The University of Nottingham
School of Computer Science and Information and Technology

Automatically generating adaptive programming exercises based on student profiles

By Chye Foong Yong

BSc (Computer & Information Systems) &

MSc (Erogonomics)

Thesis submitted to the University of Nottingham for
the degree of Doctor of Philosophy, February 2007

Acknowledgements

First and foremost I wish to thank my supervisor, Colin Higgins, for his guidance throughout the course of this research, and for the editorial feedback during the preparation of this thesis.

To the members of the LTR group, who have been very helpful in providing new ideas and support through the years: Athanasios Tsintsifas and Pavlos Symeonidis for their inspiration and advice; Geoff Gray for his technical help in extracting the marking server from CourseMarker; and the ladies of the LTR group for their support, especially in the last two years.

To the students and staff in the School of Computer Science and IT at the University of Nottingham, for their assistance and patience during the trial period.

A special thanks to Debbie Dickson for letting me stay in her house for three years, despite our initial agreement of just two weeks.

Last but not least, I am highly indebt to my family – my parents and younger brother – for their support during my five-year residence in the UK.

Thank you all.

Table of Contents

List of Figures

List of Tables

Abstract

In assessment it is usual for students to be presented with the same sets of questions, regardless of their individual ability. That is, the questions are not tailored to the students' individual performance, understanding and ability, and thus may be either too easy or too difficult for the student. However, computerised-adaptive testing (CAT) systems present questions of appropriate difficulty for the student based on their ability, allowing the student to progress at their own pace.

This thesis presents such an adaptive system that aims to encourage students to self-assess, including in the form of programming question type, and to enhance their learning. The ASAM system generates adaptive exercises for each student according to their individual profile. With the help of personalised feedback received upon submission, students can identify their weak areas. ASAM also aims to equip teachers to better monitor students' progress, especially in large classes. This can be achieved by analysing the statistics generated from the student profiles.

The ASAM system was used by some computer science students at first year undergraduate level at the University of Nottingham in 2003 and 2005. An evaluation study was carried out in each round of user trial, attempting to evaluate the system from several perspectives, including socio-technical, pedagogical and technical perspectives. The outcome reported by the studies is a positive encouragement for future research into adaptive self-assessment systems and their integration with other computerised learning tools.

Chapter 1 –

Introduction

When we learn, questions are always being asked – they may be either posed by teachers or formulated by us ourselves (Morgan and Saxton, 1994). When we ask questions such as ‘How am I doing?’, ‘Is this enough?’, ‘Is this right’, ‘How can I tell?’, ‘Should I go further?’, we are judging ourselves and making decisions about the next step. In other words, we are self-assessing (Boud, 1995).

Assessment, including self-assessment, is part and parcel of the learning process. In particular, through well-designed assessments, students are encouraged to become better able to reflect on and evaluate their own work and performance (McDowell, 1996; Erwin, 1995; Brown and Knight, 1994), as the design of assessment can enable students to be engaged in deep learning, to find an intrinsic motivation and a sense of satisfaction in their learning, to develop a range of skills and to become more aware of means of assessing and judging their performance (McDowell, 1996).

With ever-increasing class sizes, it is becoming extremely difficult for the lecturer to attend to the needs of every student. This leads to a need for computer-based test (CBT) systems, so that students can be assessed and get immediate feedback (Tsintsifas, 2002; Pollock, Whittington and Doughty, 2000; Brusilovsky and Miller, 1999; Gibbs, Habeshaw and Habeshaw, 1988).

However, standard sets of questions for all students can be a disadvantage. The questions can be too easy or too challenging to different students at different points of time, for students do not all learn at the same rate. The principal aim of this thesis is to provide a means for students to self-assess at anytime, anywhere they wish and at a suitable level.

Section 1.1 documents the objectives and aims for enabling students to self-assess as and when they wish and at their appropriate level of knowledge, and teachers to monitor students' progress, in spite of the increasing (and seemingly unmanageable) class sizes. In Section 1.2, this need to build an adaptive self-assessment system is put into perspective by examining the benefits of using CBT systems whilst also highlighting some drawbacks, leaving the outline of this thesis to Section 1.4.

1.1 Objectives and aims

It is the ultimate aim of this research to assist students to self-assess at a suitable level of difficulty.

During the research, attempts were made to answer some questions: To what extent does self-assessing help in the learning process? To what extent do adaptive self-assessments aid students in their learning, as compared to adaptive summative assessments? What form/s of feedback do students appreciate? Are marks or comments (or both) sufficient? How have researchers designed the adaptivity component in existing computerised-adaptive testing (CAT) systems? Are they easy to implement? How can student profiles be captured to enable the adaptivity to function correctly?

To answer these questions, a literature review was carried out, from both pedagogical and technical perspectives. A system was developed, entitled ASAM or “Adaptive Self-Assessment Master”, to generate personalised exercises[1] automatically for each student, based on their profile[2] (Yong and Higgins, 2003, 2004). Besides fixed response questions, this system also allows the marking of programming questions.

Students used ASAM to self-assess at the University of Nottingham. The system also aided the teaching staff to monitor students’ progress. The results gained from the students using the system to self-assess are presented, together with a socio-technical evaluation of the usage. Both types of analysis are important contributors to assess whether the objectives set out for this thesis have been accomplished.

1.2 Background and motivation

This section presents the background and motivation for delivering an adaptive system that generates personalised exercises for each student, depending on their abilities.

[1] A collection of questions is referred to here as an exercise.

[2] A student profile may include things such as the student’s background, their level of knowledge on various topics of a particular subject, the topics or subjects that they have already covered, and so on.

1.2.1 *Background*

For the last ten years, the School of Computer Science and Information Technology, at the University of Nottingham, has been using a CBT system, CourseMarker (formerly known as CourseMaster), to mark first-year students' solutions to programming exercises (Foxley *et al*, 2001; Higgins, Symeonidis and Tsintsifas, 2002). Prior to this, Ceilidh, the predecessor of CourseMarker, had been utilised for the same purpose for more than ten years.

CourseMarker allows the students to submit their solutions to programming exercises multiple times, and to obtain feedback and marks instantaneously. The maximum number of permissible submissions is determined by the teacher. This approach has been found to be beneficial to both students and teachers. The students can work on their solutions again and again, based on the feedback given by the system. Dalziel points out that regular self-testing and feedback during learning via computer-aided assessment (CAA) may significantly enhance overall learning outcomes at both discipline specific and generic levels (Dalziel, 2001).

In addition, it is noted that the marking criteria is consistent with all work submitted, which means that no one piece of work is marked unfavourably. At the same time, the teachers can free more time, which has been made available from not having to mark many sets of programming code every week, while also having an administrative tool for monitoring students' progress and detecting plagiarism. This excess time could be put more effectively into teaching.

1.2.2 *Motivation behind the research*

While there are benefits for using such a system, there are still some shortcomings. In standard CBT systems, the questions presented to the students are fixed for the cohort and are not tailored according to the students' individual performance during the test. As a result, students can be presented with questions that are either too easy or too difficult (Lilley and Barker, 2002; Eberts, 1997). As a result, researchers began to look into CAT systems. Such systems administer an examination with questions of appropriate difficulty for the student based on his/her ability (Syang and Dale, 1993).

A common form of adaptive testing involves a computer-administered test in which the presentation of each test item and the decision to stop the test are dynamically adapted to the student's performance in the test (Abdullah, 1999; Lilley and Barker, 2003). The most common stop conditions are:

- a certain number of questions have been administered;
- a time limit has been reached; and
- a certain standard error for the ability has been met.

The advantages of this form of adaptive testing are:

- it presents to the student only those test items which are likely to yield additional information;
- it can distinguish careless slips and guesswork from actual knowledge by repetitive testing; and

- it can arrive at an initial student model quite accurately in a relatively short time.

Notwithstanding these advantages, this form of adaptive testing has some drawbacks particularly as most CAT systems are used for summative assessment. Research carried out by Lilley and Barker (2002, 2003), comments that the students would be more receptive to the use of such a CAT system in a formative rather than in a summative assessment environment. This suggests that lecturers foresee problems concerning the score method used within CAT systems. In CAT systems, the final score given to a student is typically based on the number of questions answered correctly and incorrectly, as well as on the level of difficulty of these questions. As a result, students who have answered the same number of questions correctly would almost certainly have different final scores which can bring uncertainties about the “fairness” of the assessment.

Also, in their reports, Lilley and Barker highlighted that their expert evaluators (namely some lecturers in Computer Science and in English for Academic Purposes, at the University of Hertfordshire) believed that their prototype would give greater assistance in formative rather than in a summative environment. The expert evaluators suggested that formative assessments provide the lecturers with more information regarding the students’ strengths and weaknesses, since they are typically undertaken on a regular basis.

It is these concerns that bring the attention of the author to the need of an adaptive self-assessment system. The end result is a system entitled “Adaptive Self-Assessment Master” or ASAM, for short.

1.3 Scope and outline of thesis

This thesis both addresses the issues faced by educators in teaching students in ever-increasing class sizes, and the need of students' self-assessing and obtaining of subsequent feedback, especially in such conditions. Upon reviewing what kinds of computer-based educational tools are available to students (and teachers) in Chapter 2, a comprehensive understanding of the recent computer-based teaching and learning tools, with particular focus on adaptive techniques, is gained. Chapter 3 looks at the pedagogical perspective of assessment, including the different roles that formative and summative assessments play in the learning process. It has also drawn to our attention that feedback is essential, in such a way that it can enhance learning.

Chapter 4 presents a tool that can assist in the teaching and learning tasks of teachers and students, generating personalised exercises (containing some fixed response questions) for each student based on their current profile. It documents the aims and motivation for this research, as well as the design and the new tool – ASAM. Chapter 5 details an enhancement to the tool, the end result of which can also facilitate the marking of programming questions. In addition, a comparison with the more conventional CAT systems and the non-adaptive CM system is made. Chapter 6 describes some evaluation study on the use of the system. An outline of the findings and their implications are discussed too. Finally, this thesis is concluded with Chapter 7, an examination on whether the objectives set have been achieved, followed by suggestions for possible future work to extend the current research.

Chapter 2 –

Computer-Based Learning

This chapter presents a literature review of computer-based learning. Three areas of education technology are covered: the various forms of web-based education (section 2.1); the various types of computer-based tests (section 2.2); and the computerised-adaptive tests (section 2.3).

In the first section, the focus is on integrated learning systems, particularly educational portals and intelligent tutoring systems (with some example applications). A brief review on adaptive hypermedia and user modelling is also included.

The second section covers the various types of questions that computer-based tests can assess. These include fixed response and free response questions.

Next, computerised-adaptive tests are explored, specifically on how they are typically generated. At the end of the section some concerns about the usage of this type of tests are highlighted.

2.1 Web-based Education

Computers and the World Wide Web (hereafter referred to as the Web) are common tools in education, at all levels, from primary to university to lifelong learning. In particular, an ever increasing number of web-based educational applications have been developed in the last decade, as it has been viewed that such applications can be accessed at any time, anywhere – a highly valued advantage to various education stakeholders, including teachers and students (e.g. Brusilovsky, 1999). These applications are used in various aspects of education, including teaching, learning and assessments.

When educational applications first gained popularity in the 1980s, they came in a variety of forms, from simple DOS applications to multimedia courseware[1], mainly running on "standalone" computers (often using CD-ROMs) or within local networks (e.g. Eberts, 1997). Multimedia courseware, for instance, were developed using authoring tools, such as Asymetrix Toolbook[2] or Macromedia Director[3].

Despite many initial hesitations in accepting the new teaching tool, as for many other new technology implementations (Eason, 1988; Dearing, 1997), both teachers and students found many advantages to these computer-assisted instruction (CAI) applications (Eberts, 1997; Freedman and Relan, 1990; Swan and Mitrani, 1993). Among the advantages is interaction efficiency. Table 2.1 shows some advances in interaction efficiency – the number of hours required to spend on programming has significantly reduced over the years. Very often, the resultant courseware can be reused, and modification to the content is not a

[1] Courseware is special-purpose software that is authored by a subject-matter domain expert or a computer programmer to provide the teaching instructions. It can take many forms.

[2] Refer to www.asymetrix.com for more information.

[3] Refer to www.macromedia.com for more information.

difficult task. It is little wonder that CAI applications are popular. A brief history on CAI can be found in Eberts (1997).

Language	Dates Available	Programming hours needed per instruction hour provided
FORTRAN	Before July 1967	2286
Authoring in batch mode	Before October 1968	615
Authoring online	After 1968	151
Authoring with menus and prompts	1990	18
Presentation software	1994	4-6

Table 2.1: Advances in interaction efficiency (Source: Ebert, 1997)

Not only had computers begun to be used for the teaching task, they were also used for administrative tasks of instruction, such as record keeping, updating of grades, grading of tests, and so on. With these mundane tasks performed by the computers, teaching staff could use the released time more productively to, say, do what they do best, which is to provide individual help to the students and to prepare the lessons (Eberts, 1997; Johnson and Yen, 1990).

As the usage of the Web increased, web-based applications became more popular. One of the main benefits of web-based education (WBE) is its independence of location (Brusilovsky, 1999). That is to say, students do not have to be in the classroom to access the application, so long as they have access to the Web. In addition to access is also platform independent too, that is, the applications can be run on Unix, Windows or Macintosh machines.

2.1.1 *Integrated learning systems*

When the Web was first used for educational purposes, students used it to search mainly for factual information. Gradually, independent developers created facilities for discussion, chat and other tools to aid communication among web users. It was only some years later that integrated learning systems (ILEs), such as WebCT[4] and Blackboard[5], came into existence.

ILEs not only consist of learning material, but may also include the message of the day, discussion forum, some simple mode of assessments and even chat facilities. The teaching staff can choose any combination of the available facilities to form the resource base for their modules. From the educational institutions' point of view, there is also the advantage of restricting students' access to only those modules that they are taking, instead of all the resources for all modules on the ILE hosting server. This cuts down on time and resources on the implementation and hosting of learning material, particularly when there is a central hosting server in the institution, as compared to one for each department or school.

Such tools are useful to those who intend to deliver virtual university courses (as for UK eUniversities Worldwide[6], Universitas 21 Global[7] and similar e-learning service providers), as their needs include presentation of learning

---

[4] Refer to http://www.webct.com for more information.

[5] Refer to http://www.blackboard.com for more information.

[6] Refer to http://www.ukeuniversitiesworldwide.com for more information.

[7] Refer to http://www.u21global.com for more information.

material, activities, communication and administration (Brusilovsky and Miller, 2001), and their student population can be spread all over the world. All these needs can be easily made available using ILEs. This is a great improvement over the three main general ways used when distance learning was first introduced. At that time, lectures were videotaped and delivered to students at remote sites; live video and audio lectures were sent over satellite systems to sites that had a satellite hook-up; and, simple instruction were made available through the Web (Eberts, 1997). Now, resources made for the first two approaches can be delivered via the Web, since advanced technologies, such as broadband connection as well as the appropriate plug-ins for the browser, make it feasible and convenient (IDA, 2002).

Thus, one can say that the Web and computers are becoming more indispensable in education, especially for distance students.

2.1.1.1 Educational portals

WebCT, Blackboard and similar ILEs incorporate various teaching and learning tools together. While teachers can choose to use, or not to use, whichever of these tools they deem appropriate, they do not have the option of adding any third-party tools into the system. Portals are an alternative to ILEs that allow users to integrate whichever tools they want. uPortal is one such portal that implements this open concept (uPortal, 2001). It is a framework for an integrated delivery of content gathered from an assortment of information sources. Its primary function is to provide an efficient and flexible engine for assembling a presentation. Given a set of information sources (known as channels), and a recipe on how to arrange and frame them (known as stylesheets), the uPortal framework coordinates the compilation of the final document.

An advantage of portals is that they are a personalised and customised gateway designed for useful and comprehensive access to information, people and processes (UoN IS, 2003). For instance, the University of Nottingham has recently developed a student pilot portal using the uPortal technology. It is aimed to be the ‘one stop shop’ for a student, containing an easy and coherent route through to all the online information and services relevant to their life, whether academic or as a member of the Nottingham community. It provides links to the e-learning resources for each module and allows access to the administrative functions, such as notifying changes of address, or checking up on fee status. It contains a mix of channels, some of which are obligatory, but many of which may be chosen by the student, ranging from societies’ information to local entertainment. In Figure 2.1 we can see a screen shot of what the student pilot looks like in a browser window.

At the time of writing, the university is also working on a staff portal, providing similar services to support their teaching and research duties.

2.1.2 *Intelligent tutoring systems*

A large number of Web-based courses and other educational applications have been made available on the Web within the last ten years or so. The problem is that most of them are nothing more than a network of static hypertext pages (Brusilovsky, 1999). They provide the same page content and the same set of links to all users. If the student population is relatively diverse, a traditional

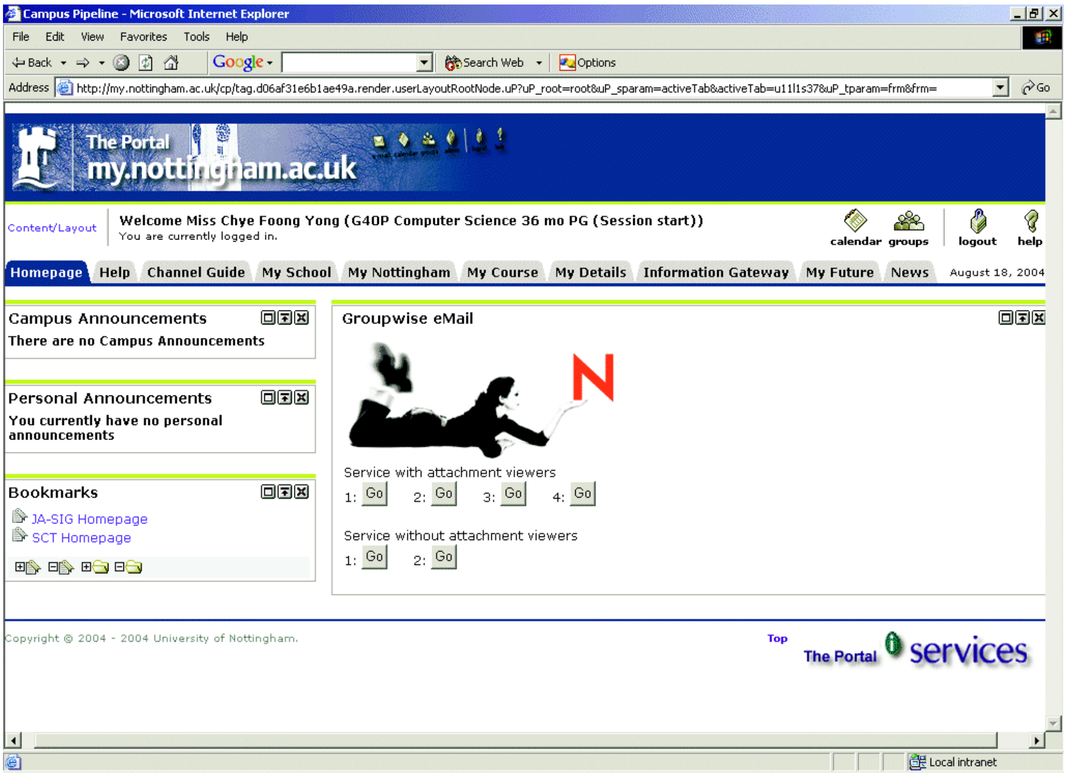


Figure 2.1: A screen shot of the student pilot portal developed by the University of Nottingham

system will "*suffer from an inability to be all things to all people*" (Brusilovsky, 2000, 2001). Eberts (1997) shares the same concern, emphasising individual differences of the students, such as how best they learn (that is, their learning styles) and whether the subject is a major or elective of the students.

Since many students usually work on their own, often from home, the development of advanced Web-based educational applications is highly desirable. As highlighted in Section 2.1.1, this is particularly important for students taking distance courses. Here intelligent and personalised assistance from teaching staff or a peer student that can be obtained in a normal classroom situation is not easy to get. Fortunately, intelligent tutoring systems

(ITSs) have come into existence (Brusilovsky and Nijhavan, 2002; Brailsford *et al*, 2001, 2002; Abdullah, 1999; Calvi and De Bra, 1997).

In fact, research and development efforts in ITSs have been evident since the late 1970s and early 1980s (Eberts, 1997). However, at that time, while there had been some notable success stories in the application of artificial intelligence to education, intelligent tutoring had relatively little impact on education and training in the real world. Corbett and colleagues suggest that there are several reasons for this lack of penetration (Corbett, Koedinger and Anderson, 1997). ITSs are expensive to develop, and until relatively recently the necessary computing power was expensive to deploy. However, Corbett and colleagues believe that "*an even more important reason can be traced to the ontogeny of the field and the consequences for software evaluation* (Corbett, Koedinger and Anderson, 1997, p. 850)." The creative vision of intelligent computer tutors has largely arisen among artificial intelligence researchers rather than education specialists.

So, what is an ITS? According to Roberts and Park (1983, in Syang and Dale, 1993), ITSs are "*instructional systems that resemble what actually occurs when a student and a teacher sit down one-to-one and attempt to teach and learn together* (p. 53)." Research on ITS thus far focuses on computer tutors that adapt to individual students based on the target knowledge the student is expected to learn and the presumed state of the student's current knowledge (Wenger, 1987, in Rickel *et al*, 2001).

Fundamentally, the components of an ITS are the subject matter to be taught, a method of representing what the student does and does not know, and the tutoring strategies (Syang and Dale, 1993; De Bra, 2002; Hartley, 1973, in Laurillard, 1993; Eberts, 1997).[8] These together enable the system to decide whether a learner is “ready” to study a new concept. For every link that appears on a Web page, the system can tell the learner whether the link leads to interesting new information, to new information the learner is not ready for, or to a page that provides no new knowledge. This idea is realised through the two most popular forms of adaptive navigation support: link annotation and link hiding (De Bra, 2002).

Due to the complexity of ITSs, Mark and Greer (1993) have observed that evaluation techniques for traditional computer programs and simpler CAI systems are frequently inappropriate for ITS. Jones and colleagues thus contend that CAI will be not be widely integrated into curricula unless it can incorporate a critical element: the ability to take into account individual preferred styles of learning (Jones, Jacobs and Brown, 1997). If the presentation of information is configured inappropriately, it is possible to demotivate the learners (Barker *et al*, 1999a, 2000 in Barker *et al*, 2002).

[8] More details on the various components of ITSs can be found in Syang and Dale (1993).

Among the many ITSs available, one such is the website developed by Boticario and Gaudioso (2000) that facilitates personalised access to Web resources. This site is based on three essential elements: new teaching materials to make the most of the medium, alternative teaching organisation adapted to the requirements of the different users (namely students, lecturers and tutors) and an interactive system, which facilitates personalised and adaptive access to information and communication requirements to each and every user at each and every moment. The system corrects and assists students in their learning tasks – this emulates the intelligent tutoring systems. At the same time, it also behaves like a Web-based adaptive education system (Brusilovsky, 1998) and is thus able to adapt the page contents and links to the user model in different ways, which is what happens in adaptive hypermedia (AH).

2.1.2.1 Adaptive Hypermedia

At the time of Brusilovsky's review on adaptive hypermedia in 1996, educational hypermedia was identified as one of the two established leaders of AH; the other leader being online information systems. The other four kinds of adaptive hypermedia systems (AHS) were online help systems, information retrieval hypermedia, institutional hypermedia, and systems for managing personalised views in information spaces. Together, the two leaders accounted for about two thirds of the research efforts in adaptive hypermedia. It was observed that online help systems and institutional hypermedia systems had received almost no attention from adaptive hypermedia researchers in the recent years.

One may ask – what exactly does an AHS do? An AHS builds a model of the goals, preferences and knowledge of each individual user, and uses this model throughout the interaction with the user, in order to adapt the hypertext to the needs of that user. For example, a student in an adaptive educational hypermedia system will be given a presentation which is adapted specifically to his or her knowledge of the subject (De Bra and Calvi, 1998), and a suggested set of most relevant links to proceed further (Brusilovsky, Eklund and Schwarz, 1998).

This is typically performed through the following process:

1. While a user is "browsing" through an adaptive hyperdocument all user actions are registered,

2. The user model is applied to classify all nodes (pages) into several groups according to the user's current knowledge and interests or goals, and

3. In order to ensure that the content of a page contains the appropriate information (at the appropriate level of difficulty or detail) the AHS will conditionally show, hide, highlight or dim conditional fragments on a page when presenting it.

An example of education hypermedia is the Interbook application developed by Brusilovsky and his team (Brusilovsky, Eklund and Schwarz, 1998). It is an authoring tool based on an approach that simplifies the development of adaptive electronic textbooks on the Web. The metaphor behind this approach is an electronic textbook, which can be any hierarchically structured hypermedia material. It uses two kinds of knowledge: knowledge about the domain being taught (represented in the form of a domain model) and knowledge about the students (represented in the form of individual student models).

De Bra and colleagues (De Bra, Brusilovsky and Houben, 1999) explained that there are three kinds of concepts in a domain model (which describes the "knowledge" contained in a hyperdocument): atomic concepts or fragments (the smallest information units), pages (composed of fragments) and abstract concepts (representing larger units of information). In Figure 2.2, we can see the various techniques used in adaptive hypermedia technologies.

2.1.2.2 User modelling (or student modelling)

A distinctive feature of an adaptive system is an explicit user model that represents user knowledge, goals, interests, and other features that enable the system to distinguish among different users (Brusilovsky and Maybury, 2002; Brusilovsky, 2001; Barker *et al*, 2002).

An adaptive system collects data for the user model from various sources that can include implicitly observing user interaction and explicitly requesting direct input from the user. It is the user model that is used to provide an adaptation effect, that is, to tailor interaction to different users in the same context. Adaptive systems often use intelligent technologies for user modelling and adaptation (Brusilovsky and Maybury, 2002), such as those mentioned in the earlier section on adaptive hypermedia.

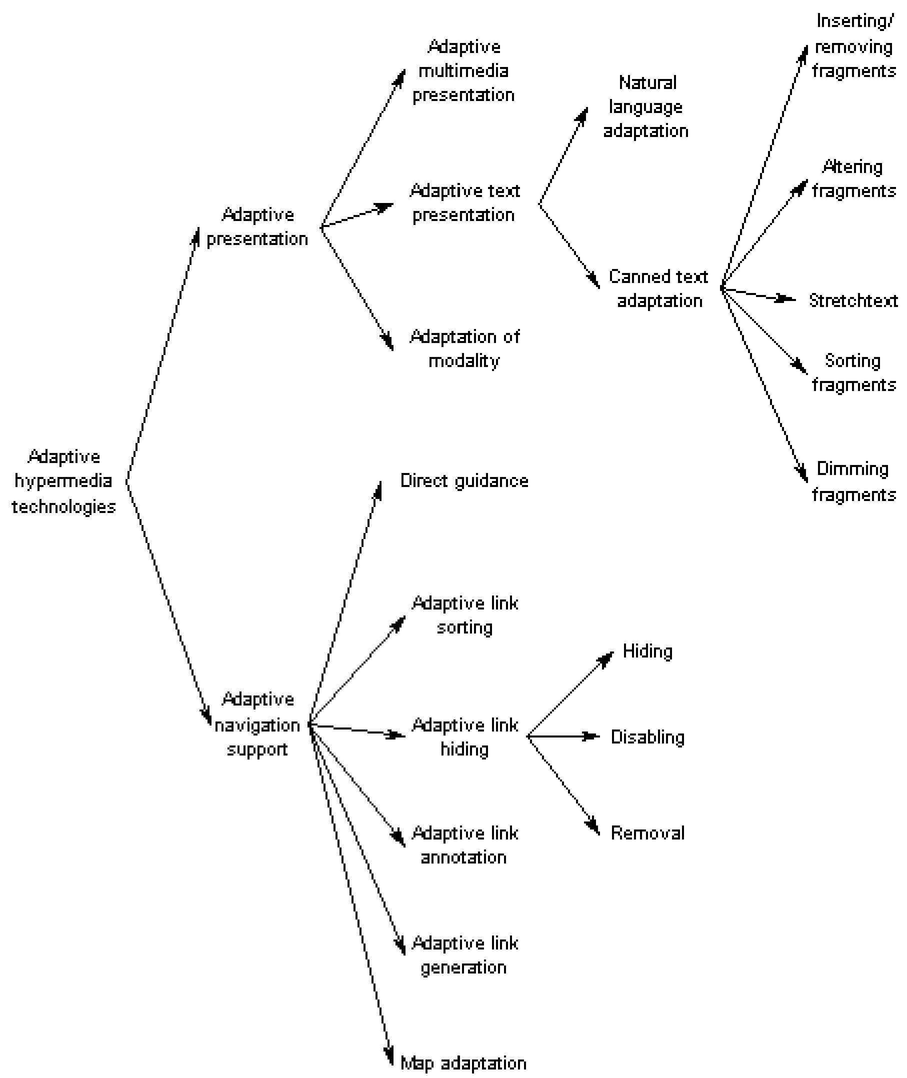


Figure 2.2: The updated taxonomy of adaptive hypermedia technologies (Source: Brusilovsky, 2001)

Brusilovsky and Maybury (2002) emphasise that there is some difference between an adaptive system and an adaptable system. Whilst an adaptive system automatically adapts to the user given a user model, an adaptable system requires the user to specify exactly how the system should be different.

Student models are an application of user modelling to the development of intelligent learning materials (Brusilovsky, 1996, 1998). Other than the above mentioned features describing the user (that is, the student), measures of performance may also be used to check on progress as the user interacts with the application, and the model may be adapted automatically to reflect any changes (Barker *et al*, 2002).

Abdullah is one of the developers that have used the approach of student modelling. In her research, student modelling has been used to develop an ITS for examination revision (Abdullah, 1999). Other applications include ANDES, an ITS that teaches Newtonian physics (Conati *et al*, 1997), and a multimedia courseware for a basic mathematics course for the General National Vocational Qualifications programme that is developed by Barker and colleagues (Barker *et al*, 2002).

There had not been a substantial student model which can be generalised to all ITSs until Syang and Dale (1993) attempted to design, test, and implement a quantitative student model called the Angel Model to measure programming abilities of undergraduate students who have taken the first Computer Science course.

Updating a student model is not a straightforward task. Different developers do it differently. Among the more sophisticated approaches, Collins and colleagues suggest the use of both granularity hierarchies and Bayesian brief networks (Collins, Greer and Huang, 1996), whilst many use just Bayesian brief networks (Eberts, 1997; Conati *et al*, 1997).

The next two sub-sections attempt to look at two ITSs in more detail.

2.1.2.3 KnowledgeTree

KnowledgeTree is a framework for adaptive online learning based on distributed re-usable learning activities (Brusilovsky and Nijhavan, 2002). The goal of KnowledgeTree is to bridge the gap between the information power of modern educational material repositories and the just-in-time delivery and personalisation power of ITS and AH technologies.

At the time of writing this thesis, the components of KnowledgeTree include three specific activity servers, a simple portal, and a simple user modelling server. All activity servers have been developed for the area of teaching programming. Each server supports authoring of a specific kind of activity and supports students' interaction with a selected activity of this kind. The WebEx system serves interactive annotated program examples (Brusilovsky, 2001); the QuizPACK serves parameterised questions (Pathak and Brusilovsky, 2002; Sosnovkey, Shcherbinina and Brusilovsky, 2003); and WADEIn serves demonstrations and exercises related with expression evaluation (Brusilovsky and Su, 2002).

All systems are self-containing servers. They are implemented on different platforms and are completely independent from a portal. WebEx is implemented using Microsoft ASP technology and served by an Internet Information Server (IIS) working on a Windows PC. QuizPACK is developed as a set of C++ CGI scripts and is served by an Apache server working on a SUN platform. WADEIn is implemented as a configurable Java applet embedded into a page generated by servlet technology from a Tomcat server.

All these servers implement some simple transparent login protocols and resource delivery protocol and thus can be used through any compliant portal.

2.1.2.4 WHURLE

WHURLE, which stands for Web-based Hierarchical Universal Reactive Learning Environment, is an XML-based ILE that is designed to adapt to individual learner profiles, thus redressing some of the major problems with most web-based learning systems (Brailsford *et al*, 2001, 2002). Two instances of such major problems are that learners are highly diverse in respect of their pedagogical requirements, and that the implementation of an effective web-based learning strategy must avoid placing a heavy technological burden on IT-naïve academics.

In WHURLE, the content is adapted to the needs of users by the use of a simplified model of translusion to create “virtual documents” visible only to the learner. Conditionality in this context is dependent on information stored in user profiles, which is used transparently to create these “virtual documents”. This allows the dynamic construction of both content and user interface to match the needs of the particular learner, in direct contrast to the imposition of inflexible systems, which is the case with most ILEs.

Information in the user profile is to be gained from both explicit and inferred sources. Examples of the former are filling out a registration questionnaire, or the scores achieved in domain-specific multiple-choice quizzes. Examples of inferred sources are the use of click-through rates, page dwell time, and the kind of information requested by the user (as defined by both content domain and pedagogic style). It is the aim of WHURLE profiling to store information about a user's domain preferences (including their abilities within these domains) and their learning style preferences, of which they may well not be consciously aware.

2.1.3 *Summary*

Since the beginning of the use of computers and the Web for educational purposes, technologies have progressed far, from standalone applications to networked applications, from static informational pages to adaptive tutoring pages. The most advantageous benefit is that the latest adaptive, web-based technology allows resources to be accessed anytime, anywhere, as well as adapting the selection and presentation of the material according to each learner's ability and needs.

2.2 Computer-based tests

Besides being a resource base for teaching and learning, educational applications have also been used for assessment purposes. The use of computer-based tests (CBTs) has increased significantly over the last few years, and there are a number of reasons why this trend is found in Higher Education. These reasons include (Pollock, Whittington and Doughty, 2000; Harvey and Mogey, 1999, in Lilley and Barker, 2002; Brusilovsky and Miller, 1999):

- Large numbers of assessments can be marked accurately and quickly.

- Time necessary to prepare these assessments is reduced by storing and reusing questions, allowing the lecturers' time to be used more productively.

- The assessment environment is brought closer to the learning environment. Software tools and web-based sources are frequently used to support the learning process, so it seems reasonable to use similar computer-based technologies in the assessment process.

- Such systems are suggested to be beneficial in terms of error reduction.

- There is the financial benefit to the institution by increasing retention rate and improving access, and also a financial benefit to the student by reducing the total hours spent on assessment.

- There is the benefit to the course itself where the students are now assessed on each topic and they must pass the assessment so all gaps in their knowledge will be filled.

- Students can tackle the assessments at their own pace which allows the course to be simultaneously delivered to a wide variety of students. Also students can get immediate feedback on their progress.

- Tutors can generate statistics more readily than if the assessment were to be conducted using traditional paper methods. As a result, a wider variety of results can be generated, and thus help improve the assessment and teaching process.

2.2.1 *Fixed response questions*

Most of the CBTs include fixed response questions of the following types:

- Multiple choice – the student selects one choice from more than one possible answers.
- Multiple response – similar to multiple choice except the student is not limited to choosing one response; he/she can select none, one or more of the choices offered.
- True/False – the student selects "true" or "false" in response to the question.
- Matching – two series of statements/words are presented and the student must match items from one list to items within the other list.
- Numeric answer – the student is prompted to enter a numeric value and this may be scored as one value for an exact answer and another score if the response is within a range.
- ‘Hotspot’ questions – a student clicks on a picture to indicate their choice. Dependant upon their choice certain feedback and grades is assigned. Some CBT marking tools provide a graphics editor to simplify specifying the choice areas.

CBT marking tools, such as QuestionMark Perception[9] (Belton and Kleeman, 2000, 2001; Danson, Dawson and Baseley, 2001) and the assessment feature of WebCT, typically mark fixed response questions.

These fixed response types of questions usually form objective tests. These tests produce student answers (or responses, or actions, or products) that can be marked objectively. With an objective test it is clear what the criteria are for marking, and these criteria can be used reliably with no scope for personal or subjective judgement (Gibbs, Habeshaw and Habeshaw, 1988).

As with other styles of assessment, objective tests have both advantages and disadvantages (Gibbs, Habeshaw and Habeshaw, 1988; Farthing and McPhee, 1999). The main potential advantages of objective tests are:

- **Improved reliability:** different markers could agree on a mark, and the mark would be a good indicator of what a student could do;

- **Ease and speed of marking:** this is especially the case when students indicate their answers by ticking possible alternatives as in the case of multiple choice questions.

- **Possibility of automating marking:** this would free the markers to do something else for their teaching duties, such as spotting students' weak points as part of revision topics, or simply to carry out their teaching duties more productively.

[9] Refer to http://www.questionmark.com/uk/perception/index.htm for more information.

On the other hand, there are disadvantages with having such objective tests:

- **Trivialisation of learning:** it is often the case that only low-level abilities are assessed by objective tests. The adoption of objective testing can result in a narrowing of the curriculum to what can be easily tested in this way. Though high-level abilities can be assessed by objective tests, this requires more imagination from the test designer. It is the poor use of objective tests which gives them a bad name rather than the intrinsic nature of the tests;

- **Possibility of guessing:** students may get the answer correctly by guessing, thus not assessing their real ability.

- **Availability of options:** students are offered a choice, instead of being asked to construct the answer.

- **Lack of feedback:** if students get an objective test item wrong then they may well want to know why. Most forms of objective test (though not all) provide them with no feedback on their learning.

Gibbs and colleagues believe that each and every one of the disadvantages can be overcome. The solution is to get the advantages of objective tests without also having the disadvantages. For instance, some use negative marking (that is, marks are deducted for incorrect answers) to counter the guessing situation. However, this technique does have the disadvantage that it can result in students being awarded negative marks (Farthing and McPhee, 1999).

2.2.1.1 Permutational Multiple Choice Question

Farthing and McPhee (1999) propose a new form of MCQ, called Permutational Multiple Choice Question (PMCQ) or “Two-stem” MCQ. A PMCQ has a two-part stem and six putative answers: two of which are keys and four are distractors (see Figure 2.3). To answer the question correctly, the student must match up each stem with the appropriate key. The two parts of the stem must ask about closely related issues. Typically PMCQs ask students to distinguish between two similar concepts. All of the options should be feasibly correct for both parts of the stem.

Q4. Many organisations plan their computer procurement policy to secure advantages. Which of the following is a characteristic of *mixed suppliers,* and which of *preferred suppliers?*

Mixed suppliers	e
Preferred suppliers	c

a.	Allows the customer to build on expertise with a given manufacturer's equipment
b.	Ensures consistency among components, such as common Human Computer Interaction
c.	Encourages competition, but avoids the overheads of repeated competitive tenders
d.	Payment may be staged over a long period of time
e.	Permits price competition and avoids lock-in to a single manufacturer
f.	All components are designed to work together, so reliability increases

Figure 2.3: An example of a PMCQ (Source: Farthing & McPhee, 1999)

In Figure 2.3, the two parts of the stem are orthogonal concepts rather than opposites, as some candidates may assume. The correct answers (e and c) have been entered for demonstrative purposes.

Unlike traditional multiple-choice questions, PMCQs are much less susceptible to students guessing the correct answer. Since there are thirty permutations of possible answers (6 x 5), the chance of getting both parts correct through random guessing is small. Students who guessed all the answers in a PMCQ test could expect a mark of only 3% (compared with 25% in a "choose one from four" test), and the likelihood of gaining a 40% pass mark in a test of ten PMCQs would be only 1:4500 (rather than approximately 1:5). Farthing and McPhee argue that these figures are at least compatible with traditional essay-type questions. Furthermore, since there is no need for guessing correction, we avoid the possibility of producing negative marks.

There are also attempts to generate exercises in an automatic way, exploiting the information that is already included in the knowledge base used in many adaptive hypermedia systems (Fischer and Steinmetz, 2000). However, it should be mentioned that this approach will not be able to create such exercises with a generally sound pedagogical strategy behind the algorithms.

2.2.2 *Free response questions*

Amongst the many types of questions, essay, open-ended, programming and diagrammatic questions are some that fall under the category of free response questions. Such question types are not naturally part of objective tests because their requirements cannot be specified sufficiently clearly to allow objective marking (Gibbs, Habeshaw and Habeshaw, 1988).

2.2.2.1 Essay questions

At the initial stage when computers were first used to aid the marking of essays, the Open University used the Tutor Marked Assignments (TMAs) mechanism by which their students receive feedback on their academic progress (Thomas and Taylor, 2000). The students were to submit their essays online, and the tutors would then mark the work with the aid of the application. At the end of the process, the students could then see the scores and TMAs, as well as feedback, on the screen. The scores and feedback would be processed automatically too.

Subsequently, more advanced technologies are used to develop the actual marking of essays. The following tools or mechanisms are some that allow the marking of free-text response questions:

- AutoMark (Mitchell *et al*, 2002);
- The Paperless School (Mason and Grove-Stephensen, 2002);
- SEAR – Schema, Extract, Analyse and Report (Christie, 1999, 2003);
- Automated Text Marker (Callear *et al*, 2001);
- e-rater™, c-rater™ (Burstein, Leacock and Swartz, 2001);
- CAPTEAP – Computer Assisted Proficiency Test of English for Academic Purposes (Aluisio and Oliveira Jr., 1999);
- TRAIDS – Tripartite Interactive Assessment Delivery System (Mackenzie, 2000);
- Miranda (Sclater and Howie, 2000);
- MacKay and Emerson’s web-based assessment of writing skills (2000);
- LSA – Latent Semantic Analysis (Landauer, Foltz and Laham, 1998)

These tools allow the teaching staff to create both summative and formative assessments with little or no human intervention.

While there are concerns about how capably a non-human marker can grade an essay, Thomas (2003) comments that there is evidence showing that acceptable scores which compare favourably with manual marking can be obtained. He has conducted tests to show that although the automatic marker consistently underestimates the final mark and, at most differs by 11% from the human markers' average, there is a good agreement. It should be noted that the difference between individual human markers could also be quite large: up to 8.5% in the case of one student.

No matter how accurately marking tools can mark essays or mathematics questions, there are some possible drawbacks, as highlighted by Pollock and colleagues (Pollock, Whittington and Doughty, 2000). In fact, these are actual observations made by students. Some students can spend too long preparing for the assessments and as a result are in danger of not completing them by the end of class time. As the students are all working at their own pace, they are on different topics and the teaching assistant/lecturer has to be able to switch from one topic to the next quite quickly. Also CAA only marks the final answer, but not the working as in traditional exams. Nonetheless, this can be addressed by students all using workbooks for the assessments, and if they feel their work is worth some marks for effort (for instance they made a silly mistake), extra marks can be given by the lecturer who checks the working.

2.2.2.2 Programming questions

While most CAA systems mark fixed response questions and some mark essay types of questions, there exist some applications that mark programming code too. BlueJ (Kölling and Rosenberg, 2001), CourseMarker and its predecessor, Ceilidh (Foxley *et al*, 2001), are three such systems.

Ceilidh is a courseware system implemented with a collection of programs and tools written in a variety of languages including shell, C and awk, which execute in a Unix environment (Foxley *et al*, 2001). It can mark various programming languages, including C, C++ and Prolog. The first version was used in 1988, and as a result of the continuous incorporation of enhancements suggested by users over the years it has become increasingly difficult to understand, maintain and support. The new CourseMarker (CM) system was subsequently developed. It is written in Java, and thus is executable on various platforms, including Microsoft Windows and Linux. CM has been mainly used for Java courses, but has also been piloted with C and C++ at some academic establishments.

Figure 2.4 depicts the improvements in CourseMarker over Ceilidh.

2.2.2.3 Diagrammatic questions

Use of diagrams in assessments is commonly found in various disciplines. For example, in Computer Science education, diagrams are used in flowcharts, entity-relationship diagrams, logic circuits and many others; in Chemistry education, chemical diagrams; and in teaching Operation Research, Petri nets.

System levels / User types	System	Course / Unit	Exercise
Student	⊗ - View system documents. ○ - Set environment properties. ○ - Customize system view. - View system MOTD*. ⊗ - Register to a course. ×	- View course documents ⊗ - View course notes. ⊗ - View course summary. ⊗ - View course MOTD. ⊗ - View total course work. ⊗ - View all students' marks. - View unit documents. ⊗ - View unit notes. ⊗ - View unit summary.	⊗ - Set up exercise. ⊗ - View exercise question. ⊗ - View exercise skeleton. ⊗ - Develop exercise. ⊗ - Submit exercise. ○ - View analytical mark and feedback ⊗ - View exercise test data (TD). ⊗ - View marks. - Execute solution: ⊗ - Interactively. ⊗ - With TD. × - With teacher's TD. × - Execute teacher's solution with student TD. ⊗ - Enquire by e-mail.
Tutor		⊗ - View students' course marks. ⊗ - View Student Statistics. ⊗ - Check course state. × - Register student to a course. ⊗ - View student's work.	⊗ - View student solutions. × - Mark solutions (manually or automatically). ⊗ - View statistics. ⊗ - View missing / submitted students. ⊗ - View tutor help. ⊗ - View plagiarism results. × - Find tutees for a specific tutor.
Teacher	⊗ - Add / Delete students, tutors, teachers. × - List tutor's tutees. × - View audit trails.	⊗ - Edit course MOTD. ⊗ - Check / Edit course properties. ⊗ - Edit unit properties.	⊗ - Edit Tutor help. ⊗ - Edit exercise question. ⊗ - Change exercise properties. ⊗ - Expunge student's work. ⊗ - Set borderline. ⊗ - Search for Plagiarism.
Developer		⊗ - Create / Add courses. ⊗ - Create / Add units.	○ - Copy / Delete exercises. × - Author exercises.
Administrator	⊗ - Edit system MOTD. ⊗ - Add / Delete Administrators. ○ - View error log. ⊗ - Register students.	⊗ - Edit course documents. ⊗ - Install new courses.	○ - Give extensions. ○ - Copy / Delete exercises. × - Author exercises.

×: Ceilidh ○: CourseMaster ⊗: Both Ceilidh and CourseMaster * MOTD: Message of the day.

Figure 2.4: User's responsibilities at different levels of system use - Ceilidh vs CourseMarker (Source: Foxley *et al*, 2001)

Tsintsifas (2002) has developed a diagram authoring extension, entitled DATsys (Diagrammatic Assessment Teaching System) that was integrated into CourseMarker. There are three parts to this extension, as illustrated in Figure 2.5:

- **Daidalos** is a diagrammatic domain modeller. Diagrammatic courseware developers use Daidalos to:
    - o Model graphically the elements, relationships and data model of a concrete domain.

Figure 2.5: The various components of DATsys

- o Change and evolve the system's behaviour by using its dynamic meta-level facilities.

- **Ariadne** is an application that uses domain libraries built in Daidalos (see examples in Figure 2.6) to allow the definition of automatically assessed graphical course-exercises. To create such exercises the developer has to enter the exercise question and its solution, parameterise the checking oracles, program the marking script and specify the application functionality of the generated diagram-editor.

- **Theseus** is the last link in the chain of diagram editors. Theseus is completely generated according to the specifications of the developer using Ariadne. It can be very simplistic, offering the minimum functionality needed by students to draw the exercise's solution, or it can be as complex as Daidalos itself. A generated Theseus can be used for a whole range of exercises or it can be customised to a specific one.

For the time being, the DATsys extension is a Java application. There are plans to make it a servlet – Theseus being the first component to be worked on

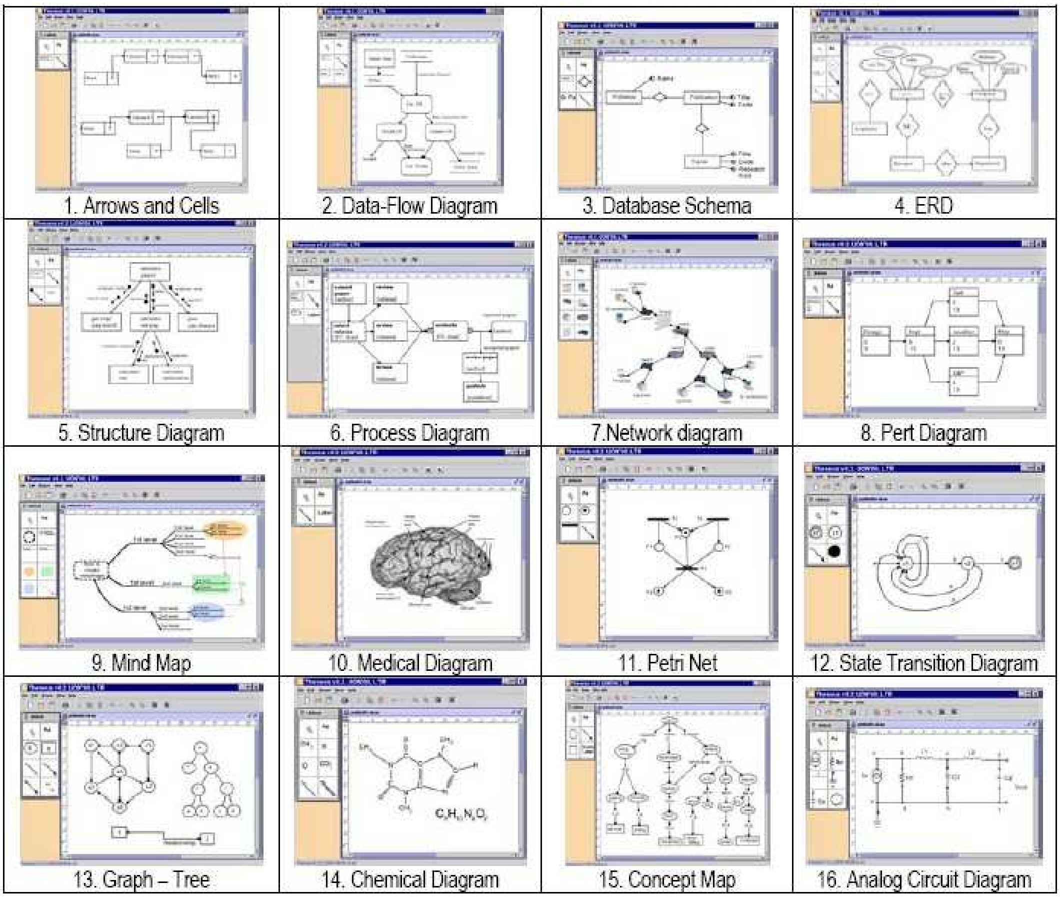


Figure 2.6: The tool libraries for the various types of diagrams supported by Daidalos (Source: Tsintsifas, 2002)

– so that it could be executed over the web, and thus compatible to the MCQ system.

2.2.3 *Summary*

The most common computer-based tests are objective-based. Nonetheless, there are an increasing number of CBT marking tools that can mark fixed response questions, such as essays and programming code. This has greatly benefited both the teachers and students, allowing students to be assessed more often so that the two groups of users can know how well they have learned. This is particularly useful in ever-growing classes.

2.3 Computerised-adaptive tests

In CBTs, the questions presented to the examinees are not tailored according to their individual performance during the test. As a result, examinees can be presented with questions that are either too easy or too difficult (Lilley and Barker, 2002; Eberts, 1997). It is exactly this concern that brings researchers to look into computerised-adaptive testing systems.

Computerised-adaptive testing is defined as a procedure that automatically administers an examination with questions (known as items in the psychometrics literature) of appropriate difficulty for the student based on the ability of the student (Syang and Dale, 1993).

A common form of adaptive testing involves a computer-administered test in which the presentation of each test item and the decision to stop the test are dynamically adapted to the student's performance in the test (Abdullah, 1999; Lilley and Barker, 2003). The most common stop conditions are:

1. A certain number of questions have been administered;
2. A time limit has been reached; or
3. A certain standard error for the ability has been met.

The advantages of this form of adaptive testing are:

- It presents to the student only those test items which are likely to yield additional information;

- It can distinguish careless slips and guesswork from actual knowledge by repetitive testing; and
- It can arrive at an initial student model quite accurately in a relatively short time.

A CAT system can be used for several purposes. For instance, in the subject area of the adaptive tests, it can be used to describe classes of problems, to describe arithmetic skills, to describe student responses to problems, and to generate problems (Abdullah and Cooley, 2000).

Though developed independently, computerised adaptive testing has found a natural home in ITSs (Wainer, 1990, in Abdullah and Cooley, 2000; Lilley and Barker, 2003). As in ITS, CAT requires some form of user modelling to keep track of students' ability. Abdullah (1999) uses premodelling through adaptive testing, which involves two stages: representing knowledge of subject domain, and creating an initial knowledge state of the student (thereby establishing the initial student model).

Some CAT systems developers have also used some other mechanisms to design their applications. Subject domains such as mathematics have been represented using the notions of "granularity hierarchies" and "prerequisite relationships" (Abdullah and Cooley, 2000). Abdullah and Cooley (2000) have included the use of Bayesian approach to probability to make inferences about students' states of knowledge so as to optimise the adaptive testing process (Collins, Greer and Huang, 1996).

Abdullah and Cooley (2000) have pointed out a possible issue on evaluating such systems. For most classes of problems, it is not feasible to expect the teacher to inspect every example. This is to say that unexpected and undesirable examples may not be revealed during the process of knowledge acquisition. Traditional program testing and additional knowledge acquisition sessions are needed to reduce the probability of errors.

All in all, CAT is potentially capable of offering higher levels of interaction than offered by a traditional CBT (Lilley and Barker, 2003).

2.3.1 *Item Response Theory*

Ideas from Item Response Theory (IRT) have been very influential in CAT systems (Wainer and Mislevy, 1990, in Abdullah and Cooley, 2000; Lilley and Barker, 2002). They have formed the basis of a system to assess student programming abilities (Syang and Dale, 1993); they have influenced Huang's content-balanced tests (Huang, 1996, in Abdullah and Cooley, 2000), and more recently, their influence can be seen in a web-based adaptive testing system in the domain of European vegetation species (Rios *et al*, 1999, in Abdullah and Cooley, 2000).

The central element of IRT is a family of mathematical functions that aims to calculate the probability of a specific student answering a particular question correctly. At present there are various IRT mathematical models for dichotomously scored questions (i.e. items), such as the One-Parameter Logistic (1-PL) Model, the Two-Parameter Logistic (2-PL) Model and the Three-Parameter Logistic (3-PL) Model.

Figures 2.7, 2.8 and 2.9 show the equations of these three mathematical models. The variable *a* represents the question's discrimination and its usefulness when distinguishing among students near an ability level θ, whereas *b* shows the difficulty of question, and *c* is the pseudo-chance or guessing parameter. The 1-PL model contains only the item difficulty parameter, *b*. The 2-PL model contains both the item difficulty parameter, *b*, and the item discrimination parameter, *a*. While the 3-PL model seems to be the most realistic of the three most common IRT models in that it acknowledges the chance response through *c*, it is affected by estimation problems.

$$P(\theta) = \frac{1}{1 + e^{-(\theta - b)}}$$

Figure 2.7: One-Parameter Logistic Model of Item Response Theory

$$P(\theta) = \frac{1}{1 + e^{-1.7a(\theta - b)}}$$

Figure 2.8: Two-Parameter Logistic Model of Item Response Theory

$$P(\theta) = c + \frac{1 - c}{1 + e^{-1.7a(\theta - b)}}$$

Figure 2.9: Three-Parameter Logistic Model of Item Response Theory

The IRT model gives the probability of answering a question correctly in terms of the interactions between b and θ. If the student is much more proficient than the question is difficult, then the probability will be larger. Conversely, if the question is much more difficult than the student is proficient, this probability will be small.

2.3.1.1 Item Characteristic Curve

When plotted, the equation of the 3-PL model has the appearance of an S-shaped curve having $P(\theta) = c$ and $P(\theta) = 1$ as horizontal asymptotes. For very low values of θ, $P(\theta)$ increases slowly. As θ approaches b, $P(\theta)$ typically increases more rapidly: the rate of increase being a maximum at $\theta = b$. (See Figures 2.10 and 2.11 for examples of an ICC for an item answered correctly and incorrectly respectively.)

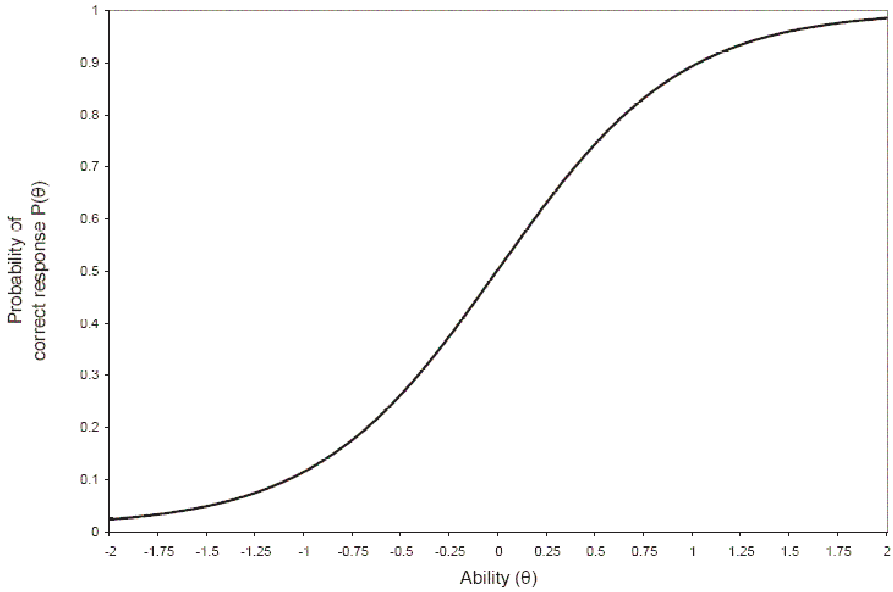


Figure 2.10: Example of an ICC for an item answered correctly (Source: Lilley and Barker, 2002)

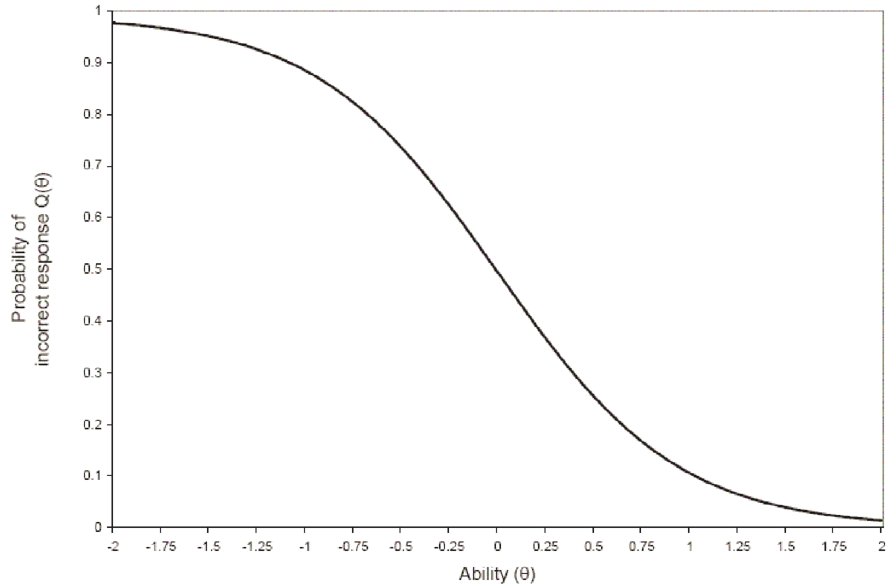


Figure 2.11: Example of an ICC for an item answered incorrectly (Source: Lilley and Barker, 2002)

As in many CAT applications that implement the IRT model (see Lilley and Barker, 2002, and Syang and Dale, 1993), the test starts with a provisional ability for the examinee. This provisional ability can be obtained either randomly or based on the most recent estimate of the examinee's ability. However, this poses a problem. The new provisional ability for the examinee can only be estimated when the examinee has answered at least one item correctly and one item incorrectly. This happens because the examinee's response likelihood curve is formed from the product of all the ICCs of items answered during the current test (see Figure 2.12 for an example of a likelihood curve), and if the examinee answered all the items correctly, the examinee's response likelihood curve is composed by the product of various $P(\theta)$ and, therefore the curve does not have a peak value. If the examinee answered all the items presented incorrectly, the examinee's response likelihood curve is calculated by the product of various Q(θ) (= 1 - P(θ)) and, consequently, the curve does not have a peak value.

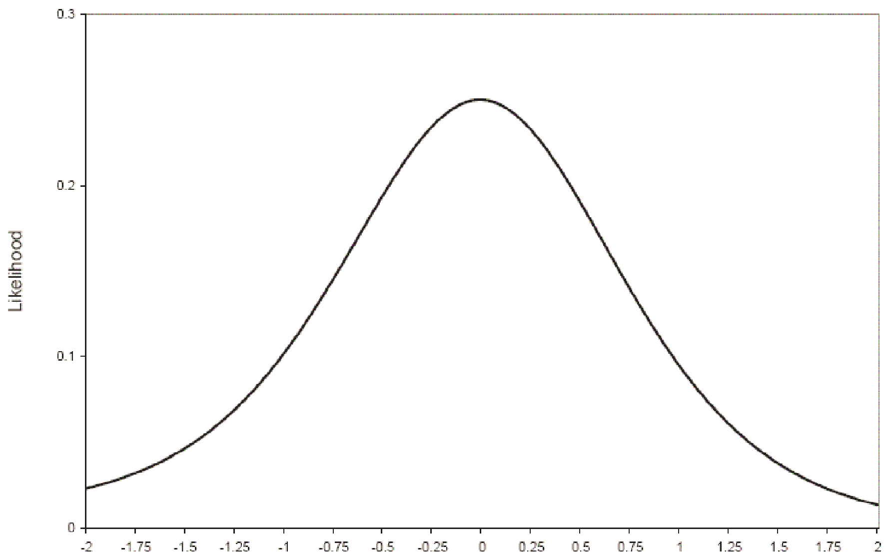


Figure 2.12: Example of a likelihood curve (Source: Lilley and Barker, 2002)

Given that the peak of the likelihood function corresponds to the most likely value of θ for a given set of responses, the most recent value of ability estimate θ is taken to be the peak value of this composite function. If a given student's ability estimate θ is the same as the level of difficulty of a question, that student has a 50% chance of answering that question correctly (Wainer, 1990).

More information on IRT can be found in Baker (2001).

2.3.2 *Concerns about the use of CATs*

As with any other types of applications, adopting CAT systems has some drawbacks, however good the system may be.

In some research carried out by Lilley and Barker (2002, 2003), the following observations have been made:

- The expert evaluators (namely, some lecturers in Computer Science and in English for Academic Purposes, at the University of Hertfordshire) believed that the prototype would give greater assistance in formative rather than in a summative assessment environment. They suggested that formative assessments provide the lecturers with more information regarding the students' strengths and weaknesses, since they are typically undertaken on a regular basis.

- It has been noted that the students would be more receptive to using a CAT in a formative rather than in a summative assessment environment. This suggests that lecturers foresee problems regarding the score method used within CAT. In a CAT, the final score given to an examinee is calculated based on the number of questions answered correctly and incorrectly, as well as on the level of difficulty of these questions. As a result, examinees who have answered the same number of questions correctly would almost certainly have different final scores, and this can bring uncertainties about the "fairness" of the assessment.

- The expert evaluators expressed the opinion that the prototype's ability to help students to detect their own potential educational needs in both summative and formative assessment environments was poor. This is because the students were unaware of the adaptive process and therefore possibly unable to learn from their mistakes. As such, it is important to make the students aware of the adaptive feature of the application, so as to maximise the potential help it gives to them.

- For the same prototype, Lilley and Barker had attempted to get some feedback from a focus group. Findings from the focus group suggested that, whilst students generally preferred the CAT set of questions to the CBT one, some students were concerned about the fact that they were unable to return to previous questions once they had submitted their responses. Furthermore, some students expressed concern that stopping conditions based on consistent standard error scores might mean that some students have tests of different lengths. The participants in the focus group feared that this characteristic might disadvantage students who initially perform badly and provide no opportunity for them to catch up.

In another study, Barker and colleagues made an attempt to configure a multimedia educational system based on a cooperative psychological student model (Barker *et al*, 2002). In this work, information and the level of tasks and questions presented to users were adapted cooperatively based upon their ability. A major limitation of their approach was the difficulty in establishing the appropriate level of the user model based on performance. This task required input cooperation from both tutor and student in order to establish the level of difficulty. Despite its pedagogical value, this method was inefficient in time and resources. Feedback to the adaptive student model was therefore slow and often for this reason it was difficult to be certain that the model had been configured optimally.

Despite all these concerns, Lilley and Barker (2002) still regard CAT potentially to be an important tool in teaching and learning in the coming years. While there are some increasing concerns of increased student numbers, and thus demand of lecturers' time, CAT is deemed a possible solution to these concerns. This is particularly supported by the drive towards the use of computer-based assessment within Managed Learning Environment in Higher Education.

2.3.3 *Summary*

While CBTs marking tools have the advantage of allowing teachers to give more tests to more students, the questions presented to the students are not tailored according to their individual performance during the test. As a result, students can be presented with questions that are either too easy or too difficult. CAT systems can overcome this problem. Despite the huge potential of such systems, there are still some considerations to be taken when deciding how to design and implement them.

2.4 Summary

The use of computers and the Web in the educational context has been considered:

- Educational technologies have progressed considerably, from standalone applications to networked applications, from static informational pages to adaptive tutoring pages. The most advantageous benefit is that the latest adaptive web-based technology allows resources to be accessed anytime, anywhere, and can adapt the materials according to each learner's ability and needs.

- Besides learning, computers are increasingly being used for assessment purposes too. Computer-based tests can include fixed response questions, such as multiple choice, as well as free response questions, such as essays and programming code. This allows teachers to assess their students more often so that the two groups of users can monitor learning progress. This is particularly useful with growing class sizes.

- Over the last few years, there have been an increasing number of research on computerised-adaptive testing systems. The advantage of such systems over CBT marking tools, is that the questions presented to the students are tailored according to their individual performance during the test, instead of being potentially either too easy or too difficult. Nonetheless, there are still some considerations to be taken into account when deciding how to design and implement CAT systems, regardless of the potential of such systems.

Having looked at the technical aspects of education, a more pedagogical approach towards assessment is explored in the next chapter.

Chapter 3 –

The Pedagogical Perspective

The previous chapter has presented a literature review of IT in education, in particular web-based education, computer-based tests and computerised-adaptive tests.

This chapter explores education from a pedagogical perspective, firstly, in Section 3.1, from the point of view of assessment, in particular its relationship with learning.

Section 3.2 considers formative and summative assessment, and the importance of the two types of assessment in the learning process, as well as the lack of emphasis on the former type. Also included is a focus on one particular type of formative assessment, namely self-assessment, by discussing its use and implications.

Section 3.3 examines how learning outcomes are classified, specifically how Bloom's Taxonomy fits into the various types of assessment, and generally, how questions are to be designed for CAA purpose.

The fourth section discusses feedback which has an important role in the learning process, if the student is to learn efficiently and effectively.

3.1 Assessment and Learning

Students need questions to learn, regardless of the source of the questions – may they be posed by teachers, or formulated by the students themselves (Morgan and Saxton, 1994). As Schwartz (2002) has observed, the students are saying: 'We want more opportunity to test our understanding', 'We want more feedback on our progress', and 'We want credit for the work we're doing during the year and we don't want everything to be based on a final exam.' We can conclude from this that questions, or rather any form of assessment, and learning are inseparable.

Sambell and colleagues have noted a comment made by one tutor, which in fact, echoes a general sentiment:

> *'Nowadays the only time the students seem to care is two weeks before the assignment's due in, when they wander in for a tutorial saying they can't find anything on the topic! …' … We see class sessions as their chance to clarify their understanding of what they have been learning; they see them as occasions for us to dish out material that they can give us back in the assignment.* (Sambell, Miller and Hodgson, 2000, p.138)

This illustrates that students see learning as part of working on their assessment, rather than assessment as part of the learning process. This is the exact opposite to the original intention of assessment, in general (as described later). Nonetheless, both students and teachers view assessment and learning as inseparable from each other.

As can be seen, assessment has a strong influence on the way students behave and approach their learning. This is further shown in many other research on student learning (e.g. Miller and Parlett, 1974, in McDowell, 1996). Through well designed assessment, students are encouraged to become better able to reflect on and evaluate their own work and performance (McDowell, 1996; Erwin, 1995; Brown and Knight, 1994). The design of assessment can enable students to be engaged in deep learning, to find an intrinsic motivation and a sense of satisfaction in their learning, to develop a range of skills and to become more aware of means of assessing and judging their performance (McDowell, 1996). All in all, we can conclude with a quotation from Erwin:

> "*deciding what to teach and assess is one issue, not two*" (Erwin, 1995, p.51).

He continues

> "*one of the best ways to improve teaching is through better testing*" (p. 53).

This is not the only reason for assessing. In two of her papers, Brown has attempted to list other reasons (Brown and Knight, 1994; Brown, Race and Rust, 1995), including:

- Students expect it.
- Students are motivated by assessment.
- It can provide feedback to learners so they can learn from mistakes and build on achievements.

- It consolidates student learning.
- Assessment is learning.
- It helps learners to apply abstract principles to practical contexts.
- It estimates students' potential to progress to other levels or courses.
- It gives teachers feedback on how effective they are being at promoting learning.
- It provides statistics for internal and external agencies.
- We have always done it.

As can be seen, there are many good reasons for taking assessment seriously. Also, with the results obtained from the assessments, we can evaluate the impact of courses or programmes on students. Erwin (1995) has suggested four analytical strategies or questions just for this purpose:

- How many students achieve expected minimum competency?
- Do students who have had the course(s) or educational experience perform better than students who have not had the course?
- Are student marks and grades related to or correlated with later outcomes?
- Do students change over time more than can be explained by the natural process of maturation?

3.1.1 *Sound Assessment*

Consideration of reasons for assessment raises the question of what can be considered as sound assessments, particularly in higher education (HE). According to Brown and Knight (1994), sound assessments in HE:

- Are valid, which means that they are related to the goals of universities, programmes and courses.
- Are targeted at multiple achievements.
- Use multiple methods.
- Assess the same "outcomes" on several occasions.
- Produce useful and informative data.
- Are properly stored and well used within universities.
- Are systemic, that is, no assessment should be an island.
- Are fit for the purpose.

In another publication, Race (1995) has recommended ten principles of assessment:

1. The purposes of assessment need to be clear.
2. Assessment needs to be an integral part of course design, not something to be bolted on afterwards.
3. Assessment methodology needs to be valid.

4. Assessment processes and instruments need to be reliable.

5. Assessment methodology needs to be feasible.

6. Assessment needs to be transparent to students, staff and employers.

7. Assessment needs to be a means of delivering feedback.

8. The overall assessment strategy needs to employ a wide range of techniques and processes.

9. The amount of assessment should be appropriate.

10. Assessment should be free of bias.

All in all, we can see that a sound assessment system does not consist of only one assessment, but a few valid and reliable assessments, which together provide feedback to both students and teacher, as well as the institution and external agencies, such as employers.

While the characteristics and principles of sound assessment are known, one may ask a key question – what are the teachers assessing? Do we know, and do the learners know? (Brown, Race and Rust, 1995) There is no point setting an assessment without knowing what to assess. Likewise, if the students are not given guidelines as to what they are to be assessed on, they will not know how to set about working on the assessment. Hence, it is important that when facing an assessment, both teachers and students know what is covered.

Also, when assessing, teachers are to note the different elements involved:

> *"Where a range of outcomes is envisaged, it is important to clarify how the outcomes will be assessed and what weighting the different elements will receive in terms of marks. One particular problem is the balance between assessment of outcomes or* product *and assessment of* process. *…[On the other hand,] where skills are used during the process of an assessment task, and valid ways of assessing them need to be carefully considered rather than simply making assumptions that, for instance, a good team report necessarily results from the deployment of a high standard of team work skills"* (McDowell, 1996, p.164).

If we are to let assessment drive the learning, as suggested by Sambell and colleagues, more explicit discussion of assessment issues are to be built into the teaching, so as to highlight what are seen as appropriate ways of tackling assignments (Sambell, Miller and Hodgson, 2002). Sambell and colleagues argue that it is important to start from where the learners actually are, rather than from where the teachers would like them to be, however frustrating or 'time-wasting' it may appear from the subject-focussed perspective.

On the other hand, in some subject areas, such as mathematics for engineering or business studies, students' lack of motivation presents tutors with considerable difficulties, and it is only by experimenting with different patterns of assessment that working solutions are found. In their paper, Farmer and Eastcott (1995) have named a range of assessment patterns that encourage productive learning activity and feedback:

- So-called 'fast feedback' tests that are peer marked in class and which encourage students to practise in their own time.

- Final assessment activities that take questions exclusively from previous feedback tests.

- Assignments that focus on information gathering and analysis and which lead students to investigate more open-ended problems.

- Design questions, often computer packages, for the students to do in their own time, where they can talk to each other and discuss the principles involved. To avoid problems of plagiarism, learners are assessed not on the design as much as on their ability to apply the designs in controlled conditions so as to solve problems. An accounts package for a small hotel might be designed by accountancy students and they would be assessed on their ability to apply it successfully to handle data given to them under examination conditions.

- Open-book assignment tests that assess students' understanding of the work set in an assignment (introduced in order to overcome problems with plagiarism).

An observation by McDowell (1996) is that innovative assessment does have the distinct advantage that it tends to generate debate and discussion about the fundamentals of assessment – what is really being assessed? Is this what should be assessed? Is the assessment equitable and fair? What are the its side-effects? In many cases, this debate also involves students. All this increases the possibility of developing open, reliable and valid assessment that enables students to learn rather than simply perform.

3.1.2 *Summary*

We have learnt that assessment has a strong influence on the way students behave and approach their learning. In the absence of assessment, learning will be incomplete. Therefore, with a sound assessment structure, we can be sure that student's learning can be enhanced.

3.2 Formative and summative assessment

The word 'assessment' is still very much understood by most people as 'summative consideration of teaching', rather than as 'formative consideration of how to improve student learning' (Borland, 2002). Nonetheless, both purposes of assessment are just as important. Summative assessment, which includes end-of-module assessment, is the assessment which produces a measure which sums up someone's achievement and which has no other real use except as a description of what has been achieved. On the other hand, the purpose of formative assessment is to get an estimate of achievement that is used to help in the learning process. This form of assessment also includes coursework where the student receives feedback that helps them to improve their next performance; discussion between a mentor and a student; and end-of-module examinations whose results are used to identify areas for attention in later modules (Brown and Knight, 1994).

Whilst assessing students' achievement at the end of a module is important, making sure that students are progressing well in the duration of the module is essential too. Even Schwartz (2002) is convinced that formative assessment, such as quizzes, provide the best opportunity for teachers to ensure that students are able to do what teachers want them to do by the end of each module, whilst final exams, with their limited scope for 'sampling' the content of each module, will clearly not be as useful for this purpose.

As mentioned earlier, summative assessment is more commonly known and used than formative assessment, and hence the latter will now be considered in more detail.

If the formative function is to succeed, the tasks set need to be carefully devised. Brown and Knight (1994) propose that tasks should:

- Be such that full feedback is generated for the students' use.
- Have criteria for successful completion available.
- Direct students' attention to the importance of drawing upon earlier experiences of this sort of activity, or to previous encounters with this type of subject matter, situation or problem.
- Ideally, have review points built in where students have to pause and either reflect themselves upon what they are doing and what they know about how to do the task, or sound out colleagues or tutors.
- Be achievable.
- Reflect the goals of the curriculum.

Though formative assessment is very much appreciated by both tutors and students, there are still mixed feelings on continuous assessment. Race has recorded some positive feelings of students on continuous assessment, including:

> *chance to excel with no time barrier; chance to develop my work; and, aware of ongoing progress; the chance to do right things I previously did wrong; involved in my work; in control.* (Race, 1995, p. 64)

In contrast, some negative feelings include *time-consuming*; *stressful*; *ongoing load*; *burdensome*; and *need to keep achieving*.

Tutors too have expressed their worries about continuous assessment (Race, 1995):

1. Learners can feel under so much pressure that their ‘want to learn’ is damaged.
2. Though ‘learning by doing’ may be involved, the range of ‘doing’ may be quite narrow (e.g. writing essays).
3. The value of feedback is often eclipsed by learners’ reactions to scores or grades.
4. Learners rarely have time or opportunity to ‘make sense’ of the experience of the feedback.
5. The opportunity for tutors to give learners good feedback is reduced by large class sizes.

6. In large classes, the chances of cheating not being noticed are higher.

7. Learners don't usually know much about the criteria being used to assess their work.

8. When there are exams and continuous assessment, learners may spend too much time on latter – and even fail their exams.

9. Student life becomes driven by assessment, instead of by learning.

10. Too little use is made of the enormous potential for learners to learn by assessing.

These worries from the tutors have the tendency to be intensified by the ever-growing class size. Therefore, the goal of developing an assessment system for a large class that would meet the dual aims of offering students more opportunities for testing themselves and for obtaining feedback, while at the same time not requiring more time and effort from teachers, is one that is probably quite widely shared. One of the options, probably the best one, to achieve the aims, is to computerise the assessment. Nonetheless, it is noted that it is not a guarantee of immediate, complete success, nor is it the easy, painless option that some teachers seem to think it is, as observed by Schwartz (2002). Some careful planning involving various stakeholders will be beneficial (Eason, 1988; Laurillard, 1993; Barker and Barker, 2002).

However complex or time-consuming, this option still gives other benefits, such as being able to provide statistics more readily than the traditional paper method. This is especially useful for a Web classroom scenario. In a Web classroom, where student-to-student and student-to-teacher communication is limited, comparing statistics is very important for both teachers and students to get the "feeling" of the classroom (Brusilovsky and Miller, 1999). For example, by comparing class average with personal grades a student can determine class rank. By comparing class grades for different tests and questions a teacher can find too simple, too difficult, and even incorrectly authored questions.

3.2.1 *Self-assessment*

Amongst the many forms of formative assessment, self-assessment is just one. While traditionally self-assessment has not been part of teaching modules, it has had an important role in learning. Students are always self-assessing (Boud, 1995). Whenever we learn, we question ourselves, 'How am I doing?', 'Is this enough?', 'Is this right?', 'How can I tell?', 'Should I go further?' In the act of questioning is the act of judging ourselves and making decisions about the next step. This is self-assessment.

Sadler goes further to explain that self-assessment is assessment by the students of their work using comparable cognitive, intellectual and pedagogic processes to that of the assessment setter (Sadler, 1989, in Taras, 2002).

Boud (1995) states that:

> *All assessment – whether conducted by teachers or by learners – involves two key elements. The first is the development of knowledge and an appreciation of the appropriate standards and criteria for meeting those standards which may be applied to any given work. Unless it is known what counts as good work, it is impossible to tell*

> *whether the specific work being considered is adequate. The second is the capacity to make judgements about whether or not the work involved does or does not meet these standards.* (p. 11)

He carries on to comment that, unfortunately, emphasis is normally given, by both staff and students, to the latter of these two elements. Engagement with the standards and their criteria is down-played to the detriment of learning. In an earlier publication of his, he proposed the following as the defining characteristic of self-assessment:

> "*the involvement of students in identifying standards and/or criteria to apply to their work and making judgements about the extent to which they have met these criteria and standards.*" (Boud, 1991, p.5)

In his later book, Boud (1995) also lists other characteristics of self-assessment:

- It refers to particular work or achievements and does not purport to represent an account of the person *qua* person or an overall state of being; the person is conceptually separated from the work.

- It is not normally public or directed at a specific person or entity, although declaring one's self-assessment can be a useful part of developing a commitment to it and receiving feedback on it.

- It has the self as agent and audience.

- It is not undertaken for the sake of catharsis, although it may be associated with completion and making whole.

- It is essentially formative and not absolute, though it can be used for summative purposes.

One may observe that self-assessment is coming to be regarded as an accepted and significant part of teaching modules because it relates to one of the central goals of a university education: enabling students to become effective and responsible learners who can continue their education without the intervention of teachers or teaching modules. Specifically, it is assumed that (Boud, 1995; Dearing *et al*, 1997):

- It is a necessary skill for lifelong learning.
- It needs to be developed in university courses.
- It is necessary for effective learning.

It is worth noting that self-assessment is not simply a substitute for activities in which students would normally be assessed by someone else, even though there are many examples of its use in such circumstances. There are many purposes for which it is used, roles it plays and contexts in which it operates (Boud, 1995):

- For individual self-monitoring and checking progress.
- As a way to promote good learning practices and learning how-to-learn skills.
- For diagnosis and remediation.
- As a substitute for other forms of assessment.

- As a learning activity designed to improve professional or academic practice.
- To consolidate learning over a wide range of contexts.
- To review achievements as a prelude to recognition of prior learning.
- For self knowledge and self understanding.

In another chapter of his book, Boud continues to remark that considerations of the learning process give support to the use of self-assessment practices, and provide indications of good practice:

- Well-constructed self-assessment activities act as prompts to deep approaches to learning. However, there are no guarantees that any strategy will reliably lead to a deep approach as it is students' conceptions which influence their actions. Concern other than those related to the learning task will always be present.
- Self-assessment encourages students to take responsibility, particularly when they are actively involved in considering criteria which are meaningful to them.
- Self-assessment activities need to take account of students' goals and experiences, whatever other goals teachers may have. To ignore the students' perspective is too give a message to them which says their concerns are not important. This will have wider consequences in terms of the value students place on these activities. Students should be actively involved in decisions about self-assessment activities.

- Self-assessment needs to engage students with central problems and issues in the field of study to equip them with subject-matter skills as well as more generic ones. Self-assessment is not context- nor content-independent.

- Self-assessment can encourage students to reflect on their learning to consolidate it and move beyond it.

In summary, well-designed self-assessment activities need to take account of:

- Learners and the experience which they bring to the task.

- The particular demands of the subject matter.

- The context in which they take place.

- The extent to which teaching modules have reflective activities incorporated as a core feature.

3.2.2 *Summary*

Whilst assessment is understood by most people as 'summative consideration of teaching', rather than as 'formative consideration of how to improve student learning', the latter plays an important role in learning too. Formative assessments, such as quizzes, provide the opportunity for teachers to ensure that students are able to do what teachers want them to do by the end of each module.

We have also learnt that students are always self-assessing, with or without tutors' prompting. With appropriate self-assessment techniques, students can become effective and responsible learners who can continue their education without the intervention of teachers or teaching modules.

3.3 Classification of learning outcomes

There are many models of classifying learning outcomes, such as RECAP, SOLO and a revision to Bloom's Taxonomy (see Stephens and Percik, 2002; King and Duke-Williams, 2001). Nevertheless, the most cited model by the CAA published community is still Bloom's Taxonomy (Bloom, 1956). Bloom proposed that a task stimulates one of three psychological domains and that the objectives concerned with learning the task could be categorised using those domains. They are the:

- **Cognitive domain**, which is related to the acquisition and application of knowledge and understanding.
- **Affective domain**, which concerns attitudes and feelings from the learning process.
- **Psychomotor domain**, which concerns manipulative or physical skills.

He then defined a classification of cognitive learning that contains ascending levels of abstraction that augment gradually from simple knowledge recall to the ability to evaluate (see Table 3.1).

Cognitive aspect of learning	**Ability**
Knowledge	To remember
Comprehension	To rephrase knowledge
Application	To apply rephrased knowledge in a novel situation
Analysis	To break a problem into its constituent parts and establish the relationship between each one
Synthesis	To combine separate elements into a whole
Evaluation	To make a judgement on the worth of something

Table 3.1: The hierarchy of the cognitive domain following Bloom's Taxonomy (1956).

Bloom's taxonomy has been used widely as a model to argue for the adoption of teaching strategies in both computer science and other disciplines (e.g. Brown, 2002; Mason and Grove-Stephensen, 2002; Moore, 1995). As such, the "task-oriented question construction wheel" (as seen in Figure 3.1) – a tool that helps educators to identify what is important to assess in relation to learning – has been designed.

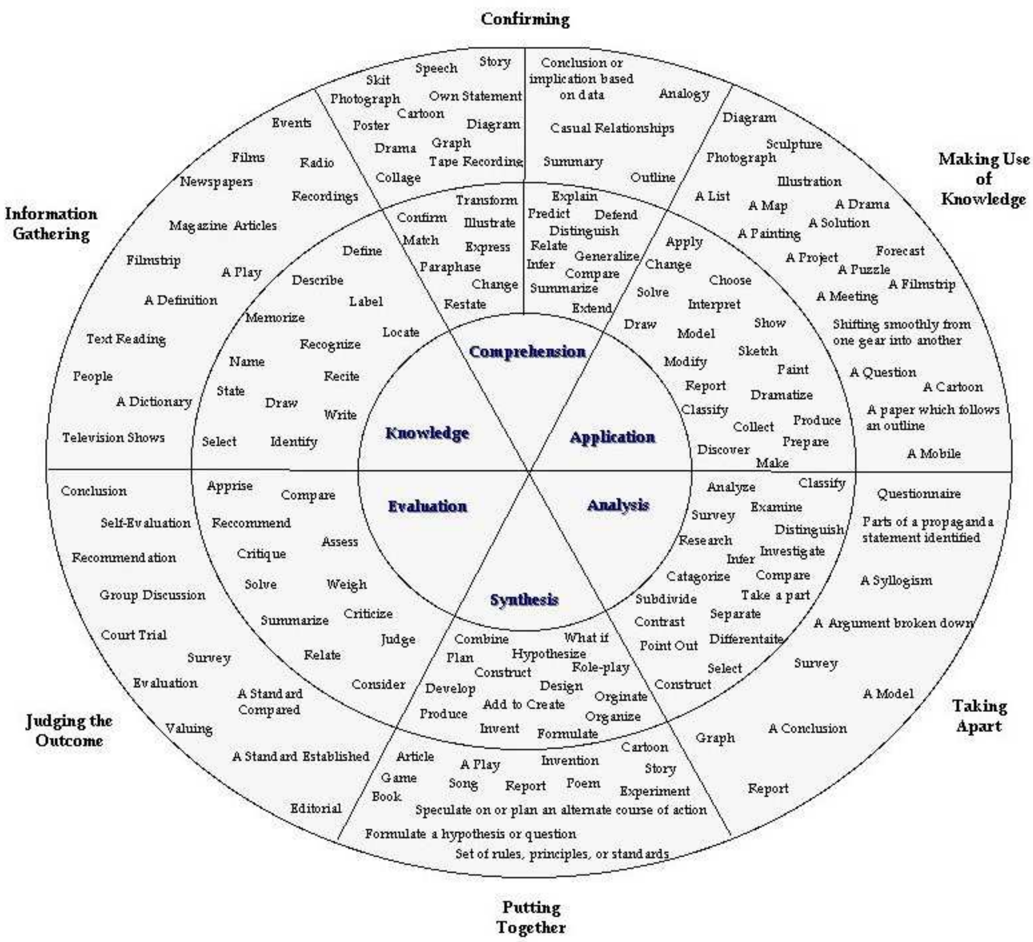


Figure 3.1: Task-oriented question construction wheel

In summary, Bloom's Taxonomy suggest that one cannot value or judge something until one

- Knows the facts.
- Understands the facts.
- Can apply the facts.
- Can take the facts apart.
- Put the facts together in such a way that new perspectives are revealed.

3.3.1 *Question design for CAA*

As discussed in the earlier chapter, CAA is becoming more popular with educators, especially in the subject areas of the hard sciences. A considerable amount of expertise in question design for CAA in the hard sciences does exist, and its best proponents choose to mix question types which (Brown, 2002):

- seek best-match answers;
- ask students to click on-screen hotspots to identify relevant points on graphs;
- get students to click on a series of arrows to demonstrate correct pathways on diagrams;
- require students to select and position labels for elements of graphs, charts or diagrams;
- ask students to put some given factors into an order of importance;

and so on, rather than just simple multiple choice questions.

In general, questions can be classified by general functions, namely questions which elicit information, questions which shape understanding, and questions which press for reflection (Morgan and Saxton, 1994). As such, it can be said that Bloom's Taxonomy is a good guideline to formulate questions.

Particularly appropriate for question design for CAA, Brusilovsky and Miller (1999) have suggested some components of a state-of-the-art question, namely the question itself (or stem), a set of possible answers, an indication which answers are correct, a type of the interface for presentation, question-level feedback that is presented to the student regardless of the answer, and specific feedback for each of the possible answers. In addition, an author may provide metadata such as topics assessed, keywords, the part of the course a test belongs to, question weight or complexity, allowed time, number of attempts, etc. This metadata could be used to select a particular question for presentation as well as for grading the answer.

3.3.2 *Summary*

Various models can be used to classify learning outcomes, among which the most cited is Bloom's Taxonomy. This model is particularly appropriate as a reference while designing questions for CAA.

3.4 Feedback in assessment

Just as assessment is important in the learning process, feedback has an essential role too. Race (1994a) has argued that how people learn best depends on the following key factors:

- The importance of 'the want to learn', or motivation.
- The fact that most learning is 'by doing', including practice, and trial and error.
- The importance of receiving feedback from other people.
- The need to make sense of what has been learned, or to 'digest' it.

He stresses that the ‘wanting, doing, feedback, digesting’ processes are best thought of in an overlapping way, rather than as a cycle of consecutive steps. He has also discussed else (Race, 1994b) the conflicts between ‘wanting, doing, feedback, digesting’ and assessment – particularly thinking of examinations. This model can be summed up visually as Figure 3.2.

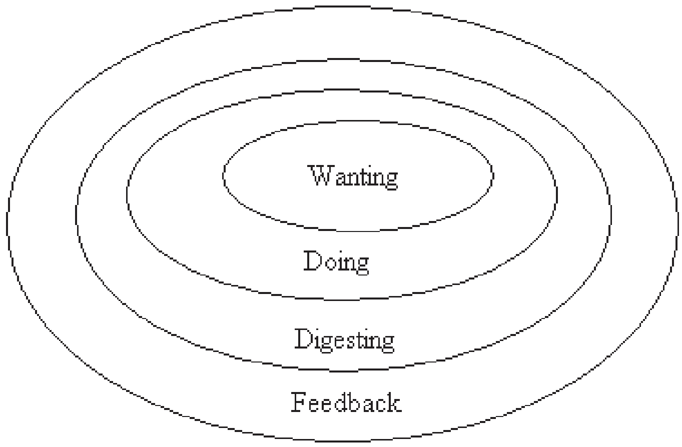


Figure 3.2: A model of the key processes needed for successful learning

This shows that student learning can be greatly supported and enhanced by useful and effective feedback (see also: Taras, 2002, 2003; McDowell, 1996; Dalziel, 2001; Brusilovsky and Miller, 1999). It can provide encouragement to students, reinforce and develop correct knowledge and reasoning, indicate areas of weakness in both learning and style, and direct future study in specific areas (Stephens and Percik, 2002). For these aims to be achieved, the feedback needs to be provided frequently, concentrated on specific areas relevant to the individual student, and received soon enough after the assessment to be considered relevant and easily assimilated. The effects improve with the level of detail in the feedback, but long-term retention rapidly declines as the length of time between the assessment and the receipt of feedback increases (Azevedo and Roberts, 1995, in Stephen and Percik, 2002; Brown, Race and Rust, 1995).

What is feedback, exactly? According to Ramaprasad (1983, in Taras, 2002) feedback is information about the gap between the actual level and the reference level of a system parameter that is used to alter the gap in some way. Brookhart and Sadler go even further to state that knowing what feedback to give assumes an understanding what caused the errors (Brookhart, 2001, in Taras, 2003) and that using feedback for future learning implies an understanding of their errors (Sadler, 1989, 1998, in Taras, 2003). Thus both of them signal that feedback is not a freestanding piece of information, but that it forms part of a learning context where all elements need to be engaged in the process. It thus follows that the understanding and using of feedback would benefit from being negotiated between the parties concerned so as to help to eliminate ambiguity and misunderstanding.

One may note that the usual options for the feedback include: simply telling if the answer is correct or not, or partially correct; giving the correct answer; and providing some individual feedback (Brusilovsky and Miller, 1999). Individual feedback may communicate: what is right in the correct answer, what is bad in incorrect and partially incorrect answers, provide some motivational feedback, and provide information or links for remediation. All individual feedback is usually authored and stored with the question.

Fone (2002) has proposed an improved assessment method that allows personalised feedback to be formulated. In a MCQ assessment, other than selecting an answer, the student is also asked to indicate the level of confidence they have in the selection, choosing: very confident/confident/uncertain/very uncertain. By asking students to supply an indication of how confident they feel about their answers in MCQs, misconceptions and insecurities can be identified. In turn, this allows personalised feedback to be given that is concise yet of high value.

Brown and Knight (1994) have even suggested that ungrading, where work is commented upon but where no mark is attached, may become the norm, freeing up the assessment system by allowing experiment and risk-taking.

No matter whether feedback is standalone or complementary to a mark, it is important to give useful feedback. Amongst 53 tips on interesting ways to assess students, Gibbs and colleagues have included a list of 20 guidelines for giving feedback, which were actually consolidated after consulting some of their students (Gibbs, Habeshaw and Habeshaw, 1988):

1. Keep the time short between the student writing and the feedback.
2. Where possibly, give instantaneous feedback.
3. Tie in the grade with the comment (i.e. not “An excellent piece of work: D”)
4. Summarise the comments and flag the fact that it is a summary.
5. Balance positive with negative.
6. Flag what is positive and what is negative.
7. Negative points should be constructive.

8. Indicate how the student can improve.
9. Follow up with oral feedback.
10. Aim for a dialogue.
11. Encourage students to evaluate themselves.
12. Encourage students to ask for feedback elsewhere (e.g. from other students or other members of staff).
13. Ask students what kind of feedback they want.
14. Make the criteria clear when setting the work and relate the feedback to the criteria.
15. Distinguish between different skills (e.g. the student may have lots of good ideas but be poor at spelling).
16. Offer help (e.g. "Would you like a refresher course on the use of the apostrophe?")
17. Give affective feedback (e.g. "It's really frustrating reading your essay because it could have been good but…" or "I enjoyed reading this…").
18. Make further suggestions (e.g. for further reading or developing ideas).
19. Distinguish between formative and summative assessments.
20. Give periodic oral feedback on rough drafts.

Despite the evidence showing that feedback can enhance learning, there seems to be an increasing phenomenon in the reduction of tutor feedback. With modularisation (that is, dividing course credits into semester-long units) and increases in class sizes, assessment patterns have changed. Often this involves streamlining coursework assessments, increasing time-constrained assessment, and examinations (Gibbs and Lucas, 1996, in Taras, 2002). This often has as a consequence a reduction in tutor feedback from assessed work (Taras, 2002).

The good news is that this can be compensated by the use of web-based learning technology (Korhonen *et al*, 2002; Brown, 2002), as the relatively new learning tool can automatically give immediate feedback to the students, with a fixed overhead cost. That is to say that regardless of the number of students, the effort to generate assessment and to give feedback is the same. Thus, even if the class size is incredibly large, students can still use the technology to obtain immediate feedback, such as formative feedback on correct and incorrect responses by the means of a screen of text including explanations of right and wrong answers. This moves CAA from a testing to a learning tool and can dramatically improve student performance. Nonetheless, one should note that it also adds a whole new dimension to the problems facing the writers of assessment items: it is very time consuming and 'uses up' questions that cannot readily be reused in summative assessment (Brown, 2002).

Another advantage of using CAA is that it can allow students an unlimited number of attempts. This, with the immediate feedback on their answers, can give students the confidence they need to engage with the problems (Grebenik and Rust, 2002).

Notwithstanding these positive arguments for feedback, both have noted that it is important to permit a feedback loop to take place (Boud, 1995; Laurillard, 1993). As Laurillard puts it, "*action without feedback is completely unproductive for a learner. As we learn about the world through acting on it, there is continual feedback of some kind, and if we can make the right connection between action and feedback, then we can adjust the action accordingly and this constitutes an aspect of learning. And it is not just getting feedback that is important, but also being able to use it.*" (p. 61)

3.4.1 *Summary*

Feedback plays an important part in the learning process, as it helps to identify the gap between the actual level and the reference level of a system parameter that is used to alter the gap in some way. This is particularly essential if students are to learn efficiently and effectively. It is also highlighted that feedback loop, which provides opportunities to allow actions based on feedback, can help enhance learning.

3.5 Summary

In this chapter, we have looked at education from a pedagogical perspective:

- Assessment has a strong influence on the way students behave and approach their learning. In the absence of assessment, learning will be incomplete.

- Whilst assessment is understood by most people as 'summative consideration of teaching', rather than as 'formative consideration of how to improve student learning', the latter plays an important role in learning too. Formative assessments, such as quizzes, provide the opportunity for teachers to ensure that students are able to do what teachers want them to do by the end of each module.

- Various approaches can be taken to classify learning outcomes, among which the most cited is Bloom's Taxonomy. This model is highly recommended as a reference while designing questions for CAA.

- Feedback plays an important part in the learning process, especially if the student is to learn efficiently and effectively. It is also highlighted that a feedback loop, which provides opportunities to allow actions based on feedback, can help enhance learning.

The next chapter will describe a proposed adaptive system that aims to encourage both tutors and students to adopt the little-used self-assessment mode.

Chapter 4 –

The ASAM System: the basic system

In the last two chapters, we have seen a review of assessment and computerised tools that have been researched and developed for assessment purposes. We can conclude that however little used is self-assessment, it enables students to self-evaluate whether they are able to do what teachers want them to do by the end of each module, and that the development of an adaptive self-assessment system is useful.

In this chapter, we shall look into a system that is designed to fill in this particular gap identified in the review – that is, a system that can encourage students to self-assess.

Section 4.1 will offer a definition of an adaptive self-assessment system and will also describe the aims and motivation behind the proposed system. Section 4.2 will include a summary of the phases involved in the design and development of the proposed system, ASAM (which stands for “Adaptive Self-Assessment Master”). The design for the ASAM system and its various components will also be documented.

4.1 Defining an adaptive self-assessment system

This section introduces the subject of adaptive self-assessment systems by defining the key concepts, followed by setting general aims and presenting the motivation behind the idea.

4.1.1 *Definitions*

In Section 2.1.2, we have seen that there is some difference between an adaptive system and an adaptable system. Whilst an adaptive system automatically adapts to the user given a user model, an adaptable system requires the user to specify exactly how the system should be different. It is the first definition that the proposed system follows.

Self-assessment, as seen in Section 3.2.1, is one of the various forms of formative assessment. Formal marks are not associated with this particular mode of formative assessment. Nonetheless, some form of feedback can be obtained in order to assist the student draw conclusions of where he or she stands in respect to the original objectives of the course or teaching module, so that he or she can attempt to improve upon their weakness(es).

Therefore an adaptive self-assessment system is to generate self-assessments that adapt to the abilities of every student, so as to help them monitor their own learning progress.

4.1.2 *Aims and motivation*

Whilst there have been research on CAT systems, work on adaptive self-assessment systems is sparse. This section discusses some of the initial and most fundamental questions that originate from the fields of assessment, feedback and CAT.

As can be seen from the last two chapters, however important self-assessment is in student learning, both teachers and students have not been putting sufficient emphasis on this mode of assessment (Section 3.2). This may explain why very few researchers work on the development of systems for self-assessment. Nonetheless, such applications are in great demand nowadays, especially since class sizes are on the increase, and teachers are finding it increasingly difficult to attend to the needs of each and every student on a one-to-one basis. Immediate feedback provided by self-assessment systems can, in a way, compensate for the lowering of teacher-student communication (Section 3.4).

Despite the significant amount of research into ITS, there is not as much research and development in adaptive assessment, and even less in assessing programming skills. Consequently, the focus of this research falls on the formative self-assessment mode for programming subjects, although the proposed application can also be used for the teaching of other disciplines.

Over the recent years, we have also seen a growth in the research and development of CAT applications. While such systems are beneficial to students since each is presented with questions of appropriate challenging levels (instead of fixed sets of questions for all students that may be either too easy or too difficult for each individual), there are still concerns over its usage. These include students' uncertainties over the "fairness" of the assessment (see Section 2.3.2).

Likewise, there has been an increase in research on systems that can mark free response questions. However, most, if not all, are not adaptive – that is, all students are presented with the same questions – and are summative. The writer often hears students comment that should there be some ways of knowing what is expected of them before the first summative assessment, it would be wonderful. In other words, they would very much appreciate informal feedback before they are graded formally. It can get harder to obtain such feedback, especially since the class sizes are on an increase.

Consequently, the author proposes a system to fill in this gap – an adaptive self-assessment system that can mark free response questions. This system, on one hand, allows students to practise questions/exercises as many times as they wish, and to get feedback upon their submissions. On the other hand, the students need not be too concerned with the "marks" as they would not be taken formally into consideration. The following sections (and the next chapter) explain further the proposed system.

4.1.3 *Summary*

This section has defined an adaptive self-assessment system, and has listed the aims and motivation behind the proposal of building one of such systems.

4.2 Design and development of ASAM

This research aims to develop a system that can help students in their learning at their own pace, and that can compensate for the feedback which they expect to receive from their tutors, in spite of large class sizes.

The design and development task involves the following phases:

1. Improving upon the CM MCQ system.
2. Developing the adaptive part of the new system.
3. Testing and refining the adaptive system.
4. Incorporating CM's capability to mark programming code into ASAM[1].
5. Another round of evaluation.

4.2.1 *Improving upon the CM MCQ system*

A few years ago, the CourseMarker (CM) MCQ system was developed as a final year project, using Java servlet technology (Slater, 2002). It was proposed as an extension to CM[2].

The system combined a graphical interface for entering questions into a repository, together with a mechanism for searching questions from particular subjects, topics and other such properties and combining them into a test. The questions were stored in an XML data format, which allowed easy data exchange, parsing and transformation. Also, the use of the repository meant that questions could be reused in any number of tests without having to re-enter any question data.

[1] Details of this development phase can be found in Section 5.1.

[2] More information on the CM system can be found in Section 2.2.2.2.

There was also a student client environment, which allowed students to take a test, either as an exercise in CM or as an individual test, which used the system's own internal authorisation system to validate users. This part of the system provided a timer mechanism to prevent students from exceeding time limit. Tests were then marked automatically by the system, and depending on the wishes of the test-setter, the student was given some form of feedback to indicate the mark obtained or just a confirmation of submission.

However, due to the time constraint of development as a final-year project, although the MCQ system then functioned reasonably well on its own, or with CM, extra effort has to be put in to make it work more robustly and flexibly with CM.

For instance, the final state of the CM MCQ system was such that the basic functions (including addition of questions into the question bank, and creation of a test) were available. However, the various components were separate. In other words, there was no single overall central medium that could allow users to easily access the various components. At that time, users had to key in the individual URLs manually. Some examples of the URL are:

- http://62.254.3.59:8080/AddQuestion.htm
- http://62.254.3.59:8080/SearchQuestions.htm
- http://62.254.3.59:8080/TakeTest?test=MyTest.xml

This had imposed an additional burden on the mental state of the user, having to remember the complicated URLs (Atkinson, 1990; Baddeley, 1976, 1993; Eysenck and Keane, 2000). Due to the complex parameters and the hosting server's case-sensitiveness of the URL address, it could be, to some extent, confusing too.

Therefore, there was a need for a generic central menu to be created (for instance, a drop-down list), for the administrator (or course developer) to access the various functions more easily, and the students to gain access to the relevant exercises/tests.

Other unresolved issues, such as making the application more generic, had to be dealt with too. As the final product of the final-year project, the majority of the servlet code was hard-coded, making it difficult to maintain the system. Therefore, one of the first tasks to improve upon the system was to make it more generic. The simplest and logical approach was to re-write the code of the entire system, incorporating the true advantage of JSP[3] programming. Whilst re-writing the system, some more improvements were to be incorporated, which include:

- integrating XSD[4] to define the XML[5] metadata, instead of using the original DTD[6] document – this is to facilitate easier maintenance and future enhancements, since XSD is the new generation of defining XML documents;

---

[3] JSP stands for JavaServer Pages. For more information, please refer to Hall (2000), etc.

[4] XML Schema Definition. For more information, please refer to Fallside (2001).

[5] EXtensible Markup Language. For more information, please refer to Bray *et al* (2000).

[6] Document Type Definition.

- replacing SAX[7] with the JAXB[8] approach as the adopted API[9] for XML while developing the application – with a large amount of XSD documents and XML elements, JAXB permits easier maintenance; and

- improving the security feature by also allowing POP3 as the medium, so that the log-in process is more secured. The original log-in list was stored in an ASCII file and the content not encrypted, and therefore could be easily hacked into. Also, this does not require students to have yet another set of user-id and password, as the POP3 authentication process can be linked to an existing authentication mechanism.

4.2.2 *Developing the adaptive part of the new system*

One of the main practical contributions of this research is to develop the adaptive capability of the CM MCQ system, which accepts data from a student model and generates personalised exercises for every student, based on their individual profile. Whilst CM MCQ was originally developed to mark fixed response questions, the adaptive self-assessments need not be limited to such

[7] Simple API for XML. For more information on SAX and other XML APIs, please refer to Armstrong *et al* (2001).

[8] Java Architecture for XML Binding. For more information, please refer to Ort and Mehta (2003).

[9] Application program interface.

question types. They can also include programming and diagrammatic[10] questions. Therefore, the main task is to incorporate the adaptivity feature into the revamped CM MCQ system, with the end product named ‘ASAM’, and to integrate CM’s capability of marking programming code into this new system. The latter is done by extracting the marking server of CM and integrating it with ASAM.

Details on the various components of the adaptive system will be covered in the next section. Fundamentally, the course developer has to create a skeleton for each questionnaire to define the structure of the self-assessment questionnaire. With this skeleton, the system will select the appropriate questions from the question bank that match each student’s ability.

The output of the system will be such that upon submission and marking, students’ marks are to be stored in the student model, so that the metadata can be used in the future, to determine their ability level for each topic when the student next log on to the system.

4.2.3 *Testing and refining the adaptive system*

Once the development of the beta version was completed, a user trial was conducted during the first part of a basic Java programming module, at the School of Computer Science and Information Technology, University of

[10] The fact that the complex diagrammatic marking application was developed as a Java application, does not permit easy integration into the new web-based adaptive system. As such, at the moment, ASAM is not capable of marking diagrammatic questions.

Nottingham. Data was gathered during and after the user trial, and later analysed.

4.2.4 *Summary*

This section has described the various tasks involved in the design and development of the adaptive self-assessment system, ASAM, which include the improvement upon the CM MCQ system; the development of the adaptive part of the new system; and testing and refining the adaptive system.

4.3 Components of ASAM

As described earlier, the adaptive self-assessment system, ASAM, comprises of several components to make the adaptive capability possible. These components are:

- A question bank[11] to store all the questions.
- An authoring tool for teachers to use in creating questions and questionnaire skeletons.
- An assessment generating system which the students are to log on to and to attempt the questions presented.
- A student model to store the profiles of individual students.

[11] This is also known as item bank in the research community.

- A marking tool to mark students' submissions and pass on the data to the student model.

- A utility tool with which teachers can obtain some simple statistics.

Figure 4.1 illustrates the interaction among these components, and the following sub-sections detail more on the various components.

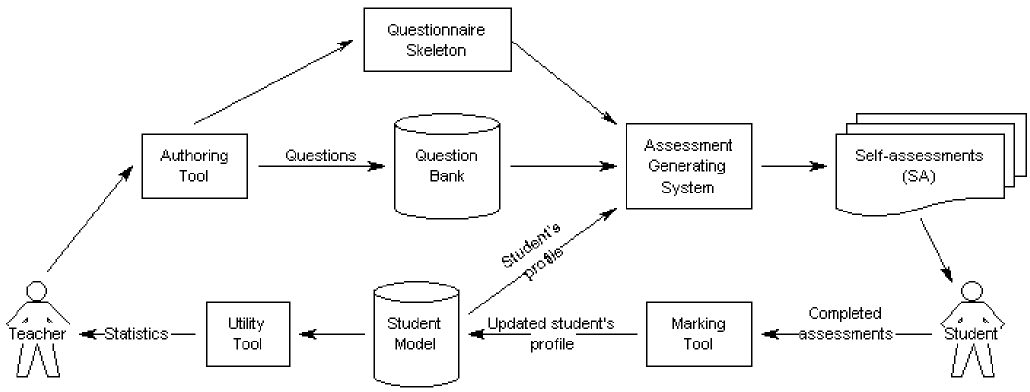


Figure 4.1: The interaction among the various components of ASAM.

4.3.1 *Question bank*

Just as the name suggests, this component of ASAM stores all the questions in the system. As seen in the literature review (Section 2.3.1), we know that the Item Response Theory (IRT) has had a great influence in the development of CAT systems. Nonetheless, it is also known to be difficult to implement in practice due to its complexity (PennState, 2000). Thus, the author has decided to use a less sophisticated approach to classify the questions in the question bank.

For each topic, the questions are grouped into three different levels of difficulty, namely beginner, intermediate and advanced. Teachers can assign the questions into the various levels of difficulty intuitively, that is, based on their intuition or experience. Should a guideline be necessary, the author recommends a reference to Bloom's Taxonomy. As seen in Section 3.3, all questions can be classified as to difficulty, corresponding to the six ascending levels of abstraction as defined in Bloom's Taxonomy (Bloom, 1956; NYIT, 2003) – the lowest three levels to 'beginner', the next two higher levels to 'intermediate' and the highest to 'advanced' (as shown in Table 4.1).

Level of difficulty	Corresponding cognitive aspect of learning, as defined in Bloom's Taxonomy
Beginner	Knowledge Comprehension Application
Intermediate	Analysis Synthesis
Advanced	Evaluation

Table 4.1: The levels of difficulty of question items and their corresponding cognitive aspect of learning

4.3.1.1 Metadata for each question

To facilitate the generation of a self assessment that matches each student's ability during each attempt, metadata, such as the following, has to be tagged to each question[12]:

- Question type (e.g. MCQ/multiple-response/numeric/programming)

[12] The XML schema for the questions is listed in Appendix C.

- Subject theme
- Level of difficulty (e.g. beginner/intermediate/advanced)
- Time expected to complete the solution

4.3.2 *Questionnaire skeletons*

The set of questions in a given questionnaire can be of either of two formats – namely fixed and adaptive – and is specified in a questionnaire skeleton[13]. The availability of two formats is to increase the flexibility in terms of the types of questionnaires that the teacher can set.

Under certain circumstances, the teacher may want all students to attempt the same set of questionnaire, and thus the first of the two formats, which is the fixed set of questions, allows the teacher to do so. A fixed questionnaire consists of questions that are pre-selected from the question bank by the teacher. This means that all students will be presented with the same set of questions.

Alternatively, the teacher can opt to let the system choose a set of questions dynamically from the question bank, so as to give each student a different questionnaire that matches the student's abilities. With this format, the teacher defines an adaptive questionnaire skeleton in advance, specifying the number of questions and the topic each question covers, with an option of also

[13] The XSL schema for the questionnaire skeleton is listed in Appendix D.

specifying the question type and pre-requisite topic. Questionnaires of this format allow students to self-evaluate (and thus learn) at their own pace.

An example of questionnaire skeletons is one with three questions, two of which are MCQs and the remaining one a programming question, while another self-assessment could have, say, ten MCQs covering various topics. There are also other possible combinations of questions in a questionnaire skeleton. For instance, a course developer wants to set a questionnaire of ten fixed questions for all students, while for another questionnaire, he may want the system to draw eight questions from the question bank dynamically, resulting in each student possibly having eight different questions for the same self-assessment, depending on their individual ability. Even if two students have attained the same level of ability for each relevant topic, they may be presented with different sets of questions, as the system makes a random selection from questions with equal difficulty.

The idea of letting teachers define the format of the questionnaires and the types of questions in each of them, is to allow them have better control over the self-assessments that are provided for the students, for instance, knowing the combination of topics in each questionnaire. This can aid the teachers in monitoring students' learning progress.

4.3.3 *Authoring tool*

To add questions to the question bank and to create questionnaire skeletons, the teacher or course developer has to use the authoring tool. Figures 4.5 to

4.9 depict some sample screenshots for some of the processes involved in achieving the two tasks.

When adding questions to the question bank, the teacher can choose to enter the metadata of each question manually via the web, or to import an existing XML file. These options are shown in Figure 4.2.

Similar options are available while creating a questionnaire skeleton (as shown in Figure 4.3). For fixed questionnaires, the teacher is to compile a list of questions by searching for them individually (as in Figure 4.4) before finalising the details of the questionnaire (as in Figure 4.5). On the other hand, for adaptive questionnaires, the teacher will have to specify the general metadata of each question, including questionnaire name, subject, topic, pre-requisite (if any) and question type, as illustrated in Figure 4.6.

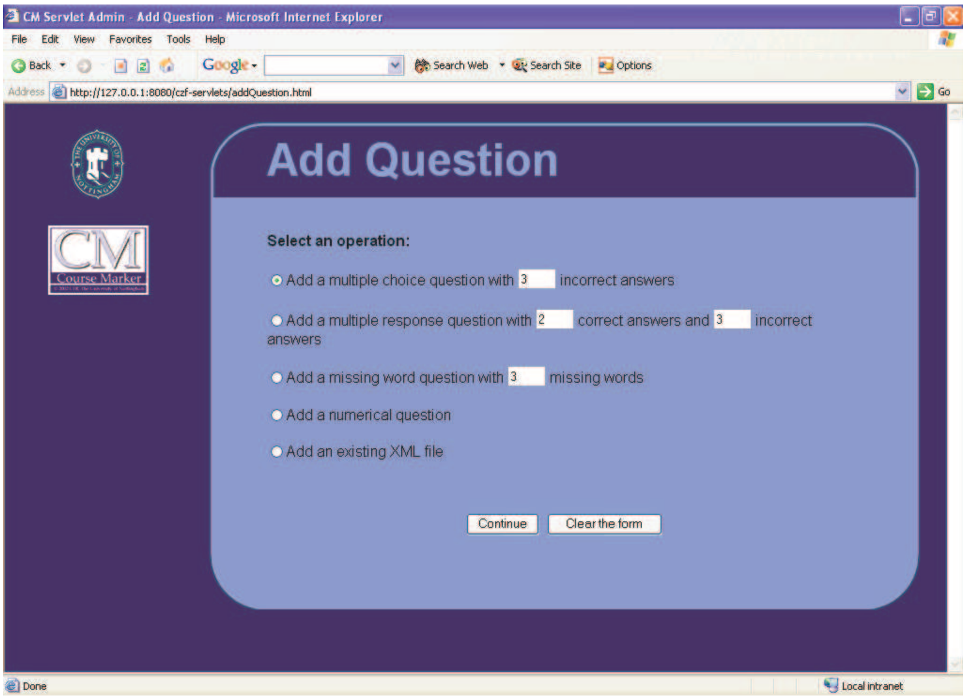


Figure 4.2: A screenshot of ASAM's authoring tool to add questions to the question bank

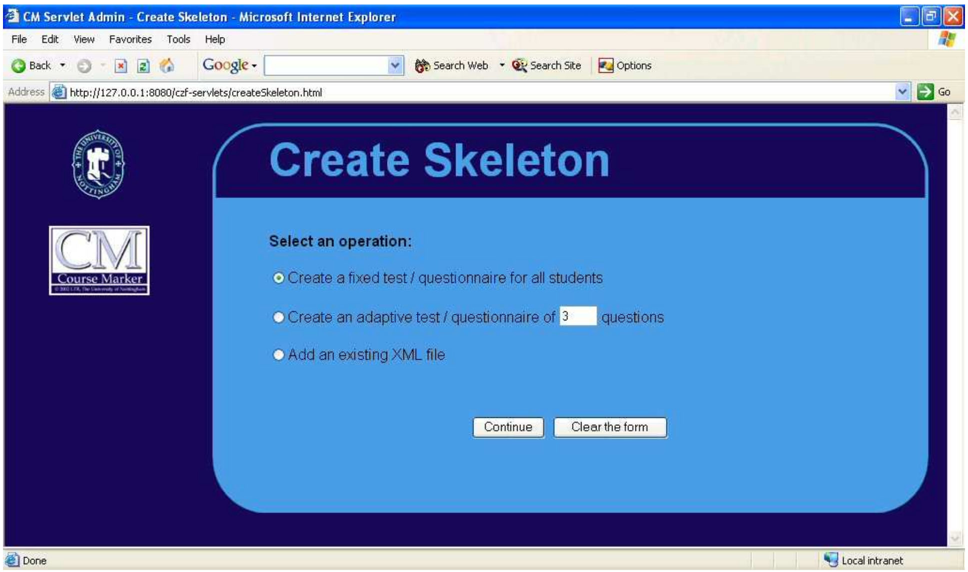


Figure 4.3: A screenshot of ASAM's authoring tool to create questionnaire skeletons

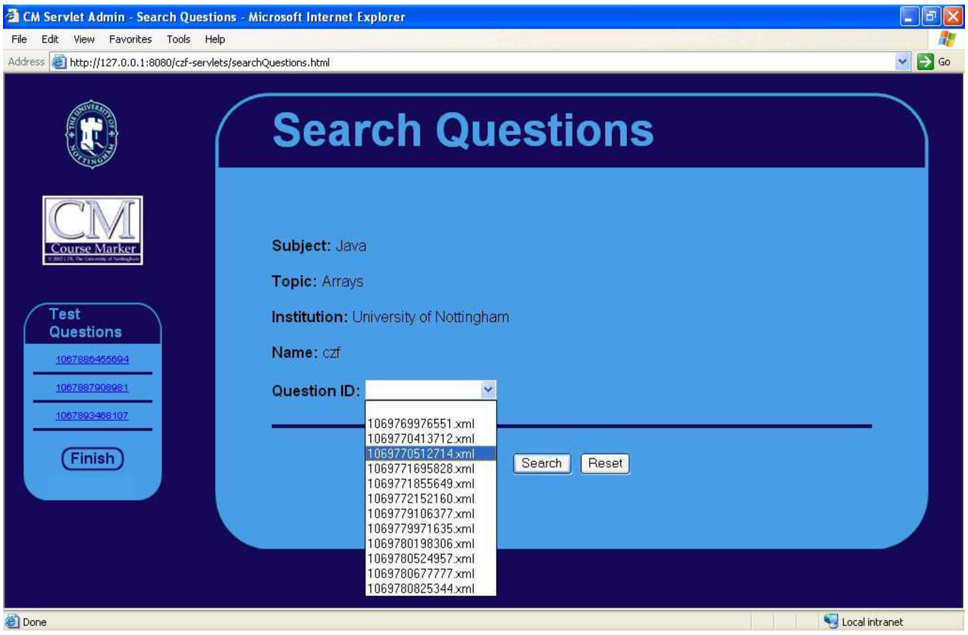


Figure 4.4: A screenshot of ASAM's authoring tool to search questions within the question bank

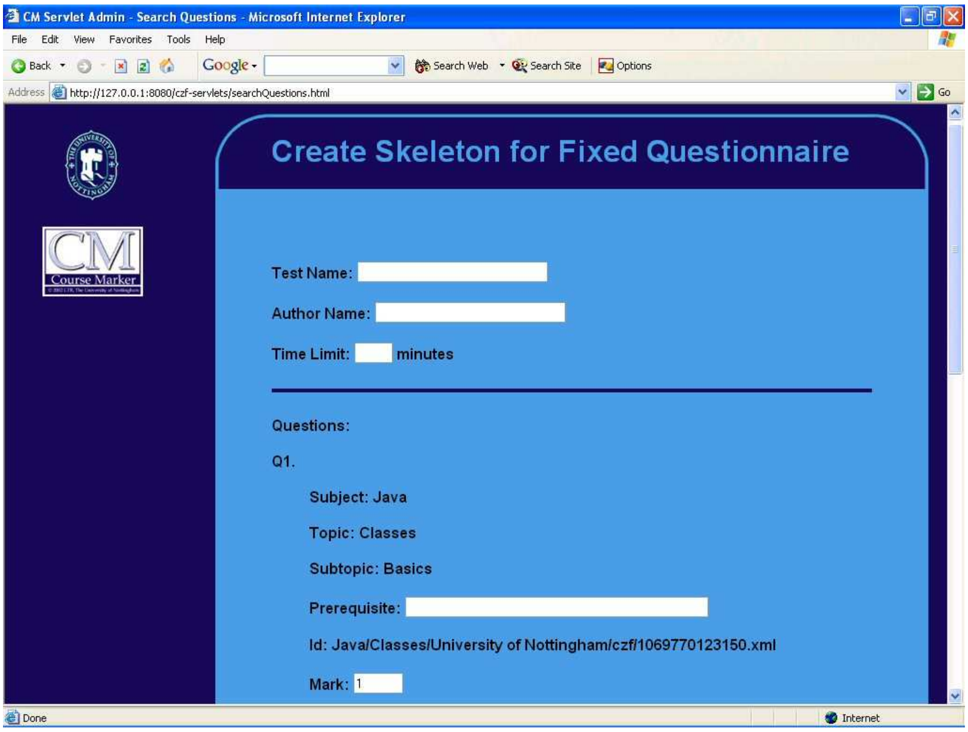


Figure 4.5: A screenshot of ASAM's authoring tool to create skeletons for fixed questionnaires

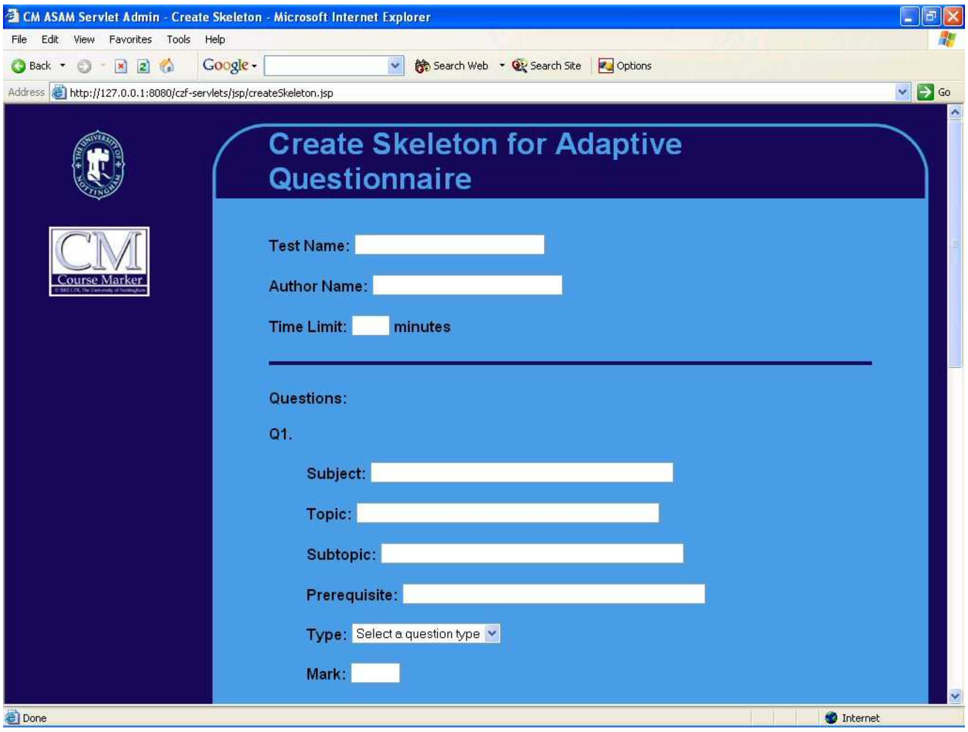


Figure 4.6: A screenshot of ASAM's authoring tool to create skeletons for adaptive questionnaires

4.3.4 *Assessment generating system*

With the questions and questionnaire skeletons in place, the system is ready for students to self-assess. To attempt a fixed self-assessment, it is very much the same as accessing one with any other assessment system. On the other hand, when students log on to the system to attempt an adaptive self-assessment, ASAM will generate a set of questions, tailoring to each student at an appropriate level of difficulty for that student's ability.

First of all, for every question in the questionnaire, a list of questions that match the student's ability level (for that topic), the topic and question type (if applicable), will be pre-selected from the question bank. It is then that ASAM chooses one from this list randomly. If there are options available for the question (for instance, in a multiple-choice question or a multiple response question), the options will be randomised, so that students cannot memorise the answer or copy the answer from fellow students, thus making sure that they do not have prior knowledge of the correct answer.

At any one point, if the pre-requisite (if any) of a question is not satisfied, the generating process will be aborted, and the student will be prompted with a message, asking him/her to attempt the questions of the pre-requisite topic first.

All the questions of each questionnaire are displayed all in one go, and an internal check is conducted to ensure that no single question appears twice in each individual questionnaire. Nonetheless, it is still possible that the same question is presented to a student at different attempts.

Depending on the size of the question bank, it is usual that each student may be presented with different sets of questions, even when their ability level is the same at each attempt.

4.3.5 *Student model*

All the profiling data and assessment results are stored in a student model. The profiling data for each student includes:

- User ID
- Ability level for each topic
- History of previous attempts

This data is stored partially in a database (which can speed up retrieval and data query even for a huge database) as well as in XML files (which mainly store student submissions as archives). While the volume of archives can be quite huge, no frequent access is required of such data. Therefore it was decided that they would be stored in a XML file for each student user.

From time to time, the system will re-adjust the ability level for each student for each topic accordingly, may it be promotion or demotion to the next level. The system implements a simple updating approach for the student profile. Three consecutive right (or wrong) answers given by the student for each topic will determine a promotion (or demotion) for his/her ability level for that topic. This is based on Abdullah and Cooley's argument (2000) that two pieces of positive evidence or two pieces of negative evidence are required to allow the tutor to reach a firm judgement. Thus, they suggest that at most three questions of a specified level of difficulty requiring a particular skill are required. The repetition (of the extra question) helps eradicate uncertainty due to the possibility of guesswork or careless slips.

However, there is a slight variation for programming questions. If there is a compilation error and if compilation is not one of the marking criteria (as specified by the question setter), the student model will not record the attempt. For it is regarded that it may not be fair to fail an attempt, simply because of a silly mistake such as a missing semicolon, which could easily cause a compilation error and would consequently affect the promotion or demotion of one's profile ability. In addition, since the mark for each programming question is in percentage, the system would consider anything below an arbitrary cutting point (for example, 80%) as a wrong attempt.

4.3.6 *Marking tool*

Besides the usual marking of questions and passing on the marking outcomes to the student model (to update the students' profile), ASAM's marking tool also returns feedback to students at the end of the marking process.

From the literature review (Section 3.4), we can conclude that immediate feedback is something students appreciate a lot, especially when they are self-assessing. Therefore, feedback is given at the end of the marking of each assessment (as shown in Figure 4.7).

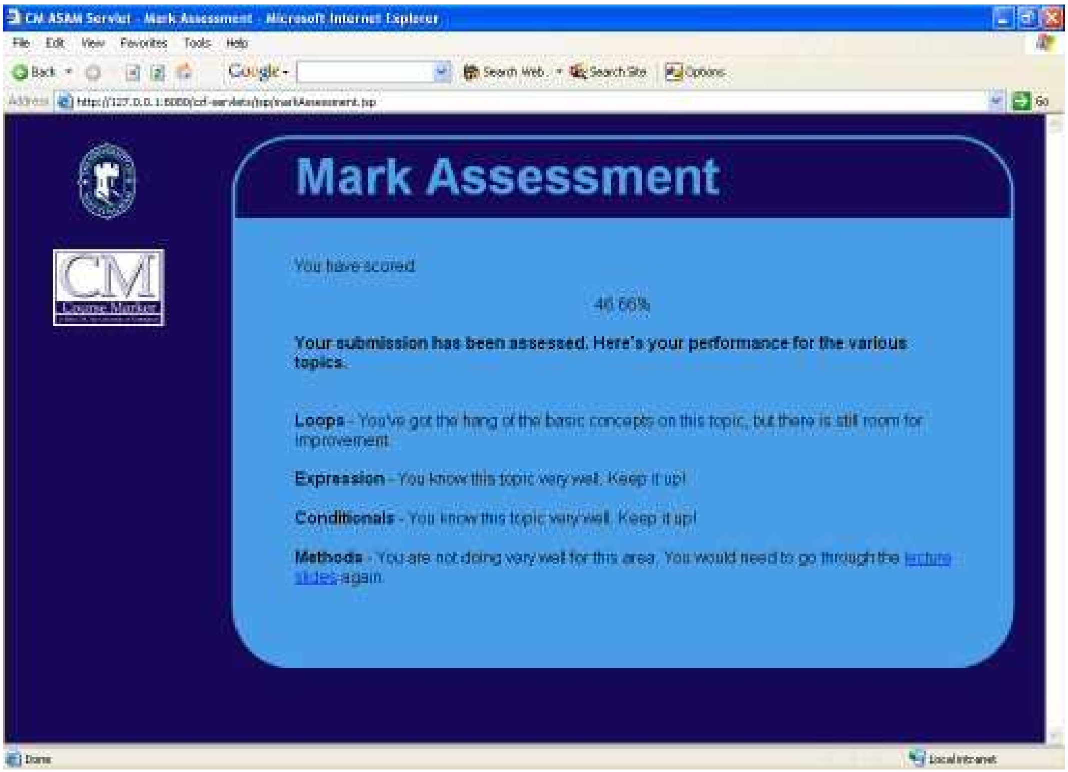


Figure 4.7: A screenshot of sample feedback returned by ASAM's marking tool

In relation to a research on a collaborative question search and play portal, Davies and Davis (2006) reckon that important design questions [for search results] are, what, of the information presented, is useful, and what other information could be shown to help users. Likewise, these two questions can be applied to the design of feedback returned to ASAM student users. It is important that information returned is useful to students; otherwise, it will not do them any good, or even worse, confuse them. However, it does not mean that elaborate or very specific feedback would imply greater usefulness either. For instance, replicating questions in feedback may not be a good idea. For one, it may encourage students to regurgitate answers for future attempts. Secondly, the idea of using a question repository is to reuse the questions in different occasions. It is often argued that increased exposure of questions would jeopardise their use in future assessment sessions (Freeman and Lewis, 1998, in Lilley *et al*, 2005).

The type of feedback[14] provided by ASAM would not be very specific. A sample feedback format would consist of some brief information of the student's ability for each topic covered in the questionnaire and some recommended links for further study. Such design is aimed to encourage students to improve upon a wider topic area, (instead of just focussing on some specific sub-topics), thus building a better foundation on the topic. As Barker and Lilley (2006) put it, one of the potential benefits [of having an automated feedback system based on a computer-aided test] was how the feedback might be used in order to help students do extra work, either remedially, or as extra challenges.

Once again from Section 3.4 of the literature review, it is suggested that repeated attempts are appreciated in self-assessment mode. As a result, ASAM is designed such that students are allowed to have unlimited number of attempts for each self-assessment.

From the XML schema for questionnaire skeleton (in Appendix D), one may notice that the ASAM feedback for each topic is specified in the questionnaire skeleton, instead of in individual questions. This implies that the feedback would not be applied to all courses or modules that may use some common questions. For instance, Courses X and Y both use certain questions, but the

[14] The feedback format may be slightly different for programming questions (see Chapter 5).

feedback returned for such questions may not be the same. This arrangement allows the questionnaire setter to customise feedback for different courses or modules or even questionnaire.

4.3.7 *Utility tool*

A utility tool has also been developed to allow the tutors to generate some simple statistics, such as the scores of all students for a particular assessment, a list of the number of attempts of the self-assessments by each student and a breakdown of the students' ability in various sub-topics. The output is data shown in the browser, or a text file. The tutor can then import the latter form of data into some third-party applications, such as Microsoft Excel or SPSS, to do further analysis. This allows the tutors to generate statistics that traditional paper assessment methods cannot provide easily, if at all (referring to Sections 2.2 and 3.2).

4.3.8 *Summary*

This section has detailed the various components of the ASAM system and the interaction among them.

4.4 Summary

In this chapter, we have seen the definition of an adaptive self-assessment system, and the aims and motivation behind the development. In addition, we have covered the tasks involved in the design and development of the ASAM system, which comprises of a question bank, an authoring tool, an assessment generating system, a student model, a marking tool and a utility tool. The interaction among the various components has also been described.

In the next chapter, we shall look in details at how the basic ASAM system is integrated with the feature of marking programming questions.

Chapter 5 –

The Enhanced ASAM System: adaptive programming exercises

The previous chapter describes the rationale behind the basic ASAM system. We have also seen how the basic system generates self-assessments with fixed-response types of questions. In this chapter, we shall look further into the ASAM system, with respect to adaptive programming exercises.

Section 5.1 will document an enhancement to the basic ASAM system, to integrate the capability of marking programming code. In Section 5.2, we will follow four fictional students on their progress when they use ASAM for self-assessing purposes. Section 5.3 will then compare ASAM with more conventional adaptive systems and the non-adaptive CourseMarker system, looking from both a student's and teacher's perspectives.

5.1 Enhancing ASAM with adaptive programming questions

As seen in the previous chapter, the basic ASAM system can generate and mark adaptive questions of some fixed response types, including multiple choice and multiple response types. However, as stated in Section 2.2.1, there are some disadvantages to fixed response questions. Therefore, in an attempt to better assess students' learning progress, or rather to allow the students to better self-assess their own progress, the inclusion of some mode of free response questions into ASAM was considered. It was then decided to incorporate the marking of programming code. This is made possible by integrating ASAM with part of CM, namely its marking subsystem. The steps involved are as follow:

1. Tagging CM programming questions in XML format;
2. Establishing a Java RMI[1] connection between ASAM and CM;
3. ASAM preparing and sending an "exercise package" to CM;
4. CM marking (and archiving) the student's submission and returning a marking outcome to ASAM; and,
5. ASAM returning appropriate feedback to student and updating the student model.

Figure 5.1 illustrates the interaction between ASAM and CM that permits the marking of adaptive programming exercises.

[1] Remote Method Invocation. For more information, please refer to Java Sun (2005).

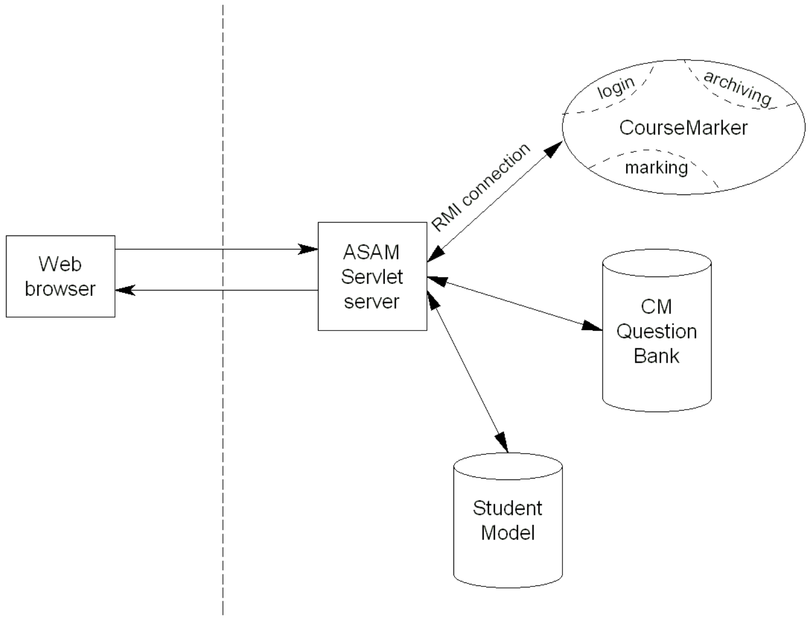


Figure 5.1: The interaction between ASAM and CM to allow the marking of programming code.

5.1.1 *Tagging CM programming questions in XML format*

In order for ASAM to recognise CM programming questions, these questions have to be tagged in XML format. However, not all information about each question is required to be converted into the recognisable format. The essential metadata includes: CM reference (namely the codes of the corresponding CM course, unit and exercise), the question details (which can be different from that presented in CM), and the filename of the skeleton Java file. Such metadata is more of a reference point to CM, rather than tagging everything about the CM questions (which may include the marking scheme, etc).

A sample XML document of a programming question is listed in Figure 5.2.

```
<?xml version="1.0" encoding="UTF-8"?>
<problem>
<head>
   <subject>CS</subject>
   <topic>Java</topic>
   <subtopic>u1</subtopic>
   <institution>University of Nottingham</institution>
   <author>
      <name>czf</name>
      <address></address>
      <email></email>
      <tel></tel>
   </author>
   <difficulty>Beginner</difficulty>
   <duration>5</duration>
   <created>
      <time>21:45</time>
      <date>19/10/2005</date>
   </created>
   <id_number>1129754738951</id_number>
</head>
<body>
    <programming>
       <question><![CDATA[The following program code attempts to read a number of shifts worked, and to compute and print the time the provided number of shifts would last.
<p>
You are to modify the code to include
<blockquote>
         the use of constant SHIFT (with a value of 14700)
</blockquote>
for the length of shift.]]></question>
       <cmref>
          <course><![CDATA[Java]]></course>
          <unit><![CDATA[u2]]></unit>
          <exercise><![CDATA[packing1]]></exercise>
       </cmref>
       <skeletonfile><![CDATA[Packing.java]]></skeletonfile>
    </programming>
</body>
</problem>
```

Figure 5.1: A sample XML document of a programming question.

5.1.2 *Establishing a RMI connection between ASAM and CM*

The procedure for the selection of adaptive programming questions according to students' abilities is the same as that for fixed response questions, more information on which can be found in Section 4.3.4. However, when a student clicks on the "Submit" button for an adaptive programming questionnaire, ASAM will attempt to establish a RMI connection with CM, passing on the identity of the student user to CM for authentication.

Whilst it was also possible to incorporate CM's marking subsystem into the web-based ASAM application – i.e. running CM's marking subsystem as part of the serlvet server - this approach could slow down the marking process, as it would use up a lot of the servlet server's resources. This phenomenon could potentially intensify when there are many students trying to submit at the same time. The idea of using the RMI mechanism is supported by Foxley *et al* (2001), for they consider RMI more effective than other client/server methods and relatively easy to use as it makes the network transparent to the programmer. Hence, it was decided that CM's marking subsystem was to be run independently from the web-based ASAM system.

5.1.3 *ASAM preparing and sending an "exercise package" to CM*

Once CM authorises the RMI connection, ASAM will prepare an "exercise package" of the student's submission for each ASAM programming question[2] and send it to the CM marking subsystem. The "exercise package" includes the question's CM corresponding course code, unit code and exercise code, as well as the student's submitted source code and all other necessary files (if any) for submission to a CM's exercise (as depicted in Figure 5.2) – the number of files required depends on the configuration of each CM exercise.

CM corresponding Course	CM corresponding Unit	CM corresponding Exercise	Student's submitted source code	Any other necessary files

Figure 5.2: ASAM's exercise package for each programming question that is to be sent to CM's marking subsystem.

5.1.4 *CM marking the student's submission and returning a marking outcome to ASAM*

Upon receiving the "exercise package" from ASAM, CM will process its usual marking of programming exercises, the outcome of which will be returned to ASAM. In addition, a copy of every submission to CM, together with its corresponding marking result, will be stored in CM's archiving subsystem. Hence, if the teacher or anyone else wishes to examine the student's

[2] A CM exercise is equivalent to an ASAM question.

submission for some reason at some point in the future, they can access CM's archiving subsystem.

The assessment of CM exercises is carried out by configuring a set of metric tools that perform various quality checks upon submission of a student's solution (Higgins *et al*, 2002). The available metric tools permit the assessing of typographic layout, dynamic execution, and program features.

For typography, CM's marking subsystem checks the program layout, indentation, choice and length of identifiers and usage of comments. All typography parameters can be customised on a per exercise basis.

However, typographical layout is not considered a major concern for ASAM self-assessment – though one may argue that it is best to cultivate a good habit at an early stage of learning – and hence, this feature is not taken into consideration for submission to ASAM. In addition, students may find it not as easy a task to format the indentation on a web browser as when compared to using a more powerful editor.

For dynamic execution, the marking subsystem runs the student's program several times with various sets of test data. This verifies whether the student's program is correct and whether it satisfies the stated specification. The marking subsystem gives a wide range of options concerning the flow, control and order in which the dynamic tests are executed.

For program features, the student's source code is checked for special features that are exercise dependent. For instance, an exercise set on a unit that teaches the difference between "while" and "for" statements would typically include a feature test to ensure the appropriate use of each construct.

5.1.5 *ASAM returning appropriate feedback to student and updating the student model*

Upon receiving the marking outcome (in the form of a tree) from CM, ASAM will analyse the marking result tree and return a revised feedback and marks to the student. It was decided not to simply return the CM marking result to the student, but to adapt its feedback for ASAM users, since some feedback returned by CM may not be overly useful[3], such as:

- *"% of correct indentation of braces feedback: Too much bad indentation of braces"*
- *"Average characters per line feedback: Enough characters per line"*
- *"% uncommented closing braces feedback: Try decreasing the amount of uncommented closing braces"*

On many occasions, the student's source code may seem perfectly all right to human eyes; however, the CM's marking subsystem may return some obscure feedback, (and a lower grade) on, for example, the typographical layout.

[3] There is evidence that different batches of students, year after year, have some difficulties understanding some of CM's feedback, as there are always similar repeated queries being sent to the helpdesk for the G51PRG module at the University of Nottingham. There is even a webpage dedicated to help students in addressing some of the issues - http://www.cs.nott.ac.uk/~bxm/teaching/g51prg/help/ [last accessed: 28 Jul 2006].

Hence, ASAM attempts to personalise the feedback by presenting it in a different manner. Figure 5.3 displays a sample result tree from CM in a graphical format, and Figure 5.4 a sample screenshot of ASAM's feedback and marks for an adaptive programming question.

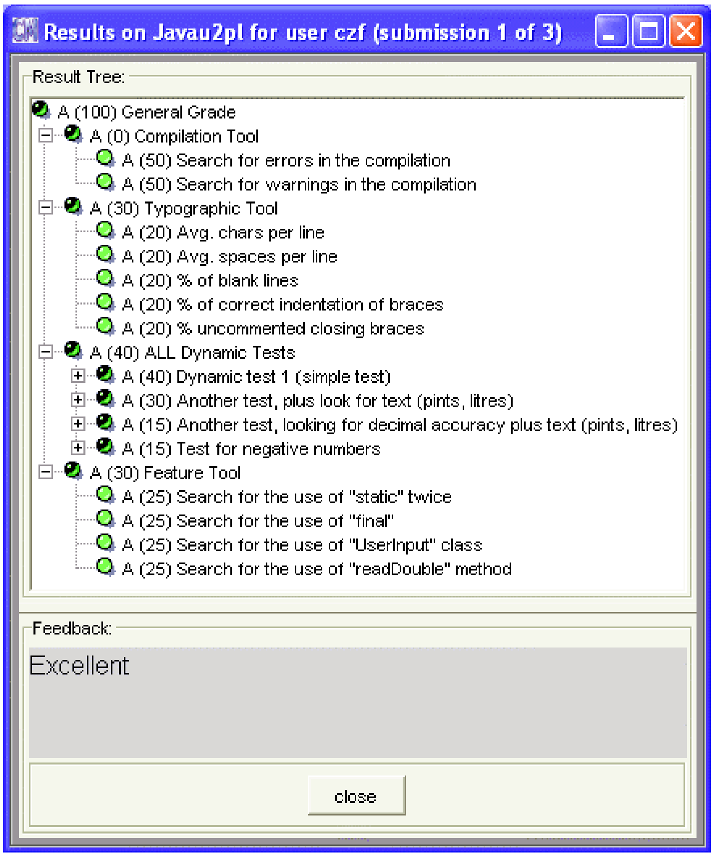


Figure 5.3: A sample CourseMarker result tree in graphical format.

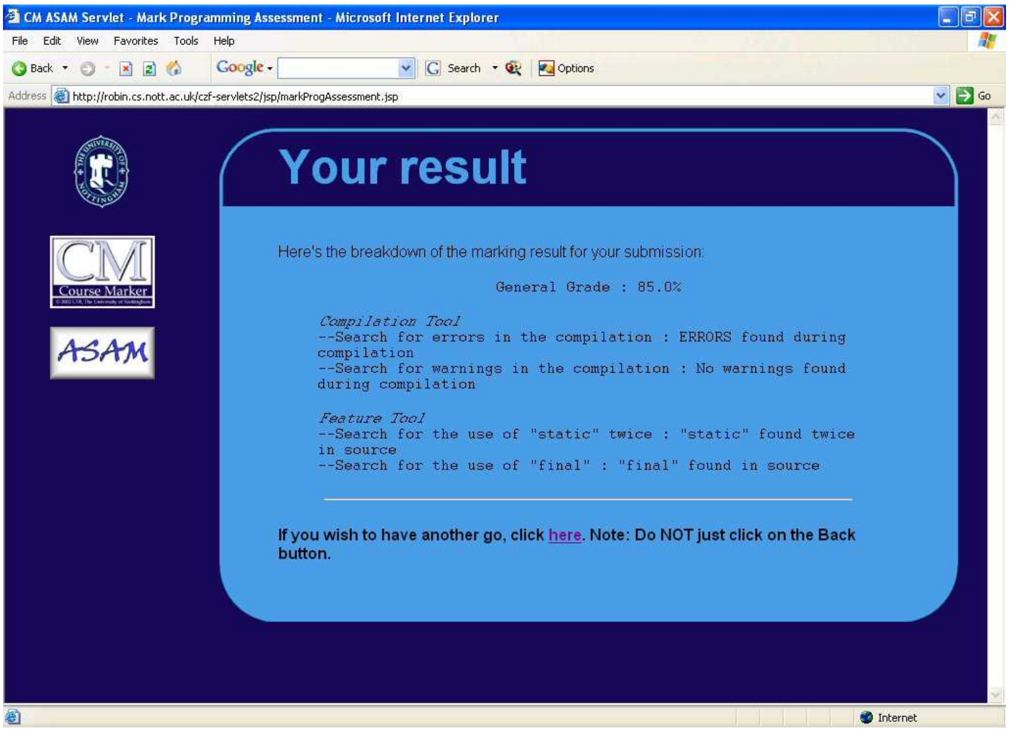


Figure 5.4: A sample feedback returned by ASAM for a programming question.

With respect to keeping track of students' profile, the enhanced ASAM system will update the student model[4] as it would for any fixed response questionnaire – that is, three consecutive right/wrong answers for each topic would determine a promotion/demotion to the next/previous level. Hence a sufficiently high cut-off mark is needed to make sure that the student has answered the programming question well enough, and at the same time, to give some leeway for minor errors – the teacher is reminded to keep this (cut-off mark) in mind when designing programming questions for use in ASAM. For the purpose of the testing for this research, it was decided that any mark above 80% will grant a correct entry (for each programming question) in the student

[4] More information on the student model can be found in Section 4.3.5.

model, whereas anything below 80% (inclusive) will be considered as an incorrect entry. The reason for such decision is that it is not envisioned that ASAM is going to be used for assessing students on complex or multiple Java concepts, but more on the simple and individual ones, such as programming basics, conditionals, loops, and arrays. Therefore, a relatively high cut-off mark has been set for the testing of this research.

In an attempt to emulate CM, it is recommended that an adaptive programming questionnaire is to consist of only one programming question. This is so as not to cause any confusion for teachers and students who use both ASAM and CM. Moreover, one question allows the student to easily associate the returned feedback to that question. Otherwise, the situation can be confusing with too many questions, especially since a web browser is the medium of communication; or, an alternative user interface may be required.

5.1.6 *Summary*

This section has documented how ASAM is incorporated with the feature of marking programming code, which is made possible by integrating with CM's marking subsystem.

5.2 More on Student Model

The ASAM student model consists of elements of two techniques that are widely used in adaptive hypermedia systems, namely the overlay model and the stereotype model. The overlay model measures the student's knowledge within a given domain whilst the stereotype model classifies individuals according to their background and abilities (Eklund *et al*, 1997).

In the current implementation, students are classified into one of three categories – namely Beginner, Intermediate or Advanced – for their ability level in any one topic. A self-assessment questionnaire may contain any number of topics and questions, as is determined by the lecturer in the questionnaire skeleton. An adaptive questionnaire skeleton may then have dependencies to target its questions at student's abilities in certain topics, depending upon scores – or rather groupings – stored in the student profile. These topic groupings are independent – for instance, a learner may be a Beginner in one topic but Advanced in another.

5.2.1 *Four scenarios*

To have a clearer picture of how the ASAM system adapts the self-assessment questionnaires to each student according to their ability, this section attempts to demonstrate how the following four fictional students progress as they use the system:

- A very bright student (Brian)
- A very weak student (William)
- A student who learns from mistakes (Matthew)
- A student who tends not to learn from mistakes (Nathan)

It has to be highlighted that everyone starts from a common point, which is the Beginner level when they use ASAM for the very first time, and that all questions in each questionnaire are displayed in one go, that is, all at the same time in a single servlet page. These questions are selected randomly from the question bank, with the criteria of matching to the student's ability profile for the corresponding topic.

Imagine there exists an adaptive self-assessment questionnaire which covers three topics with three questions for each topic, and the four fictional students each make three attempts at this questionnaire. Let's say that Brian, a **very bright student** attempts the adaptive self-assessment questionnaire with ASAM, and he manages to answer all questions correctly for each of the three attempts to the questionnaire. Table 5.1 illustrates how Brian's ability profile (as captured by the student model) progresses after each attempt. Since every three consecutive correct answers for each topic[5] would promote a student to the next ability level (Advanced being the highest), it can be seen that Brian is promoted after every attempt of the adaptive questionnaire, and hence, is given questions of greater difficulty during the subsequent attempt.

Topic	**Before Attempt 1**	**After Attempt 1**	**After Attempt 2**	**After Attempt 3**
Fundamentals	Beginner	Intermediate	Advanced	Advanced
Conditionals	Beginner	Intermediate	Advanced	Advanced
Loops	Beginner	Intermediate	Advanced	Advanced

Table 5.1: A summary of how the ability profile of a very bright student, Brian, progresses after three attempts of an adaptive questionnaire.

On the other hand, there is another student, **William, who is very weak** in Java programming and attempts the same questionnaire as Brian and also three

[5] As described in Section 4.3.5.

times. However, he fails to answer any of the questions correctly, and thus stays at the Beginner level, even after the third attempt, as shown in Table 5.2. Consequently, he is presented with questions of the lowest difficulty level every time. However, depending on the size of the question bank, it is unlikely that he would be presented the same set of questions every time, thus allowing him to attempt different questions.

Topic	Before Attempt 1	After Attempt 1	After Attempt 2	After Attempt 3
Fundamentals	Beginner	Beginner	Beginner	Beginner
Conditionals	Beginner	Beginner	Beginner	Beginner
Loops	Beginner	Beginner	Beginner	Beginner

Table 5.2: A summary of how the ability profile of a very weak student, William, progresses after three attempts of an adaptive questionnaire.

When **Matthew, who learns from mistakes,** attempts the same adaptive questionnaire (also for three times), his ability profile may be different as in Table 5.3. At each attempt, he would be presented with questions of varying levels of difficulty for different topics, which match his ability profile as captured by the student model. For instance, at the second attempt, his questions for the “Fundamentals” topic could be of the Intermediate level whereas for “Conditionals” and “Loops”, of the Beginner level.

Topic	Before Attempt 1	After Attempt 1	After Attempt 2	After Attempt 3
Fundamentals	Beginner	Intermediate	Intermediate	Advanced
Conditionals	Beginner	Beginner	Intermediate	Intermediate
Loops	Beginner	Beginner	Intermediate	Advanced

Table 5.3: A summary of how the ability profile of a student, Matthew, who learns from mistakes, progresses after three attempts of an adaptive questionnaire.

Now consider another student, **Nathan who does not learn from mistakes**, likewise having three goes with the same adaptive questionnaire; his ability profile may follow a different pattern, as shown in Table 5.4. For instance, he may have mastered the very basics of Java programming, but has not fully grasped the looping skills and yet does not learn from mistakes for that particular topic. As a result, he was even demoted from the Intermediate level to the Beginner level after the third attempt of the questionnaire since he had answered three consecutive "Loops" questions incorrectly. That would mean Nathan would be given easier questions on "Loops" at the fourth attempt than at the third attempt.

Topic	Before Attempt 1	After Attempt 1	After Attempt 2	After Attempt 3
Fundamentals	Beginner	Intermediate	Advanced	Advanced
Conditionals	Beginner	Beginner	Beginner	Intermediate
Loops	Beginner	Beginner	Intermediate	Beginner

Table 5.4: A summary of how the ability profile of a student, Nathan, who does not learn from mistakes progresses after three attempts of an adaptive questionnaire.

5.2.2 *Summary*

This section has explored a bit more on ASAM's student model. Four scenarios are described to demonstrate how the ability profiles of different calibred students progress as they use ASAM to help in their self-assessment. It also shows how this affects the kind of questions they are presented when they log on to an ASAM adaptive questionnaire.

It is worth noting that these four fictional students by no means represent the whole user population, but just some examples of possible cases. Nonetheless, it is hoped that this helps to illustrate how four different students with different abilities may progress differently as they use the ASAM system.

5.3 Comparison with more conventional adaptive assessment & non-adaptive CM

Having seen how students' ability profiles are being updated in ASAM, one may wonder – in what ways the ASAM system is different from more conventional adaptive assessment and non-adaptive CM. This section attempts to answer this question from a student's perspective, followed by that of a teacher.

5.3.1 *From a student's perspective*

When using **more conventional adaptive assessment systems**[6], a student

[6] More details of such systems can be found in Section 2.3.

would usually be presented with one question at a time. As nothing is known about the student prior to the administration of the first question at every attempt, the algorithm is generally started by selecting a question of average difficulty as the first item, adjusting the difficulty level after every attempted question. A probable drawback is that the users may not know in advance how many questions are involved in a questionnaire, as the assessment is only terminated when a stop condition is met (as listed in Section 2.3). More often than not, such systems are used for summative purposes and often compared to paper-and-pencil tests. These systems tend to offer only fixed response questions, such as multiple choice and multiple response question types, and hence the student could possibly answer questions correctly by guesswork. As a result, this may not necessarily reflect the student's real understanding of the topic. Depending on the systems' setup, students may be able to access the application wherever they are via the Internet, or they may have to use the client software installed on specific computers. This may limit the availability of the adaptive assessment application.

On the other hand, **non-adaptive CM** allows students to have some practice with programming code type of questions within a limited number of attempts permitted for each exercise. The marks returned by CM often contribute, partially or wholly, to students' final grade – it would usually be the score of the last submission for each exercise no matter whether it is higher or lower than those of the previous submissions. As all students are presented with the same questions, individuals may find the questions too easy or difficult. While CM has a web-based version of the client interface, it is not as versatile as the standalone client application which requires installation onto the computers.

This limits where, and potentially when, students can work on their submission.

ASAM aims to bring together the advantages of both more conventional adaptive assessment systems and non-adaptive CM by offering both fixed response and programming questions. Students can practise their programming skills on their own, for as many or few times as they wish, without any worry that the marks will affect their final grades. At the same time, students are presented with questions that match their ability profile as captured by ASAM at the moment they log on. In this way, they are challenged at the appropriate level, and therefore not feeling put off by neither too challenging nor too boring questions. On the other hand, some students may find it too rigid and tedious (as was commented by some students during a survey conducted by the author – see Section 6.3.4 for more information). This may be because all students have to start from the Beginner level when they first use the ASAM system.

It is to be noted that unlike the more conventional adaptive systems, all questions are displayed in one go. As the name of the system suggests, ASAM is mainly used for self-assessing purposes. However, its main purpose is not just to help students self-evaluate and monitor their learning progress, but also to let them have more opportunities to practise as well as getting some form of feedback. The latter is particularly important in large classes, for which the teacher may not be able to provide quality feedback for every student. As ASAM is a web-based application, it permits students to use it anywhere, anytime they wish to. That is, for instance, they can choose to access the application at midnight in the comfort of their home or hostel room.

A summary of this sub-section can be found in Table 5.5.

5.3.2 *From a teacher's perspective*

Having seen how the three kinds of systems can affect students' experience, we now look at these systems from a teacher's perspective.

With **more conventional adaptive assessment systems**, the teacher is likely to find it relatively easy to set fixed response questions. S/he may not have to set a structure for each CAT, but this also means that s/he would not have control over the topics that the questions would cover. Consequently, the more conventional adaptive assessment system may simply select questions from the question bank even though the topics have not been taught.

Given the various components of the **non-adaptive CM** to be considered when setting CM programming exercises, the teacher may find the task (of setting such questions) a challenge, particularly with the current CM interface. For instance, even an expert user may require an hour to set a CM question (and to make sure it works properly) before releasing the question for students to attempt. Nevertheless, it is very likely that s/he would appreciate the existence of the archiving subsystem which can be used to help monitor the students' learning progress just as they attempt questions for various programming concepts. With some analysis on the students' submissions, one may be able to spot where their weaknesses are, or even their programming and/or submission habits. For example, some students may just submit and see what errors would appear, then trouble-shoot and submit again, so as to repeat the process till they are satisfied or use up all their allowed submissions.

More conventional adaptive assessment	Presented with one question at a time, starting off with a question of average difficulty Mainly for summative purposes May not know how many questions are involved; the assessment is terminated when a stopping condition is met Fixed response questions – possible to guess the answer ⇒ may not necessarily reflect student's real understanding/ability Depending on the systems' setup, it may be available on the Internet or on local network
Non-adaptive CM	Programming questions Limited number of attempts permitted for each exercise Mainly for summative purposes Standard questions for all ⇒ can be either too easy or too difficult Though CM has a web version of the client interface, it is not as versatile as the standalone client version
ASAM	Both fixed response and programming questions No pressure – can do as many or few times as they wish, without affecting their formal grades (though brighter students may find it rigid) Everyone starts at the Beginner level when they use ASAM for the first time Display all questions in one go Mainly for self-assessing purposes Good for obtaining quality feedback, even in classes where the teacher-student ratio is high ASAM is fully web-based, and hence, can be accessed anywhere, anytime the student may wish to

Table 5.5: Comparison with more conventional adaptive assessment and non-adaptive CM from a student's perspective

On the other hand, for **ASAM**, with more advantages that come with it, the teacher may have to put in more effort at the beginning to get the system up and running. S/he would have to set more questions than for non-adaptive CM. However, s/he can modify a question to create a few others by varying the degree of difficulty slightly. Again, with the archive of students' submissions, the teacher can examine how well the students have learnt a programming concept. In addition, the teacher has the flexibility of deciding the depth of the hierarchy for questions stored in ASAM, which in turn affects the amount of details of the students' abilities captured in the student model. This is because ASAM captures the profile of students only at the topical level (i.e. the second level of the hierarchy). This is best illustrated with an example. For instance, the teacher may decide to monitor students' progress to the details of the various programming concepts, such as "Conditionals" and "Loops". Hence, s/he would have to assign the module as the top level – in this case, "Java" – as depicted in Figure 5.5(a). Alternatively, if s/he is only interested in the overall progress with respect to the module, the top level can be "Computer Science", as shown in Figure 5.5(b). In this case, only students' ability for "Java" will be captured.

Other than the questions, the teacher has to set a questionnaire skeleton, which specifies whether the self-assessment questionnaire is of a fixed or adaptive format, as well as the number of questions, their question types and topics covered in the questionnaire. This allows the teacher to have a better control over the self-assessment (and feedback) that students receive.

Table 5.6 summarises the differences between the three types of systems as observed by a teacher.

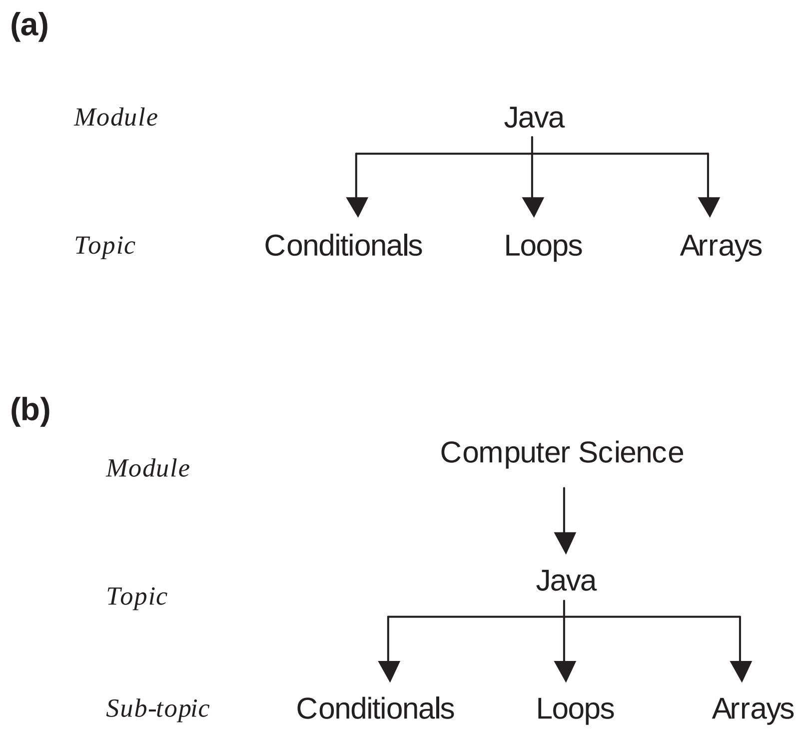


Figure 5.5: Different possible degrees of levels defined in ASAM

5.3.3 *Summary*

We have just taken a look at ASAM from a student's and teacher's perspectives, comparing with more conventional adaptive systems and the non-adaptive CourseMarker system.

More conventional adaptive assessment	Likely to find it easier to set fixed response questions Do not have to set a questionnaire skeleton
Non-adaptive CM	Can be a challenge to set CM questions Can better monitor how well students have learnt a programming concept
ASAM	More questions to set, but can vary the degree of difficulty slightly for "a question" Can better monitor how well students have learned a programming concept Can decide the depth of the hierarchy for questions stored in ASAM Have to set a questionnaire skeleton

Table 5.6: Comparison with more conventional adaptive assessment and non-adaptive CM from a teacher's perspective

5.4 Summary

In this chapter, we have covered the enhancement to the basic ASAM system that allows the marking of programming questions. This was made possible by integrating CourseMarker's marking subsystem with ASAM. We have also looked at the student model further, and through four scenarios, we have examined how ASAM adapts to the use by four fictional students, as well as some comparisons with more conventional adaptive systems and the non-adaptive CourseMarker system, looking from both a student's and teacher's perspectives.

In the next chapter, we shall look at an evaluation study for the ASAM system, and the results gathered from the study, as well as some discussions and implications drawn from the findings.

Chapter 6 –

Evaluation

In the last two chapters, we have seen the rationale behind the developed ASAM system and a detailed description of the system.

The design and development of an application will inevitably involve many technical issues, but it must also meet user and organisational requirements. An effective socio-technical system depends upon the compatibility of the two sub-systems, namely the social and technical sub-systems. Eason (1988) suggests to add a set of human-related criteria to the normal list of technical design criteria rather than to replace it, and thus to ensure that systems are designed and developed that are both technically sound and will fit the organisational context.

Therefore, this chapter will describe an evaluation study (Section 6.2) carried out on the ASAM system, attempting to evaluate the system from several perspectives (Section 6.1). The results gathered from the study are documented too (Section 6.3). This leads onto some discussion on the preliminary use of the system (Section 6.4).

6.1 Objectives

This chapter addresses two main objectives:

- To evaluate the feasibility and usefulness of the ASAM system for the support of students' learning process
- To evaluate the usefulness of the system for supporting the teaching tasks of the teachers

An evaluation study has been conducted in an attempt to address two hypotheses:

- That the ASAM system does help users in their learning. In other words, there is a significant improvement in their marks after using the ASAM system.
- That the ASAM system has correctly profiled the students' ability.

The methodology of the evaluation study is presented in Section 6.2, with the findings of the study in Section 6.3. Implications and discussion are consequently drawn upon the findings, as documented in Section 6.4.

The objectives for the ASAM system have been to offer the students a tool for self-assessment, and the teachers a tool for support in their teaching tasks, in particularly in circumstances where the class size is large. Therefore, this chapter aims to assess the system, to see if these objectives have been achieved.

6.2 Methodology

The ASAM system was introduced to the students for the first time in the academic year 2003/4. It was used by first-year Computer Science students

(N=199) for the first part of one compulsory module "Introduction to Programming" (equivalent to one sixth of the first semester work), at the School of Computer Science and Information Technology of the University of Nottingham. During that first semester, the students were given two self-assessments (of mainly fixed response questions) to attempt.

In addition to the self-assessments, the students were to take part in three MCQ tests[1], as part of the formal assessment. It is to be pointed out that the two self-assessments were only given to the students after the first MCQ test, with one before each of the last two tests and that the first self-assessment is of a fixed format, whereas the second is of an adaptive format. In all cases, once a self-assessment is set, students can attempt it at any time during the course of the module. In Table 6.1, this information on the two self-assessments has been summarised, together with the number of questions and topics covered in each of self-assessments.

Besides the actual use of the ASAM system, students were asked, some weeks after the first semester, to complete a survey on the usage of the ASAM system, the findings of which are documented in Section 6.3.3.

A similar round of evaluation was conducted with the first year students in the academic year 2005/6. They were also asked to complete a survey on the usage (or non-usage) of the ASAM (this time with programming type of questions). The survey results can be found in Section 6.3.4.

[1] The MCQ tests, except for the first one that was carried out on paper, were conducted using the ASAM system.

Self-Assessment (SA)	SA1	SA2
When?	Before MCQ test 2	Before MCQ test 3
Format	Fixed	Adaptive
No. of questions	15	6
Topics covered	Statements/expression Conditionals Loops Methods	Methods Arrays

Table 6.1: A summary on the two self-assessments set for the 2003/4 cohort

6.3 Findings

As students were not obliged to use the ASAM system, not all of them chose to use it. All in all, 89 students had used the system at one point or another during the semester, whereas 85 others did not use it at all[2]. The following sub-sections document the findings obtained from the usage and non-usage.

6.3.1 *Use of the ASAM system*

Table 6.2 shows the number of times students had logged on to the ASAM system for each of the two self-assessments and submitted their answers. It is observed that the students had used the ASAM system for their self-assessment more often before the third MCQ test than before the second MCQ test.

[2] These numbers included only those who had taken all MCQ tests. In other words, those who had only participated in less than three tests are not included in these figures.

	SA1	SA2	Total
Before MCQ Test 2	180	0*	180
Before MCQ Test 3	55	242	297

Table 6.2: The number of times students logged on to the ASAM system for each self-assessment and submitted their answers

6.3.2 *Analysis and comparison of MCQ tests' mean scores*

Table 6.3 lists the mean scores of the two groups of students who took part in all three MCQ tests (with Figure 6.1 showing this in graphical form).

Mean Score	**N**	**MCQ1** ***(std dev)***	**MCQ2** ***(std dev)***	**MCQ3** ***(std dev)***
ASAM users	89	53.72% *(13.60)*	44.18% *(15.67)*	44.33% *(15.63)*
ASAM non-users	85	50.06% *(14.67)*	40.80% *(13.00)*	36.41% *(13.79)*

Table 6.3: Comparison of mean scores (and standard deviations) of students who took part in all three MCQ tests

* Since SA2 was set after the second MCQ test.

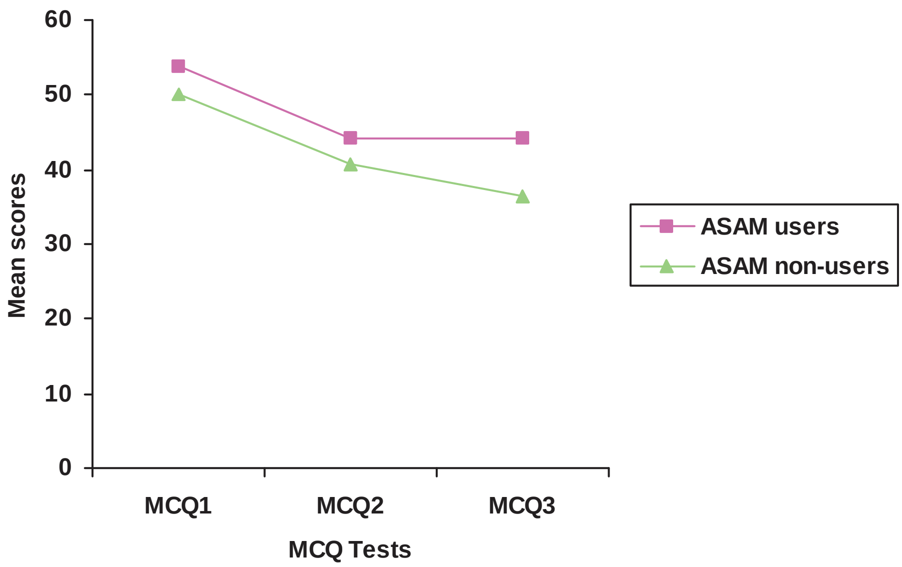


Figure 6.1: Comparison of the mean scores of students who took part in all three MCQ tests

It is observed that the two groups follow a similar trend, that is, a drop in the mean score from MCQ1 to MCQ2, followed by a slight increase or a drop for MCQ3. It is worth noting that there was some slight improvement for the ASAM users in the third MCQ test, whilst the ASAM non-users scored a lower average mark. This suggests that the non-user group struggled to keep up with their learning, such that their average scores for the third MCQ test had dropped further.

Looking at Figure 6.1, the gap in the mean scores between the two groups has widened after the third MCQ test, as compared to the difference at the start of the comparison. From this, it appears that ASAM may have helped its users, at least, maintain their level of ability in programming skills.

An ANOVA mixed design (of one repeated measure and one unrelated factor) analysis was carried out on the data. The Multivariate tests showed significant differences for the scores (Wilks' Λ=0.592, $F(2,171)=58.868$, $p<0.001$). However, it cannot be concluded that there is any significant differences in the interaction between the scores and the ASAM usage/non-usage (Wilks' Λ=0.975, $F(2,171)=2.197$, $p>0.001$).

It implies that the null hypothesis of the ASAM system having no effect on students' learning progress cannot be rejected. Hence, while we cannot conclude that the use of the ASAM system has some definite effect on the students' scores, there is no evidence that it is detrimental to student learning, and the raw data indicates some improvement in students' scores.

Besides the mean scores for the three MCQ tests, we have the overall scores for the first part of the module for the two groups, as shown in Table 6.4. The overall score (100%) for the formal assessment consisted of the on-going coursework (10%), two big exercises (40%), the three MCQ tests (30%) and two real-time examinations (20%) – all of these were carried out using CourseMarker.

Mean Score	**N**	**Overall Score *(std dev)***
ASAM users	89	64.87% *(16.53)*
ASAM non-users	85	58.72% *(17.17)*

Table 6.4: Comparison of mean overall scores (and standard deviations) of students who took part in all three MCQ tests

An attempt to find out whether there was a relationship between students' mean overall score and their profile that was captured by the ASAM system was carried out by calculating the Pearson correlation coefficients (r). Table 6.5 shows the values of the Pearson correlation coefficients calculated. Since the last two MCQ tests were conducted using the ASAM system, all students have been profiled in the student model of the system. Whilst the correlation coefficient for the ASAM non-users ($r(83)=0.293$, $p<0.01$) is lower than that for the ASAM users($r(87)=0.550$, $p<0.01$), it is observed that the correlation coefficient of $r=0.468$ ($p<0.01$) is comparatively high for the entire cohort. This shows that there is a significantly moderate degree of linear relationship between the student profile captured in the ASAM student model and the mean overall scores of students. Thus, we can say that there is some evidence supporting the hypothesis that the ASAM system has correctly profiled students' ability.

	Pearson Correlation Coefficient (r)
ASAM users	0.550
ASAM non-users	0.293
All 2003/4 students who took part in all three MCQ tests	0.468

Table 6.5: Pearson correlation coefficient between the mean overall score and the students' profile captured by ASAM

6.3.3 *Survey results I*

A short survey[3] was also conducted some weeks after the first semester of 2003/4. The aim of the survey was to find out how students felt about the use of the ASAM. In total, 28 students responded. Among the respondents, 21

[3] A copy of the survey 2003 survey can be found in Appendix E.

had used the ASAM system for self-assessment, whereas 7 had not. 16 of the "user" respondents indicated that they found the system useful, whereas the five others had expressed otherwise. When asked what type of feedback they preferred, all 21 of the "user" respondents chose both marks and comments. A summary of these statistics is shown in Table 6.6.

	Response	**N**
Used ASAM?	Yes	21
	No	7
Found ASAM useful?	Yes	16
	No	5
Preferred feedback format	Marks only	0
	Comments only	0
	Both marks and comments	21
	None	0

Table 6.6: Some results gathered from the survey conducted in 2003/4

The students were also asked an open-ended question for their opinions on why the system was useful. The general comments included:

- It helps to indicate their progress during the course of the module.
- They can obtain immediate feedback, upon clicking on the "Submit" button.
- It helps to identify the areas they need to improve upon.

6.3.4 *Survey results II*

Another short survey[4] was conducted shortly after the availability of the ASAM system in the first semester of 2005/6. An adaptive self-assessment questionnaire was set, which consisted of a single programming question. Nonetheless, it could draw from the question bank of a range of programming questions, including some debugging type of questions. The aim of the survey was to find out how students felt about the use of the ASAM (or, why they chose not to use the system) to help them practise their programming skills. In total, 19 students responded. Among the respondents, 9 had used the ASAM system for self-assessment, whereas 10 had not. The programming background (prior to taking the module) of the respondents is summarised in Figure 6.2.

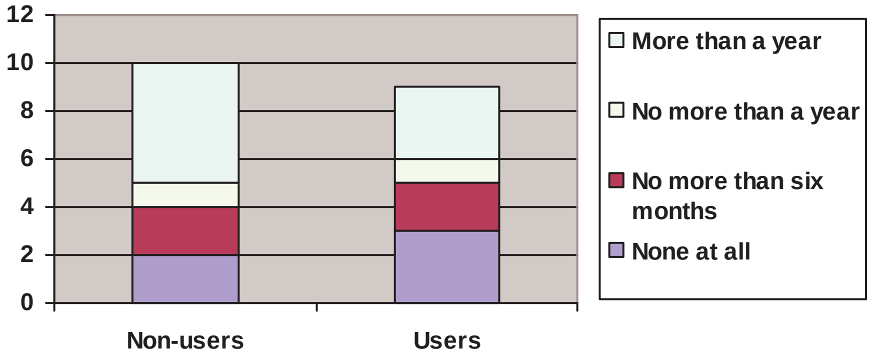


Figure 6.2: Programming experience (prior to taking the G51PRG module) of survey respondents (2005/6)

[4] A copy of the survey 2005 questionnaire can be found in Appendix F.

The “user” respondents were asked whether they thought the feature of generating different programming questions (of varying levels of difficulty) useful. The mean for the Likert-scale item (with 1 being “not useful at all” to 5 “very useful”) was 4, with a standard deviation of 1.414. The responses for the survey question on whether using the system encouraged them to want to do more programming was mixed (4 said “yes”, 3 “no” and 2 “maybe”) whilst all but one “user” respondents thought it was a good practice to self-assess using the system.

On the other hand, only 2 (out of 9) of the “user” respondents said that using the system did help them, in some ways, understand programming better. Again, there was a mixed response for the question on whether the users thought the system would reinforce their learning of the programming skills (with 3 votes on each option, namely “yes”, “no” and “don’t know”). When asked whether they found the feedback returned by the ASAM system useful, there was a fairly even split between “yes” and “no” votes. Last but not least, all but one “user” respondents said they would at least consider recommending the system to their friends. A summary of these statistics is shown in Table 6.7 and Table 6.8.

	Response	N
Did you use the ASAM system to help you in your revision?	Yes	9
	No	10
Do you think the feature of generating different programming questions (of varying levels of difficulty) useful?	1	1
	2	0
	3	2
	4	1
	5	5
Did using the system help you understand programming better?	Yes, a lot	0
	Yes, a little	2
	No, not really	6
	No, definitely not	1
Does using the system encourage you to want to do more programming practices on your own?	Yes	4
	No	3
	Maybe	2
Do you think it is a good practice to self-assess using the system?	Yes	8
	No	1
	Don’t know	0
Do you think using the system will reinforce your learning of the programming skills?	Yes	3
	No	3
	Don’t know	3

Table 6.7: Some results gathered from the survey conducted in 2005/6 (I)

	Response	N
Did you find the feedback returned by the ASAM system useful?	Yes, a lot	1
	Yes, a little	4
	No, not really	3
	No, definitely not	1
Would you recommend the ASAM system to your friends?	Yes	4
	No	1
	Maybe	4

Table 6.8: Some results gathered from the survey conducted in 2005/6 (II)

The survey also asked the non-users why they did not choose to use the ASAM system, and the reasons given included:

- *"I don't find it that helpful – too much like CourseMarker."*
- *"I thought that the lecture notes plus my own notes and the other literature that I have would be enough."*
- *"It's due to time constraints."*
- *"I found it not useful – it requires both to think of logical errors and syntax at the same time which is in my opinion not what I would like to do. When I write my own code, I try avoiding syntax errors and even if I miss any the compiler tells."*
- *"I need to log on and log off all the time to move on and that takes time."*
- *"I didn't require revision for material covered so far in the course."*

When all survey respondents were asked for general comments, the responses included:

- *"There is no opportunity to debug your effort, or even look over where the mistakes specifically were. Once you have submitted the first time, there is no going back, as it were."*[5]

- *"From my point of view, ASAM is a really useful tool for self-assessment for those people who would like to get involved more into studying programming as well as revising their current knowledge. Moreover, it could replace CourseMarker for those who would prefer to have more practice."*

- *"It would really help if the user could answer more than one questions on each log-on."*

- *"I did have a look and thought that the system being web-based was restrictive: the HTML <textarea> is not flexible enough to program in. No syntax highlighting, auto-indenting, "tab" is very difficult to achieve, etc."*

- *"Although I didn't use the system to revise from, I can see how the system would be useful to practise coding with."*

[5] The writer suspects that the respondent was making comparison with CourseMarker, which allows users to submit multiple times.

- *"I don't like looking at someone's code for syntax errors because I find a compiler to be a useful tool for this."*
- *"Although I didn't use the system to revise from, I can see how the system would be useful to practise coding with."*

6.3.5 *Summary*

In this section, we have just seen the findings collected from the usage (and non-usage) of the ASAM system, with some comparison with the previous cohort, as well as the data gathered from two surveys conducted in 2003/4 and 2005/6. Whilst we cannot conclude that the use of the ASAM system has some definite effect on the students' scores, there is some evidence supporting a moderate positive relationship between students' mean overall score and their profile captured in the ASAM system. The results gathered from both surveys suggest that the "users" respondents had positive feedback though the second survey had a more mixed response.

6.4 Implications and discussion

Having viewed the findings gathered from the usage and the subsequent survey conducted, some implications and discussion can be drawn upon a few issues. These can be categorised into socio-technical, pedagogical and technical issues.

6.4.1 *Socio-technical issues*

As mentioned earlier in the chapter, a successful implementation of a new system is dependent on the handling of the socio-technical issues involved.

It was observed during the first round of evaluation that there was a fairly high uptake at the beginning of the implementation. Although it was the first time that students were introduced to the ASAM system, there was not much hesitation in the uptake of the system at the beginning – there were almost 90 submissions just on the first two days of implementation. This is contrary to what usually happens in many other implementations of new systems, which encounter a lot of resistance from the intended users (Eason, 1988). This may be due to the initial interest students tend to have in any tool that can help in their learning. Also, it could have helped that the availability of the new system was made publicly known at the official website for the module.

Nevertheless, this excitement from the students lasted for only two days, after which there were only a handful of submissions per day, until the day before the second MCQ test. This is in line with the typical last-minute preparation by students, which is also suggested in the second round of user trial.

Upon seeing the benefits from the first self-assessment after the MCQ2 test, students were motivated to use the system more. According to the system's log, there was a surge in usage in the last two days before the last MCQ test (which was held in the afternoon), particularly in the last five hours – 179 submissions during that period. This shows students' appreciation of such applications for their revision. This is especially so since they can gain access to the web-based application anytime, anywhere, as compared to

CourseMarker, which they can only gain access to when the computer laboratories are open (unless they install the client on their own machine or use the web interface which is inferior to the client).

Also, Table 6.3 suggests that this frequent use of the ASAM system had some links with the improvement in the third MCQ test results.

During the second round of evaluation, there has been some comment made by some students, which highlights that the brighter students might find the system too rigid and/or tedious. One reason could be that unlike the standalone CM client application, ASAM does not facilitate the compiling of programming code via the browser before submission. The writer would then suggest the students to compile their code before submission, just as they would do for any paper submission.

Alternatively, it could be due to the design that all students have to start from the Beginner level when they first use the ASAM system. This can potentially be resolved by allowing the teacher or possibly the students themselves to set their starting ability level, after which the student model could promote or demote them accordingly upon their subsequent attempts.

6.4.2 *Pedagogical issues*

A self-assessment system is of no use unless it adds value to the teaching and learning task of teachers and students. Thus, this sub-section is dedicated to the pedagogical aspect of the socio-technical system.

From the findings, we can conclude that self-assessment is generally beneficial to students, may it be fixed or adaptive, as it allows students to evaluate their own ability informally at their own pace, thus motivating them to learn better (Yong and Higgins, 2004).

Note that for the user trial in 2003/4, the self-assessment set before the MCQ2 test was of fixed mode, that is, all students were given the same set of questions at all times. However, the second self-assessment, which was set before the last MCQ test, was adaptive, that is, questions were chosen dynamically from the question bank, according to individual students' ability. From this and the improvement rates as seen in Section 6.3.2, we can conclude that adaptive self-assessment adds value to their learning as they are presented with questions according to their ability level, instead of those that are either too easy or too difficult (Yong and Higgins, 2004).

Whilst the application adds value to the students' learning, it is worth noting that the teacher has a great role to play – he/she has to use his/her own discretion to assign the appropriate level of difficulty to every question. If some guideline is ever needed, the teacher could consult Table 4.1. Therefore, the success of the application in selecting questions of appropriate level is dependent on the teacher's judgement in assigning the appropriate level of difficulty to each question. Of course, the teacher can gain insights about the questions during the usage and adapt the questions themselves for later years.

When SA2 was introduced during the first round of user trial, a small experiment was carried out to see students' reactions to the type of feedback they received. The feedback format for the self-assessments was changed from purely marks, to just comments on student ability and what they could do next (later to both marks and comments). When the marks-only feedback was replaced by comments, there were some instant requests from some students, to re-introduce the marks feedback. Once that was added to the comments – instead of replacing them – students were more content. This contentment can be further proven by the results of the first survey, whereby all ASAM "user" respondents had indicated their preference of both marks and comments, as compared to either of the two feedback types, or even none.

It must be highlighted that at any one point, no feedback to individual questions (except for programming questions) is given. That is to say, there is no indication whether individual questions have been answered correctly. The reason for this decision is in hopes that students will be inspired to identify for themselves which questions they have answered correctly or are unsure about, rather than to spoon-feed them, or cause them to focus on only certain sub-topics. The aim is to encourage students to be confident and versatile in each sub-topic.

As two of the MCQ tests were also conducted using the ASAM system in 2003/4, some useful statistics could also be obtained from the system – these could otherwise be tedious to obtain (Kleeman, 2000; Aston *et al*, 2004). A breakdown on the MCQ test results could subsequently be obtained, in terms of topics. This aided in identifying the weaker students, as well as the weaker topics of the entire cohort, and in turn, in preparing revision lectures.

As an illustration, Table 6.9 shows a breakdown of the mean score for each sub-topic covered in the second MCQ test. From this, it can be concluded that students were fairly confident in the Fundamentals of Java Programming. However, they did need more assistance in the other areas, particularly Methods. Thus, the lecturer could choose to emphasis on Methods, for instance, during the forth-coming revision lectures.

Sub-topic	Number of questions	Mean score
Fundamentals	2	61.9
Expressions	2	45.6
Loops	8	44.8
Conditionals	5	46.6
Methods	8	33.8

Table 6.9: A breakdown of mean score for each sub-topic covered in the second 2003/4 MCQ test

Whilst most CAT systems generate questions one at a time, the ASAM system lists all questions in each self-assessment all in one go. This allows students to have a good idea of how many questions they will need to attempt in each self-assessment, and thus an estimation of how much time they have to spend for each question. Some may argue that this could lose the benefit of being adaptive on-the-fly. Alternatively, to make a better use of the adaptive feature of ASAM, teachers can choose to set shorter self-assessment questionnaires, for instance, six questions in the case of SA2 for the first evaluation study, or one question in the second user trial.

In the second survey, only a couple of students reported that using the ASAM system – when it was then able to mark programming questions – helped them understand programming better. The writer reckons that the reason for such low percentage could be that the students had not used the adaptive application enough to build up an appropriate ability profile and hence found the questions too easy or too similar to those in CM, which they had been using for weekly coursework. Interestingly, the two students who gave positive response had some programming experience prior to taking the compulsory module for all first-years (one with no more than six months of experience and another more than a year of experience). This may suggest – though not statistically significant – that students do find ASAM potentially useful in helping them learn how to program. For the least, the majority of those who used ASAM thought that the feature of generating different programming questions (of varying levels of difficulty) was useful, and some students considered ASAM as a possible replacement of CM. It seems that such a tool is highly regarded as a useful learning tool by students, especially the weaker ones in large classes where quality feedback can be scarce.

In a research by Vizcaíro and his colleagues (Vizcaíro *et al*, 2000), it was noted that one reason teachers think why students do not write good programs is that students do not think about the possible solutions to the problems and which of them is best. The writer thinks that this argument is, in some ways, supported by some student comments for the second survey. As one student remarked, "There is no opportunity to debug your effort… Once you have submitted the first time, there is no going back…" On the contrary, the writer regards debugging type of questions can test, or even train, the students on this – hopefully challenging them to ponder further before "declaring" that their code is submissible. Such questions can be particularly useful for weaker students. Also, it can challenge even an experienced programmer with regard to their programming skills, which can be beneficial in the long run.

With the inclusion of programming questions in ASAM, it is now not just an application for objective testing, which may not accurately assess students' ability since questions could possibly be attempted by guesswork. This inclusion addresses one of the limitations highlighted by Baker and Lilley (2006). One of their staff feedback expressed concerns related the use of objective testing as the only method with their approach (i.e. automating feedback for a computer-aided test).

In one of their recent papers, Ashton and Thomas (2006) discuss a number of issues that, if addressed, could narrow the gap between teaching, learning and assessment:

- The need to use the same software tools through the education process;
- The need to assess what the student has learnt rather than what is easy to assess;
- To consider the possibility of assessing qualitatively not quantitatively, and
- To consider the possibility of assessing the application of knowledge, rather than its acquisition.

The paper looked at how learning, teaching and assessment can become misaligned resulting in an education system that does not support student learning. In the writer's opinion, all of the issues mentioned above can be addressed with the use of ASAM. Of course, input from teachers cannot be underestimated.

Much research argue that information that is assessed but never put to use during the learning process may be difficult to retrieve and use when the need arises in the real world (Schank and Kass, 1996). Software such as CM and ASAM encourage students to practise what they have learned during the course of the module. Even though they may have difficulty in achieving their goal (e.g. solving a problem or attaining a higher level of achievement or ability), it is hoped that they would be encouraged/prompted to search for further assistance, say from friends, tutors and/or other people (Schank and Kass, 1996). The ultimate aim is to inspire students to practise more (Draaijier and Boter, 2005), may it be with ASAM or on their own. For those who may find ASAM too boring or rigid, they could still try some other mode of self-assessment.

6.4.3 *Technical issues*

Having covered the socio-technical and pedagogical issues, we should not ignore the technical aspect of the implementation.

Generally, the system worked fine and performed as expected. However, when the ASAM system was used for the last two MCQ tests, there were some problems with the login speed and reliability of the network during the initial and final stages of the second MCQ test. This occurred when the students were trying to log on all at the same time. Also, six students had encountered problems while attempting to submit their answers. After some upgrades to the ASAM system, the situation had improved in the last MCQ test. At other times, there had not been any problem using the system.

Actually, this problem is not unique to the implementation of the ASAM system – Mackenzie (2000) had encountered this too when he and his colleagues were attempting to use the TRAIDS Assessments system on a university-wide project. Nonetheless, as the ASAM system is aimed to run across the network for summative assessments too (for the benefits of the statistics to be obtained later for analysis), it acts as an incentive to resolve the casual problems in the network perhaps, more rapidly than would otherwise have been the case.

6.4.4 *Summary*

In this section, some implications and discussion that can be drawn from the findings of the evaluation studies are presented, and these cover three types of issues, namely socio-technical, pedagogical and technical issues.

6.5 Summary

Whilst there were some setbacks during the evaluation studies, there had been many more positive results from the usage:

- The web-based system allows students to self-assess anywhere, anytime.
- The self-assessments generated by ASAM add value to students' learning process, especially with the adaptive self-assessments.

- Students think it is a good practice to self-assess using the ASAM system, so much so that they would at least consider recommending the system to their friends.

- The immediate feedback that ASAM returns to the students can help them identify their learning progress, and consequently, weaker areas.

- The reports generated by the ASAM system can help teachers identify various issues, including the weaker areas of individual students and of the entire cohort, and the self-assessing patterns of each student.

It can be concluded that the objectives for developing the ASAM system (as set out in Section 4.1.2) have been met. Nonetheless, it is suggested that more evaluations should be conducted in the future, as results from future years will give statistics that enable more rigorous assessment of the system.

Chapter 7 –

Conclusions

A novel system has been conceived, designed and built that allows students to self-assess at a suitable level of difficulty – particularly, it is one with the capability to mark adaptive programming questions. While self-assessment enables students to reflect on their own progress and performance, questions that are too easy or too challenging bring little benefit. Therefore, self-assessing questions of an appropriate level of difficulty are required. The system also empowers the teaching staff with statistics, which can otherwise be difficult to obtain, to help them monitor students' progress.

The ASAM system developed during this research demonstrates how adaptive self-assessments can help students monitor their own progress – in this case, more specifically on their programming skills – with help from the immediate feedback given on each topic upon submission. The system is appropriate for all educational levels, but is particularly useful for weaker learners and those studying in institutions where class size is too big for the teaching staff to attend to the needs of each and every student as carefully as they might like.

Specifically, feedback from ASAM would benefit very low performing students who are more likely to need help most. Barker and Lilley (2006) point out that feedback system is valued more highly in the context of formative, rather than summative, assessment. Hence it is important to have applications like ASAM to provide feedback at an early stage of one's learning process, rather than when it may be too late at summative assessment, such as at examinations.

In addition, with the reports generated from the student profiles, the lecturer can identify the weak areas of the students (either of individuals or of the entire cohort) so that they can consider re-structuring their lectures or tutorials to address these areas.

The results of this research have illustrated the usefulness of self-assessment in the learning process. The students that used ASAM found that it added value to their learning. This was particularly true with the adaptive self-assessment that catered to individual students' abilities. The immediate feedback that ASAM returned to the students upon submission helped them identify their learning progress for each topic, and consequently, weaker areas. It also suggested what students could do next. At the same time, the reports generated by the ASAM system helped the teaching staff identify various issues, including the weaker areas of individual students and of the entire cohort, and the self-assessing patterns of each student. These were used to pinpoint which areas they could focus on during their lectures. For instance, when some analysis were done on students' submitted programming code – as stored in the archiving subsystem of the CouseMarker application – students' programming and submission habits could be observed. From this, undesirable habits could be identified and addressed if necessary.

The following sections highlight some of the important issues raised during this thesis together with areas that are considered innovative and some potential future work.

7.1 The gap in the educational technologies

Chapter 2 has documented the existing computerised tools for educational purposes. These include intelligent tutoring systems that are able to provide learning materials matching students' level of knowledge. While similar mechanisms are used for assessing students, they are mainly for summative assessments.

Chapter 3 has highlighted the importance of self-assessing in the learning process. However, it is not highly regarded by many students and lecturers. Therefore, there is a need for a self-assessing system, which can adapt exercises according to students' abilities, to help put self-assessment back in perspective. Specifically, there is no adaptive system that can facilitate the marking of programming questions.

7.1.1 *Adaptive self-assessing with programming questions*

The main contribution of this work is an adaptive self-assessment system, with some emphasis on use for computer science education as it permits the marking of programming code. With the ASAM system, one can set composite questionnaires for programming students, that is, mixed modal questionnaires that may include any combination of fixed response questions and programming questions. Nonetheless, the application is not limited for use in this discipline – any other discipline can use the features/question types they deem appropriate. Likewise, statistics/reports generated by the system is beneficial to any discipline.

As mentioned earlier, a particular feature of this adaptive self-assessment system is that it can mark programming code – this was the first of such adaptive applications. It was made possible by integrating with the marking subsystem of the CM application. This allows students to have as few or as many practices as they wish on their programming skills at an appropriate level while not having to worry that it would affect their final grade and at the same time, receiving some immediate "formal" feedback for their submission. This is particularly helpful to students in large classes, for not all of them might be able to get personalised attention from their teacher as much as both parties would like to. With the archive of students' submissions, especially for the programming code, the teacher can opt to analyse exactly where students go wrong, and use this information to assist in their planning for revision lectures.

7.1.2 *Student profiling*

Another interesting input is a less sophisticated and feasible approach to updating student profiles. This approach is based on the stereotype model and is designed such that three consecutive correct (or incorrect) answers submitted by students to each topic will promote (or demote) their level of ability (from Beginner at the lowest to Advanced at the topmost) for that topic. It is an attempt to implement Abdullah and Cooley's theory (2000), which argues that two pieces of positive evidence or two pieces of negative evidence are required to allow the tutor to reach a firm judgement. It is then suggested that at most three questions of specified level of difficulty requiring a particular skill are required. The repetition (of the extra question) helps eradicate uncertainty due to the possibility of guesswork or careless slips. This approach is comparatively less sophisticated than the complex Bayesian brief network that has been used by many researchers in the adaptive hypermedia field.

7.2 Meeting the objectives

This section revisits the objectives set out in the fourth chapter and demonstrates that the key objective has been accomplished. This was to design and build an adaptive self-assessment system that adds value to the teaching and learning task of teachers and students, in particular with the feature of being able to mark programming questions (a form of free response questions, which can assess better than objective testing), and in classes where the teacher-student ratio is high.

For this research, an application has been developed that generates personalised self-assessments containing questions that match individual students' ability. In other words, the questions are aimed to be not too challenging nor too easy for individual students, for such questions tend not to benefit the students.

Chapter 5 has elaborated more on the feature of marking programming questions and the student model. An comparison has been made with the more conventional CAT systems and the non-adaptive CM system.

Chapter 6 has reported on how students appreciated the use of the adaptive system. From the result of the first evaluation study, whilst we could not conclude that the use of the ASAM system had some definite effect on the students' scores, there was some evidence supporting a moderate positive relationship between students' mean overall score and their profile captured in the ASAM system. Even after the user trials had ended, there were requests from students, wanting to continue with the self-assessing. Though the responses for the second survey were not many, the majority of the 'user' respondents could see some potential benefits that ASAM could give them in the long run – one even suggested the possibility of replacing CM with ASAM. With the statistics obtained from the student profile stored in the student model, the teaching staff could identify students' weaknesses, thus enabling them to re-adjust their lecture/tutorial structures to address these problematic areas. Thus, it can be concluded that the use of adaptive self-assessment systems, such as ASAM, can benefit both students and teachers in their learning and teaching task.

It could be worth noting that whilst both CM and the basic ASAM system – without the capability to mark programming questions – can run perfectly well independently on both Windows and Unix platforms, for some obscure reasons, a Java RMI connection could not be established between CM and ASAM on the Windows platform. Therefore, for the time being, the enhanced ASAM system can only be operated on the Unix platform.

7.3 Future work

Whilst amongst the main contributions of this work has been the creation of an adaptive self-assessment system for programming students, there are numerous possible directions that can be researched to continue this work. This section highlights some that could be achieved in the more immediate future.

7.3.1 *Modifying approach to upgrade students' profile*

In Chapter 4, it was mentioned that the strategy adopted by ASAM to promote (or demote) student ability is three consecutive right (or wrong) answers to promote (or demote). This strategy was initially adopted when only fixed response questions were incorporated. Now that ASAM can assess students with programming questions, a possibility arises – would a 100% correct submission (for the programming questions) deserve an immediate promotion, and likewise for a 100% wrong submission? This is suggested because the use of programming questions – one of the free response question types – can assess students better than fixed response questions (Gibbs, Habeshaw and Habeshaw, 1988; Farthing and McPhee, 1999), and therefore, the marks for each question can possibly tell us more about the student's programming skill. It could be that a score of 80%-100% may justify an immediate promotion, and 0%-30% an immediate demotion; otherwise, the original strategy can be retained. While this is still an idea, once tested, it can potentially pave the way for research on other free response question types, and thus help to profile students better.

7.3.2 *Adding more question types*

Currently, there are only a few question types available, including simple Java programming and some fixed response types. Further inclusions, such as GUI programming[1], diagrammatic and the remaining fixed response types[2], can be made to the marking facility of the ASAM system. Whilst CourseMarker is capable of marking diagrammatic questions, the complex Java application would require some effort to make it Web-compatible. Thus, it may have to be set as a separate project. Fixed response question types, such as hotspot, drag-and-drop, matching and/or even permutational multiple choice question (see Section 2.2.1.2), can be included too, so that teachers can have a greater variety of question types (and hence can better assess the students).

7.3.3 *Incorporating lesson plans*

A possible additional feature to the ASAM system that educators will appreciate is the automated progression of a test or self-assessment questionnaire from one topic to another – an idea drawn from intelligent tutoring systems. In tackling this task, the problem of the justification of any structuring of a syllabus may have to be ignored. Though there may be, from some given point of view, an optimal way of structuring a syllabus, the author agrees with Abdullah and Cooley (2000) that it is a subjective matter to be

---

[1] The newer version of CM (i.e. version 4) can mark students' GUI (Graphical User Interface) programming code, as part of a recently completed PhD research of a member of the author's research group.

[2] These may include fill-in-the-blank, hotspot and drag-and-drop types of questions.

determined by an expert teacher. As such, there is a need for a facility that allows different educators to define their own lesson plans, even if they teach the same subject.

At the moment, when using ASAM, the teacher or course developer is responsible for specifying the pre-requisite (if any) for each question in the self-assessment, and this is indicated in the questionnaire skeleton. This feature can be further enhanced by allowing the teacher to specify a lesson plan in advance, so that subsequent self-assessments attempted by students will "consult" this lesson plan for any pre-requisite. This lesson plan can form a basis for checking the pre-requisites for questions of all topics, instead of having the teacher specify the pre-requisites for each and every question. In such a case, it is only until students have demonstrated sufficient evidence for attaining a particular level of knowledge on a certain topic, that they will be given exercises of the next topic.

If it were to adopt WHURLE's concept (Brailsford *et al*, 2001, 2002), the ASAM system will have a lesson plan at the topmost level (as shown in Figure 7.1), followed by the level of exercise(s) for each lesson. At the bottommost are the individual questions.

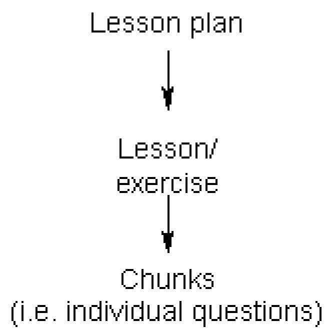


Figure 7.1: Concept of self-assessments in ASAM (similar to that of WHURLE)

A possible scenario for a Java programming teaching module is: before an exercise for the topic “Inheritance” is generated on-the-fly for a student to attempt, it has to be checked that the particular student has attempted the exercise on “Classes and Objects” and has obtained satisfactory marks. Likewise, a course developer may have a lesson plan in which “Conditionals” is taught before “Loops”. In this case, each student must obtain a reasonable score for the topic “Conditionals”, before the exercise on “Loops” will be generated.

However, if it is envisioned to use a common question repository shared among different institutions or departments, a single lesson plan may not be ideal for use by all the different groups. The author wishes to stress that there is no definite good or bad lesson plans – just preferences of individual teaching staff. An alternative can be: not to associate any pre-requisite to questions or sub-topics, that is, just having plain questions for various topics and no links between topics while storing lesson plans separately.

7.3.4 *Conforming to a common standard for assessment content*

At the time of designing and developing the ASAM system, there was no "stable" common standard that could easily integrate programming questions[3]. As a result, a standalone XML schema for the question item was used. Therefore, the current state of the question items in the question bank is such that only users of the ASAM system can make use of the questions. If there were a need to add any questions that may be available from other sources, the educator would have to input these questions manually into the system. This is not very conducive, especially if there are tens or even hundreds of items in question. Therefore, it is worth replacing the XML schema of ASAM with that of a universal standard, so that it can permit the sharing of question items in the question bank with other universities without limiting the use to a single application (such as ASAM) or any commercial tool (such as QuestionMark Perception). This conformation is also in line with the objectives of the

[3] A PhD research was originally initiated by a member of the author's research group to work on one of such common standard that can accommodate free response questions, including programming questions. However, it is now abandoned due to some personal reasons.

existing national question bank projects, such as the e3an[4], LTSN-ICS[5] QUILS[6] and COLA[7] projects.

Since towards the end of last century, there have been attempts from all over the world to standardise the definition of metadata for assessment content. Examples of resultants are the IMS QTI[8] and IEEE LOM[9] standards. Although many attempts have been made to apply to these standards, they are not all success stories (Sclater, Low and Barr, 2002). Despite this, over the recent years, there are signs of great potential for success (Lay, 2004; Sclater and MacDonald, 2004). With the latest version (which is version 2) of the QTI standard and the existence of the many question bank projects, the sharing objective looks increasingly promising. To facilitate the sharing of questions among educators in UK universities (and possibly across the world), it is necessary to conform the metadata standard to a universal one.

---

[4] Electrical and Electronic Engineering Assessment Network. Refer to http://www.e3an.ac.uk for more information.

[5] Learning and Teaching Support Network – Centre for Information and Computer Sciences. Refer to http://www.ics.ltsn.ac.uk for more information.

[6] QUestions for Information and Library Science. Refer to http://www.ics.ltsn.ac.uk/ILS/quils.html for more information on the project.

[7] COLEG OnLine Assessment. Refer to http://www.cetis.ac.uk/profiles/uklomcore/toia_cola_metadata_v1p2.doc for more information on the project.

[8] Question and Test Interoperability. Refer to http://www.imsglobal.org/question/index.cfm for more information on the specification.

[9] Learning Object Metadata. Refer to http://ltsc.ieee.org/wg12/index.html for more information on the standard.

7.3.5 *Interoperability*

Assessment is just one part of the learning process, which also includes acquisition of knowledge, etc. Therefore, it would be good to use ASAM alongside other teaching and learning tools, so as to provide a better judgement of students' learning achievement. Together with other learning tools, the resultant would be a more comprehensive resource for learning, and hence we can achieve better teaching and learning outcomes. Ashton and Thomas (2006) would support development of such tool.

With the many ITSs developed by researchers all over the world – such as KnowledgeTree (see Section 2.1.2.3) and WHURLE (see Secion 2.1.2.4) – there is a potential for such systems to share the student profiles with the ASAM system. This can prevent too many wheels from being re-invented. Also, with the individual adaptive systems operating separately, the potential of learning support is not maximised. The knowledge about the students captured in ITSs is usually about the perceived knowledge of the students, but not their actual knowledge level. If there is also some tool (such as ASAM) to assess the students, the lecturers can be more certain about where the students stand in terms of gaining knowledge in the subject area. Likewise, if there were just an adaptive self-assessment system, learning support for the student is limited, in terms of feedback. An ITS would complement this.

This can be accomplished through what is known as collaborative student models (Barker *et al*, 2002; Brusilovsky, 1996). Figure 7.2 depicts a possible set-up of the collaboration. As such, a standard format for the student model is desirable. This can be achieved through some collaboration work with other research groups, such as one of Minerva[10]'s projects, ADAPT[11].

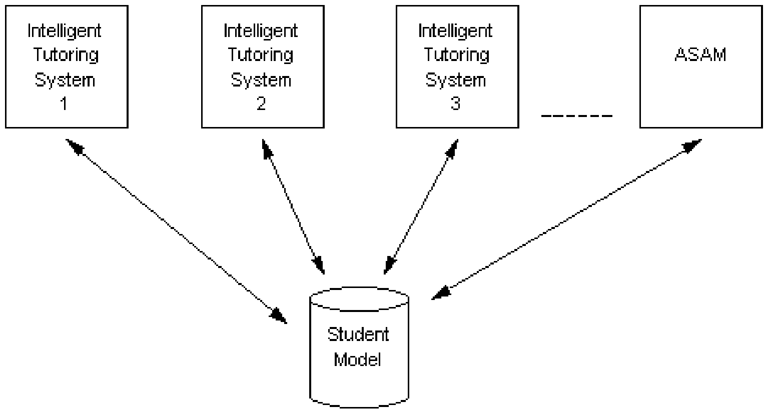


Figure 7.2: ASAM sharing of the student model with other intelligent tutoring systems.

A potential resultant is such that data about the students (e.g. their marks from ASAM and learning progress from a ITS) are passed from various applications to the student model, to assist in the formulation of the student profile. This is in hope that the various activity applications can then make use of the information, in a useful manner. Ideally, each activity application is a blackbox in this big system, that is, it is not of concern to the various

[10] The Minerva Action seeks to promote European co-operation in the field of Information and Communication Technology (ICT) and Open and Distance Learning (ODL) in education. Refer to http://europa.eu.int/comm/education/programmes/socrates/minerva/ind1a_en.html for more information

[11] This is a European project that aims to establish a European platform of standards (guidelines, techniques and tools) for user modelling-based adaptability and adaptation, in the sense of the new paradigm of intelligent human-computer interaction, based on the new generation of ODL tools, towards individualisation of the learning process. Refer to http://wwwis.win.tue.nl/~acristea/HTML/Minerva/ for more information.

developers to know how the other activity applications in the system are to utilise the data.

Therefore, together with the conforming of the XSD to a common standard for assessment content (as suggested in Section 0), the student model can pave the way for making ASAM interoperable with other computerised educational tools, such as ILEs and discussion forums (Draaijer and Boter, 2005). Potentially, with careful selection of other tools, the composite teaching/learning system can be a useful tool for both students and teaching staff, of various subjects/disciplines. In other words, it may be possible to have customised adaptation (of different tools) for each subject/discipline or even different modes of teaching. For instance, if group work is an important aspect of a teaching module (as in Doolan and Baker, 2005), inclusion of appropriate tools that would facilitate group work could be considered as part of the composite system.

Nonetheless, it is worth noting that interoperability issues, such as those highlighted by White and Davis (2000) should not be ignored nor underestimated.

7.4 Closing remarks

This research has demonstrated the advantages of proactive self-assessing and feedback. It allows students, as well as teachers, to reflect on their learning progress and performance. Nonetheless, students tend to find self-assessment time-consuming and burdensome, particularly since they do not count towards the final score of the module. This has been proven wrong by the use of the ASAM system.

To design and develop a useful adaptive self-assessment system, one has to consider a number of factors, including: the definition of question items and assessment content, the classification of question items (according to difficulty), the capturing and storing of the profiling data, the feedback returned to the students and the support to the teaching staff. Last but not least, teachers and students alike are to be encouraged to use the system. Therefore, user interface (as well as implementation strategy) is very important – the benefits of a reliable and powerful application is lost if the user struggles to carry out their day-to-day tasks (Golightly, 2006).

References

Abdullah, S C, 1999, Premodelling for Examination Revision through Adaptive Testing, *3rd Human Centred Technology Postgraduate Workshop*, 30 Sep – 1 Oct 1999, Brighton. Available at: http://www.cogs.susx.ac.uk/lab/hctw99/download/html/sophiana.html [Accessed 21 Apr 2003]

Abdullah, S C, and Cooley, R E, 2000, Using Constraints to Develop and Deliver Adaptive Tests, *4th Computer Assisted Assessment Conference,* 21 – 22 June, Loughborough.

Aluisio, S M and Oliveira Jr, O N, 1999, An Innovative Computer Assisted Proficiency Test of English for Academic Purposes, *3rd Computer Assisted Assessment Conference,* 16 – 17 June, Loughborough.

Armstrong, E, Ball, J, Bodoff, S, Carson, D B, Evans, I, Fisher, M, Fordin, S, Green, D, Haase, K and Jendrock, E, 2001, *The Java(TM) Web Services Tutorial* [online]. Sun Microsystems, Inc: Sun Developer Network Site. Available at: http://java.sun.com/webservices/docs/1.3/tutorial/doc/index.html [Accessed 31 May 2004]

Ashton, H S, Beevers C E, Schofield D K and Youngson, M A, 2004, Informative Reports – Experiences from the Pass-IT Project, 8th *Computer Assisted Assessment Conference*, 6 – 7 July, Loughborough.

Ashton, H and Thomas, R, 2006, Bridging the Gap between Assessment, Teaching and Learning, *10th Computer Assisted Assessment Conference*, 4 – 5 July, Loughborough.

Atkinson, R L, *et al*, 1990, *Introduction to Psychology* (10th ed). US: Harcourt Brace Jovanovich.

Baddeley, A D, 1976, *The Psychology of Memory*. New York: Basic.

Baddeley, A, 1993, *Your Memory: A User's Guide* (2nd ed). London: Prion (Multimedia Books Ltd).

Baker, Frank B., 2001, *The Basics of Item Response Theory* (2nd ed). Available at: http://ericae.net/irt/baker/ [Accessed 21 Nov 2005]

Barker, T and Barker, J, 2002, The evaluation of complex, intelligent, interactive, individualised human-computer interfaces: What do we mean by reliability and validity?, *Proceedings of the European Learning Styles Information Network Conference*, University of Ghent, June 2002.

Barker, T, Jones, S, Britton, C and Messer D, 2002, The use of a co-operative student model of learner characteristics to configure a multimedia application, In *User Modelling and User Adapted* Interaction, 12 (2/3), pp 207-241.

Barker, T and Lilley, M, 2006, Measuring Staff Attitude to an Automated Feedback System based on a Computer-Aided Test, *10th Computer Assisted Assessment Conference*, 4 – 5 July, Loughborough.

Belton, M and Kleeman, J, 2000, Latest Advances in Perception, *4th Computer Assisted Assessment Conference*, 21 – 22 June, Loughborough.

Belton, M and Kleeman, J, 2001, New Developments in Perception Designed to Meet the Changing Face of CAA, *5th Computer Assisted Assessment Conference*, 2 – 3 July, Loughborough.

Bloom, B, 1956, *Taxonomy of Educational Objectives*. New York: McKay.

Borland, K W, Jr, 2002, Towards a culture of assessment, In: Schwartz, P and Webb, G (eds), *Assessment: case studies, experience and practice from Higher Education*, pp. 97-104.

Boticario, J G, and Gaudioso, E, 2000, Adaptive Web site for distance learning, *Campus-Wide Information Systems*, 17 (4), pp. 120-128.

Boud, D, 1991, *Implementing Student Self Assessment* (2nd ed). HERDSA Green Guide 5: Campbelltown: Higher Education Research and Development Society of Australasia.

Boud, D, 1995, *Enhancing Learning through Self Assessment*. London: Kogan Page.

Brailsford, T J, Moore, A, Stewart, C D, Zakaria, M R, Choo, B S and Davies, P M C, 2001, Towards a framework to effective web-based distributed learning, Poster, *10th International World Wide Web Conference*, Hong Kong, 1-5 May.

Brailsford, T, Stewart, C D, Zakaria, M R and Moore, A, 2002, Autonavigation, Links and Narrative in an Adaptive Web-Based Integrated Learning Environment, *11th International World Wide Web Conference*, Honolulu, Hawaii, 7-11 May.

Bray, T, Paoli, J, Sperberg-McQueen, C M and Maler, E, 2000, *Extensible Markup Language (XML) 1.0* (2nd Edition), W3C Recommendation, Oct 2000.

Brown, S, 2002, What to do about John?, In: Schwartz, P and Webb, G (eds), *Assessment: case studies, experience and practice from Higher Education*, pp. 32-38.

Brown, S and Knight, P, 1994, *Assessing Learners in Higher Education*. London: Kogan Page.

Brown, S, Race, P and Rust, C, 1995, Using and Experiencing Assessment, In: Knight, P (ed), *Assessment for Learning in Higher Education*, pp. 75-85.

Brusilovsky, P, 1996, Methods and techniques of adaptive hypermedia, *User Modeling and User-Adapted Interaction*, 6 (2-3), pp. 87-129.

Brusilovsky, P, 1999, Adaptive and Intelligent Technologies for Web-based Education, In: Rollinger, C and Peylo, C (eds.), *Künstliche Intelligenz, Special Issue on Intelligent Systems and Teleteaching*, 4, pp. 19-25.

Brusilovsky, P, 1998, Adaptive Educational Systems on the World-Wide-Web: A Review of Available Technologies. In: *Proceedings of Workshop "WWW-Based Tutoring" at 4th International Conference on Intelligent Tutoring Systems* (ITS'98), San Antonio, TX, August 16-19.

Brusilovsky, P, 2000, Adaptive Hypermedia: From Intelligent Tutoring Systems to Web-based Education, In: Gauthier, G, Frasson, C and VanLehn, K (eds.), *Intelligent Tutoring Systems*, Lecture Notes in Computer Science, vol. 1839 (Proceedings of 5th International Conference on Intelligent Tutoring Systems, 19-23 June, Montreal, Canada), Berlin: Springer Verlag, pp. 1-7.

Brusilovsky, P, 2001, WebEx: Learning from examples in a programming course, In Fowler, W and Hasebrook, J (eds.) In: *Proceedings of WebNet'2001, World Conference of the WWW and Internet*, Orlando, FL, AACE (2001) 124-129.

Brusilovsky, P, Eklund, J and Schwarz, E, 1998, Web-based Education for All: A Tool for Development Adaptive Courseware, In: *Computer Networks and ISDN Systems*, 30, 1-7 (1998), pp. 291-300.

Brusilovsky, P and Maybury, M T., 2002, From Adaptive Hypermedia to the Adaptive Web, *Communications of the ACM*, 45 (5), pp. 31-33.

Brusilovsky, P and Miller, P, 1999, Web-based Testing for Distance Education, In: De Bra, P and Leggett, J (eds), *Proceedings of WebNet'99*, World Conference of the World-Wide Web and Internet, Honolulu, HI, 24 – 30 Oct, AACE, pp. 149-154.

Brusilovsky, P and Miller, P, 2001, Course Delivery Systems for the Virtual University, In: Tschang, F T and Senta, T D (eds.), *Access to Knowledge: New Information Technologies and the Emergence of the Virtual University*, Amsterdam: Elsevier Science and International Association of Universities, pp. 167-206.

Brusilovsky, P and Nijhavan, H, 2002, A Framework for Adaptive E-Learning Based on Distributed Re-usable Learning Activities. In: Driscoll M and Reeves T C (eds.) *Proceedings of World Conference on E-Learning*, E-Learn 2002, Montreal, Canada, 15 – 19 Oct, AACE, pp. 154-161.

Brusilovsky, P and Su, H-D, 2002, Adaptive visualisation component of a distributed web-based adaptive educational system, In *Proceedings of International Conference on Intelligent Tutoring Systems* (ITS 2002), Biarritz, France, June 2002.

Burstein, J, Leacock, C and Swartz, R, 2001, Automated Evaluation of Essays and Short Answers, *5th Computer Assisted Assessment Conference*, 2 – 3 July, Loughborough.

Callear, D, Jerrams-Smith, J and Soh, V, 2001, Using Constraints to Develop and Deliver Adaptive Tests, *5th Computer Assisted Assessment Conference*, 2 – 3 July, Loughborough.

Calvi, L and De Bra, P, 1997, Proficiency-Adapted Information Browsing and Filtering in Hypermedia Educational Systems, *User Modeling and User-Adapted Interaction*, 7, pp. 257-277.

Christie, J R, 1999, Automated Essay Marking – for both Style and Content, *3rd Computer Assisted Assessment Conference*, 16 – 17 June, Loughborough.

Christie, J R, 2003, Automated Essay Marking for Content – Does it Work?, *7th Computer Assisted Assessment Conference*, 8 – 9 July, Loughborough.

Collins, J A, Greer, J E and Huang, S X, 1996, Adaptive Assessment using Granularity Hierarchies and Bayesian Nets, In: Frasson, C, Gauthier, G and Lesgold, A (eds), *Intelligent Tutoring Systems*, Third International Conference, ITS'96, Canada, June 1996, Lecture Notes in Computer Science 1086. Berlin Heidelberg: Springer-Verlag, pp. 569-577.

Conati, C, Gertner, A S, VanLehn, K and Druzdzel, M J, 1997, On-Line Student Modeling for Coached Problem Solving Using Bayesian Networks, In *Proceeding of the Sixth International Conference on User Modeling*, pp. 231-242. Available at: http://www.pitt.edu/~conati/um97.ps [Accessed 05 Apr 2003]

Corbett, A T, Koedinger K R and Anderson, J R, 1997, Intelligent Tutoring Systems, In Helander, M, Landauer, T K and Prabhu, P (eds), *Handbook of Human-Computer Interaction*, 2nd ed., pp. 849-874.

Dalziel, J, 2001, Enhancing web-based learning with computer assisted assessment: Pedagogical and technical considerations, *5th Computer Assisted Assessment Conference,* 2 – 3 July, Loughborough.

Danson, M, Dawson, B and Baseley, T, 2001, Large Scale Implementation of Question Mark Perception (V2.5) – Experiences at Loughborough University, *5th Computer Assisted Assessment Conference*, 2 – 3 July, Loughborough.

Davies, W M and Davis, H C, 2006, QuestionBuddy – A Collaborative Question Search and Play Portal, *10th Computer Assisted Assessment Conference*, 4 – 5 July, Loughborough.

De Bra, P, 2002, Adaptive Educational Hypermedia on the Web, *Communications of the ACM*, 45(5), pp. 60-61.

De Bra, P, Brusilovsky, P and Houben, G-J, 1999, Adaptive Hypermedia: From Systems to Framework, *ACM Computing Survey*, 31(4). Available at: http://www.acm.org/pubs/articles/journals/surveys/1999-31-4es/a12-de_bra/a12-de_bra.pdf [Accessed 21 Nov 2005]

De Bra, P and Calvi, L, 1998, AHA! An open Adaptive Hypermedia Architecture, *The New Review of Hypermedia and Multimedia*, vol. 4, pp. 115-139

Dearing R. *et al*, 1997, *Higher Education in the Learning Society: Report of the National Committee of Inquiry into Higher Education*, London: HMSO and NCIHE Publications.

Doolan, M A and Barker, T, 2005, Evaluation of Computing Students Performance, using Group Based Learning Online and Offline, *9th Computer Assisted Assessment Conference*, 5 – 6 July, Loughborough.

Draaijer, S and Boter, J, 2005, Questionbank: computer supported self-questioning, *9th Computer Assisted Assessment Conference*, 5 – 6 July, Loughborough.

Eason, K, 1988, *Information Technology and Organisational Change*.

Eberts, R E, 1997, Computer-Based Instruction, In Helander, M, Landauer, T K and Prabhu, P (eds), *Handbook of Human-Computer Interaction*, 2nd ed., pp. 825-847.

Eklund, J, Brusilovsky, P, and Schwarz, E, 2002, Adaptive Textbooks on the World Wide Web, *3rd Australian World Wide Web Conference*, Lismore NSW, 5-9 May. Available at: http://ausweb.scu.edu.au/proceedings/eklund/paper.html [Accessed 21 Jul 2006]

Erwin, T D, 1995, Attending to Assessment: A Process for Faculty, In: Knight, P (ed), *Assessment for Learning in Higher Education*, pp. 49-59.

Eysenck, M W and Keane, M T, 2000, *Cognitive Psychology: A Student's Handbook* (4th ed). London: Psychology Press.

Fallside, D C, 2001, *XML Schema Part 0: Primer*, W3C Recommendation, May 2001. Available at: http://www.w3.org/TR/2001/REC-xmlschema-0-20010502/ [Accessed 21 Nov 2005]

Farmer, B and Eastcott, D, 1995, Making Assessment a Positive Experience, In: Knight, P (ed), *Assessment for Learning in Higher Education*, pp 87-93.

Farthing, D W, and McPhee, D, 1999, Multiple choice for honours-level students? A statistical evaluation, *3rd Computer Assisted Assessment Conference*, 16-17 June, Loughborough.

Fischer, S and Steinmetz, R, 2000, Automatic Creation of Exercises in Adaptive Hypermedia Learning Systems, *Hypertext 2000 Proceedings*, pp. 49-55.

Fone, W, 2002, Improving Feedback from Multiple Choice Tests, *Proceedings of 7th Annual SIGCSE Conference on Innovation and Technology in Computer Science*, (Tips and Techniques Session), 24-28 June, Aarhus, Denmark, pp 196.

Foxley, E, Higgins, C, Hegazy, T, Symeonidis, P and Tsintsifas, A, 2001, The CourseMaster CBA System: Improvements over Ceilidh, *5th Computer Assisted Assessment Conference*, 2 – 3 July, Loughborough.

Gibbs, G, Habeshaw, S and Habeshaw, T, 1988, *53 Interesting Ways To Assess Your Students* (2nd ed). Bristol: Technical and Educational Services.

Golightly, D, 2006, HCI in the media: the usability of Lotus Notes, *The Ergonomist*, April 2006, pp. 1-2.

Grebenik, P and Rust, C, 2002, IT to the rescue, In: Schwartz, P and Webb, G (eds), *Assessment: case studies, experience and practice from Higher Education*, pp. 18-24.

Hall, M, 2000, *Core Servlets and JavaServer Pages*. US: Sun Microsystems Press and Prentice Hall.

Higgins, C., Symeonidis, P. and Tsintsifas, A., 2002, The marking subsystem of CourseMarker, In: *Proceedings of Integrating Technology in Computer Science Education* (ITiCSE) 2002, Denmark, 24 – 26 June, pp. 46-50.

Java Sun, 2005, *Java Remote Method Invocation (Java RMI)*. Available at: http://java.sun.com/products/jdk/rmi/ [Accessed 21 Sep 2005]

Jones, P, Jacobs, G and Brown, S, 1997, Learning styles and CAL design: a model for the future, *Active Learning*, 7, December, pp. 9-13.

King, T and Duke-Williams, E, 2001, Using Computer-Aided Assessment to Test Higher Level Learning Outcomes, *5th Computer Assisted Assessment Conference*, 2 – 3 July, Loughborough.

Kleeman, J, 2000, Getting More Meaning from the Results of CAA Discussion forum introduction, *4th Computer Assisted Assessment Conference*, 21 – 22 June 2000, Loughborough.

Kölling, M and Rosenberg, J, 2001, Guidelines for Teaching Object Orientation with Java, *Sixth Annual Conference on Innovation and Technology in Computer Science Education Proceedings,* Canterbury, 25-27 June.

Korhonen, A, Malmi, L, Myllyselkä, P and Scheinin, P, 2002, Does it Make a Difference if Students Exercise on the Web or in the Classroom?, *Proceedings of 7th Annual SIGCSE Conference on Innovation and Technology in Computer Science*, 24-28 June, Aarhus, Denmark, pp. 121-124.

Landauer, T K, Foltz, P W and Laham, D, 1998, Introduction to Latent Semantic Analysis, *Discourse Processes*, 25, pp. 259-284.

Laurillard, D, 1993, *Rethinking university teaching: A framework for the effective use of education technology*. London: Routledge.

Lay, S, 2004, Question and Test Interoperability: introducing version 2 of the IMS QTI specification, *8th Computer Assisted Assessment Conference*, 6 – 7 July, Loughborough.

Lilley, M and Barker, T, 2002, The Development and Evaluation of a Computer-Adaptive Testing Application for English Language, *6th Computer Assisted Assessment Conference*, July 2002, Loughborough.

Lilley, M and Barker, T, 2003, An Evaluation of a Computer Adaptive Test in a UK University Context, *7th Computer Assisted Assessment Conference*, 8 – 9 July, Loughborough.

Lilley, M, Barker, T and Britton, C, 2005, Automated Feedback for a Computer-Adaptive Test: A Case Study, *9th Computer Assisted Assessment Conference*, 5 – 6 July, Loughborough.

MacKay, B R and Emerson, L, 2000, Web-based assessment of writing skills, *4th Computer Assisted Assessment Conference*, 21 – 22 June, Loughborough.

Mackenzie, D, 2000, Production and delivery of TRIADS Assessments on a university-wide basis, *4th Computer Assisted Assessment Conference*, 21 – 22 June, Loughborough.

Mark, M A and Greer, J E, 1993, Evaluation Methodologies for Intelligent Tutoring Systems, *Journal of Artificial Intelligence and Education*, 4(2/3), pp. 129-153.

Mason, O and Grove-Stephensen, I, 2002, Automated free text marking with Paperless School, *6th Computer Assisted Assessment Conference*, July 2002, Loughborough.

McDowell, L, 1996, Enabling Student Learning Through Innovative Assessment, In: Wisker, G and Brown, S, *Enabling Student Learning: Systems and Strategies*, pp. 158-165.

Mitchell, T, Russell, T, Broomhead, P and Aldridge, N, 2002, Towards Robust Computerised Marking of Free-Text Responses, *6th Computer Assisted Assessment Conference*, July 2002, Loughborough.

Moore, I, 1995, Staff and Educational Development for Assessment Reform: A Case Study, In: Knight, P (ed), *Assessment for Learning in Higher Education*, pp. 95-109.

Morgan, N and Saxton, J, 1994, *Asking Better Questions: Models, techniques and classroom activities for engaging students in learning*.

NYIT, 2003, *Bloom's Taxonomy* [online]. New York: New York Institute of Technology Educational Enterprise Zone. Available at: http://www.nyiteez.org/EDIN777/bloom.htm [Accessed 21 Apr 2003]

Ort, E and Mehta, B, 2003, *Java Architecture for XML Binding (JAXB)* [online]. Sun Microsystems, Inc: Sun Developer Network Site. Available at: http://java.sun.com/developer/technicalArticles/WebServices/jaxb/index.html [Accessed 31 May 2004]

Pathak, S and Brusilovsky, P, 2002, Assessing student programming knowledge with web-based dynamic parameterised quizzes, In: *Proceedings of ED-MEDIA 2002 – World Conference on Educational Multimedia, Hypermedia and Telecommunications*, Denver, CO, 24-29 June, AACE.

PennState, 2000, *Academic Testing: Item Response Theory Approach* [online]. Pennsylvania: The Pennsylvania State University, Schreyer Institute for Teaching Excellence. Available at: http://www.uts.psu.edu/Item_Response_Theory_frame.htm [Accessed 17 Feb 2004]

Pollock, M J, Whittington, C D and Doughty, G F, 2000, Evaluating the Costs and Benefits of changing to CAA, *4th Computer Assisted Assessment Conference*, 21 – 22 June, Loughborough.

Race, P, 1994a, *The Open Learning Handbook*, 2nd ed. London: Kogan Page.

Race, P, 1994b, *Never mind the Teaching – Feel the Learning*, SEDA Paper 80, SEDA Publications, Birmingham, UK.

Race, P, 1995, What has Assessment Done for Us – and to Us?, In: Knight, P (ed), *Assessment for Learning in Higher Education*, pp. 61-74.

Rickel, J, Lesh, N, Rich, C, Sidner, C L and Gertner, A, 2001, Building a Bridge between Intelligent Tutoring and Collaborative Dialogue Systems, In: *Proceedings of Tenth International Conference on AI in Education*, May 2001, pp. 592-594.

Sambell, K, Miller, S and Hodgson, S, 2002, Let's get the assessment to drive the learning, In: Schwartz, P and Webb, G (eds), *Assessment: case studies, experience and practice from Higher Education*, pp. 137-143.

Schank, R and Kass, A, 1996, A Goal-Based Scenario for High School Students, *Communications of the ACM*,39(4), pp. 28-29.

Schwartz, P, 2002, Gain without pain?, In: Schwartz, P and Webb, G (eds), *Assessment: case studies, experience and practice from Higher Education*, pp. 25-31.

Sclater, N and Howie, K, 2000, The Scottish Computer Assisted Assessment Network, *4th Computer Assisted Assessment Conference*, 21 – 22 June, Loughborough.

Sclater, N, Low, B and Barr, N, 2002, Interoperability with CAA: Does it Work in Practice?, *6th Computer Assisted Assessment Conference*, July 2002, Loughborough.

Sclater, N and MacDonald, M, 2004, Developing a National Item Bank, *8th Computer Assisted Assessment Conference*, 6-7 July, Loughborough.

Slater, C, 2002, *Multiple Choice Question system for CourseMaster* (Final Year Dissertation), University of Nottingham, School of Computer Science.

Sosnovsky, S, Shcherbinina, O and Brusilovsky, P, 2003, Web-based Parameterized Questions as a Tool for Learning, In: *Proceedings of World Conference on E-Learning*, E-Learn 2003, Phoenix, AZ, USA, 7-11 Nov, AACE, pp. 309-316.

Stephens, D and Percik, D, 2002, Constructing a Test Bank for Information Science based upon Bloom's principles, *Innovations in Teaching and Learning in Information and Computer Sciences*, 2(1). Available at: http://www.ics.ltsn.ac.uk/pub/italics/issue1/stephens/009.html [Accessed 21 Nov 2005]

Syang, A and Dale, N B, 1993, Computerized Adaptive Testing In Computer Science: Assessing Student Programming Abilities, In: *Proceedings of the 24th SIGCSE Technical Symposium on Computer Science Education*, 18-19 Feb, Indianapolis, USA, pp. 53-57.

Taras, M, 2002, Using Assessment for Learning and Learning from Assessment, In: *Assessment and Evaluation in Higher Education*, 27(6), pp. 501-510.

Taras, M, 2003, To Feedback or Not to Feedback in Student Self-assessment, In: *Assessment and Evaluation in Higher Education*, 28(5), pp. 549-565.

Thomas, P, 2003, The Evaluation of Electronic Marking of Examinations, In: *Proceedings of the 8th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education (ITICSE 2003)*, 30 June – 2 July, Thessaloniki, Greece, pp. 50-54.

Thomas, P and Taylor, D, 2000, Reducing the distance in distance education, *4th Computer Assisted Assessment Conference*, Loughborough.

Tsintsifas, A, 2002, *A framework for the computer based assessment of diagram based coursework* (PhD thesis), University of Nottingham, School of Computer Science.

UoN IS, 2003, *Portal*, University of Nottingham, Information Services. Available at: http://www.nottingham.ac.uk/is/about/projects/portal/intro.html [Accessed 06 Feb 2004]

uPortal, 2001, *uPortal architecture overview*. Available at http://mis105.mis.udel.edu/ja-sig/uportal/architecture/uPortal_architecture_overview.pdf [Accessed 21 Nov 2005]

Vizcaíro, A, Contreras, J, Favea, J and Prieto, M, 2000, An adaptive collaborative environment to develop good habits in programming, In: Gauthier, G, Frasson, C and VanLehn, K (eds.), *Intelligent Tutoring Systems*, Lecture Notes in Computer Science, vol. 1839 (Proceedings of 5th International Conference on Intelligent Tutoring Systems, 19-23 June, Montreal, Canada), Berlin: Springer Verlag, pp. 262-271.

Wainer, H, 1990, *Computerized Adaptive Testing (A Primer)*. New Jersey: Lawrence Erlbaum Associates, Publishers.

White, S and Davis, H, 2000, Creating large-scale test banks: a briefing for participative discussion of issues and agendas, *4th Computer Assisted Assessment Conference*, 21 – 22 June, Loughborough.

Yong, C-F and Higgins, C, 2003, Automatically Creating Personalised Exercises based on Student Profiles, In: *Proceedings of Integrating Technology in Computer Science Education* (ITiCSE) 2003, Greece, 30 June – 2 July, pp 236.

Yong, C-F and Higgins, C A, 2004, Self-assessing with adaptive exercises, *8th Computer Assisted Assessment Conference*, 6 – 7 July, Loughborough.

Appendix A –

List of abbreviations

AH	adaptive hypermedia
AHS	adaptive hypermedia system
CAA	Computer-aided assessment
CAI	computer-assisted instruction
CAT	computerised-adaptive test
CBA	computer-based assessment
CBT	computer-based test
CM	CourseMarker
GUI	Graphical User Interface
HE	higher education
ILE	integrated learning environment
IRT	Item Response Theory
ITS	intelligent tutoring system
MCQ	multiple-choice questions

PMCQ	permutational multiple choice question
RMI	Remote Method Invocation
TMA	tutor marked assignment
UM	user modelling
WBE	web-based education

Appendix B –

List of websites

ADAPT – a Minerva project	http://wwwis.win.tue.nl/~acristea/HTML/Minerva/
Asymetrix Toolbook	http://www.asymetrix.com
Blackboard	http://www.blackboard.com
COLA	http://www.cetis.ac.uk/profiles/uklomcore/toia_cola_metadata_v1p2.doc
e3an	http://www.e3an.ac.uk
IEEE LOM	http://ltsc.ieee.org/wg12/index.html
IMS QTI	http://www.imsglobal.org/question/index.cfm
LTSN-ICS	http://www.ics.ltsn.ac.uk
Macromedia Director	http://www.macromedia.com
The Minerva Action	http://europa.eu.int/comm/education/programmes/socrates/minerva/ind1a_en.html
QuestionMark	http://www.questionmark.com/uk/perception/index.htm
QUILS	http://www.ics.ltsn.ac.uk/ILS/quils.html

UK Universities Worldwide	http://www.ukeuniversitiesworldwide.com
Universitas 21 Global	http://www.u21global.com
WebCT	http://www.webct.com

Appendix C –

XML schema for Questions

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
        elementFormDefault="qualified">

  <xs:element name="problem" type="ProblemType"/>

  <xs:complexType name="ProblemType">
    <xs:sequence>
      <xs:element ref="head"/>
      <xs:element ref="body"/>
    </xs:sequence>
  </xs:complexType>

  <xs:element name="head">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="subject" type="xs:string"
minOccurs="0"/>
        <xs:element name="topic" type="xs:string"
minOccurs="0"/>
        <xs:element name="subtopic" type="xs:string"
minOccurs="0"/>
        <xs:element name="institution" type="xs:string"
minOccurs="0"/>
        <xs:element ref="author" minOccurs="0"/>
        <xs:element name="difficulty">
          <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:enumeration value="Beginner"/>
            <xs:enumeration value="Intermediate"/>
            <xs:enumeration value="Advanced"/>
          </xs:restriction>
          </xs:simpleType>
        </xs:element>
        <xs:element name="duration" type="xs:string"/>
        <xs:element ref="created"/>
        <xs:element name="id_number" type="xs:string"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="id_number" type="xs:string"/>
  <xs:element name="author">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="name" type="xs:string"/>
```

```
        <xs:element name="address" type="xs:string"/>
        <xs:element name="email" type="xs:string"/>
        <xs:element name="tel" type="xs:string"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="created">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="time" type="xs:string"/>
        <xs:element name="date" type="xs:string"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name="body">
    <xs:complexType>
      <xs:choice>
        <xs:element ref="multichoice"/>
        <xs:element ref="multiresponse"/>
        <xs:element ref="missingword"/>
        <xs:element ref="numeric"/>
        <xs:element ref="programming"/>
      </xs:choice>
    </xs:complexType>
  </xs:element>
  <xs:element name="multichoice">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="question"/>
        <xs:element ref="correct"/>
        <xs:element ref="incorrect"
maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="multiresponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="question"/>
        <xs:element ref="correct" minOccurs="2"
maxOccurs="unbounded"/>
        <xs:element ref="incorrect"
maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="missingword">
    <xs:complexType>
      <xs:sequence>
        <xs:sequence>
          <xs:element ref="text"/>
          <xs:element ref="word"/>
        </xs:sequence>
        <xs:sequence minOccurs="0" maxOccurs="unbounded">
          <xs:element ref="text"/>
```

```
          <xs:element ref="word" minOccurs="0"/>
        </xs:sequence>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="numeric">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="question"/>
        <xs:element ref="correct"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="programming">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="question"/>
        <xs:element ref="cmref"/>
        <xs:element name="skeletonfile"
type="xs:string"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="question" type="xs:string"/>
  <xs:element name="correct" type="xs:string"/>
  <xs:element name="incorrect" type="xs:string"/>
  <xs:element name="text" type="xs:string"/>
  <xs:element name="word" type="xs:string"/>

  <xs:element name="cmref">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="course"/>
        <xs:element ref="unit"/>
        <xs:element ref="exercise"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="course" type="xs:string"/>
  <xs:element name="unit" type="xs:string"/>
  <xs:element name="exercise" type="xs:string"/>
</xs:schema>
```

Appendix D –

XML Schema for Questionnaire Skeleton

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           elementFormDefault="qualified">

  <xs:element name="skeleton" type="SkeletonType"/>

  <xs:complexType name="SkeletonType">
    <xs:sequence>
      <xs:element name="name" type="xs:string"/>
      <xs:element name="author" type="xs:string"/>
      <xs:element name="type">
        <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:enumeration value="Fixed"/>
          <xs:enumeration value="Adaptive"/>
        </xs:restriction>
        </xs:simpleType>
      </xs:element>
      <xs:element name="time_limit" type="xs:integer"/>
      <xs:element ref="questions"/>
      <xs:element ref="feedback"/>
    </xs:sequence>

  </xs:complexType>

  <xs:element name="questions">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="question_info"
maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name="question_info">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="subject" type="xs:string"/>
        <xs:element name="topic" type="xs:string"/>
        <xs:element name="subtopic" type="xs:string"/>
        <xs:element name="prerequisite" type="xs:string"
```

```
minOccurs="0"/>
        <xs:choice>
          <xs:element name="question_id"
type="xs:string"/>
          <xs:element name="question_type">
            <xs:simpleType>
            <xs:restriction base="xs:string">
              <xs:enumeration value="Multiple Choice"/>
              <xs:enumeration value="Multiple Response"/>
              <xs:enumeration value="Missing Word"/>
              <xs:enumeration value="Numeric"/>
              <xs:enumeration value="Any Type"/>
            </xs:restriction>
            </xs:simpleType>
          </xs:element>
        </xs:choice>
      </xs:sequence>

    <xs:attribute name="mark" type="xs:integer"/>
    </xs:complexType>
  </xs:element>

  <xs:element name="feedback">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="topic_feedback" minOccurs="0"
maxOccurs="unbounded"/>
      </xs:sequence>
    <xs:attribute name="value">
      <xs:simpleType>
      <xs:restriction base="xs:string">
          <xs:enumeration value="Yes"/>
          <xs:enumeration value="No"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    </xs:complexType>
  </xs:element>

  <xs:element name="topic_feedback">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="beginner" type="xs:string"/>
        <xs:element name="intermediate"
type="xs:string"/>
        <xs:element name="advanced" type="xs:string"/>
      </xs:sequence>
    <xs:attribute name="for_subj" type="xs:string"/>
    <xs:attribute name="for_topic" type="xs:string"/>
    </xs:complexType>
  </xs:element>

</xs:schema>
```

Appendix E –

Survey (2003) questionnaire on ASAM

This survey is to collect users' views on the usage of the adaptive self-assessment system for the module G51PRG, in the first semester of 2003/4. It would be greatly appreciated if you could take some time (about 5 minutes) to complete the survey. Thank you.

Q1. Have you used the adaptive self-assessment system in the last semester?

- [ ] Yes*(Go to Q2)*
- [ ] No*(Go to end of survey)*

Q2. Did you find it useful?

- [ ] Yes
- [ ] No

Why?

Q3. What kind/s of feedback do you prefer?

- Marks only
- Feedback on various topics only
- Both marks and feedback on various topics
- None of the above

Q4. Any other comments?

Thank you for your time.

Appendix F –

Survey (2005) questionnaire on ASAM

This survey is to collect users' views on the usage of the adaptive self-assessment system for the module G51PRG, in the first semester of 2005/6. It would be greatly appreciated if you could take some time (about 5 minutes) to complete the survey. Thank you.

Q1. Did you use the ASAM system to help you in your revision?

Yes*(Go to Q3)*

No*(Go to Q2)*

Q2. Why did you not use the ASAM system?

(Go to Q10)

Q3. Do you think the feature of generating different programming questions (of varying levels of difficulty) is useful?

(1-5 = "not useful at all"-> "very useful")

- 1 (not useful at all)
- 2
- 3
- 4
- 5 (very useful)

Q4. Did using the system help you understand programming better?

- Yes, a lot
- Yes, a little
- No, not really
- No, definitely not

Q5. Does using the system encourage you to want to do more programming practices on your own?

- ☐ Yes
- ☐ No
- ☐ Maybe

Q6. Do you think it is a good practice to self-assess using the system?

- ☐ Yes
- ☐ No
- ☐ Don't know

Q7. Do you think using the system will reinforce your learning of the programming skills?

- ☐ Yes
- ☐ No
- ☐ Don't know

Q8. Did you find the feedback returned by the ASAM system useful?

- ☐ Yes, a lot
- ☐ Yes, a little
- ☐ No, not really
- ☐ No, definitely not

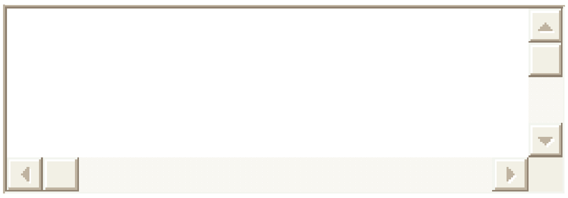
Q9. Would you recommend the ASAM system to your friends?

- ☐ Yes
- ☐ No
- ☐ Maybe

Q10. Prior to the G51PRG module, what was your programming experience?

- ☐ Had not done any programming at all
- ☐ Had been programming for no more than six months
- ☐ Had been programming for no more than a year
- ☐ Had been programming for more than a year

Q11. Any other comments?



Thank you for your time.