Journal of Information Technology Education Volume 1 No. 2, 2002

Empowerment in IT Education

R.J. Dawson and I.A. Newman
Loughborough University, Loughborough, UK

R.J.Dawson@lboro.ac.uk I.A.Newman@lboro.ac.uk

Executive Summary

Educating undergraduate students to be capable IT professionals poses several significant problems. Firstly, there is the question of what knowledge should be taught in a subject where new areas of interest can appear, flourish and then largely disappear within a few years. Secondly there is the question of how the subject should be taught to try to ensure that the students not only acquire knowledge which enables them to pass their University examinations but can put that knowledge into practical use in their subsequent careers.

This paper presents a method that has been used by the authors successfully over a number of years to supplement conventional lecture and laboratory classes. The approach is designed to convince the students that they are capable of working in groups on problems that they have not encountered previously. The students are asked to tackle open ended problems which, typically, require the students to both apply some of the knowledge that they have been taught and acquire new knowledge for themselves. The tasks are designed to allow students at different levels of ability to succeed in achieving something worthwhile while also ensuring that even the most able and well motivated students will still be left with unfulfilled challenges.

The paper first introduces the concept of empowerment, explaining why it is important in the context of preparing students for working with IT. The particular importance of the need to prepare students for continual changes in the subject is stressed and the consequential requirement to develop a range of skills which will promote and encourage 'lifelong learning' is explained.

The methods that have been used by the authors to empower the students are then explained. The techniques that are used to get the students to reflect on what is happening and thus to internalise the experience are also presented. The ideas are amplified using four case studies. Two of these directly relate to undergraduate education, one for students studying computer science as at least half of their degree, the other for students for whom computer science comprises only about one fifth of their curriculum content. A third case study explains how the same ideas were successfully used in an industry setting to help empower students who had not had this type of experience in their degree course. The final case study is used to show that the empowerment technique can be used successfully at other educational levels and that IT based tasks provide an excellent motivational environment for a very wide range of people. In each case study, evidence is offered to show that the approach has been effective in empowering the students and has had beneficial effects from the viewpoint of outside observers.

The case studies are used to illustrate how the approach being advocated draws on, yet also differs from, the interpretivist approach to education. The paper concludes by summarising the essential features of the approach and commending it as a worthwhile and enjoyable approach for both staff and students.

Editor: Linda Knight

Keywords: empowerment, IT skills undergraduate education,

Introduction

This paper addresses a subject which, at first sight, may not seem directly relevant to IT education. The first part of the paper, therefore, explains what we mean by empowerment and why we think it is highly relevant, worthwhile and fun! The remainder of the paper shows how students can be empowered with examples drawn from our combined teaching experiences over many years (Dawson 2000; Dawson and Newsham, 1997a & b; Dawson, Newsham and Fernley, 1997; Jackson and Dawson, 1999; Newman, Dawson and Parks 2000; Newman, 2001a & b; Newsham and Dawson, 1996, Parish and Newman, 1999).

Most of our experience relates to university students undertaking programmes with a significant IT content, so this provides the main focus for the paper. However, we also have experience of teaching, and using IT to empower, University students for whom IT is not a major element of their degree and also people in industry and children in schools.

This paper presents three related messages:

- ‘Empowerment’ is as important an issue in education as acquiring ‘knowledge’, although less time is devoted to it in the curriculum;
- IT provides an ideal environment in which to promote empowerment;
- Empowering students provides a means of ‘protecting’ them against the rapid changes that take place in IT.

Background: What is Empowerment and Why is it Important?

This section explains what we mean by ‘empowerment’ and why we think it should be a focus for IT related teaching. We argue that empowerment is at least as important as knowledge acquisition and that IT is an ideal vehicle to empower people studying a variety of subjects at different educational levels.

A dictionary definition taken from a Pocket Oxford Dictionary originally published in 1947 is: “Empower: to authorize or enable; empowerment: the act of enabling”. Definitions in other dictionaries are similar and have not varied significantly over time. In our case we are enabling students to cope with the unexpected in an IT context, a skill that seems to be desirable (Dyba, 2000; Sandburg and Vinberg, 2000).

The relative stability of the dictionary definition, contrasts sharply with rapidly changing experiences of IT. This creates a problem for IT teachers and is the main reason that we are championing the empowerment approach. Basically, because of the rapid changes that are taking place in all IT related subjects, we believe that the most useful attribute we can give students is the confidence to find their own solutions to any given IT problem. The pragmatic value of promoting this idea was first recognized when it was observed that students undertaking an intern year in industry in the middle of their computing degree programmes were frequently being asked to carry out tasks for which they had not specifically been educated. For example, interns were expected to be proficient in a range of programming languages other than the languages they had been taught. However, it was simply not possible to add extra teaching to cover all the programming languages and other computing topics that they might encounter in industry. Yet, despite this, we found that, according to the employers, our interns were, in the main, rising to the challenge. The employers also indicated that our students were more capable, in this respect, than those from some of the other 'computing' degree programmes. Questioning the employers and the students to try to discover the reasons for this suggested that at least part of the reason was a second year project based course (module). In this module the students were set tasks which involved them writing a program in a language they had not been taught and working as part of a group to accomplish an open ended task. In both cases they were expected to find out the information they required for themselves and were, therefore, neither surprised nor worried when they were expected to do this again in their intern year.

This strongly suggested that open ended exercises which required the students to learn and apply new things were a good idea. Taken with the observation that most commercial and industrial situations involved groups of people working on a project, it also suggested that group work needed to be encouraged. These ideas were reinforced by the observation that many IT related problems were highly practical and that theoretical knowledge, as such, was often of very little help in a practical situation (Dawson and Newsham, 1996). It was, therefore, decided to endeavour to develop the students' skills and confidence so that they could acquire the necessary expertise when they needed it.

The aim was to develop the following 'attributes' for a competent professional IT practitioner:

1. the ability to learn about something new from existing information;
2. the confidence to try out new things and to learn from the experience;
3. the skills to find and recognize possible solutions to problems;
4. the confidence to be able to select and execute a sensible course of action;
5. the ability to explain and justify what has been done.
6. the ability to work as an effective member of a team.

It is our contention that empowerment, which is the acquisition of these six attributes, is the most effective assistance we can give students to prepare them for their subsequent careers.

The term 'empowerment' has been used by at least one other organisation (Psoinos, Kern, Smithson, 2000) but this was specifically concerned with the benefits that accrued from providing staff with appropriate computer based support tools, a subject that has been addressed by several other authors (e.g. Callahan and Khatsuriya, 1998; Joshi and Rui, 2000). Our use of the term does include theirs, as one of the case studies will illustrate, but is considerably more far reaching.

As indicated by skills 1 and 2 in the list above, the concept of empowerment carries with it the requirement for experiential learning. This has been widely discussed in the literature (e.g. Armarego, Fowler & Roy, 2001; Brophy & Alleman, 1991; Brown, 1994; Callahan & Khatsuriya, 1998; Friedman & Kahn, 1994; Kolb, 1984). However, the focus for empowerment is less on the knowledge that is gained by the experience and more on the ability to gain knowledge. Nevertheless, because it is necessary for the students to gain knowledge, and understanding, and to recognise that they have done so, there must be an experiential component that relates to subjects that the students are also studying by other means. In order to reinforce the empowerment concept we have found it most effective to choose subjects (topics) within the degree programme that most students find difficult. The intention is to demonstrate to the students the close relationship between 'doing' and 'learning' in IT.

The Problem: Preparing Students for Working with IT

The problems encountered in preparing students for the wide range of software, hardware, concepts and approaches in industry are universal in IT education because of the rapidly changing nature of the subject. This makes it difficult for lecturers to teach the subject in a conventional way if the aim is to prepare students for a career involving IT. We suggest that the need for IT related skills is increasingly being seen as important and that this too poses problems for conventional teaching approaches.

Having a good knowledge of a subject, and being able to impart that knowledge, are considered important aspects of being a good teacher. It is, therefore, taken for granted that there is a body of knowledge that needs to be learned, with the learning being assessed in some way. This is particularly true at the university level where most of the assessment process is devoted to testing knowledge and understanding.

Thus, IT educators tend to focus on the knowledge that needs to be acquired by students and published syllabuses tend to concentrate on the units of knowledge to be taught. Also the definition of a profession is usually related to the knowledge that members are expected to have before they can be accepted as adequately qualified. This is certainly true in the well established professions such as medicine, law and accountancy which all demand the achievement of an acceptable academic standard, demonstrated by passing examinations, as part of their acceptance criteria. Closer to IT, software engineers have been discussing what is needed to make their discipline into a profession for some years (Ford and Gibbs, 1996; Bagert and Mead, 2001; Thompson and Edwards, 2001), and are attempting to identify the underlying 'Body of Knowledge' required (SWEBOK, 2001).

However, IT differs from conventional subjects in its rapidly changing nature.

Firstly, there is the continuously changing technology. Many topics seen as important little more than a decade ago have become irrelevant because of the changes in hardware and software. For example, sophisticated algorithms to manage memory and disc storage space were routinely taught to computer science undergraduates when computers offered 640K bytes of memory and 100Mbytes of disc. Now it is usually much cheaper to buy additional Megabytes of memory and Gigabytes of disc than employ a programmer to write code to improve memory management.

Secondly there are the changes in the terminology. New jargon words and phrases are created, come into widespread use, but then disappear. What is worse, new phrases are regularly coined for the same underlying concept (e.g. Management Information System; Executive Information System; Decision Support System).

These changes frequently drive student expectations of what should be taught, reflecting the latest fashion in the IT industry. For example, student demand for the programming language they are first taught has moved from FORTRAN to Pascal to C to C++ and now Java (Cowling 2001). By comparison with other disciplines this is really very unusual.

The problem facing the conventional lecture based approach is further complicated by the increasing recognition that the acquisition of skills is also an important component of IT education. As an example, the UK Quality Assurance Agency (QAA) published guidelines to be used to assess university computing courses. These contain three skill sections summarized below (Computing 1999):

- Computing related cognitive abilities: the abilities to apply computing knowledge when identifying, modeling, designing, building and evaluating systems and to reflect on experiences and on the professional, moral and ethical issues involved.
- Computing related practical abilities: the abilities to specify, design, construct and evaluate computer-based systems; assess alternatives, evaluate risks and use appropriate tools.
- Transferable skills: the abilities to retrieve and use information, work in a team and develop personal learning capabilities.

Methodologies for Teaching Empowerment

If knowledge changes quickly and provable skills are also required there is clearly a need to find a way of supplementing conventional teaching. The traditional method of skills training was to apprentice a student to a 'master'. Engineers, carpenters and plumbers were all trained through apprenticeship schemes, and accountants and lawyers had similar arrangements. However, apprenticeship schemes have largely been discontinued as they were seen as expensive and incompatible with the rapid rate of change in society. Since one of the main catalysts for the rapid change is IT, this apparently makes the apprenticeship system unworkable as a method for skills and knowledge training in IT.

A possible alternative is the closely related idea of learning by experience. Experiential learning is not new (Kolb, 1984). However, considering the extent of the knowledge and skills needed and the very limited time the students have, it seems infeasible to expect students to acquire all they need by random experimentation.

Empowerment, the strategy that we have adopted, utilizes experiential learning. However it focuses the students so that they get the benefits of learning by experience but are monitored and directed to avoid wasting a lot of time in unproductive experimentation. In this way lecturers can pass on their knowledge and skills to the students in the style of the master and apprentice but considerably faster as befits the pace of the modern world. However, as will be discussed later, it is always necessary to give the students sufficient time to allow them to have the chance of making mistakes and to reflect on what they have experienced.

This empowerment is achieved by creating a project environment that forces the students to:

- apply some of the knowledge they have learned in ‘real’ situations;
- discover that real projects are not well ordered but are ill-defined and subject to change, making it impossible to identify all requirements initially;
- cope with the 'impossible' tasks of changing requirements and customer expectations;
- work with others in situations where the synergy of the team is as important as individual effort;
- recognize that success in a project is about satisfying people;
- think for themselves, to reflect on what happens and improve performance.

From this experience the students should learn to believe in themselves and realize that they can do things that they have not been taught.

To create this environment, students are required to tackle a real problem that they can relate to, using technology that is perceived by them as 'cutting edge'. This motivates them to take an active part in their own learning experience. As an example, at the time of writing, 'the Internet' represents the cutting edge. This means that, if the students carry out a suitable project successfully, as illustrated in the first case study, they will feel they are in a good position for getting a job or for setting up in business on their own. The student experience is further enhanced by providing a simulated ‘real world’ environment for the project where requirements are ill formed and tend to grow, where customers do not know what they want and change their minds regularly, and where the hardware and software of development platform do not remain stable.

The task the students undertake needs to be sufficiently challenging to motivate them, but not so difficult that they feel it is impossible and do not even start. Selecting a suitable task represents a considerable challenge where there are large numbers of students and their abilities vary significantly. While group working can reduce the apparent disparities it does not eliminate them and a substantial variation between the ‘best’ and ‘worst’ groups is still possible. To overcome this problem we use a generic task and allow each group to negotiate the deliverables. Making the students determine the scope of the task they undertake means that they cannot be assessed on ‘absolute’ achievement. Instead, assessment is based on getting them to report what they did, to **reflect** on the process and to report the results of that reflection. A group is typically assessed on their answers to four questions:

- What were the requirements and how well are these met by what you produced?
- What more could you have done?

- What should you have done differently and why?
- How would you do things differently in the future?

Each individual is also asked to report on the way the group worked and the contributions each individual group member made to the group. The individual is assessed on the quality and thoughtfulness of their report. The students are encouraged to work as a team, but occasionally, some groups agree to ask that an individual be marked down for failing to make an adequate contribution. On one occasion, reported in the first case study, one group actually 'fired' one of its members for failing to do enough work.

As part of the task the students are expected to apply some of the theory they have been taught. In most cases they discover they did not fully understand what they had been told because they cannot put it into practice in the given situation. To help improve their understanding, intermediate deliverables are requested which relate to particular taught areas (e.g. Entity Relation diagrams; Use Cases). These are marked and feedback is given on the misunderstandings and errors made. They can then be given the opportunity of reviewing their work and submitting a revised version as part of the final deliverable. The concept that failure is a very effective teacher is an important part of our approach to empowerment. However, it is essential to ensure that overall students do not perceive themselves as failing. The task of the lecturer is to re-motivate them and make sure that they do succeed.

The Case Studies

This section of the paper uses case studies to help illustrate the ideas presented above. Two of the studies focus on students at University and a third on post University Industrial training. To illustrate the wider applicability of the empowerment techniques, there is also a brief discussion, of the use of the technique to help 13 and 14 year old educational under-achievers in a secondary school. The case studies also show that empowerment can be combined with a variety of more focused educational goals. In each case, the background is first given, the approach is described and the outcomes are explained. Each of the scenarios is drawn directly from a real situation in which one, or both, of the authors participated.

The 'Educational Objectives' section of the examples below identifies the skills that were being targeted using the skills classification listed above. It also indicates the knowledge areas that were intended to be reinforced. To maximize the effectiveness of the process, the students are usually given some initial knowledge, and also some criteria for making decisions about which knowledge areas are likely to be relevant. The progress made by the students is monitored and additional instruction is given where there are indications that individual groups or the whole class are either not making adequate progress or are moving in an undesired direction. On other occasions, rapid progress made by the students has caused a change in the staff perception of reachable outcomes. The aim is to maximize the effectiveness of the learning experience for the students rather than to 'hit' all of the specified targets. It is assumed that the students will undertake a number of different empowerment exercises during their course and that if particular 'optional' objectives are not achieved in one exercise they can, if required, be made mandatory in a subsequent exercise.

A Large Group with a variety of Different Backgrounds and Experiences

Background

This case study involves more than 150 undergraduate students undertaking a module in the second semester of the second year of their degree programme. This module constituted one twelfth of the years' study for all of the students. More than a third of the group was taking a Computer Science degree while the remainder was taking 'joint honours' programmes in either 'Computing and Management' or 'Information and Computing'. About half of the students were planning to undertake an 'Industrial Training' (intern) year before starting on the final year of their degree.

Although all of the students taking this module had taken other common modules both in the first year of their degree programme and in the first semester of the second year, there appeared to have been relatively little mixing between the students following different programmes.

Educational objectives for the module

As discussed above, the overall objective is to improve the ability of the students to work effectively in groups and to take responsibility for managing an ill-defined problem to a successful conclusion, gaining new knowledge as, and when, they need it i.e. to empower them ready for the experiences that they will encounter when they leave University both for their intern year and, later, at the end of their programme. These, however, are not easily measurable objectives. More specific objectives must also be set. For each requirement, an indication is given of the method used to measure the success, or otherwise, of the student groups.

Requirements negotiation and capture in a 'fuzzy' environment (Computer related cognitive skill). The student groups were given an incomplete specification and asked to negotiate with their clients (the course team) to agree what the actual requirements were for that group. The basic problem was common to all of the groups but the detail was open to negotiation and their success was measured by the scope of what they attempted and the degree to which their achievements matched the agreed requirements.

Project management (Computer related cognitive skill), including the planning of the task, subdividing it into smaller tasks (sections), allocating the sections to members of the group, identifying decision points, quantifying potential risks and documenting the process. Every group was required to use software which had been developed in-house to produce and maintain a project management website (PMWS) which was inspected on a weekly basis from week 4 of the 11 week project. What was claimed on the website was validated against their final deliverables which contained reports analyzing one of the risks they had considered and reflections on the process that they had followed.

Use of software and documentation of which they had no prior knowledge and which was subject to change (Computer related practical skill). The in-house system was used to ensure that no student could previously have had experience. New versions of the software, with additional features were released three times during the project. This software was to be used to produce the PMWS and also as a storyboarding tool to demonstrate the proposed functionality of the system they designed. The quality of the sites they produced and the way in which they used the software were the assessment criteria in this case. The work they had done was also compared with the objectives that they set themselves and with the comments that they made in other reports. One group chose to assess the risk of not using the final version of the in-house software in their product and obtained an excellent mark.

Reflection (Computer related cognitive skill). The students were required to think about the process of developing a demonstrator system and of reporting progress via a project management website. The groups were asked to produce a final report which reviewed what they had done, what had worked and what could be improved. Each individual was also required to produce a report assessing how the individuals in the group had contributed to the overall result.

Understanding of theory and technology previously taught (knowledge area plus Computer related practical skills). The problem specified allowed the students to put into practice and improve their understanding of subject material taught in modules they had previously taken. As the quality of the product produced was part of the assessment, this improved understanding was also assessed. For example, the design and implementation of the database system, the working of a dynamic web site on different hardware and browser platforms and the use of a UNIX system of which most had had little experience was all part of the requirement specified and all became part of the assessment.

Overall there was also a desire to try to get the students from different degree programmes to mix more and to recognise that, in real situations, it was necessary to draw on a variety of different skills. It was also intended that they should recognise the benefits which accrue from group working.

Approach

The task which the students were set was to:

- Form teams of from 5 to 7 students;
- Act as potential sub-contractors to a large company, designing a demonstrator system for two complementary 'products' using pre-specified software:
  - a support system for IT help desk staff dealing with queries about hardware and software problems from employees of an organisation;
  - a self help enquiry system which would let users with problems look to see if their problem has already been solved;
- Produce and maintain a project management website (PMWS) using pre-specified software, the website would be inspected weekly from week 4;
- Design and implement, using Microsoft Access, a database for the information required to provide the two services above, produce a report describing the design, with ER diagrams and data table formats and demonstrate the database in action (deliverables due in week 6);
- Storyboard the interface to the proposed systems at the demonstration in week 6 and produce a report describing the interfaces (deliverable due in week 6) This allowed us to compare their database designs with their system designs.;
- Make a proposal, in the form of a report, for the enhancements they would propose for the system that they would deliver in week 11 (deliverable due in week 6);
- Demonstrate the proposed web based interface to their systems and provide a report describing their system and assessing how it met the objectives (deliverables due in week 11);
- Show how the group had worked using the PMWS at the demonstration in week 11 and via a report assessing the operation of the group.

The course was scheduled to have one contact hour per week which was used to answer questions and to make general announcements, including giving notice of any changes in requirements that applied to all of the groups. The formal academic requirements, in terms of the timing and contents of the reports and demonstrations were not changed, but the scenario was altered, corresponding to the sorts of changes that occur naturally in most real projects. The groups could also contact staff via e-mail and arrange face to face meetings if they wished. About half of the groups availed themselves of the latter opportunity and most successfully negotiated variations in the requirements to suit their perceived capabilities.

This format allowed us to simulate some real world constraints, such as delivering the local software late, promising features that were made available eventually but too late to be used, changing some of the details of the requirements, asking them to document what had been agreed but not always being available when they tried to contact us. The students were given a substantial amount of feedback: in the meetings arranged by the groups; in lectures; by e-mails to the whole cohort; via the feedback system in the PMWS; via e-mail. They could expect a reply within a day or two, provided that they phrased their questions appropriately by giving us options for possible answers. They also received feedback verbally during the first demonstration and in writing, via e-mail, with a provisional mark that they were told could be improved if they rectified the problems we had identified. This option was taken up by several of the groups.

Outcome

Several of the objectives were demonstrably met for most students. For example, 18 of the 26 groups were ‘mixed’, containing members from at least two of the constituent programmes and their reports indicated that they had gained from working in these mixed groups. Three of the single programme groups suggested either verbally or in their reports that they might have performed better if they had had additional skills. Comments were typically made in the reports about the different skill sets which different people brought to the problem.

Requirements capture: Virtually all of the groups in their reports commented on the difficulties of requirements capture and claimed that they had begun to understand some of the complexities of real world requirements elicitation.

Project management: all of the groups successfully built and used a PMWS to display what they had been doing.

Use of new software: as noted in the previous section, they all managed to use the software for the PMWS and all but one of the groups used it as the basis for prototyping the look and feel of the final systems.

Knowledge/skill gain: obviously, the existence of web based systems running on Unix, indicated that at least some members of every group had gained relevant knowledge and practical skills. The fact that most of the groups (16 of the 26) also failed to produce properly documented designs for the database as part of deliverable 1 but then ten of them resubmitted designs that were a substantial improvement with deliverable 2 suggests that the objective of reinforcing database knowledge and experience was also largely met.

Overall: All but two of the groups felt they succeeded. Even these two groups obtained a pass grade for the module. One of the two was a ‘forced group consisting of four people who would, or could, not join a self-selected group and they were probably content to just pass. The other was a group which became demotivated when a member left the University after four weeks. By contrast, 19 of the groups included in their ‘reflection’ comments which suggested that the module had been one of the most challenging and interesting they had attempted.

The winning 6 person group went on to win an IBM sponsored inter-university competition to which their success gained them entry. It is worth noting that this group did not include any of the top ten students in any of the degree programmes indicating that ‘groupwork’ skills triumphed over individual excellence. A discussion of the student experiences of the project which included extracts from the comments that appeared in their reports was presented at a conference (Newman, Dawson and Parks, 2000).

Commercial Example

Background

This case study involves graduates of computer science courses in their first months of employment at the Plessey Telecommunications Company (now part of GEC-Siemans Communications) in the 1980s and 1990s. The company trained groups of up to 15 newly recruited software engineers in a two week full time in-house training course known as the “Software Life Cycle” course (Dawson and Newsham, 1997b). The purpose of the course was to “knock the graduates into shape” and was developed because managers had noted that their new graduate recruits, although often very knowledgeable in the subject, lacked the practical skills to handle real problems in a real working environment.

Educational objectives

Many of the educational objectives were similar to those of the first case study. The graduates needed to learn the skills of working effectively in a team, to learn how to deal with inadequate and ambiguous specifications, to work with procedures and constraints with which they were not familiar, to cope with change and above all to learn how to learn from their experiences. When the courses were first started it was found that few graduates had any experience of attempting to use these skills, though by the 1990s the majority of graduates were at least familiar with team working and many had experienced "real world" project conditions.

The company course aimed to give essential experience of:

Team working: Where participants had to organize their own team structures and communication.

Project management: Where as well as organizing themselves participants had to conform to processes and procedures laid down by a member of staff acting the part of their manager.

Inadequate requirements: Where the teams had to discover what had not been specified, to suggest alternative solutions and negotiate what they would eventually deliver.

Changing requirements: Where new or changed requirements were given on a daily basis.

Changing working conditions: Where the procedures to be followed or the development platform also changed on a regular basis.

Use of newly acquired knowledge: Where the software of the development platform involved using a language learned on a previous short course only a few weeks beforehand and which the graduates had had no opportunity of using in an exercise of any size.

Reflection: Where all teams had to identify and present what they had learned from the course.

There was no formal assessment of the teams or individuals but at the end of the course the teams had to demonstrate what they had produced and give a presentation on what they had learned to the managers of the departments they were due to work in after the course. Feedback on their progress based on observations by the training staff was also given to the students and managers at the end of the course.

Approach

The graduates were given a half day introduction informing them of the realities of the real world, warning them of the problems and where projects usually go wrong. They were then organized into teams of 3 to 5 members and given a project to undertake over the next 9 working days. All teams worked on the same project which was to develop a piece of software it was claimed the company needed. The teams worked during company hours with some supervision at all times. The training staff used role play to provide experience of managers, customers, quality inspectors and technical advisors throughout the project. As in the real world, customers were presented as having different ideas and different understanding that would change as the course progressed.

Feedback was given in the first instance after two days of the project when each team had to meet their "manager" to present in detail what solution they were proposing to develop. Cross examination would inevitably show that their solutions were not as well thought out as they believed and that far too many assumptions had been made. Subsequent meetings would review progress and allow for changes to be negotiated. Short notice meetings with the 'Quality Inspector' were used to highlight any failings in their working standards and processes. Finally the hardware and software development platform was ‘altered’ from time to time to simulate the hazards of the real working environment.

The last half day of the course consisted of a “wash up” session for which the graduates' real managers were invited. The teams demonstrated their software and then gave presentations of how well they had coped on the project, highlighting what they had learned. The course training staff also gave their review of the projects, indicating the good and bad points of the projects and teams.

Outcome

The course was well appreciated by both the graduates and their managers. The software produced as the project deliverables was rarely fully working as planned by the teams. For many of the graduates it was often quite a shock to find out how little they were prepared for the real working environment. Comments such as “You go into a meeting thinking you know everything and come out realizing you know nothing” were not uncommon. Nevertheless they all felt they had learned from the experience. For many, the importance of questioning everything they were given and, just as importantly, questioning their own assumptions was the key lesson and all learned to be more realistic in their expectations. The course certainly succeeded as feedback showed they would be better prepared for such a project next time – this was particularly appreciated by their managers as the next project would be in the real working environment!

Although there was no formal assessment the review session, and the fact they were presenting to managers, meant that all participants took the task of reflecting on what they had achieved very seriously. The fact that the teams worked alongside each other in the laboratory and could observe the other teams’ experiences as well as their own helped promote the learning experience. The trainer’s review at the end would reinforce this as everyone could relate to comments made about other teams as well as their own. The trainers’ comments on how the teams had managed themselves, how they had worked together and how they had coped with the problems imposed helped the students learn these “soft” aspects of their course as well as the “hard” facts of which project deliverable worked the best.

The company was able to observe a definite improvement in their new graduate recruits in the 1990s as more and more universities made teamwork part of their degree programmes and attempted to provide some element of real world working. Nevertheless the company course was still appreciated by managers who continued to send their new recruits to the course every year, and it was not until a major company restructuring removed the in-house software training facility that the Life Cycle course came to an end.

The Systems Engineering Programme Modules

Background

In 1991 the University was approached by a major international defence contractor who wished to recruit graduates with 'systems' skills. Their concept was of students gaining a broad knowledge across several disciplines and being able to provide the 'glue' which bound together project teams containing mainly specialists in a single discipline. The University agreed to create a suitable course and passed on the responsibility for designing it to a group of academics drawn from the five 'main' subjects which would be covered in the course (Electronic, Mechanical and Aeronautical Engineering; Computer Science and Ergonomics). For economic reasons the students would be expected to take existing modules alongside specialist students in these subjects and would also take mathematics, materials science and management modules with other engineering students. The main question for the design team was to provide the 'systems' content which would comprise about one sixth of the degree for the first three academic years (there would also be an intern year in the middle of the programme and a final academic year in which the students would 'specialise' in one of the main subjects). The first intake of 38 students started in October 1992 and 33 students graduated in July 1997 (Parish and Newman, 1999).

Educational Objectives

The systems modules needed to introduce the students to the concepts and applications of the ‘systems process’ and ‘systems thinking’ (knowledge and skill). It was also decided that these modules needed to provide an overall framework for the degree programme which motivated the students, gave them a sense of identity and overcame the diversity, lack of coherence, and absence of explicit systems content in the other modules. The perceived danger was that the high workload and broad spread of topics being covered would lead to students becoming demotivated because they would see themselves as ‘second class citizens’ in each of the specialist subjects.

More specific objectives included:

- the development of teamworking skills (empathy, communication, delegation, cooperation);
- the elicitation of requirements in design and development situations;
- problem solving skills (brain storming, analysis and synthesis);
- confidence in tackling new subjects (breadth) and in understanding details (depth);
- the ability to see problems in context (holism).

Approach

In the first two academic years, the systems modules provide a combination of taught ‘theory’ (systems processes and systems thinking and tools to support systems activities) and practical applications of the theory undertaken in small groups. The practical work mainly consists of open ended tasks which the students are given a relatively short time (2-6 elapsed weeks) to research. At the end of the period every group must prepare and give a presentation. Throughout there is a strong focus on reflection:

- how did our presentation go?
- what can be learned from the performance of other groups?
- what processes did we use and how could they be improved?
- would other methods of presentation be more effective?
- how can we get a better result with no more effort?

In addition, there are a few tasks which seek to demonstrate to the students that it s possible to acquire new skills quickly when the need arises. Computing examples are particularly effective for this purpose since most of the students can see the potential, marketable, value of the skills even if they do not believe that they can ever acquire them. As an example, they have been asked to learn to use an operating system (UNIX) which they had not used before, modify a badly written C program to add new features and produce a website from scratch, all in four one hour sessions in a computer laboratory. To their surprise almost all of them find that they can do it!

In the third year they are expected to tackle a single problem and see it through from requirements elicitation to a prototype ‘product’ which they must present and demonstrate to senior executives from their sponsoring companies as well as to academics. They are also expected to write up what they have done, reflecting on the process, and the outcome, and providing documentation so that the project can be continued in subsequent years.

Outcome

Five cohorts have now completed the course and results have been excellent, not just in the systems modules themselves, but in the specialist subject modules. The ‘systems’ students regularly outperform on average, specialist students taking the same modules. The students are well motivated and, when asked, usually explain it by saying that the systems students tend to support one another much more than is the case with students on other degree programmes. The industrial sponsors have also been pleased with the outcome. However, perhaps the most convincing result has been the number of firms who are now actively seeking to sponsor systems students on, and recruit the graduates from, the programme.

One to One in a School

Background

This particular example relates to a time when computers were not readily available in secondary schools (ages 11 to 16, grades 7 to 11) in the UK. Some computers were provided for teaching IT, but these were mainly reserved for the more able students who were expected to take external examinations in the subject. Using a computer was, in part, treated like a reward which was focussed on the 'best' students. Students who were not academically able, or who had behavioural problems, were very unlikely to be allowed access. This case study examines the results of providing six pupils, all of whom were under-achievers who had behavioural and attendance problems, with the opportunity to use IT to enhance their learning experiences. The pupils were supported by a teacher in whom they had already established a degree of trust (the pupils believed that she 'was on their side').

Educational Objectives

Encouraging them to produce work of their own using a computer as support for the English curriculum; improving self-image; reducing disruptive behaviour and absences. In this case the skills that they might be expected to gain were mainly the ‘other related IT skills’: keyboarding and familiarity with word processing and drawing packages.

Approach

Each pupil in turn was allowed three or four sessions of about one hour using a MAC Plus computer, with one to one help available to show them how to use a word processor and a drawing package.

Outcome

All six of them successfully produced a piece of work of their own. In every case the essay that they wrote was more extensive than they had ever produced previously, and in three of the cases it was much more imaginative than any of their teachers expected. During the weeks when the sessions were taking place both attendance and behaviour were significantly improved. The status of these children improved amongst their peers, and pupils who had previously looked down on them because they were in a class of under-achievers volunteered to join the group. The sessions took place in the spring and the benefits, in terms of motivation, attendance and behaviour, appeared to last, for all but one of the pupils, until the summer holidays. Even the ‘failure’ was generally regarded by other teachers as having been much happier and less prone to disruptive behaviour for several weeks after he completed his piece of work.

Analysis

The case studies presented above are being reported ‘after the event’ so it is possible to know what has been achieved. In practice, in most cases, what has been achieved is essentially what was expected.

However, not all of the students acquired all of the skills and knowledge that were identified as targets. Also, on occasions, feedback from the students indicated that the goals had been set too high, or too low, so they needed to be revised during the course of the exercise. Perhaps the most appropriate analogy for the process used in the 'empowerment' method is the systems development process advocated in the Dynamic Systems Development Methodology (Stapleton 2000). This recommends an iterative approach and suggests that two groups of requirements should be identified: the essential and the desirable. A successful system is defined as one that fulfils all of the essential requirements and some of the desirable ones. Similarly, a successful empowerment exercise empowers and also, necessarily, enables the students to gain or reinforce some skills and some knowledge.

The approach to empowerment used is largely based on an interpretivist's philosophy (Galliers, 1992). This, in a teaching context, means that students learn by participating in an experience from which lessons can be drawn. The lessons learned are subject to different interpretations and have to be considered in the context of the experience. This contrasts with the more common positivist philosophy where hard facts and techniques are taught which are learned by the students and applied to solve problems (Galliers, 1992). An interpretivist approach to teaching and learning is not new and has been applied in university environments before, but in many cases problems have been reported that many students, more used to a positivist approach, find the lack of clear-cut answers and direction in the interpretivist approach confusing and demotivating (Banks 2001). The approach we have used to teach empowerment, however, overcomes these problems in the following way:

- The approach used is not a pure interpretivist approach. In each case the students are required to apply some knowledge previously taught. This positivist goal often becomes the initial focus for the students, some of which may not see any other benefit to what they are doing at first. The interpretivist lessons then learned can be built on top of this positivist achievement.
- The project work undertaken is to produce some form of system. This in itself is a strong motivating factor; students like to produce something tangible, especially if they feel their product is worthwhile in itself.
- The objective of creating a real world experience is explained at the beginning. Students are generally keen to make a useful contribution to their future employers so they see the opportunity to do well in an environment likely to be similar to that of their future employment as a worthwhile challenge.

The main goal of empowerment, for the students to learn how to learn, is not at first appreciated by the students. However, the other goals are understood and these act as sufficient motivation. The ability to learn from their experience is generally not appreciated until the end of the course and often not until well after the course. It is not uncommon for past students, now in the work place, to reveal they have subsequently found this teaching of empowerment to be the most useful part of their degree course.

Conclusions

This paper has shown that 'empowerment' is an interesting and worthwhile aim. We have also provided evidence that it can assist in overcoming the problem of fast changing knowledge and varying perceptions on the expected outcomes of teaching. It provides the skills and self belief that are essential in industry and is an effective way of preparing students for the workplace.

The essence of the teaching methodology is to provide a realistic and challenging task that stretches students beyond their experience and then encourages them to think and reflect on what they have learned. The approach relies on new ideas coming forward which are not currently included in the curriculum to motivate and challenge the students. This means that the project that forms the basis for the empowerment exercise needs to be changed regularly. The IT environment is ideal for this type of teaching precisely because it is changing so rapidly. It will always provide opportunities to create new problems that challenge the students.

The method itself works satisfactorily and has been shown to be capable of being mapped onto a variety of circumstances. It is also a method for solving the crises created by shortages in 'new' skills and knowledge areas (Damsguard and Scheepers, 2000). We commend it to you as it is a useful and enjoyable experience for the students that will be particularly appreciated when they get to the workplace.

References

Armarego, J., Fowler, L., & Roy, G. (2001). Constructing Engineering Knowledge: development of an online learning environment. *Proceedings of the Fourteenth Conference on Software Engineering Education and Training*, The IEEE Computer Society, Los Almitos California, USA, 258-267.

Bagert, D., & Mead, N. (2001). Software Engineering as a Professional Discipline. *Computer Science Education* 11, 73-87.

Banks, D.A. (2001). Reflections on Interpretivist Teaching with Positivist Students. *Proceedings of Informing Science 2001*. Retrieved March 21, 2002 from the World Wide Web http://ecommerce.lebow.drexel.edu/eli/pdf/banksebkrefle.pdf

Brophy, J., & Alleman, J. (1991). Activities as instructional tools: A framework for analysis and evaluation. *Educational Researcher*, 20, 9-23.

Brown, A. L. (1994). The Advancement of Learning. *Educational Researcher*, 23, 4-12.

Callahan, J.R., & Khatsuriya, R.R. (1998). Web-Based Issue Tracking for Large Software Projects. *IEEE Internet Computing*, Sept-Oct 1998.

Computing (1999). The UK Higher Education Funding Council Quality Assurance Agency Benchmark for Computing. Retrieved March 21, 2002 from the World Wide Web http://www.qaa.ac.uk/crntwork/benchmark/computing.pdf

Cowling, A. (2001). Teaching Data Structures and Algorithms in a Software Engineering Degree: some experiences with Java. *Proceedings of the 14th Conference on Software Engineering Education and Training*, IEEE Computer Society, 247-257.

Damsguard J., & Scheepers, R. (2000). Managing the crisis in intranet implementation: a stage model. *Information Systems Journal* , 10, 131-149.

Dawson R. (2000). Twenty Dirty Tricks to Train Software Engineers. *Proceedings of the 22nd International Conference on Software Engineering*, ACM Press, New York, 209-218.

Dawson, R.J., & Newsham, R.W. (1996). Very Knowledgeable - But Not a Lot of Use: Turning Computer Science Graduates into Professionals at GPT. *Proc. Professional Awareness in Software Engineering '96* , Royal Society, London, 229-240.

Dawson, R.J., & Newsham, R.W. (1997a). So You Learned to Program at University. No Wonder the Software is Late. *Proceedings of the Psychology of Programming Interest Group*, PPIG9 , Sheffield, 1-6.

Dawson, R.J., & Newsham, R. (1997b). Converting Computer Science Graduates into Professionals. *The Responsible Software Engineer*, Springer, London , 321-331.

Dawson, R.J., Newsham, R.W., & Fernley, B.W. (1997). Bringing the "Real World' of Software Engineering to University Undergraduate Courses. *IEE Proceedings in Software Engineering* , 144(5-6) , pp 287-290.

Dyba, T. (2000). Improvisation in Small Software Organisations. *IEEE Software*, 10 (5)

Ford, G., & Gibbs, N. (1996). A Mature Profession of Software Engineering. CMU/SEI-96-TR-004.

Friedman, B., & Kahn, P.H. (1994). Educating computer scientists: Linking the social and the technical. *Communications of the ACM*, 37(1), 65-70.

Galliers, R. (1992). Information Systems Research - Issues, Methods and Practicality Guidelines. Blackwells Scientific, Oxford, UK.

Jackson, T., & Dawson, R.J. (1999). The Need for Computer Scientists to Receive Training on People Skills. *Collected Papers of the Psychology of Programming Interest Group, 11th Workshop* , University of Leeds, PPIG11 , 67-91.

Joshi, K., & Rui, A. (2000). Impact of information products on information system users’ job satisfaction: an empirical investigation. *Information Systems Journal*, 10, 323-345.

Kolb, D. (1984). Experiential Learning, Experience as a source of learning and development. Prentice Hall, New York.

Newman, I.A. (2001a). Using the Internet to Help Meet Learning Goals in a Conventional University Environment - Some Experiences. *Proceedings of the 2001 Symposium on Applications and the Internet*, IEEE Computer Society , San Diego, USA, 56-60.

Newman, I.A. (2001b). The Project Management Website as a Tool to Encourage Students to Think , *7th IFIP World Conference on Computers in Education: Networking the Learner* , UNI-C, WCCE 2001 , Copenhagen, 216.

Newman, I.A., Dawson, R.J., & Parks, L.M. (2000). Reflections on the Process: Some Experiences of Teaching Students to Think About How They Produce Software in a Real Environment. *Proceedings of the Fifth International Conference on Software Process Improvement Research, Education and Training*, British Computer Society, INSPIRE V, 25-36.

Newsham, R.W., & Dawson, R.J. (1996). Experiences Teaching a First Programming Language at GPT. *Proceedings of 8th Annual Workshop, Psychology of Programming Interest Group* , KaHo Sint Lieven, PPIG 8 , 133-138.

Parish, D.J., & Newman, I.A. (1999). Educating Systems Engineers in the University sector. *IEE Engineering Science Education Journal*, 8(4), 169-175.

Pocket Oxford Dictionary (1959). The Pocket Oxford Dictionary of Current English . Fourth Edition, Oxford University Press.

Psoinos, A., Kern, T., & Smithson, S. (2000). An exploratory study of information systems in support of employee empowerment. *J. of Information Technology*, 15, 211-230.

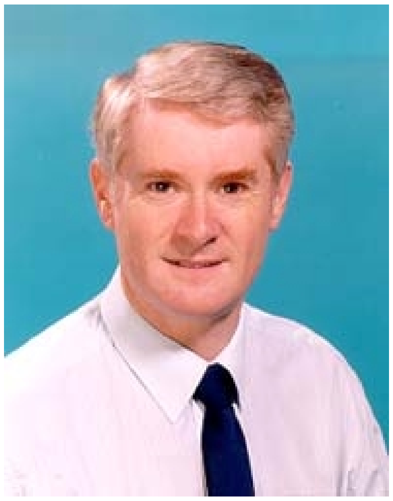
Sandburg, K.W., & Vinberg, S. (2000). Information technology and learning strategies in small enterprises. *Behaviour and Information Technology*, 19 (3), 221-227.

Stapleton, J. (2000). DSDM, Dynamic Systems Development Metho - the method in practice. Addison Wesley.

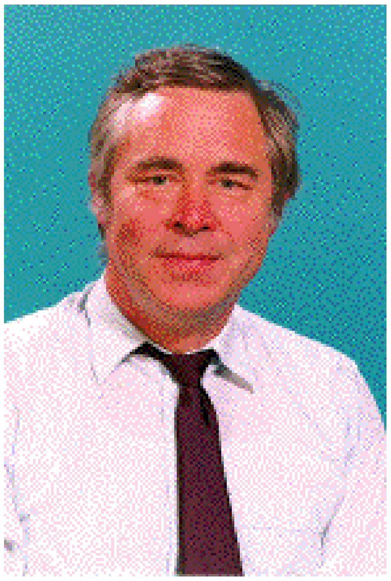
SWEBOK (2001). The Guide to the Software Engineering Body of Knowledge. Retrieved March 21, 2002 from the World Wide Web http://www.swebok.org   Stoneman Version 0.95.

Thompson, J., Edwards, H.. (2001). Achieving a world wide Software Engineering Profession. *Proceedings of the 14th Conference on Software Engineering Education and Training*, IEEE Computer Society, 67-74.

Biographies



Ray Dawson has been a Senior Lecturer in Computer Science at Loughborough University, UK, since 1987. Previously he worked for ten years as a software engineer at Plessey Telecommunications and before that he was at Nottingham University where he obtained his bachelors and masters degrees. He is currently researching into software project management and business and industry working practices, especially for the development of information systems and in information systems metrics and cost analysis. His teaching is concerned with information systems and project management. Ray Dawson is a member of the British Computer Society and a Chartered Engineer.



Ian Newman has been a member of staff at Loughborough University since 1973. Initially appointed as a Senior Lecturer in Computing in the Department of Mathematics he is now a Reader in Computer Science. Prior to moving to Loughborough he was the Senior Systems Analyst in the Computing Centre at Nottingham University for seven years. His Bachelors degree and Doctorate were in Physics and were both obtained from Nottingham University. Ian has researched and taught many different subjects but most relate to the effective use of computer systems to support people in doing their jobs. Currently his research and teaching are both focused on methods of encouraging learning amongst undergraduates and postgraduates.