactive system for Computer Science assignments. It was based on the algorithm of the earlier versions, but took a backwards step with regard to interactivity in that it did not offer instant diagnostic feedback to students. The purpose of this apparent retrograde move was to generate an E-Grader variant that could interface with assignments received by SOL, a process described under the next subheader.

**Step 5: Input and output requirements of E-Grader in the Unisa environment**

The next step in the development of E-Grader was to determine the input and output requirements of the required system. Due to the non-interactive nature of the process, input and output processes required multiple sub-processes and various conversions and translations.

**Input requirements:**
The system had to receive input from manually-gathered electronic versions of COS321-6 assignments. As previously stated, the SOL system is a conduit for electronic receipt of assignments; it is not an interactive system. Electronic assignments were received from the SOL system, mainly in MS Word format, and had to be converted manually to files in a text (.txt) format, and divided into separate files for each question before they could be batch processed by E-Grader.

**Output requirements:**
Output of quantified scores had to be in a text format to be exported to a spreadsheet program (MS Excel or QuattroPro) in order for the electronically-generated results to be compared with manual scores allocated by hand-marking the same questions. Assessment of the responses for COS321-6 is summative assessment (see Section 3.2.1.4) because assignments are formally graded, and these marks usually contribute to the student's final mark for admission to the final examination.
Step 6: Building the first operational *E-Grader* prototype

During the year 2001, students had to submit three assignments for the module COS321-6 (Operating Systems and Architecture). The feasibility of the program was initially tested on three questions from the first assignment.

The algorithm initially used for the feasibility testing of the rainbow-question was adapted to:
- include keywords of the correct answers to three questions of Assignment 1,
- read and mark the answers from the files containing the student's answers, and
- store the marks in a text file.

The algorithm was translated into PHP code, and three different programs (one for each question) were produced. Figure 6.5 indicates the basic operation of the algorithm, showing how it iteratively repeats the marking process for each answer, and within each answer tests for each relevant keyword.

Keywords were hard-coded into each program, and the keyword searches were expanded by adding regular expressions (see Section 5.6.1) to search for a range of variations of a word (i.e. different ways of spelling or conjugating). The programs were designed to grade answers by using the same criteria applied in hand-grading by a trained external marker. Although the keywords were expanded with regular expressions, only a simple keyword search was done, without interpreting the semantics of the responses.

6.3 Implementation of the *E-Grader* prototype in a real-world situation

An *intervention* (see Plomp's development research model in Section 6.1), is a real-world introduction of a product into a specific situation to change the usual mode of operation within that situation. The first intervention for this study consisted of the application of the *E-Grader* prototype, which was designed and developed to conduct electronic grading of open-ended textual responses, in the Unisa context. The purpose was to test the system against the current hand-marking system.
Figure 6.5: The basic algorithm of E-Grader
In Section 6.3.1 the first implementation of *E-Grader* on three questions from the first assignment of COS321 of 2001 is discussed, while the second implementation of *E-Grader* on three questions from the third assignment of 2001 is discussed in Section 6.3.2. Data was extracted from assignments submitted electronically via SOL and compared with grades obtained by hand-marking the same answers by looking for instances of keyword matching and then assigning scores.

### 6.3.1 The first operational *E-Grader* - implementation on assignment questions

The inclusion of an electronic grading system into the current system was proposed in Step 2 of Section 6.2.1 in order to bypass the tedious, time consuming and expensive manual marking processes described in Step 1, thereby eliminating the need for human external markers.

After the algorithm was translated into PHP code, different programs (one for each question to be marked electronically by *E-Grader*) were produced. The implementation of these programs on the answers to three questions from the first assignment of COS321-6 is discussed next.

### 6.3.1.1 Assignment 1 of COS321-6 in 2001

Forty five out of 192 students submitted the first assignment via SOL. After these assignments were hand-graded (together with assignments received via conventional mail) by an external marker, three of the questions were also marked by *E-Grader*. Raw data of this implementation is presented in Appendix B.

Assignments received through SOL were individually converted from MS Word-format (.doc) to text-format (.txt) files using Linux’s *Advanced Text Editor*. These files were further broken down into separate files for each question to be graded by *E-Grader*, and these files were then stored in a directory to be processed in a batch. Questions, total marks, keywords and model answer used for grading these three questions of Assignment 1 with *E-Grader* are presented in Tables 6.1.1, 6.1.2 and 6.1.3 respectively.
<table>
<thead>
<tr>
<th>Question</th>
<th>Model answer [3 Marks]</th>
<th>Keywords</th>
</tr>
</thead>
<tbody>
<tr>
<td>Explain the main differences between operating systems for mainframe</td>
<td>Personal computer operating systems are not concerned with fair or maximal use of computer facilities. Instead, they try to optimize the usefulness of the computer for an individual user, usually at the expense of efficiency. Mainframe operating systems are concerned with fair or maximal use of computer facilities. Systems need more complex scheduling and I/O algorithms to keep the various system components busy. These systems accommodate many users at the same time, and make use of time-sharing. Efficient utilization of the CPU is very important, as well as file and resource protection.</td>
<td>(one</td>
</tr>
<tr>
<td>Question</td>
<td>Model answer</td>
<td>Keywords</td>
</tr>
<tr>
<td>----------</td>
<td>--------------</td>
<td>----------</td>
</tr>
<tr>
<td>Several popular microcomputer operating systems provide little or no means of concurrent processing. Discuss some of the major complications that concurrent processing adds to an operating system.</td>
<td>A method of time sharing must be implemented to allow each of several processes to have access to the system. This method involves the preemption of processes that do not voluntarily give up the CPU (by using a system call, for instance) and the kernel being reentrant (so more than one process may be executing kernel code concurrently). Processes and system resources must have protections and must be protected from each other. Any given process must be limited in the amount of memory it can use and the operations it can perform on devices like disks. Care must be taken in the kernel to prevent deadlocks between processes, so processes are not waiting for each other's allocated resources.</td>
<td>(time, S, shar, processor, preempt, kernel, reentrant, synchronizing, context, switch, control, protect, integrity, data, memory, deadlock)</td>
</tr>
</tbody>
</table>
Text box 6.1 illustrates how two different responses to Question 2.2 were marked by E-Grader by using the keywords in Table 6.1.2, and consequently produced 4 marks out of 6 for each answer. Words for which E-Grader awarded marks are shown in bold. Marks were awarded for variations and conjugations of keywords as described in Section 5.6.1.2.

**Question 2.2:** Describe what multiprogramming is [6]

**Response 1:**
"Multiprogramming is utilised in operating systems to **improve the CPU utilisation**. This is accomplished through **job scheduling** where the processor would accept a job from the job queue and start executing it until the job has to **wait for some other process to complete** such as an I/O request. While the jobs is idle, the CPU is assigned another job from the queue etc. In this way, several processes are completed simultaneously."

**Response 2:**
"Multiprogramming makes efficient use of the CPU by overlapping the demands for the CPU and its I/O devices from various users. It attempts to **increase CPU utilization** by always having something for the CPU to execute."

---

Text box 6.1: How E-Grader marked different responses to the same question

Even though the word "utilisation" was spelled differently in the two responses, both were awarded a mark since the regular expression "(utili(s|z)ation)" considers both. Furthermore, both "improve the CPU utilisation" and "increase CPU utilization" were marked correct since the regular expression "(improve)|(optimi(s|s)|maximi(z|s))(maximi|e(s|s)|CPU|ls| (utili(s|z)ation)" will consider:

- improve CPU (utilisation or utilization)
- optimise or optimize CPU (utilisation or utilization)
- maximise or maximize CPU (utilisation or utilization)
- increase CPU (utilisation or utilization)
- efficient CPU (utilisation or utilization)

The "ls*ls*S*ls*" part in the regular expression used in front of the keyword "CPU" considers a space followed by any letter or word (or no letter or word) followed by another space or no
space, before the word CPU (see Example box 5.4 in Section 5.6.1.2 (2)). Therefore, "the CPU" in Response 1, as well as only "CPU" in Response 2, both received a mark.

Marks obtained by each student for each question were written to three files (one for each question) in a comma delimited format (with commas (",", " ) inserted between the student number and mark obtained) to ensure that the data could be imported into a spreadsheet file.

After removing redundant data from these three text files (see Section 5.6.3.1) each file contained only the student number and mark allocated by E-Grader.

Following this intervention, the marks assigned by the external marker, and marks of the same questions as electronically assessed by E-Grader, were combined in a spreadsheet file for the purposes of comparison and control. Results obtained by assessment with E-Grader are considered to be intermediate outcomes in line with Plomp's development research model (see Figure 6.1).

A notable problem occurred with electronic assessment of the first question, namely, "Explain the main differences between operating systems for mainframe computers and personal computers?". It was found that this type of question, where differences between concepts had to be discussed, was not suitable for assessment with E-Grader. The answers were not structured in such a way that they could be split into two separate parts (one on operating systems for main frames, and another on operating systems for personal computers). Unlike an intelligent human marker, the automated assessment mechanism could not distinguish which features related to which type of computer. After experimenting with various possible ways of grading this answer, it was decided to allocate marks for mentioning the concepts on which the two operating systems differ, such as number of users, time-sharing, file protection, user convenience, and CPU utilisation. However, the inadequacies in this approach were recognised by the researcher.

The next step was to evaluate the fairness and effectiveness of the assessment conducted by E-Grader. The validity and reliability of the intervention is investigated in the next subsection.
6.3.1.2 Evaluation of the first implementation

Statistical techniques for measuring validity and reliability of assessment instruments are based on measures of agreements and differences, and range in complexity from correlations and analyses of variance, to factor analyses and multi-variant analyses. The standard approaches of measuring validity and reliability are derived from psychometrics, a discipline particularly concerned with the development of personality measures and intelligence tests. The underlying concepts of validity and reliability in psychology and education are both based on the concept of precision or accuracy (Brown, Bull & Pendlebury, 1997).

(1) Validity

The validity of an assessment instrument or tool is the extent to which it measures what needs to be measured. "Validity is a form of truth-seeking. It is often described as the match between what is intended to be measured and what is measured" (Brown, Bull & Pendlebury, 1997: 239). For this intervention, the correlation (agreement) between grades allocated by a human grader and grades allocated by E-Grader was calculated. The scores of the human grader, who was an official and qualified grader, were taken as the norm, and the correlation of E-Grader's scores with this norm, was taken to be an indication of the validity of the electronic assessment. The raw data is presented in Appendix B.

Due to the problem mentioned in the previous subsection, a very low correlation ($r = 0.16$) was found between the marks allocated by the external marker and E-Grader for Question 1.2 (see Figure 6.6). However, statistically significant correlations on the 0.1% level were found when comparing the marks allocated by the external marker to the marks obtained by E-Grader for the other two questions:

- Question 2.1: $r = 0.83$, $p < 0.001$
- Question 2.1: $r = 0.85$, $p < 0.001$

where $p$ is the probability of such a correlation occurring by chance alone, and where $p < 0.001$ means that this probability is less than 0.001, i.e. 0.1%.

The correlations listed above and shown graphically in figure 6.6, indicate that E-Grader offers high validity when used to assess appropriately structured questions.
(2) Reliability

Reliability of an assessment instrument or tool is the extent to which it yields consistent results over time and over different applications. The fact that *E-Grader* assessments on certain types of questions (those that did not consist of comparisons of concepts), correlated significantly with a human grader’s scores, can be taken as an indication of reliability for those type of questions.

Another measure of reliability in assessment, is a measure of the agreement between different assessors. Only one human assessor was used for this first implementation of *E-Grader*, thereby producing only one correlation for each question. In order to calculate the measure of agreement between more than one assessor, this aspect was addressed in the investigation of the third assignment. For this application, the same answers were also marked by the researcher herself, i.e. these answers were assessed by two different human markers and electronically assessed by *E-Grader*. These correlations are presented in Section 6.3.2.
6.3.2 The second implementation of E-Grader on assignment questions

The algorithm used for the first operational implementation was adapted for this intervention, assessing textual answers to three questions from the third assignment. Three more programs were thus generated, each containing the relevant keywords for the corresponding questions.

6.3.2.1 Assignment 3 of COS321-6 in 2001

As with the first assignment, 45 out of the 192 students submitted the third assignment via SOL. As in the case of the first operational intervention, assignments received through SOL were converted from MS Word-format (.doc) to text-format (.txt) files, and were further broken down into separate files for each question to be marked by E-Grader, and were then stored in a separate directory. The revised programs were executed on these files, and the marks (outcomes) stored in a similar way as was done with the first operational implementation (Section 6.3.1). Raw data of this implementation is presented in Appendix B.

The style of Question 5a differs from the previous questions in that an example was explicitly requested. The question is “Discuss the use and working of pipes in UNIX. Give an example of the use of pipes.” (see Table 6.2.2). The program was adapted to award one mark for a legitimate example, and three marks for the description of “pipes”.

Questions, total marks, keywords and model answers used for the implementation are presented in Tables 6.2.1, 6.2.2 and 6.2.3 respectively.

6.3.2.2 Evaluation of the second implementation

(1) Reliability

Reliability relates to the consistency of results over time and in different applications. The high correlations obtained in the investigation of the third assignment confirm the results obtained on different questions in the first assignment.

As stated earlier, a further measure of reliability in assessment is a measure of agreement between assessors. “There is plenty of evidence on the disagreement between assessors, even when using marking schemes” (Brown, Bull & Pendlebury, 1997: 234).
<table>
<thead>
<tr>
<th>Question</th>
<th>Model answer</th>
<th>Keywords</th>
</tr>
</thead>
<tbody>
<tr>
<td>What is an acyclic graph</td>
<td>A graph is a generalization of a tree where a node may have more than one parent. In a sense, therefore, a graph is a tree that has been corrupted. An acyclic graph is a graph without any cycles, i.e. a node cannot be its own ancestor. Therefore, no edge leaving a node will eventually loop back to the same node or any previously visited node.</td>
<td>(general)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>more</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(file</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(without)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>S*s</td>
</tr>
</tbody>
</table>

Table 6.2.1: Question 1.4

<table>
<thead>
<tr>
<th>Question</th>
<th>Model answer</th>
<th>Keywords</th>
</tr>
</thead>
<tbody>
<tr>
<td>Discuss the use and working of pipes in UNIX. Give an example of the use of pipes.</td>
<td>The pipe is the interprocess communication (IPC) mechanism most characteristic of UNIX. A pipe allows for a one-directional steam of bytes between 2 processes. The first process writes into one end of the pipe, and the other process reads from the other end of the pipe. A pipe is traditionally implemented as an ordinary file. There are however some exceptions: It has no name in the file system and it is created by the pipe system call. The size of a pipe is fixed, and any process which attempts to write into a full pipe is simply suspended. The following example of the use of a pipe (there are many) is to list the contents of your current directory, sort it and display it one page at a time: Is</td>
<td>sort</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(one un</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(transfer</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(constant</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(constant</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ordinary</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(local</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For the example:   vertical</td>
</tr>
</tbody>
</table>

Table 6.2.2: Question 5a
<table>
<thead>
<tr>
<th>Question</th>
<th>Model answer</th>
<th>Keywords</th>
</tr>
</thead>
<tbody>
<tr>
<td>Explain how the <em>socket mechanism</em> is seen to be an extension of the</td>
<td>In UNIX the pipe is implemented as a special case of the <em>socket</em> mechanism. The socket mechanism provides a general interface not only to facilities such as pipes, which are local to one machine, but also to networking facilities. As such the socket mechanism can be used by unrelated processes. A socket is the endpoint of communication and a socket in use normally has an address bound to it. There are several socket types in UNIX, e.g. stream sockets.</td>
<td>pipe <em>s</em> <em>S</em> <em>s</em> <em>S</em> <em>s</em> special <em>s</em> case <em>s</em> <em>S</em> <em>s</em> <em>S</em> <em>s</em> socket <em>(general)</em> interface <em>s</em> <em>S</em> <em>s</em> <em>S</em> <em>s</em> <em>S</em> <em>s</em> pipe ✓ general <em>s</em> interface ✓ (bi)both <em>s</em> <em>S</em> <em>s</em> *(direct)*way ✓ network *(different)*separate <em>s</em> <em>s</em> <em>S</em> <em>s</em> <em>S</em> <em>s</em> <em>S</em> <em>s</em> <em>s</em> <em>s</em> machines ✓ communicating <em>s</em> <em>s</em> stream <em>s</em> <em>s</em> <em>S</em> <em>s</em> <em>S</em> <em>s</em> <em>S</em> <em>s</em> <em>s</em> <em>s</em> <em>s</em> <em>s</em> <em>S</em> <em>s</em> <em>S</em> <em>s</em> <em>S</em> <em>s</em> address ✓ unrelated <em>s</em> processes <em>s</em> processes <em>s</em> <em>S</em> <em>s</em> <em>S</em> <em>s</em> <em>S</em> <em>s</em> <em>S</em> <em>s</em> <em>S</em> <em>s</em> <em>S</em> <em>s</em> <em>S</em> <em>s</em> <em>S</em> <em>s</em> <em>S</em> <em>s</em> <em>S</em> <em>s</em> <em>S</em> <em>s</em> <em>S</em> <em>s</em> not <em>s</em> related ✓ end <em>s</em> <em>S</em> <em>s</em> point <em>S</em> <em>s</em> <em>S</em> <em>s</em> <em>s</em> <em>s</em> communication ✓</td>
</tr>
</tbody>
</table>
In order to measure the reliability of the second implementation of *E-Grader* by using two sets of control data, answers were hand-marked by the same external marker used for Assignment 1, and the same answers were also marked by the researcher herself who, as previously stated, is one of the lecturers on the COS321 team. These marks were stored in a spreadsheet file to be compared with the marks allocated by *E-Grader*.

As in the case of the second and third questions of Assignment 1, statistically significant correlations on the 0.1% level were found (see Figure 6.7) when comparing the marks allocated by the external marker to the marks obtained by *E-Grader* for the other two questions, and even higher correlations were found when comparing the marks allocated by the researcher to the marks obtained by *E-Grader*:

- \( r = 0.84, \ p < 0.001 \)
- \( r = 0.83, \ p < 0.001 \)
- \( r = 0.85, \ p < 0.001 \)

where \( p \) is the probability of such a correlation occurring by chance alone, and where \( p < 0.001 \) means that this probability is less than 0.001, i.e. 0.1%.

![Figure 6.7: Correlations between E-Grader and two control data sets](image-url)
6.4 Discussion and summary

This discussion refers briefly to relevant aspects from previous chapters, relating to proposed extensions of the design, implementation and interventions addressed in this chapter.

Chapter 6 described the cyclical design and development process, implementation and outcomes of a prototype software system called E-Grader. The chapter was structured according to Plomp's model, and then Pressman's phases within Plomp's processes. E-Grader was implemented as the first intervention of this study. The system was successfully applied as an electronic assessment (e-assessment) system within the Unisa assignment marking environment. The system enabled automatic grading of textual assignment answers, and stored grades thus obtained in a text-file. Comparison of these grades obtained by e-assessment with control grades obtained by manual assessment indicated a significant correlation. E-Grader was implemented as a support tool for the lecturer to reduce labour and to enhance productivity, and the results reported in this chapter indicated that it does so effectively. It was not designed to provide feedback to the learner in order to improve the learning process, i.e. it was designed as an educator-centred system, not a learner-centred system.

Chapter 2 described a literature review conducted to ensure that development efforts undertaken as part of this study were based on sound principles of contemporary learning theory. This study focused on the implications of learning theories (such as behaviourism and constructivism) on technology and learning, and showed that current instructional and learning theories are moving towards constructivism. Moreover, technology lends itself to the implementation of constructivist learning. Current instructional design and development of learning systems indicate a shift in this direction, therefore, it was decided to extend E-Grader to produce a learner-centred system based on such principles, an approach to be addressed in Chapter 7.

In a constructivist learning environment (discussed in Section 2.2.2.2(2)), the focus shifts from the educator to the learner. The goal of constructivist learning is not the accurate transmission of content from the instructor to the learner, but rather, learners are encouraged to construct their own version of the content. Constructivists believe that learning is more meaningful if learners are allowed to explore independently, while making errors and learning from them. A learning situation should be created in which learners can learn through experimentation, collaboration and sometimes failure. Educators and learners are considered as collaborative
co-constructors of meaning and knowledge. Learners should also communicate with each other about their learning in order to conceptualise their thoughts. Constructivists believe that assessment should be embedded in the context of learning, and should focus on what has been constructed by the learner during the learning process i.e. formative assessment (see Section 3.2.1.4 for a differentiation between summative and formative assessment). Formative assessment provides feedback to learners on their progress during a course, in ways that enable them to improve their learning. Caution was expressed in Section 3.3.2.1 that assessment of constructivist learning is considerably more complex than assessing traditional learning, because constructivist learning is based on critical thinking and active participation in problem-solving. It was proposed in Section 3.4.2.2 that the interactive nature of the WWW should be utilised for implementing active constructivist learning.

Chapter 4 proposed a move towards automated scoring, and that validation tools for formative assessment of textual responses should be developed. It was proposed that teaching should always include effective assessment of learners' work, and feedback by educators. Most of the automated grading systems currently used for web-based textual inputs (investigated in Chapter 4) provide diagnostic feedback to learners.

In contrast to constructivist principles, assessment conducted by E-Grader for this study and described in Chapter 6, was summative since assignments which counted towards the final mark for the COS321-6 course, were formally graded. Furthermore, the system is not learner-centred, and does not provide supportive feedback to learners. Therefore, the system requires extension in an attempt to produce a more learner-centred, interactive, web-based system based on certain constructivist principles.

An extended system was therefore designed to focus on the learning process by supplementing the scoring mechanism with immediate feedback to enhance the learning process. The resulting system, called Web-Grader, is discussed in the next chapter.
References - Chapter 6


Plomp, T. 2002. *Some reflections on development research (DR)*. Notes for a lecture presented at the University of Pretoria. Enschede, the Netherlands: University of Twente.


Chapter 7:

*Web-Grader: development, implementation and evaluation*

### 7.1 Introduction

Chapter 6 described the cyclical design and development process of *E-Grader*, as well as an intervention and its outcomes. *E-Grader* was designed and developed as a tool to aid lecturers, and it laid the foundation for *Web-Grader*, which had as its prime purpose interactive support for students.

In this chapter the software development process and implementation of the second variant of the prototype system are described. This variant, called *Web-Grader* was developed as an interactive web-based extension of the first variant (*E-Grader*). The system read in students' short responses to open-ended questions, assigned marks, and offered substantive feedback (see Figure 7.1). The system differed notably from *E-Grader* in three respects:

- Firstly, it offered interactivity in that students interacted directly with the system as end-users working on-line.
- Secondly, a mark was assigned, as with *E-grader*, but furthermore, personalised feedback supported the student in the form of specific comment clearly related to his/her answer, including textbook references and suggestions on how to improve it.
- Thirdly, the system was not used for formal grading, but as a student study support tool.

![Figure 7.1: Interactive Web-Grader](image-url)
Closely aligned with Plomp's (2002) development research model, incorporating design, intervention and outcomes, the work described in this chapter comprises the design process of Web-Grader, its application in a real-world situation (the intervention), and the consequent outcomes of the intervention. Once again Pressman's view is addressed within a phase of Plomp's model.

Section 7.2 focuses on the third phase of Pressman's (1997) generic phases of software development (described in Section 5.5) namely the maintenance phase that focuses on change. The steps of the definition and development phases were re-applied in this phase, in the context of the existing software.

In Section 7.3 the implementation of Web-Grader as a study support tool in a real-world situation is addressed, and the results (outcomes) of this intervention are discussed.

7.2 Design and development process of the Web-Grader prototype

In line with Pressman's generic phases of software development, the steps of the definition and development phases were re-applied in the development process of Web-Grader. Since it was an extension of E-Grader, the real-world problem (analysed in Step 1 of Section 6.2.1); the required new system (proposed in Step 2); and the subject matter and target group (identified in Step 3) remained unchanged, these steps were not re-applied. No further feasibility testing of the very first algorithm used for E-Grader (Step 4 of Section 6.2.2) was required in the development process of Web-Grader. The re-application of Steps 5 and 6, however, played a major role in this extended development. Whereas E-Grader was implemented and tested in 2001, Web-Grader was developed and tested in 2002.

In Section 7.3, where the implementation of Web-Grader is discussed, a textbox namely Textbox 7.1 is provided, which describes a typical user interaction with the system. It may be of help to the reader to glance briefly at this textbox in order to contextualise the discussion of the design and development process in this section.
7.2.1 Definition phase: What was built?

The basic algorithm of E-Grader was extended by adding a graphical user interface (GUI) for student use. The system was further extended by connecting it to the Apache web-server, and converting it into a database-driven web site using the MySQL relational database discussed in Section 5.7. Figure 7.2 represents this dynamic interaction.

![Diagram of Database-driven Web-Grader]

Figure 7.2: Database-driven Web-Grader

The idea of writing test items before designing instruction, i.e. starting the design process by deciding on how performance is to be assessed - as proposed by the Dick & Carey (1996) model as well as by Prestera (2002) - was applied in the design of Web-Grader. The system was constructed as a self-assessment system that graded learners' open-ended textual responses to questions, and that also provided textual feedback as a means of supporting the process of learning.

7.2.2 Development phase: How was the system built?

7.2.1.1 Re-application of Step 5: Input and output requirements

The first step in the extension of E-Grader to Web-Grader was to determine the input and output requirements of the new system. As with the first variant, the system was designed to be used by students enrolled for COS321. It was tested in an intervention with the class of 2002.
(1) Input requirements

- Lecturer-input

An administrator interface was required for inputting, editing, updating and deleting the learning content of the web-site. Questions from previous years' assignments and examination papers had to be presented on the web site, and a lecturer had to store data such as usernames, questions, model answers, maximum marks, hints on where to find the answers, and keywords in a database.

- Student-input

Students should be able to select and answer questions, as few or as many as they please, in any sequence and as often as they wish. The learner should be able to enter open-ended answers into a text box, submit the content, and be graded immediately. If not satisfied with marks obtained, students should be able to re-submit answers.

(2) Output requirements

The system was designed to provide immediate, contextualised feedback (discussed in more detail in Point 5 of Step 6) based on the input of the student.

7.2.1.2 Re-application of Step 6: Building the Web-Grader prototype

(1) Design of Web-Grader's algorithm

The algorithm of E-Grader was extended by using HTML, PHP, and JavaScript (discussed in Sections 3.2.2 and 5.7) for adding a GUI. It was further extended by adding access control and a database connection. After logging in, and choosing a question from the menu, the student could enter and submit the answer. Answers were graded interactively by the system, providing immediate feedback.

Figure 7.3 shows the basic algorithm of the main program (named cosresult.php) used for grading and providing feedback. The source code of this program is presented in Appendix A.
Choose question → Input answer

cosresult.php

- Get keywords for answer from database
- Get maximum mark for question
- Convert keyword list to an array of keywords

- Get answer from user
- Convert answer to an array of words contained in the answer

- Initialise total to 0
- Compare answer [x] to keyword [y]

- Display results and feedback

- Save answer
- Redo answer
- Answer another question

Database

Figure 7.3: Basic algorithm of Web-grader
(2) Design and development of a graphical user interface

General tools used to design a GUI were discussed in Section 5.7.2.1. Web-Grader's design and development process, using some of these tools, is described here.

» Dialogue chart

First, the dialogue between the screens was modelled by using a dialogue chart (see Figure 7.4).

![Dialogue chart diagram]

Figure 7.4: Web-Grader's dialogue chart

» Icons, menus and windows

Hypertext and hyperlink, iconic menus, dialogue-boxes and windows were discussed in Section 5.7.2.1. Figure 7.5 shows some of the icons used in Web-Grader.

Figure 7.6 shows how a combination of hypertext and hyperlink, iconic and pop-up windows was implemented in Web-Grader.
Figure 7.5: Icons used in Web-Grader

Pop-up window

Iconic menus
Hypertext links

Figure 7.6: A combination of different types of windows

202
Pop-up windows were used in *Web-Grader* for:

- Providing an option between *<Go back and Save>* and *<Exit the system>* when leaving the system (see Figure 7.7 (1)).
- Confirming that marks were saved into the database (see Figure 7.7 (2)).
- Displaying feedback when a student received zero marks for a question (see Figure 7.7 (3)).
- Displaying the benefits of using a cyber classroom (see Figure 7.7 (4)).

![Image](image_url)

(1) Exit the system  
(2) Save the mark and answer

![Image](image_url)

(3) Feedback when receiving zero marks  
(4) Benefits of using a cyber classroom

**Figure 7.7**: Pop-up windows used in *Web-Grader*

- **User interface controls/components**

  User interface controls such as text boxes, radio buttons, check boxes, drop-down lists, buttons, etc. were discussed in Section 5.7.2.1. *Web-Grader* made use of text-input boxes and submit-buttons on screens where students enter:

  - usernames and passwords (see Figure 7.12 (2) and (3)),
  - messages to be posted on the notice board (see Figure 7.12 (10)), and
  - answers to the questions (see Figure 7.12 (13)).
After choosing a question from the menu, a text-input box was displayed where the student could key in the answer to the chosen question. This answer was submitted by clicking on the <Submit> button (see Figure 7.12 (13)). A pop-up window or another screen containing feedback and animations depending on the mark obtained for the answer was then displayed. Answers could be saved or re-done, or another question could be chosen by clicking on the relevant icon.

- **Graphic resolution**

Graphic resolution was discussed in Section 5.7.2.1. Web-Grader was designed for 800 000 horizontal pixels by 600 000 vertical pixels in a 17 inch diagonal display.

(3) **Design and development of access control**

The system was password-protected (see Section 3.2.1.2) to ensure that it was accessible only to valid students. A list containing student numbers of authorised students was stored in a text file. On the first visit to the web site, students created a login and password before being allowed to operate, using their student number for a username, and choosing their own password.

(4) **Database design and development**

The relational database model and MySQL were discussed in Section 3.2.2.7, and database development tools were discussed in Section 5.7.2.3. With this background, Web-Grader's database and table structures are discussed.

The name of the relational database was *php_martjie*, and consisted of three tables (see Figure 7.8):

- **tblusers**, the first table, was used for authentication, to store date of the student's last login, and to count how many times each user visited the site.
- **CC_store** was used to store each student's username, mark (grade) for each answer, date submitted, and a flag ("Y" or "N") to indicate whether the answer was intentionally saved by the student, or saved by a salient agent in the program.
- **CC_Quesans** was used to store question numbers, questions, total marks, model answer and keywords.
Database name = *php_martjie*

<table>
<thead>
<tr>
<th>tblusers</th>
<th>CC_store</th>
<th>CC_Quesans</th>
</tr>
</thead>
<tbody>
<tr>
<td>username</td>
<td>Qnumber smallint</td>
<td>Qnumber smallint</td>
</tr>
<tr>
<td>password</td>
<td>username varchar(20)</td>
<td>question varchar(100)</td>
</tr>
<tr>
<td>date</td>
<td>date</td>
<td>total_marks smallint</td>
</tr>
<tr>
<td>counter</td>
<td>int</td>
<td>model_answer blob</td>
</tr>
<tr>
<td></td>
<td></td>
<td>book varchar(100)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>page varchar(50)</td>
</tr>
</tbody>
</table>

**Figure 7.8: Database structure used for Web-Grader**

(5) **Design and development of interaction and feedback**

It was stated in Section 4.2 that all learning sessions, whether in a contact-teaching environment, at a distance, delivered electronically, or any other form, should provide opportunities for learners to interact with the educator, control the learning process, and provide reverse feedback, receive customised feedback about the knowledge being constructed and interpreted. These principles were applied in the design and development of interaction and feedback in Web-Grader.

- **Interaction and communication**

  On-line communication (discussed in Section 3.2.1) was incorporated within Web-Grader by providing an e-mail facility, as well as a public notice board where students could make spontaneous contributions in the form of messages and comments to the lecturer and to other students. This is known as reverse feedback.

- **Learners’ control of the information process**

  The intention with Web-Grader was that students should be able to visit the site asynchronously at any time of day or night. Furthermore, they should be able to answer questions in any order, and receive feedback on a particular question immediately after submitting an answer. They should be permitted to redo answers until satisfied with the mark obtained, and to view their own record at any time.
Assessment and feedback about the knowledge being constructed

Plowman's (2002) model for guided interaction, based on the scenario of a family member reading a picture book with a young child while offering guidance (see Section 3.4.2.1), was combined with Black and William's (1998) view of formative assessment (see Section 5.7.2.2) in designing interaction and feedback in Web-Grader. Table 7.2 shows how Black and William's elements of feedback system for formative assessment, were used in the design of feedback to students who obtain more than zero but less than full marks for a question.

<table>
<thead>
<tr>
<th>Black and William's elements of formative assessment</th>
<th>Feedback provided by Web-Grader</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data on the actual level of some measurable attribute.</td>
<td>The mark obtained was provided immediately after submitting the answer to a question.</td>
</tr>
<tr>
<td>Data on the reference level of that attribute.</td>
<td>The submitted answer was displayed with an indication of which words received marks - these words were displayed in a contrasting colour.</td>
</tr>
<tr>
<td>A mechanism for comparing the two levels, and generating information about the gap between them.</td>
<td>A list was displayed of all the keywords that were not mentioned in the answer.</td>
</tr>
<tr>
<td>A mechanism by which the information can be used to alter the gap.</td>
<td>The name of the prescribed book and the page numbers where the answer can be found were displayed.</td>
</tr>
</tbody>
</table>

Table 7.2: Black and William's elements applied in Web-Grader

Figure 7.9, an extension of Figure 7.3, shows the basic algorithm of Web-Grader's main program with differentiated feedback based on input (see Appendix A for the source code of this program).
Figure 7.9: Basic Web-Grader algorithm with differentiated feedback based on input
Students obtaining zero marks for an answer, received only an indication of where to find the answer in the book. This information was provided in a pop-up window containing an animation. No hints or clues were provided (see Figure 7.7 (3)).

Feedback for answers receiving more than zero marks but less than full marks was enhanced with a special effect (programmed in JavaScript) consisting of cascading light bulbs, and an animation of a booklet being paged through (see Figure 7.10).

Students receiving full marks for a question were rewarded by cascading exploding stars programmed with JavaScript and a picture of a student holding a cup (see Figure 7.11). The answer submitted by the student was displayed, with an indicating which words received marks by displaying those words in a contrasting colour.

Web-Grader gave learners the opportunity to re-do questions until satisfied with their mark. At this point the student could store the final answer (and its associated mark) in the database by clicking on an icon displaying an image of a stifly (see Figures 7.5 and 7.6). On saving the record, a pop-up window appeared, containing an animation to confirm that the record had been saved (see (see Figure 7.7(2)).
Figure 7.10: Feedback for receiving more than zero marks but less than full marks

Figure 7.11: Feedback for receiving full marks
Figure 7.12: Screen shots of the COS321-6 CyberClassroom2002
(5) Choose between posting a message to the noticeboard or reading the noticeboard

(10) Enter a message into the text input box

(11) Read the noticeboard

(12) Menu containing the questions to be answered

(13) Enter the answer to the chosen question into the text input box

(14) Zero marks awarded to the answer

Figure 7.12 (continued): Screen shots of the COS321-6 CyberClassroom2002
Result and feedback

Question 2: Describe what multiprogramming is (13 Marks)

You did not mention:
- program scheduling
- a swap waiting list
- acquire more resources

Your total mark for Question 2 is:
10 out of 13

(15) Mark awarded is more than zero but less than full marks

Your total mark for Question 2 is:
13 out of 13

(16) Full marks are awarded for the answer

Save

(17) Save the mark and answer

View record

(18) View record

Send e-mail

(19) Send e-mail

Exit the system

(20) Exit the system

Figure 7.12 (continued): Screen shots of the COS321-6 CyberClassroom2002
7.3 Implementation of the Web-Grader prototype in a real-world situation

The second intervention of this study relates to the real-world implementation of the Web-Grader prototype as a virtual classroom. On-line virtual communities (such as virtual classrooms, virtual libraries, virtual outings and virtual laboratories) were discussed in Section 3.2.1.6. Benefits of a virtual classroom, such as providing instructional content anywhere in the world (distance-independence) using any computer platform (device-independence), at any time of the day or night (time-independence) were mentioned. Virtual communities are also called cyber communities. Web-Grader was a move in this direction in that it provided opportunities for students to study at their own pace according to their own schedule at any on-line location world-wide. Moreover, they could post messages and comments on the notice board, thus communicating with the lecturer and fellow students.

The generic software engine of the Web-Grader prototype was content-free and was developed as a database-driven system that can be used to produce various different cyber classrooms. Various classrooms in different domains can use the system to access data in content-specific databases. In this way, the learning content of each cyber classroom would be provided by the relevant database, i.e. its functionality can in future be extended beyond the domain of COS321-6.

The cyber classroom produced by Web-Grader for this study was called the COS321-6 CyberClassroom2002, and it was implemented as an interactive web-based study support tool with an innovative informal theme. The aim of this classroom was to involve students in collaborative activities by promoting interaction, involvement and engagement (discussed in Section 3.2.1) and to use self-assessment as means of promoting learning.

To set the context of the upcoming section relating to the implementation of Web-Grader as a cyber classroom, Text box 7.1, containing a description of a typical user interaction with the system, is provided. This is done to set the scene for the ensuing discussion in Section 7.3.1. The bracketed numbers, e.g. "(6)" refer to the subfigures of figure 7.1.2.
After connecting to the web site, a first-time user could create a login by entering his/her student number as a username, and a password of his/her own choice into an input-box (see Figure 7.12 (2)).

Thereafter, each typical work-session commenced with the student logging in (1). After successful authentication, he/she proceeded to the welcome page (6), and from there to the home page (7) containing the main menu. The home page offered many possible routes for navigating through the system:

- A first time user would probably first visit the introduction page (8) where certain rules and procedures were set out.

- Other users would probably go straight to the main menu (12) where a series of questions was presented. To respond to these questions, the student would click on one of the hyperlinked questions. This resulted in the display of an input box (13), where the student’s response to the question could be entered. The answer was keyed in, edited and submitted by clicking on the <Submit> button. A pop-up window or another screen containing detailed feedback and animations (depending on the input and the mark obtained for the answer) was displayed instantly: (14), (15) or (16). The student could then decide to save the answer its associated grade, or to re-do the answer, or to choose another question by clicking on the relevant icon.

- All pages in the system contained links to:
  - The introduction page (8).
  - The menu containing a series of questions to be answered (12).
  - The public notice board, where students could read messages posted by other students or the lecturer (11), or post a message to the notice board (10).
  - An e-mail facility – students could send e-mail to the lecturer or other students (19).
  - A page containing the student’s record (18).
  - Exit the system (20).

Text box 7.1: A typical student interaction with the system
7.3.1 Implementation of Web-Grader as a cyber classroom

The cyber classroom was available on the WWW as a study support tool, accessible to authorised users, on the web site of Unisa’s School of Computing at the URL:

www.osprey.unisa.ac.za/cos/cosindex.html

The intervention was conducted during late 2002 when students were encouraged to use the classroom as part of their preparation for the end of the year examination. All students enrolled for COS321-6 were invited to use the cyber classroom as a study support tool. This invitation was included in tutorial letters and was also displayed on the Osprey web site. More than 60 out of 187 students enrolled for the module, made use of the cyber classroom. Of these 60, 21 were given the opportunity of using it as a special concession. Unisa students need to earn 100 credits (by submitting specified assignments during the year) in order to obtain admission to the examination. COS321 students who had not achieved this by the deadline, and still needed up to 30 credits after submitting at least two assignments, were allowed to use the cyber classroom in order to obtain additional credits.

Assessment in the cyber classroom consisted of a combination of summative and formative assessment (discussed in Section 3.2.1.4). Although marks obtained were formally graded in the intervention, and counted towards the final mark for examination admission (summative assessment), the system was primarily designed to implement learner-centred formative assessment.

For ethical reasons (Cohen, Manion & Morrison, 2001), students were informed of the implications of their involvement. It was stated in a tutorial letter, as well as on the home page of the system, that the cyber classroom formed part of a research project and that results would be published. Students were assured that only collective results, and no individual responses would be published (see Figure 7.13), i.e. anonymity would be preserved.
Text box 7.2 illustrates how *Web-Grader* marked two different responses to one of the questions on the cyber classroom. Words for which *Web-Grader* awarded marks are shown in bold.

**Question 1:**
What are the main functions of an operating system? [6 Marks]

**Keywords:**

- manage[control] \( \backslash s^{*} \backslash s^{*} \backslash s^{*} \backslash s^{*} \) resources[resource \( \backslash s^{*} \backslash s^{*} \backslash s^{*} \backslash s^{*} \backslash s^{*} \) service],
- allocates \( \backslash s^{*} \backslash s^{*} \backslash s^{*} \backslash s^{*} \backslash s^{*} \) resources[resources \( \backslash s^{*} \backslash s^{*} \backslash s^{*} \backslash s^{*} \backslash s^{*} \backslash s^{*} \backslash s^{*} \backslash s^{*} \backslash s^{*} \backslash s^{*} \)],
- allocate[allocate \( \backslash s^{*} \backslash s^{*} \backslash s^{*} \backslash s^{*} \) resource,
- fair[efficient],
- provide \( \backslash s^{*} \backslash s^{*} \backslash s^{*} \) services,
- application \( \backslash s^{*} \) program,
- higher \( \backslash s^{*} \backslash s^{*} \) (level/abstraction)

**Response 1:**
The main functions of an operating system is to **provide services** to the user and to create the environment in which the user can execute programs in an **efficient** way. The operating system also **manage the computer resources** and acts as an intermediary between the user and the hardware. The o/s **allocates the needed resources**, which include hardware and software resources, needed by the user when using **application programs** such as word processors, compilers etc. In short, operating systems exists to solve the problem of creating a usable computing system.
Web-grader's feedback to Response 1:
You did not mention:
- higher level

Your total mark for Question 1 is 5 out of 6

Response 2:
It provides an environment for a computer user to execute programs on computer hardware in a conventional and efficient manner. As an application program it serves two major functions: supervision of the execution of user programs to prevent errors and improper use of the computer, and management of the I/O devices. It allocates resources of the computer as needed to solve the problem given. The allocation process should be as fair and efficient as possible.

Web-grader's feedback to Response 2:
You did not mention:
- provide services
- higher level

Your total mark for Question 1 is 4 out of 6

Text box 7.2 (continued): How Web-Grader marked two different responses a question

7.3.2 End-user-involvement in the cyclical development process of Web-Grader

As mentioned in Section 6.1, development research entails the cyclical process of improving a product while using it. The rapid prototyping model (discussed in Section 5.4.1), used for the development of Web-Grader, requires active user participation in the development process. Pressman's (1997) third phase, namely maintenance, focuses on change that is associated with:

- error correction,
- adjustments required as the software's environment evolves, and
- changes due to enhancements brought about by changing customer requirements.
This section describes how the system was improved by incorporating user-input into the cyclical development of Web-Grader.

7.3.2.1 Peer evaluation

Four colleagues evaluated Web-Grader prior to student use and their suggestions were used to improve the system. This peer evaluation resulted in the following modifications:

- Correction of spelling errors.
- Improvement of the Home Page by placing the menu in a more prominent centred position.
- The <Save>, <Redo> and <Answer another question> options displayed at the top of the page that returns results after submission of an answer, were duplicated at the bottom, so that users could invoke these options without scrolling back up.
- Initially, the cursor was not automatically positioned inside the input-boxes used on various screens. When evaluators started entering information without first positioning the cursor inside the box, the input-text did not appear in the box. A further script written in JavaScript was written to solve this problem.

7.3.2.2 End-user (learner) evaluation

In line with Plomp's (2002) development research model (see Figures 1.1 and 6.1), feedback from students was used to implement modifications and improvements in the cyclic development process of Web-Grader. Students started using the system in July 2002. The prototype question bank contained ten questions from previous years' assignments, thus comprising a compact revision aid of the COS321 syllabus. They were encouraged to provide reverse feedback, which resulted in the following evolutionary improvements:

- A logout-option was incorporated into the system after some students complained about problems in exiting the system. The first message placed on the notice board was "How do I log out of here?".
Although students were advised to use their student number as a username, it was discovered that some students used their first names instead. The form was then updated to display the word "studentnumber" by default in the username-section of the input box used to create a login.

After receiving some complaints from students who had forgotten to save their work before leaving the system, the logout-out option was extended by displaying a pop-up window, reminding them to save. It provided an option between <Go back and Save> and <Exit the system> (see Figure 7.7 (1)). Students were also advised to check their record before progressing to another question or before leaving the system, to ensure that answers were saved.

Some students experienced problems with saving certain answers. It was mentioned in Section 5.7.2.1 that a dialogue box (a type of window that presents choices to the user, and provides a graphical means of inputting or displaying information) supplements the interaction in the main window, and that it is considered by the GUI as a secondary window. This caused a problem with the pop-up window used for confirming the <Save> action. Most browsers cannot open a new secondary window while the current one is still open. Consequently, if a student did not close the pop-up window after saving an answer, he/she could not save another answer before closing the first window. The solution to this problem was to re-program the window to stay open for a few seconds before closing automatically.

### 7.3.3 Results

The main purpose of the real-world implementation of Web-Grader was to formatively evaluate the concept of using a cyber classroom as an interactive study support tool. Feedback from students who participated in the intervention was used for this formative evaluation, and is presented in Section 7.3.3.1. The findings described in this section are both qualitative and quantitative. In Section 7.3.3.2, a further outcome, namely, the performance of these users in the COS321 examination of 2002 is discussed.
7.3.3.1 Feedback to the lecturer

Students e-mailed and phoned to express appreciation for the opportunity of using an innovative tool, mentioning its value as a means of learning support. Since they were unaccustomed to receiving immediate feedback, many students were concerned about what to do next, even after they had successfully used Web-Grader to gain exam admission. Most of them needed personal reassurance from the lecturer that the marks obtained online by entering and submitting the answers were indeed sufficient for admitting them to the examination. Such uncertainty is evident in student messages under Subheader “2. Information and enquiries related to examination admission” in this section.

The public notice board also served as a vehicle for providing feedback and communication in the form of messages and comments to the lecturer and to fellow students. Messages posted on the notice board during 2002 could be divided into five categories:

1. Appreciation:
   - “I like what you’re doing here. What are you using to do the marking, a simple keyword match algorithm or a language heuristic? I see huge potential if the user interface for the lecturers is simple and doesn’t require a small program to be written to determine the correct answer.”
   - “This type of online questioning is a good idea. It allows one to refine the answers to questions through the feedback mechanism, so that the answers are concise, and contain only the key ideas.”
   - “Thanks for allowing me to take this test and earn credits to write exams at the end of the year. I thank you all who helped me in this regard.”

2. Information and enquiries related to examination admission:
   - “I did the questions in the cyberclassroom. You asked me to do it because I only accumulated 95 credits. Please let me write my exam.”
   - “I have done the questions that you had requested me to do in the cyber classroom, for admission to the examination. My credit for the assignments were insufficient they were 91. Please consider giving me admission to the exam.”
   - “I have completed the questions and I think it went better than the last time.”
   - “Hi, due to my assign.3 not being marked I only accumulated 86 credits. I have done the questions that you requested me to do in the cyberclassroom, so as to obtain...”
more credits. I have saved the answers in the cyberclassroom, must I also e-mail it to you? Your consideration in granting me admission to the exam will be greatly appreciated. Thank you."

- "I have completed all ten questions on the CyberClassroom page. I hope this will be sufficient to give me the credits that I require for examination entrance. I will contact you tomorrow (28.08.2002) to find out."

3. **Problems related to system usability:**

- "How do I log out of here now?"
- "Hi, As I have accumulated only 91 credits for the assignments, you have asked me to do the questions of the CyberClassroom but I'm having problem to save. Please help me as I really need to have admission to the exams to finish my BSc this year. Please help. I have managed to save only one result."
- "Picked up a problem using IE 5.5 trying to Save, worked on Netscape 4.08"
- "I tried to answer the questions in the cyberclassroom2002, unfortunately I had problems in submitting answers to question 2 to 10. I don't know what is the problem."
- "As I had accumulated only 87 credits I did the cyber2002 questions and obtained good marks but I am unable to save it and thus cannot view my record."

Note: As stated in Section 7.3.2.2, this feedback from students resulted in correction of the errors identified.

4. **Concern about the effectiveness of the marking mechanism:**

- "Word variants seem to be a bit of a problem, however that can be fixed by using a more comprehensive dictionary. I also experience trouble in the way I structured my sentences, for instance in the use of an indirect pronoun. The first sentence does not resolve "allocate resources": (sentences simplified) - The OS manages the resources of the computer, so as to fairly allocate them...I had to change it to - The OS manages the resources of the computer, so as to fairly allocate the resources..."
- "What mechanisms do you have in place to avoid abuse? i.e. what stops me dumping keywords in a block of text."
- "The key word search engine, in my opinion, can be improved a bit."
- "This system is not all that good; it does not allow one to phrase the answers the
way one likes. The answers could be correct but it marks them wrong because it cannot find the phrases it’s looking for. It taught me to write bad English to earn marks."

5. General questions posted on the notice board:
   - "Has assignment 1 been marked? I have not received my marks yet."

All the messages posted on the notice board were addressed to the lecturer. Students did not post messages to each other, nor did they attempt to start discussions. Most students appreciated the opportunity of gaining additional credits towards examination admission.

Where students identified problems relating to system usability, these were noted and used to improve the system (see Section 7.3.2). From the feedback received regarding the efficiency of the marking mechanism, it is clear that more research is needed on the intricacies of the search mechanism.

### 7.3.3.2 Performance of students in the 2002 COS321-6 examination

Of 187 students enrolled for COS321-6 during 2002, 129 students wrote the end-of-year examination. Table 7.3 indicates the overall performance of students in this exam. Of these 129, 57 were cyber classroom users and 72 were non-users. Table 7.4 is an extension of Table 7.3 that indicates the performance of cyber classroom users in this examination.

<table>
<thead>
<tr>
<th>104 students (81%) passed the examination</th>
<th>23 students (18%) passed with distinction</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>81 students (63%) passed without distinction</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>25 students (19%) failed</td>
<td></td>
</tr>
</tbody>
</table>

Table 7.3: Overall performance of students in the COS321-6 examination.
These tables indicate the utility and value of Web-Grader in enhancing learners' performance.

The following can be concluded from Table 7.4:

- The majority (57%) of all the distinctions came from cyber classroom users (13 out of 23 distinctions).
- The distinction rate of cyber classroom users (23%: 13 out of 57 users) was higher than the overall distinction rate (18%: 23 out of 129 students) and the distinction rate (14%: 10 out of 72 users) of the non-users.
- The pass rate of the cyber classroom users (77%: 44 out of 57) is comparable to the overall pass rate (81%).

Of the 104 students who passed the examination, 18 students did not have exam admission by the official deadline. They gained admission by using the cyber classroom to obtain additional credits (see Section 7.3.1). Had they not been permitted to write the examination, the pass rate would have been much lower (67% instead of 81%). Table 7.5 indicates the performance of 21 students who had not earned 100 credits by the deadline and used the cyber classroom specifically to obtain additional credits. Only 3 of them did not use the cyber classroom, and consequently were not allowed to write the examination. Almost all of the students who gained exam admission by using the cyber classroom, passed the
examination. The only exception was one student who qualified for a supplementary examination, but did not make use of this opportunity.

<table>
<thead>
<tr>
<th>21 students were allowed to obtain additional credits toward examination admission</th>
<th>18 of these 21 students used the cyber classroom</th>
<th>2 passed with distinction</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 obtained 65% - 74%</td>
<td>10 obtained 49% - 64%</td>
<td>1 failed</td>
</tr>
<tr>
<td>3 did not use the cyber classroom</td>
<td>These 3 did not write the examination</td>
<td></td>
</tr>
</tbody>
</table>

Table 7.5: Performance of students using the cyber classroom to obtain admission to the 2002 examination

7.4 Theory and application in Web-Grader

The research and development undertaken for this study culminated in Web-Grader. Not only has it demonstrated itself effective in improving learners' performance, providing feedback and reducing labour-intensive practices, but it also demonstrated practical application of multi-disciplinary theory. Table 7.6 summarises various educational aspects addressed during the course of this study, which were incorporated into the design of Web-Grader, while Table 7.7 summarises certain software engineering principles implemented in the design and development process. This summary contains only the principles that guided the design of the system, and does not include references to the tools (discussed in Section 5.7.2.1) used to produce the interface.
<table>
<thead>
<tr>
<th>Features of Web-Grader</th>
<th>Description</th>
<th>Sections</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access control</td>
<td>Only enrolled learners should be allowed to participate (Malikowski, 1997; Rasmussen, Northrup &amp; Lee, 1997).</td>
<td>3.2.1.2</td>
</tr>
<tr>
<td>Time-independence</td>
<td>Enquiries can be made from anywhere in the world by using any computer platform at any time of the day or night (Khan, 1997).</td>
<td>3.2.2.1</td>
</tr>
<tr>
<td>Asynchronous</td>
<td>Communication via e-mail and messages on a notice board.</td>
<td>3.2.1.3</td>
</tr>
<tr>
<td>communication</td>
<td></td>
<td>3.2.2.1 (1)</td>
</tr>
<tr>
<td>Interactive learning</td>
<td>Instruction is a purposeful interaction aimed to increase learners’ knowledge and skills. Within this context, simply publishing web pages with links to other digital resources does not institute instruction (Ritchie &amp; Hoffman, 1997).</td>
<td>3.4.2</td>
</tr>
<tr>
<td>Feedback and redoing</td>
<td>Feedback generally refers to educators’ response to learners’ efforts. Meaningful feedback by educators on learners’ efforts can provide scaffolding to the learning process (Plowman, 2002).</td>
<td>3.2.1.5</td>
</tr>
<tr>
<td>of answers</td>
<td>All learning sessions should give and receive feedback about the knowledge being constructed (El-Tigi &amp; Branch, 1997).</td>
<td>3.4.2</td>
</tr>
<tr>
<td></td>
<td>Feedback should enable learners to revise their work and learn from their mistakes (Kintsch et al, 2001; Black &amp; Wiliam, 1998).</td>
<td>4.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4.2.2.4(2)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5.7.2.2</td>
</tr>
<tr>
<td>Immediate feedback</td>
<td>Commercial software products used for automatic essay scoring and automatic grading of computer programming assignments provide immediate diagnostic feedback (Burstein &amp; Marcu, 2000b; Kintsch et al, 2001; Dessus, Lemaire, &amp; Vermer, 2000; Higgins et al, 1999; Foxley, E., Higgens, C. &amp; Burke, E. 2000.).</td>
<td>4.2.1.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4.2.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4.3.2</td>
</tr>
<tr>
<td>Models for guided</td>
<td>The design of technological interactivity has much to learn from human interaction (Plowman, 2002). Black and William’s (1998) four elements making up the feedback system.</td>
<td>3.4.2.1</td>
</tr>
<tr>
<td>interaction</td>
<td>The purpose of learning is to close the gap between achievement and desirable goals (Black &amp; Wiliam, 1998).</td>
<td>5.7.2.2</td>
</tr>
<tr>
<td>Dynamic content</td>
<td>Dynamic HTML is used for displaying information only when it is needed (Needleman, 1997). The main reason for using a server-side scripting language (such as PHP) is to provide dynamic content to a web site’s users, i.e. content that changes according to a user’s needs or inputs.</td>
<td>3.2.2.7</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3.2.2.8 (3)</td>
</tr>
<tr>
<td>Features of Web-Grader</td>
<td>Description</td>
<td>Sections</td>
</tr>
<tr>
<td>------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>Interaction</td>
<td>Programs that run without immediate user involvement are not interactive (Whatis?com, 2002). Interaction means the collaboration between a user and the computer in order to perform a certain task (Marcus, Smilonich &amp; Thompson, 1995).</td>
<td>3.3.3, 5.7.2.2</td>
</tr>
<tr>
<td>Formative assessment</td>
<td>Formative assessment provides feedback to learners on their progress during a course, in ways that enable them to improve their learning (Nicoletto, 2001).</td>
<td>3.2.1.4</td>
</tr>
<tr>
<td>Constructivism</td>
<td>A recent move towards using constructivist theories of learning (Plowman, 2002). Constructivists object to decontextualised evaluation and the separation of evaluation environment from the learning environment (Jonassen, 1992).</td>
<td>3.4.2.1, 3.3.2.1</td>
</tr>
<tr>
<td>Hexa-C Metamodel</td>
<td><em>Web-Grader</em> evidences the three inter-related elements most appropriate for its purpose and function namely creativity, customisation and aspects of constructivism (De Villiers, 1999; 2002).</td>
<td>2.3.2.2 (3)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Features of Web-Grader</th>
<th>Description</th>
<th>Sections</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rapid prototyping</td>
<td>A technique used for quickly building a functioning, but incomplete, model of the information system by using rapid application tools (Whitten, Bentley &amp; Dittman, 2001). It enables the developer to create a working model of the intended software, and to test and modify it. A prototype typically evolves into the final version of the system or application through an iterative process (Chao, 1998)</td>
<td>5.2.2.1</td>
</tr>
<tr>
<td>End-user involvement in the cyclical development of the system</td>
<td>Rapid prototyping is commonly used in interactive multi-media design due to its utility of testing prototypes on end-users. This reduces the risk of abandoning a finished product due to its failure to meet user needs (Chao, 1998; Hoffman and Margerum-Leys, 2002).</td>
<td>5.4.</td>
</tr>
<tr>
<td>Pressman’s generic view of software engineering.</td>
<td>Three generic phases, regardless of application area, project size, or complexity: (1) The definition phase focuses on <em>what</em>. (2) The development phase focuses on <em>how</em>. (3) The maintenance phase focuses on <em>change</em>.</td>
<td>5.5</td>
</tr>
</tbody>
</table>
7.5 Discussion and summary

Positive feedback and exam performance indicate that students benefitted from the cyber classroom. Although some expressed concern about the mechanism of the search technique, they appreciated being allowed to redo answers, and receiving immediate feedback on the marks obtained. The high pass rate (17 out of 18) of those students who initially did not qualify for admission to the examination, indicates the need amongst students for such a facility. As noted in Section 7.3.3.2, its use as a supplementary means of exam admission resulted in a significantly higher pass rate for COS321-6 students and several distinctions. Since tertiary institutions in South Africa are rewarded by Government on the through-put rate of students, it is of utmost importance to implement new strategies for improving pass rates.

The Web-Grader prototype, which was used for the second intervention of this study, did not only produce positive outcomes in the COS321 examination, but it was also designed in line with the current movement of instructional design and development towards interactive learning. The need to move away from static educational web pages to dynamic pages in order to produce more effective interactive web-based learning was identified in Chapter 3. This was accomplished in Web-Grader by storing learning content in a database, thereby providing individualised feedback based on input. Furthermore, Web-Grader was an attempt to improve the traditional labour-intensive approach of grading students' work and providing detailed comments to each individual student, as well as to integrate the feedback process with an enriched learning process (as proposed in Sections 3.4.1, 3.4.2, and 4.2).

Although Web-Grader does not address all of the drawbacks of current educational software mentioned in Section 3.3.3.3, it was a positive move in the direction of addressing the problems unique to distance education environments mentioned in Section 6.2. Examples of such problems are studying in isolation, a lack of immediate feedback on material submitted to the university, and a need of students to study at their own pace according to their own time schedule.

The evaluations described in Section 7.3.2 identified some deficiencies, which were corrected, and further modifications were done as enhancements. Further research is required on intricacies of the search mechanism, but overall Web-Grader performed well as a study support tool. Whereas E-Grader supported academics by reducing work load in assessment of textual answers, Web-Grader supported both academics as a productivity tool and students in its role as an interactive learning support system.
It was stated in Section 3.4.1.3 that development of policies, methods, and technologies for alternative assessment should keep pace with the development of constructivist learning environments. Although Web-Grader was not a pure constructivist system, it was a move towards constructivism - in the sense that it addresses Fetherston's (2001) challenges regarding constructivist assessment (see Section 3.4.1.3). He challenges educators to construct assessment systems that can assist learners to find out what they know or do not know. He believes that current assessment practices should change from the traditional assignment/exam system, to systems based on giving learners greater opportunity to portray what they know. He further believes that the interactive and multimedia capabilities of the WWW have the potential to play a meaningful role in the provision of constructivist assessment, and that the utilisation of these capabilities can lead to many creative assessment strategies.
References - Chapter 7


Plomp, T. 2002. *Some reflections on development research (DR)*. Notes for a lecture presented at the University of Pretoria. Enschede, the Netherlands:University of Twente.


Chapter 8: Summary and conclusion

This chapter commences with a brief summary of what has been achieved. In Section 8.2 the research questions posed in Chapter 1 are answered, followed by recommendations for further research in Section 8.3. A final conclusion is drawn in Section 8.4.

8.1 What has been achieved

The needs and context analysis conducted on web-based learning in Chapter 3, indicated the necessity for web-based learning to move away from static educational web pages to dynamic web pages in order to produce more effective interactive learning. In line with Plomp’s (2002) development research model (see Figure 1.1 in Section 1.6.1), a cyclical design and development process was used to produce two variants of a prototype system used for e-assessment of textual inputs. To ensure that the system produced for this study is based on sound theoretical principles, and that an appropriate match is achieved between technology and learning theory, findings from the investigation undertaken in Chapter 2 on the implications of learning theories (such as behaviourism and constructivism) on technology and learning, were incorporated into the development.

The cyclical design and development process and interventions undertaken for this study comprise the design, development, implementation and evaluation of two variants (E-Grader and Web-Grader) of a prototype assessment system that enables automatic marking of open-ended textual responses.

The first variant called E-Grader, was developed to initially test the concept of electronically assigning marks to questions from students’ assignments entailing short open-ended textual answers.

The second variant, an interactive web-based extension of the first variant, called Web-Grader, was implemented as a cyber classroom. This prototype was implemented as a study support tool that assigned marks to open-ended answers keyed in by students, and provided interactive feedback on how to improve their efforts. The questions were extracted by students from a
database containing a sample of questions from previous assignments. *Web-Grader* was, as proposed in Chapter 3, an interactive learning system that:

- focused on learning as a process,
- provided opportunities for communication and interaction,
- integrated assessment with instruction, thereby enabling performance to be sampled repeatedly over time, and
- was designed according to cognitive principles

In line with the cyclic design process of development research, findings of the evaluations were used for corrections and enhancements. Results of students' reverse feedback indicated appreciation of the tool. Certain problematic issues were identified, which were modified accordingly by the researcher (as designer and programmer).

In Section 8.2, the results (outcomes) of these interventions are used to illustrate the answers obtained to the research questions posed in Section 1.2.

### 8.2 Answers to the research question

The main research question that guided this study was "*To what extent can e-assessment of textual inputs enhance web-based learning?*"

In order to answer this question, the two subquestions posed in Section 1.2.2 are addressed.

The first subquestion, "*How do the marks (grades) obtained using the first variant of the prototype e-assessment tool correlate with marks assigned by human assessment?*", was answered in Chapter 6 by comparing the scores obtained by *E-Grader* to hand-marked assignments. It was found that *E-Grader* returned grades that correlate statistically significantly with those assigned by an external marker. This can be interpreted as an encouraging indication that the e-assessment tool can be used in the real world without undue risk of obtaining grades radically different to grades obtained by hand-marking.
Chapter 8: Conclusion

The second subquestion, "What was the value of interactive feedback provided by the second variant (i.e. the web-based version) of the prototype to learners?", was answered in Chapter 7. This system was not only an attempt to alleviate the traditional labour-intensive load of distance educators who are required to give elaborate diagnostic feedback and comments to each individual student, but it was also an attempt to integrate the feedback process with an enriched learning process. Positive feedback from students indicated that they found the system useful. Positive examination results obtained by users of the cyber classroom, suggest that the learning process was indeed enriched and that students benefited from using the system. It was further found that students used the system to interact on specific issues with the lecturer, and that it was not used as a vehicle to interact with each other.

What can be concluded with regard to the main research question "To what extent can e-assessment of textual inputs enhance web-based learning?"

Firstly, from the perspective of distance educators, it appears that e-assessment of textual inputs can be used effectively and relatively safely as an aid to grading assignments and to ease the burden of providing individual feedback to students. From the perspective of the student, web-based assessment tools can enhance learning by providing not only asynchronous access to the learning environment, but also immediate supportive feedback to their inputs.

8.3 Further research

Recommendations for further research include the following:

- Using the generic, content-independent Web-Grader prototype to present other cyber classrooms.

- Extension of the pattern-matching technique - this can be done by refining the current keyword search technique by:
  - including more regular expressions, and
  - adding third party software containing artificial intelligence techniques.
Chapter 8: Conclusion

- Implementing the grading system on computer programming assignments.

- Extension of the feedback mechanism by adding links to other web sites on the WWW containing resources related to the content of the question, thus providing multiple perspectives.

- Students' textual inputs in the form of contributions to discussion forums and e-mails can be monitored and analysed electronically over a period of time.

- Although students were unaware of it, all their scores (even scores not intentionally saved by them) were automatically recorded and stored by the system. This information can be used for measuring aspects such as learning gain.

- After completion of the second intervention at the end of 2002, many of the new students enrolled for COS321-6 in 2003, requested the continuation of the cyber classroom. Results from this implementation can be compared to results from the COS321-6CyberClassroom2002.

8.4 Conclusion

This study produced two variants of a prototype e-assessment tool that both use pattern-matching to assess open-ended text-based inputs. It was found that both these variants function according to fixed principles that remain constant over time and in different applications. They implement "anonymous marking", thereby reducing the effects of stereotyping and prejudice, which increases objectivity and reliability. From the high correlation between scores assigned by E-Grader and marks allocated by human graders, as well as the learning support benefit and positive examination results of cyberclass users, it can be concluded that this tool provides valid and reliable assessment of short open-ended textual input.
Appendix A: Main program used by Web-Grader

The main program developed by the researcher to be utilised by Web-Grader for automatic grading of open-ended textual responses to questions, is provided here. This program interacted with 16 other programs (also developed by the researcher).

```php
<? php //cosresult.php 23/03/2002 Web-Grader - main program used for grading
<html>
<script language="JavaScript">
<!--
function myopen(filename){mywindow = window.open(filename,);

function winopen(filepluspar){
window.open(filepluspar, "newWindow",
"toolbar=no,location=no,directories=no,status=no,menubar=no,scrollbars=no, resizable=yes, copyhistory=yes,width=400,height=430")
}

function winopen3(fileplusparameters){
window.open(fileplusparameters, "newWindow", "toolbar=no,location=no,directories=no,status=no,menubar=no,scrollbars=no, resizable=yes,copyhistory=yes,width=220,height=250")
} //-->
</script>

<body Background="/cospics/foggy2.gif">
	<table border="0">
	<tr><td><img src="/cospics/bbanner.gif" valign=top width="100%">
	</td></tr></table>
<
	echo ' 
	</center><table> <tr>
	<td background="/cospics/blokkies.gif">
	<a href="/coshome.phtml?user='.$_user.'">
	<img src="/cospics/bhome.gif" border="0" /></a></td>
</tr>

<td background="/cospics/blokkies.gif">
<a href="/coswelcome.phtml?user='.$_user.'">
<img src="/cospics/bwelcome.gif" border="0"></a></td>
</tr>

<td background="/cospics/blokkies.gif">
<a href="/cosboard.php?user='.$_user.'">
<img src="/cospics/bnotice.gif" border="0"></a></td>
</tr>

<td background="/cospics/blokkies.gif">
<a href="/cosmenux.php?user='.$_user.'">
<img src="/cospics/banotherk.gif" border="0"></a></td>
</tr>

<td background="/cospics/blokkies.gif">
<A HREF="/cosanswerx.php?user='.$_user.'&Qtel=16"/>
<img src="/cospics/bredok.gif" border="0"></a></td>
</tr>
</table>
</body>
```

234
Appendix A: Main program used by Web-Grader

```php
<?
$db = mysql_connect("localhost", "root");
mysql_select_db("php_martjie", $db);
$total = 0;
$str_query = "select * from CC_QuesAns where Qnumber=$Qtel";
$resultr = mysql_fetch_array(mysql_query($str_query));
$resultx = $resultr["Keywords"]; // split into words
$array = split(",",$resultx);
$tot = count($array);
$target = $Answer;
$flag = "0";
$max_query = "select * from CC_QuesAns where Qnumber = $Qtel";
$max_result = mysql_fetch_array(mysql_query($max_query));
$question = $max_result["question"]; // get question
$max = $max_result["total_marks"]; // get max marks
$half_max = $max/2;
$book = $max_result["book"]; // get book page
$page = $max_result["page"]; // get page

echo ' <TABLE cellpadding="8" border="0" valign=TOP width=100%> <TD align=center valign=top width="80%"> ';

function kyk ($Sitem1)
{ // compare input words with keywords
  global $max, $half_max, $total, $target, $flag;

  $pattern = $Sitem1;
  if (preg_match("/$pattern/", $target))
  {
    $total = $total + 1;
  } else if if not found
  {
    $flag = $flag + 1;
    if (($flag == 1) and ($total < $max))
    {
      print( '<img src="/cospics/bril.gif" valign=top align=center height="10%" width="10%">');
      echo ' <br><br> ';
      echo '<FONT SIZE="5%"> ';
      echo "You did not mention:";
      echo '</FONT> ';
    }
  }
}
```
if ($total >= 0) {
    echo '<LI><FONT COLOR="red">';
    $karray = split('\|\|', $item1);
    $kkarray = split(' ', $karray[0]);
    $tel = count($kkarray);
    if (count($kkarray) > 1) {
    } else {
        echo "$kkarray[0] $extra<br>");
    }
    echo '</LI></FONT>";
}
}
} # end of function kyk

function kyk_not (&$item2) {
    # search for "not" before a keyword
    @global $total, $target;
    $pattern = " not ($item2)";
    if (preg_match("/\$pattern/i", $target)) {
        echo "your total mark was $total, but one mark was subtracted for using 'not' before $item2<br>");
        $total = $total - 1;
        echo "total is now $total.<br>");
    }
} # end of function kyk_not

echo '<FONT COLOR="blue"SIZE="5%"> <b>Q</b>';
echo "Question $Qtel: $Rquestion"
    echo '<br>';
echo "[{$max Marks}]"
    echo '</BR><BR></b></FONT>'

# format output
$replacement = " ";
$newpattern = "", "
$newresult = preg_replace("/\$newpattern+/i", $replacement, $resultx));
$narray = explode(" ", $newresult);
$narray = array_reverse($narray);
$n_target = explode ("-", $target);
$x = 0;
$reg_vlag = 0;
$out = 0;

while ($x < count($n_target)) {
    $y = 0;
    $isKeyword = 0;
    $here = 0;
    while ($y < count($n_array)) {
        $target2 = $n_target[$x];
        $pattern2 = $n_array[$y];
        if (!preg_match("/\$pattern2/i", $target2) and !preg_match("/\$pattern2/i", "\$*")) {
            $isKeyword = 1;
            $here = $y;
            $reg_vlag = $reg_vlag + 1;
        }
    }
}
} $y = $y + 1;
}
if ($isKeyword == 0) {
    echo "$u_target[$x]." ";
} else
{
  echo '<FONT COLOR=blue><b>
    echo "$u_target[$x];
    echo '<b>';
    echo '</b><FONT>';
    $u_array[$b] = "xxxxxx";
  }

$x = $x + 1;
}
if ($reg_vflag >= 1) {
  echo '<br>;
  print('<img src="/cospics/reg.gif" border="0" height="5" width="5">');
  echo '<br>;
  echo '<table border="4" bgcolor="white" width="80%" valign=BOTTOM>';
  echo '<tr><td valign=BOTTOM>';
  echo '<CENTER>';
  array_walk($aArray,'kyk');</n  array_walk($aArray,'kyk_not');</n  echo '</br>;
  if (($flag > 0) and ($total == 0) ) {
    $out = 1;
    echo "Your total is very low!";
    echo "<script language="JavaScript">\n    winopen 'coszerox.php?user=".$user."&Qtel=".$Qtel."&target=".$target."&total=".$total."</script>";
    exit;
  }
  if ($total > $max) ($total = $max);
  echo '<FONT color="red" size="5">';
  echo "$total out of $max";
  echo "Your total mark for Question $Qtel is:";
  echo '<br>;
  echo '<br>
  echo '</FONT> </CENTER>'';
  echo '<br></tr></tbody>
  echo '</table>';
  $target = urlencode($target);
  echo "<a href="javascript:winopen3('cosav1.php?user=".$user."&Qtel=".$Qtel."&target=".$target."&total=".$total."')"<img src="/cospics/bioustif2.gif" border="0" align="center"></a>";
  echo '</td><br>);
  if ($total == $max) {
    echo '<TD width="20%" rowspan="2" bgcolor="ea8d6b">'';
    echo '<br><center>';
    include '/xplode.inc';
    print('<img src="/cospics/prys.gif" align="center">');
    echo '</br></center>'';
    echo '</TD></TR>'';
  }
$$today = date("Y-m-d");
$sql = "INSERT INTO CC_store (Qnumber, username, date, Sanswer, mark, flag) VALUES('$Qtel', '$user', '$today', '$target', '$total', 'N');"
$result = mysql_query($sql);
if (($total > 0) and ($total < $max)) {
    echo '<td rowspan="2" bgcolor="#e8d6b">';
    echo '<br><br><br><br>'.
    include './/bulb.inc';
    echo '<br>';
    print('<img src="/cospics/anlpages.gif" valign="top" align="right">');
    echo '<br><br><br><br>';
    echo "Please consult <b><i>$book $page</i></b> and ";
    echo '<a HREF="/cospanswerx.php?user='. $user .'&amp;Qtel='.$Qtel.'">try again</a>';
    echo '<br><br><br><br>';
    echo '</td></tr>';}
echo '<TR> <TD COLSPAN="1"bgcolor="#bab8f9">&nbsp;';
echo '<a href="cosmenux.php?user='. $user .'"> <img src="/cospics/banother.gif" border="0"></a>';
echo '<img src="/cospics/Spacer.gif" border="0">';
echo '<a href="cosanswerx.php?user='. $user .'&amp;Qtel='.$Qtel.'"> <img src="/cospics/bredo.gif" border="0"></a>';
echo '</TD></TR>';
echo '</TABLE></strong></font></CENTER>';echobr>echobr>echoa href="mailto:conramm@unisa.ac.za" <img src="/cospics/email2k.gif" border="0">';
echo '<a href="/coshome.phtml?user='. $user .'"> <img src="/cospics/homek.gif" border="0" height="15%" width="15%"></a>';
echo '</center>';?〉</body></html>
### Appendix B: Raw data - Implementation of E-Grader

#### Assignment 1

<table>
<thead>
<tr>
<th>Student</th>
<th>Question 1.2</th>
<th>Question 2.1</th>
<th>Question</th>
<th>External marker</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>E-grader</td>
<td>E-grader</td>
<td>E-grader</td>
<td>External</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>3</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>7</td>
<td>3</td>
<td>1</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>3</td>
<td>2</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>11</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>12</td>
<td>0</td>
<td>2</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>13</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>14</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>15</td>
<td>0</td>
<td>3</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>16</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>17</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>18</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>19</td>
<td>3</td>
<td>3</td>
<td>7</td>
<td>6</td>
</tr>
<tr>
<td>20</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>21</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>22</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>23</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>24</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>25</td>
<td>1</td>
<td>3</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>26</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>27</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>28</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>29</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>30</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>31</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>32</td>
<td>2</td>
<td>2</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>33</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>34</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>35</td>
<td>1</td>
<td>3</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>36</td>
<td>3</td>
<td>1</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>37</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>38</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>39</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>40</td>
<td>0</td>
<td>2</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>41</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>42</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>43</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>44</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>45</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>0</td>
</tr>
</tbody>
</table>

* For ethical reasons, student numbers are not provided
### Assignment 3

<table>
<thead>
<tr>
<th>Student *</th>
<th>E-Grader</th>
<th>External marker</th>
<th>Self</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Q1.4</td>
<td>Q5a</td>
<td>Q5b</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>11</td>
<td>2</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>12</td>
<td>1</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>13</td>
<td>2</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>14</td>
<td>1</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>15</td>
<td>2</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>16</td>
<td>1</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>17</td>
<td>1</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>18</td>
<td>2</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>19</td>
<td>3</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>20</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>21</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>22</td>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>23</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>24</td>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>25</td>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>26</td>
<td>2</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>27</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>28</td>
<td>2</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>29</td>
<td>2</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>30</td>
<td>3</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>31</td>
<td>3</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>32</td>
<td>3</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>33</td>
<td>3</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>34</td>
<td>2</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>35</td>
<td>3</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>36</td>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>37</td>
<td>1</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>38</td>
<td>2</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>39</td>
<td>3</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>40</td>
<td>3</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>41</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>42</td>
<td>1</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>

* For ethical reasons, student numbers are not provided
Bibliography


Available: http://webct.uga.edu/hostsys/cumrec/cumrec98.htm


241


246


Plomp, T. 2002. Some reflections on development research (DR). Notes for a lecture presented at the University of Pretoria. Enschede, the Netherlands: University of Twente.


Cliffs, New Jersey: Educational Technology Publications.


249


250