The South African Institute for Computer Scientists and Information Technologists

ANNUAL RESEARCH AND DEVELOPMENT SYMPOSIUM

23-24 NOVEMBER 1998
CAPE TOWN
Van Riebeeck Hotel in Gordons Bay

Hosted by the University of Cape Town in association with the CSSA, Forestry South Africa University for CHET and The University of Natal

PROCEEDINGS

EDITED BY
D. PEKOV AND L. VENTER

SPONSORED BY

ABSA Group
The South African Institute for Computer Scientists and Information Technologists

ANNUAL RESEARCH AND DEVELOPMENT SYMPOSIUM

23-24 NOVEMBER 1998
CAPE TOWN
Van Riebeeck hotel in Gordons Bay

Hosted by the University of Cape Town in association with the CSSA,
Potchefstroom University for CHE and
The University of Natal

GENERAL CHAIR: PROF. G. HATTINGH, PU CHE
PROGRAMME CO-CHAIRS:
PROF. L VENTER, PU CHE (Vaal Triangle), PROF. D. PETKOV, UN-PMB
LOCAL ORGANISING CHAIR: PROF. P. LICKER, UCT - IS

PROCEEDINGS

EDITED BY
D. PETKOV AND L. VENTER

SYMPOSIUM THEME:
Development of a quality academic CS/IS infrastructure in South Africa

SPONSORED BY

[Logo]

ABSA Group
FOREWORD

The South African Institute for Computer Scientists and Information Technologists (SAICSIT) promotes the cooperation of academics and industry in the area of research and development in Computer Science, Information Systems and Technology and Software Engineering. The culmination of its activities throughout the year is the annual research symposium. This book is a collection of papers presented at the 1998 such event taking place on the 23rd and 24th of November in Gordons Bay, Cape Town. The Conference is hosted by the Department of Information Systems, University of Cape Town in cooperation with the Department of Computer Science, Potchefstroom University for CHE and and Department of Computer Science and Information Systems of the University of Natal, Pietermaritzburg.

There are a total of 46 papers. The speakers represent practitioners and academics from all the major Universities and Technikons in the country. The number of industry based authors has increased compared to previous years.

We would like to express our gratitude to the referees and the paper contributors for their hard work on the papers included in this volume. The Organising and Programme Committees would like to thank the keynote speaker, Prof M.C Jackson, Dean, University of Lincolnshire and Humberside, United Kingdom, President of the International Federation for Systems Research as well as the Computer Society of South Africa and The University of Cape Town for the cooperation as well as the management and staff of the Potchefstroom University for CHE and the University of Natal for their support and for making this event a success.

Giel Hattingh, Paul Licker, Lucas Venter and Don Petkov
# Table of Contents

<table>
<thead>
<tr>
<th>Author(s)</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lynette Drevin</td>
<td>Activities of IFIP wg 11.8 (computer security education) &amp; IT related ethics education in Southern Africa</td>
<td>1</td>
</tr>
<tr>
<td>Reinhardt A. Botha and Jan H.P. Elof</td>
<td>exA Security Interpretation of the Workflow Reference Model</td>
<td>3</td>
</tr>
<tr>
<td>Willem Krige and Rossouw von Solms</td>
<td>Effective information security monitoring using data logs</td>
<td>9</td>
</tr>
<tr>
<td>Carl Papenfus and Reinhardt A. Botha</td>
<td>A shell-based approach to information security</td>
<td>15</td>
</tr>
<tr>
<td>Walter Smuts</td>
<td>A 6-Dimensional Security Classification for Information</td>
<td>20</td>
</tr>
<tr>
<td>Philip Machanick and Pierre Salverda</td>
<td>Implications of emerging DRAM technologies for the RAM page Memory hierarchy</td>
<td>27</td>
</tr>
<tr>
<td>Susan Brown</td>
<td>Practical Experience in Running a Virtual Class to Facilitate On-Campus Under Graduate Teaching</td>
<td>41</td>
</tr>
<tr>
<td>H.D. Masetha, T.A Dandadzi</td>
<td>Quality Academic Development of CS/IS Infrastructure in South Africa</td>
<td>49</td>
</tr>
<tr>
<td>Philip Machanick</td>
<td>The Skills Hierarchy and Curriculum</td>
<td>54</td>
</tr>
<tr>
<td>Theda Thomas</td>
<td>Handling diversity in Information Systems and Computer Science Students: A social Constructivist Perspective</td>
<td>63</td>
</tr>
<tr>
<td>Udo Averweg and G J Erwin</td>
<td>Critical success factors for implementation of Decision support systems</td>
<td>70</td>
</tr>
<tr>
<td>Magda Huisman</td>
<td>A conceptual model for the adoption and use of case technology</td>
<td>78</td>
</tr>
<tr>
<td>Paul S. Licker</td>
<td>A Framework for Information Systems and National Development Research</td>
<td>79</td>
</tr>
<tr>
<td>K. Niki Kunene and Don Petkov</td>
<td>On problem structuring in an Electronic Brainstorming (EBS) environment</td>
<td>89</td>
</tr>
</tbody>
</table>
Derek Smith: Characteristics of high-performing Information Systems Project Managers and Project Teams

Lucas Venêter: INSTAP: Experiences in building a multimedia application

Scott Hazelhurst, Anton Fatti, and Andrew Henwood: Binary Decision Diagram Representations of Firewall and Router Access Lists

Andre Joubert and Annelie Jordaan: Hardware System interfacing with Delphi 3 to achieve quality academic integration between the fields of Computer Systems and Software Engineering

Borislav Roussev: Experience with Java in an Advanced Operating Systems Module

Conrad Mueller: A Static Programming Paradigm

Sipho Langa: Management Aspects of Client/Server Computing

T Nepal and T Andrew: An Integrated Research Programme in AI applied to Telecommunications at ML Sultan Technikon

Yuri Velinov: Electronic lectures for the mathematical subjects in Computer Science

Philip Machanick: Disk delay lines

D Petkov and O Petkova: One way to make better decisions related to IT Outsourcing

Jay van Zyl: Quality Learning, Learning Quality

Matthew O Adigun: A Case for Reuse Technology as a CS/IS Training Infrastructure

Andy Bytheway and Grant Hearn: Academic CS/IS Infrastructure in South Africa: An exploratory stakeholder perspective

Chantel van Niekerk: The Academic Institution and Software Vendor Partnership

Christopher Chalmers: Quality aspects of the development of a rule-based architecture

Rudi Harmse: Managing large programming classes using computer mediated communication and cognitive modelling techniques
Michael Muller: How to gain Quality when developing a Repository Driven User Interface

Elsabe Cloete and Lucas Venter: Reducing Fractal Encoding Complexities

Jean Bilbrough and Ian Sanders: Partial Edge Visibility in Linear Time

Philip Machanick: Design of a scalable Video on Demand architecture

Freddie Janssen: Quality considerations of Real Time access to Multidimensional Matrices

Machiel Kruger and Giel Hattingh: A Partitioning Scheme for Solving the Exact $k$-item 0-1 Knapsack Problem

Ian Sanders: Non-orthogonal Ray Guarding

Fanie Terblanche and Giel Hattingh: Response surface analysis as a technique for the visualization of linear models and data

Olga Petkova and Dewald Roode: A pluralist systemic framework for the evaluation of factors affecting software development productivity

Peter Warren and Marcel Viljoen: Design patterns for user interfaces

Andre de Waal and Giel Hattingh: Refuting conjectures in first order theories

Edna Randiki: Error analysis in Selected Medical Devices and Information Systems
MANAGING LARGE PROGRAMMING CLASSES USING COMPUTER MEDIATED COMMUNICATION AND COGNITIVE MODELLING TECHNIQUES.

R. G. Harmse and G. Pullen
Department of Business Information Systems,
Port Elizabeth Technikon, Private Bag X6011, Port Elizabeth, 6001, email Rudi@ml.petech.ac.za

Abstract

This paper describes a proposed solution to address some of the problems encountered when attempting to teach programming to a large number of students with diverse backgrounds and vastly different levels of technological exposure. A conceptual framework of a lecturer support system, which addresses the diagnosis of student problems and the prescription of corrective action, is outlined. This is followed by a description of a prototype of the electronic communication subsystem, which provides for student / lecturer interaction, with regard to practical assignments. The prototype uses Internet standards to provide a flexible medium of communication to overcome the limitations of paper based assignments, while still providing a rich feedback mechanism (virtual paper).

Introduction

In order for effective learning of practical skills to take place student abilities and problems need to be successfully diagnosed. Individual diagnosis and help has been found to be the ideal situation, with explanations being clear and focused. Providing these ideal conditions in large, heterogeneous classes, where the complex skill of programming has to be taught, is very difficult.

Teaching of programming

Computer laboratory work resembles that of science and mathematics. Therefore it could be expected that effective strategies used in these fields, may also be found in computer programming courses. Studies conducted into the teaching of mathematics, chemistry, programming and general education (Hanna and Ryan, 1985; Hofstein, Ben-Zvi and Carmeli, 1990; Lehrer, Guckenberg, and Sancilio, 1988; Clements, 1990; Hawkins, 1993) has revealed certain factors present when effective learning took place. These findings are summarised here:

- Educators made their lessons more interesting.
- New material was emphasised instead of review.
- Students learned well in environments where they were personally well known.
- Effective educators were more sensitive to their students' needs and learning difficulties.
- Educators successfully diagnosed the abilities of their students and adopted appropriate teaching strategies.
- The best instructors offer individualised diagnosis and help.
- Students were encouraged to co-operate with each other.
- Questions were asked at a higher cognitive level.
- Explanations were clear and focused.
- The educators had confidence in and control over the learning material.
- Programming techniques were best learnt when taught in a mediated way.
Various factors add to the complexity of the educational environment that is being faced by lecturers and students in South African technikons when it is attempted to facilitate the learning of programming skills. Lecturers are faced with increasing numbers of students of diverse cultural backgrounds as well as various levels of technology and mathematics exposure. The programming skills that the students need to master are also complex and involve many interrelated concepts.

These problems make it difficult to provide the ideal conditions for learning, such as clear and focused explanations, diagnosis of student abilities, adoption of appropriate teaching strategies, as well as individualised diagnosis and help. The following section addresses the lecturer's task as manager of this complex learning environment.

Management by objectives

The lecturer functions as manager of the learning resources and learning environment. Since the time and resources available to lecturers is always limited, they should naturally concentrate on doing the work that flows most directly from their role as managers of learning resources (Davies, 1973). To see more clearly the management task of the lecturer it is useful to consider the work done by a lecturer in terms of the four functions of management:

i. Planning. This is the work done when establishing learning objectives.

ii. Organising. This involves the arranging and relating of the learning objectives in such a way that the learning objectives are attained in the most effective and efficient way possible.

iii. Leading. This is the motivating, encouraging and inspiring of the students to allow them to attain the objectives most effectively. This also includes the giving of feedback to the students regarding their progress in meeting the objectives as determined by the controlling function.

iv. Controlling. This is when the lecturer determines whether the organising and leading functions are successful in meeting the set objectives. If the objectives are not being met the lecturer should assess and regulate the situation.

These four functions form a cycle of interrelated activities. Together these four functions define the area of an educator's professional competence and expertise. Of course these functions do not only involve things to be done, but also decisions to be taken. These decisions involve the determination of precise and concrete objectives, followed by the selection of a course of action from among alternatives which is most likely to lead to a successful outcome (Davies, 1973). Planning happens before controlling, but plans do not achieve themselves. Planning guides the allocation of resources while controlling allows the monitoring of the progress being made. Decisions at this level should rely on quantified comparisons with established and agreed standards of performance. Controlling involves the making of complex, structured decisions, especially when these decisions involve the taking of corrective or remedial action.

The lecturer can only perform the management tasks effectively if he or she can determine whether the students are achieving the learning objectives that have been set. Once the objectives have been defined and the learning resources organised, the main concern of the lecturer is managing in terms of the results achieved. If the objectives are being met the course is progressing well, if objectives are not met the lecturer has to reorganise and take corrective action.

The performance review is the most crucial in the whole learning objectives cycle. This is when the progress the students have made in realising the learning objectives is reviewed. During this review the students are faced with the reality of what they have achieved. This enables them to see where they have gone wrong and where they need help. Strengths and weaknesses can be isolated and identified. The lecturer can then decide what special guidance and tuition is necessary and how a student’s performance can be improved and his/her potential realised by the next review. Of course the review is only meaningful if the learning of the students has been accurately assessed and the results communicated to the student.
The following section describes a conceptual framework for providing the above mentioned capabilities.

**Conceptual framework of a lecturer support system**

This section briefly describes the overall framework for the lecturer support system. The system should consist of various modules that can interact with each other and allow for integration with other components to extend functionality. The main framework consists of three parts:

- The curriculum objectives model;
- the individual student models; and
- the communications module.

The lecturer uses the curriculum objectives model to develop the original curriculum and its objectives. This model serves as a course guide to the students and also provides the basis to compare student progress against.

When an assignment is evaluated, the lecturer updates the student model. This model is compared with the intended objectives as defined in the curriculum model. The student model helps the lecturer determine which students have problems. This model is also available to the student, letting the student know exactly how he/she is progressing and what is required to improve. The student model is constructed using cognitive modelling techniques used in intelligent tutoring systems. The interpretation of student assignments is the responsibility of the human instructor. This is to avoid the limitations of intelligent tutoring systems, which suffer from inadequate reasoning capabilities, slow development times and a lack of flexibility.
The communications module provides for the actual communication of practical assignments between the lecturer and students. A prototype of this module has been developed and will be discussed in detail later.

**Electronic lecturer/student interaction**

This section describes previous approaches to the submission of practical assignments and some of their problems. This is followed by a discussion of the role of the prototype system within the conceptual framework for handling programming assignments. This prototype specifically addresses the communication problems encountered with large programming classes as far as practical work is concerned, within the framework of the larger lecturer support system. This subsystem also provides the capability to communicate with the individual student, therefore allowing some degree of individualisation of instructional intervention.

**Previous situation**

Traditionally the student code was printed out by the student, submitted to the lecturer and then evaluated by the lecturer or an assistant. Feedback was done by correcting code segments and making suggestions where needed in the code. This was easy to do on paper and worked reasonably well. The problem with this system is that while feedback was relatively easy, it was difficult for the lecturer to find all of the mistakes, especially in larger more complex programs. A popular solution to this method was to use E-mail to send the code to the lecturer. This allowed the lecturer to run the code on his/her computer to find any problems, as well as allowing the student to be situated anywhere.

The problem with using the E-mail system is that it limits the useful feedback that accompanies the code, which could be provided on paper. This makes it difficult for the person who submitted the code to understand what he/she did wrong. An additional problem is that this inhibits the individual from learning from positive criticism about his/her coding technique. There are many electronic submission systems currently available, but they have numerous problems. Most of these systems require specialised client software. This limits the user’s physical location to those machines that have the software installed on them. The advantage of printed paper over the current electronic systems is that it’s easier to scribble comments all over the pages.

Managing printouts can cause problems when there are many students submitting large assignments. Hundreds of pages may be needed and in will take the lecturer hours to print out. Printing costs are also a contributing factor.

It is therefore desirable to create a transparent computer-based student/lecturer feedback system. This system must specifically cater for the needs in the programming environment, mainly to provide for feedback to assignments, and this has to be done using open standards, which make it easy to integrate with other systems, make changes and add extra functionality.
The following section defines Computer-Mediated Communication (CMC) and examines how it can be used in solving the communications problem. This is followed by a discussion of the prototype system, which uses CMC techniques, that facilitates the communication of practical assignments between the student and the lecturer electronically, while improving on the usefulness of feedback in the current system.

**Computer Mediated Communication**

Computer-Mediated Communication (CMC) is the usage of computer networks to facilitate the exchange of information between people. This may be in the form of text, images, audio and video. CMC supports both synchronous and asynchronous communication. Synchronous communication permits information to flow in both directions simultaneously (e.g. telephone call), while asynchronous communication only permits information to flow in one direction at a time (e.g. e-mail).

**Categories of CMC**

Santoro (1995, p. 14) divides CMC into three broad categories. This is done by distinguishing the nature of the human-computer interaction and the role taken by the computer in the mediation process.

The first category is characterised by the computer only acting as a communications medium to support direct human to human communication. This category is referred to as computer based conferencing or simply conferencing. Studies show that his category of CMC incorporates the aspects of written and spoken communication (Santoro, 1995, p. 16). E-mail is a good example of this category since it plays such a passive role, only acting as the medium over which the communication takes place.

The second category is called informatics, and here the computer plays a more active role as the maintainer of organised information. The information is supplied by humans as well as later retrieved by them. Remote databases and online public-access library catalogues (OPAC’s) are examples of such systems. (See figure 4).

The computer plays an even greater role in the third category called computer-assisted instruction (CAI). The computer is responsible for structuring, managing and presenting the information.

**Traits of CMC**

**Asynchronous Nature**

Many of the CMC systems used are of an asynchronous nature (e.g. E-mail). This allows the users to take their time to respond to questions. Therefore, instead of rushing replies well thought out replies can be sent. A problem with this is that the users are not pressed to respond as they would in "normal" synchronous communication.

**Time and Distance Constraints**

One of CMC’s most important benefits is that it liberates instruction from time and distance constraints (Berge and Collins, 1995, p. 3). This aspect of CMC allows students to learn any subject matter from any location irrespective of their time zone or geographic location. This trait also causes a diversity of cultures and approaches to arise (Lewis, Whitaker and Julian, 1995, p. 17).
Personal Development

CMC gives everyone the ability to express their viewpoint and gives the users confidence in their own abilities. It also promotes self-discipline and responsibility in its users. Unfortunately, not all students are capable of adequately managing their own learning and therefore this could act as a disadvantage (Berge and Collins, 1995, p. 5).

Effects of media usage

An aspect that is also very important is the fact that most of CMC's instructional use is text based. This requires the users to be able to express themselves in text, a requirement that many people are not able to fulfill. The problem deepens when one realises that most of the writing is done in English, which might be the user's second or third language. Another trait of the textual nature of many CMC systems is that social queues are lost (Berge and Collins, 1995, p. 4).

Prototype student/lecturer communication system

The principles used to design a new communications system for use by the students and the lecturer will now be discussed. The system must be capable of handling files, since this is easiest for the students and storage purposes. The system also needs to be asynchronous in nature, making the system more adaptable to the students' and lecturers' timetables. The final aspect that must be taken into consideration, is the type of network classroom that will be required, and whether the current infrastructure can support such a network classroom environment. The institution currently has a WAN (Wide Area Network) installed which is connected to the Internet and can easily support a web-based CMC solution (see figure 4).
The reason that a WWW based system was chosen is that it offers asynchronous communication on the existing infrastructure and is more flexible than a pure e-mail based system. A WWW based system also allows for greater content, expandability and easy access. As well as integration opportunities with online teaching materials and internet-based co-operative learning initiatives.

The system consists of four components:
- The evaluation application;
- client application for students;
- web server and
- data repository.

The different components will now be individually discussed.

**Evaluation Application**

The evaluation application must be capable of formatting the code to make it more viewable. This could be as simple as using different colours for keywords, but there is scope for the future use of intelligent agents, which attempt to locate and highlight potential problem areas. There should also be a capability for the lecturer to add comments and hyperlinks to the assignments for the benefit of the student. Finally the evaluation should provide integration and easy access with the student-profiles, both for querying and updating. These capabilities could be implemented using one of two basic methods.

Firstly the evaluation application could be developed using a programming language like VBA (Visual Basic for Applications). VBA allows a software developer to customise and add functionality to all Microsoft Office applications. This drastically reduces development time, and Microsoft Word is a perfect application on which the evaluation application can be based, since it already has most of the needed functionality built into it. An additional advantage of using such an approach is that it makes it easy to link office applications with one another e.g. connect a Microsoft Access database with a Microsoft Excel spreadsheet. A disadvantage of this approach is that it requires Microsoft Office to be installed on the computer on which the evaluation will take place.

The second approach would be to develop the application from scratch. This would ensure that no unnecessary code is present and would not be limited to the capabilities of VBA and Microsoft Office. This would, however, also require all functionality to be developed with no recourse to the capabilities already present in Microsoft Office.

It was decided to implement the initial prototype of the application using VBA and Microsoft Word. This allowed a functional prototype to be developed using native capabilities in Microsoft Word. This prototype has all of the main features required of the evaluation application. Additionally, it is capable of exporting evaluated assignments to HTML format, including hyperlinks and comments to the assignments.

**Viewing Application**

The students use the viewing application to view the assignments after evaluation. A Java capable web browser can act as this application since the evaluated assignments are made available to the students in HTML format. This has the advantage that no additional software is required on the students' computers and additionally allows the students to access the communications system from any WWW enabled computer via the Internet.
Web Server

The web server handles all of the web browsers' transactions. This is done using cgi or asp scripts, especially when generating dynamic HTML pages. All of the communications to and from the web server is secured by using the SSL (secure sockets layer) protocol.

Data repository

A database is used to store student profiles, practical assignments as well as any information pertaining to them. An advantage of storing all of the assignments in a database is that it enables the lecturer to have instant access to all of the assignments. This enables the lecturer to pick up trends, and give more informed feedback during the evaluation process. The students also benefit here because the database acts as a repository for them as well, making it impossible for them to lose old assignments.

A disadvantage of using this system is that it firstly requires a machine for the database to run on and secondly, that the database will grow fairly large. However, even though it might become large it should not have many concurrent users therefore the performance should not degrade too much. The reason why the assignments are stored within the database instead of storing pointers to external files within directories, is that a directory based solution relies on the operating system for maintaining security, while security can be enforced via the database irrelevant of which operating system it is running on.

The system's appearance

The system is at the moment in prototype phase, but will appear to users as a web based e-mail system, however it will not make use of e-mail. The reason for this is that all of the assignments are permanently stored in a relational database instead of an e-mail server. An advantage of using this web-based e-mail like interface is that it is a familiar paradigm making it easier for the users to use.

Student perspective

This section will discuss the steps that the student must go through to submit an assignment. The student initiates the process by typing in the full URL (Uniform Resource Locator) of the evaluation system within his or her web browser (e.g. www.netech.ac.za/studentevaluation.html). The opening screen will ask for information to authenticate the student, for example the student's student number as well as a password. Once authentication has taken place a new page is displayed which lists all of the students evaluated practical assignments. In addition to this the student is given the option to upload his or her newly completed assignment.

If the student chooses the upload option a new page will be displayed. The page should consist of a text box area in which the student can type any text comment as well as browse, submit and cancel buttons. The browse button will be used to select the assignment that the student wants to send. The submission button will send the assignment and message to the database. Once the assignment has been uploaded the server generates a unique token of receipt which is sent to the student. This unique token can be used as proof of submitting the assignment, preventing repudiation at a later date.

If the student chooses to view already reviewed assignments they can do so on the second screen. This screen provides a list all of the student's reviewed assignments in addition to a submission option for new assignments. The student must simply click on one of the assignments listed and a new screen will be displayed. On this screen the browse will display any comment associated with the assignment as well as giving the user an option of downloading.
Lecturers' perspective

The lecturer also connects to the system using a web browser. Once the lecturer has been authenticated the second screen is once again displayed. This second screen does however, differ from the student's in that it displays all of the assignments which the students have submitted. The lecturer can then retrieve any of the listed assignments by following the same steps that the students must.

The system automatically knows to whom the evaluated assignments belong and therefore the lecturer follows the student submission procedure when returning the evaluated assignment. However for cases where the system is unavailable to the students, and they have to physically give their assignments to the lecturer on disk, the lecturer can specify to whom the assignment belongs. This is accomplished by the use of a student number field on the submission form.

Technical perspective

The initial screen is a static-HTML page, which can be linked to a cgi or asp script. These scripts make use of the submitted information to authenticate the user by comparing it to the student information within a database. The second screen is a dynamic HTML page that is generated by the scripts using information extracted from the database. Submission makes use of the web browser's ability to retrieve the needed files on the local machine as well as encoding them, while cgi or asp scripts on the web server handle the decoding of the file(s) and storage into a database. Before the files are stored one of two things happen. Either the student’s student number is added to the filenames, or the student number is added to the file as the first line. Once the file(s) have been stored within the database public key cryptography is used to generate a unique token, which is, in turn, sent to the web browser. The web browser will then prompt the user for a location to save it.

If a student clicks on one of the assignments listed on the second screen, the scripts will request the appropriate information about that assignment from the database. The cgi or asp scripts which are located on the web server then generate a new HTML page which contains any comments from the lecturer as well as "links" to all of the evaluated assignments. At the same time a field within the database is set which indicates that the user has read that specific assignment. If the student now clicks on a "link" the scripts request the specific files from the database. The files are then encoded on the web server and sent to the web browser, which in turn decodes the files and saves them to disk.

The only difference between what happens when a lecturer submits an evaluated assignment is that the student’s student number is used by the system to associate the assignment with the student within the database. The number is obtained by either being filled in by the lecturer, or by looking at the filename or first line of the assignment.

Conclusion

The CMC system improves communication and allows individualisation of instructional intervention, but with the large numbers of students it will be difficult to provide meaningful feedback based on the student's overall progress, rather than simply providing feedback on immediate errors. It is proposed that cognitive modelling techniques, which are used in intelligent tutoring systems, can be utilised to gain a more accurate picture of the development of each individual student. The limited capabilities of artificial intelligence techniques utilised in intelligent tutoring systems limit the capabilities of these systems, therefore the proposed solution makes the interpretation of student assignments the responsibility of the human instructor. The choice of instruction is also the responsibility of the instructor. The computer system serves as the communication medium, as well as a storage system and analysis tool of student development. The system
provides the instructor with the required detail regarding a student's cognitive development over time and provides a comparison with the intended development.

References


