TEACHING DATABASE ANALYSIS AND DESIGN IN A WEB-BASED CONSTRUCTIVIST LEARNING ENVIRONMENT

Thomas M. Connolly, Carolyn E. Begg
School of Computing, University of Paisley, High Street, Paisley, United Kingdon

Keywords: Database analysis and design, online learning, reflective practitioner, constructivist learning environments, project-based learning.

Abstract: The study of database systems is typically core in undergraduate and postgraduate courses related to computer science and information systems. However, there are parts of this curriculum that learners find difficult, in particular, the abstract and complex domain of database analysis and design, an area that is critical to the development of modern information systems. This paper reflects on these difficulties and describes an approach for teaching database analysis and design online motivated by principles found in the constructivist epistemology, which helps to overcome these difficulties and provides the learner with the knowledge and higher-order skills necessary to understand and perform database analysis and design effectively as a professional practitioner. The paper presents some preliminary results of this work and reflects on the findings.

1 INTRODUCTION

The database is now the underlying framework of the information system and has fundamentally changed the way many companies and individuals work. This is reflected within tertiary education where databases form a core area of study in undergraduate and postgraduate courses related to computer science and information systems, and typically at least an elective on other data-intensive courses. The core studies, typically, are based on the relational data model, SQL, data modeling, relational database analysis and design and, increasingly, object-relational concepts. This supports industry where the object-relational DBMS is the dominant data-processing software currently in use. The core relational theory is a mature and established area now in relation to other parts of the computing curriculum. However, there are parts of this curriculum that learners find difficult, in particular, database analysis and design.

Mohtashami and Scher (2000) note that pedagogical strategies for teaching database analysis and design traditionally follow a similar modality to that of other technical courses in computing science or information systems. A significant amount of technical knowledge must be imparted with the teacher becoming a ‘sage on stage’ and the learners passive listeners. This is the objectivist model of learning, which views learning as the passive transmission of knowledge from the teacher to the learner, heavily criticized for stimulating surface learning and knowledge reproduction. In contrast, the central tenet of the constructivist view is that learning is an active process where new knowledge is constructed based on the learner’s prior knowledge, the social context, and the problem to be solved. In this paper, we describe a teaching approach that we have used to teach database analysis and design online motivated by principles found in the constructivist epistemology to help provide the learner with the knowledge and higher-order skills necessary to understand and perform database analysis and design effectively as a professional practitioner.

In the following section, we outline the high-level pedagogical aims of our database modules and consider some of the difficulties that arise in achieving these aims. In the subsequent section, we examine related work on constructivism and constructivist learning environments. In Section 4, we discuss how we have applied constructivist principles to our teaching of a database module delivered fully online. In Section 5, we present some early findings from our approach followed by some conclusions.

2 PEDAGOGICAL AIMS

The database modules in the School of Computing at the University of Paisley have the following general educational aims:

- Develop a sound understanding of the principles and underpinning theory related to the study of database systems.
- Assist the development of a business ethos in the student that emphasizes fitness for purpose as the guiding principle in the design, development, and assessment of database systems and their components.
- Enable the student to take a disciplined approach to problem definition, and to the specification, design, implementation, and maintenance of database systems.
- Develop critical, analytical, and problem-solving skills and the transferable skills to prepare the student for graduate employment.
- Assist the student to develop the skills required for both autonomous practice and team-working.
- Create awareness of the continuing development of database technologies and applications and the need for continued study, reflection, and development throughout a career as a database professional.

In themselves, these aims are not unusual and are typical for many undergraduate database modules. Our modules have a vocational orientation and we expect our graduates to become professional database practitioners typically in a multi-disciplinary environment. Previous approaches to educating database designers and, more generally, software designers model scientific and engineering methodologies, with their focus on process and repeatability. In general, this approach is based on a normative professional education curriculum, in which students first study basic science, then the relevant applied science, so that learning may be viewed as a progression to expertise through task analysis, strategy selection, try-out, and repetition (Armarego, 2002). While students tend to cope well using this approach with many of the theoretical and practical components of the core database curriculum, for example, understanding the properties of the relational data model, the basics of SQL, and using a relational DBMS, one area that tends to be problematic is the abstract and complex domain of database analysis and design (for the purposes of this paper, we use the term database analysis and design to encompass system definition, requirements collection and analysis, conceptual database design, logical database design, and physical database design). A comparable problem has been identified with object-oriented analysis and design, which is also highly abstract (Hadjerrouit, 1999), requirements engineering (Bubenko, 1995), and software design and testing (Budgen, 1995).

While databases have become so essential to organizations, some students become deceived by the simplicity of creating small databases using products such as Microsoft Access and believe they can create more complicated databases just as easily. Unfortunately, the resulting databases are hard to use, barely meet system requirements, and are difficult to redesign. In addition, students require skills to work in a project team, skills to apply appropriate fact-finding techniques to elicit requirements from the client (both ‘soft’, people-oriented skills), skills to conceptualize a design from a set of requirements (‘soft’, analytical and problem-solving skills), skills to map a conceptual model to a logical/physical design (‘hard’, technical skills), and skills to reflect and review intermediate designs, particularly where information complexity is present (a combination of ‘soft’ and ‘hard’ skills). These are different skills from learning SQL, knowing the components of an ER model, or being able to recite the properties of the relational model. Students often have considerable difficulty comprehending implementation-independent issues and analyzing problems were there is no single, simple, well-known, or correct solution. They have difficulty handling ambiguity and vagueness, which can arise during knowledge elicitation. They can also display an inability to translate classroom examples to other domains with analogous scenarios, betraying a lack of transferable analytical and problem-solving skills. These problems can lead to confusion, a lack of self-confidence, and a lack of motivation to continue.

Software engineering (and therein database analysis and design) has been described as a *wicked problem*, characterized by incomplete, contradictory and changing requirements, and solutions that are often difficult to recognize as such because of complex interdependencies (DeGrace and Hulet Stahl, 1998). According to Armarego (2002), there is an educational dilemma in teaching such problems in software engineering because:

- complexity is added rather than reduced with increased understanding of the problem;
- metacognitive strategies are fundamental to the process;

- a rich background of knowledge and intuition are needed for effective problem-solving;
- a breadth of experience is necessary so that similarities and differences with past strategies are used to deal with new situations.

Schön (1983) argues that the primary challenge for designers is how to make sense out of situations that are puzzling, troubling, and uncertain. According to Schön the following are some of the key problems in teaching design:

- It is learnable but not didactically or discursively teachable: it can be learned only in and through practical operations.
- It is a holistic skill and parts cannot be learned in isolation but by experiencing it in action.
- It depends upon the ability to recognize desirable and undesirable qualities of the discovered world. However, this recognition is not something that can be described to learners, instead it must be learned by doing.
- It is a creative process in which a designer comes to see and do things in new ways. Therefore, no prior description of it can take the place of learning by doing.

As an additional complexity, to provide more flexible modes of study and capture new markets, tertiary education is providing more modules and courses in an online format, resulting in students who are geographical dispersed and have diverse backgrounds. While online learning has many advantages ("anytime, anywhere, anypace") there are also disadvantages such as increased setup costs, more responsibility is placed on the learner who has to be self-disciplined and motivated, increased workload on students and staff, non-involvement in the virtual community may lead to feelings of loneliness, low self-esteem, isolation, and low motivation to learn, which in turn can lead to low achievement and dropout, and dropout rates tend to be higher than in traditional face-to-face programs, often 10 to 20 percentage points higher (Connolly and Stansfield, 2006). To address these issues we require a different approach to traditional (face-to-face) teaching methods. Figure 1 provides a representation of the types of knowledge and skills required to undertake database analysis and design and the associated problems.

The above discussions suggest an alternative approach to teaching database analysis and design may overcome some of the above difficulties and in the next section we examine one such approach that we have found useful.

3 PREVIOUS WORK

While traditional education has been guided by the paradigm of didactic instruction, which views the learner as passively receiving information, there is now an emphasis on *constructivism* as a philosophical, epistemological, and pedagogical approach to learning. Cognitive constructivism views learning as an active process in which learners construct new ideas or concepts based upon their current/past knowledge. The learner selects and transforms information, constructs hypotheses, and makes decisions, relying on a cognitive structure to do so. In addition, constructivism asserts that people learn more effectively when they are engaged in constructing personally meaningful artifacts. Social constructivism, seen as a variant of cognitive constructivism, emphasizes that human intelligence originates in our culture. Individual cognitive gain occurs first in interaction with other people and in the next phase within the individual (Forman and McPhail, 1993). These two models are not mutually exclusive but merely focus upon different aspects of the learning process.

According to Gance (2002) the main pedagogical components commonly associated with these models are:

- A cognitively engaged learner who actively seeks to explore his environment for new information.
- A pedagogy that often includes a hands-on, dialogic interaction with the learning environment (eg. designing a database is preferred to being told how to design a one).
- A pedagogy that often requires a learning context that creates a problem-solving situation that is realistic.
- An environment that typically includes a social component often interpreted as actual interaction with other learners and with mentors in the actual context of learning.

Dewey argued that knowing and doing are intimately connected and that learning occurs in the context of activity when an individual attempts to accomplish some meaningful goal and has to overcome difficulties in the process. Schön (1983) describes professionals as individuals who make this connection between knowing and doing through reflective practice, suggesting that professionals learn to think in action and learn to do so through their professional experiences. For Schön, practitioners (in our case, database designers) have their own particular knowledge codes fully embedded within their practices. They apply tacit knowledge-in-action, and when their problems do not yield to it, they *reflect-in-action*, using the languages specific to their practices. When they evaluate the event afterwards, they *reflect-on-action*, using the language of practice, not the language of science. In this way, professionals enhance their learning and add to their *repertoire* of experiences, from which they can draw in future problem situations. Schön believes that it is this ability to reflect both in, and on, action that identifies the effective practitioner from less effective professionals. For Schön the ideal site of education for reflective practice is the 'design studio' where, under the direction of a master practitioner serving as coach, the novice learns the vocabularies of the professional practice in the course of learning its 'operational moves'.

These arguments suggest that students can only learn about design by doing design, and rely less on overt lecturing and traditional teaching. This approach requires a shift in the roles of both students and teachers, with the student becoming an apprentice, exploring and learning about the problem in the presence of peers (who may know more or less about the topic at hand) and the teacher moving from being the 'knowledgeable other' towards becoming a facilitator, who manages the context and setting, and assists students in developing an understanding of the material at hand (Koehler and Mishra, 2005).

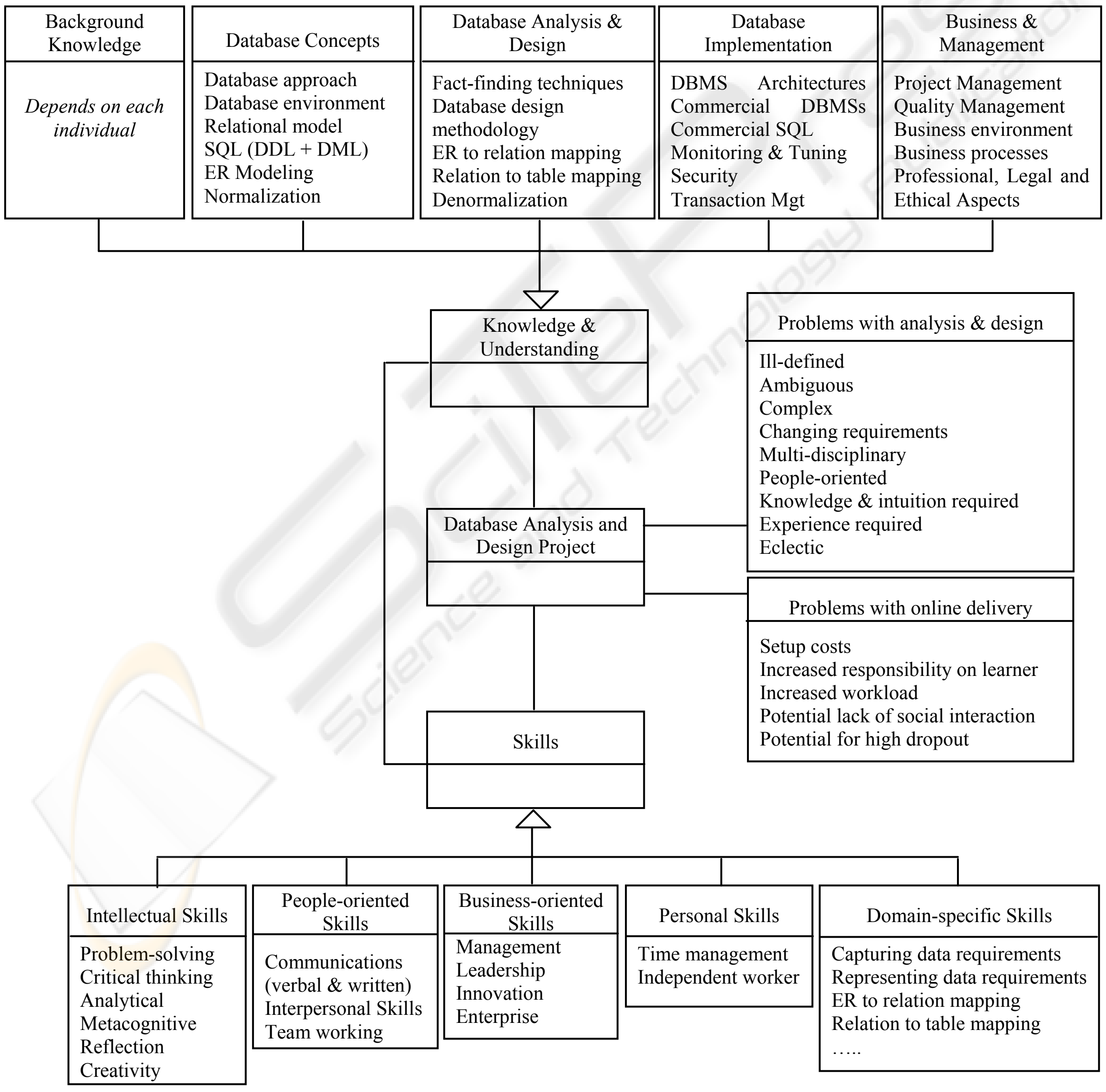

Figure 1: Types of knowledge and skills required to undertake database analysis and design.

3.1 Constructivist Learning Environments

Many researchers have expressed their hope that constructivism will lead to better educational software and better learning (for example, Brown, Collins, & Duguid, 1989). They emphasize the need for open-ended exploratory authentic learning environments in which learners can develop personally meaningful and transferable knowledge and understanding. This has led to the development of guidelines and criteria for the development of a *constructivist learning environment* (CLE) - “a place where learners may work together and support each other as they use a variety of tools and information resources in their guided pursuit of learning goals and problem-solving activities” (Wilson, 1996, pp. 28).

According to Ben-Ari (2001) constructivist principles have been more influential in science and mathematics education than in computer science education. However, there are examples of the application of constructivism within computer science from the development of Logo – a programming language for schoolchildren (Papert, 1980), the teaching of programming (Pullen, 2001), computer graphics (Taxén, 2003), object-oriented design (Hadjerrouit, 1999), communication skills in computer science (Gruba and Søndergaard, 2001), to collaborative learning using the Web (Connolly, Stansfield, & McLellan, 2005).

Project-based learning (PBL) is a constructivist approach to learning knowledge and skills through a process structured around projects with complex and authentic tasks, objectives, questions, and problems. In PBL, the teacher (facilitator) is available for consultation and plays a significant role in modeling the metacognitive thinking associated with the problem-solving processes. These reflect a *cognitive apprenticeship* environment (Collins *et al*., 1989) with *coaching* and *scaffolding* (e.g. offering hints, reminders, and feedback) provided to support the learner in developing metacognitive skills. As these skills develop, the scaffolding is gradually removed. The intention is to force learners to assume as much of the task on their own, as soon as possible. A further important element is *debriefing*, which provides the opportunity for learners to consolidate their experience and assess the value of the knowledge they have obtained in terms of its theoretical and practical application to situations that exist in reality.

4 APPLYING THESE PRINCIPLES

We have developed a Web-based CLE using the above principles built around the cognitive apprenticeship model and project-based learning to teach some of the database modules in our undergraduate/postgraduate courses. A more complete discussion of the CLE can be found in Connolly and Begg (2006), however, in this paper we focus on the use of the online CLE for the Fundamentals of Database Systems (FDBS) module, a core module in the School’s MSc Information Technology course, a conversion course for non-computing graduates. The students taking this module are reasonably experienced learners although not experienced in computing.

The FDBS module runs in a traditional face-to-face mode for full-time and part-time cohorts and since session 2001/2 in a fully online format for a part-time cohort. Since session 2002/3, we have used a CLE for the online cohort, which typically consists of 15-25 students, all from similar professional backgrounds. Scaffolding is provided through the teacher (facilitator) as well as through the creation of visualizations for a number of database concepts (eg. ER modeling, normalization, mapping an ER model to relations) and lower-level online units covering the relevant module material. When the students encounter problems they can drill down to the relevant material or use the higher-level visualizations. In the early stages, asynchronous online tutorials are run to discuss worked examples covering activities that groups would have to undertake as part of database analysis and design. It is important that students fully understand these examples and can apply the principles in the different contexts they will find themselves in.

The students self-select themselves into groups of size 3-4 and each group chooses a project that is of interest to all group members. These projects are generally from small businesses in the West of Scotland, which has the added advantage of benefiting these businesses and thereby the local economy. The facilitator provides background advice to ensure that a group does not take on a project that is too large or complex or alternatively too trivial. Students are encouraged to keep sufficiently detailed and formal records of their work and, in particular, the decisions made with supporting justifications. They are also encouraged to frequently reflect on these decisions and the processes that led to the decisions both as a group and as individuals. To support the notion of cognitive preference (Connolly, MacArthur, Stansfield, & McLellan, 2006), each group/individual is given scope to use whatever tools they feel most appropriate and most comfortable with. The FirstClass VLE is used for the online material as well as providing email facilities and discussions boards, both public (ie. available to the students and facilitators) and private (a student-only discussion area). Interestingly, while groups initially use these basic facilities, they also develop their own wikis and blogs, while using Skype/mobiles and instant messaging for more urgent communication. Groups use laptops and PDAs for recording meetings with the clients and the facilitator.

Support is provided by the facilitator as and when necessary but this is only in an advisory capacity: groups are not provided with solutions or partial solutions but are instead directed to where appropriate information can be found. This reinforces the principles of constructivism and emphasizes to the students that they are acting as professional database design consultants and have to act in this capacity. Debriefing is conducted at the end for all parties (facilitators, students, and clients) to reflect on the learning outcomes and to reflect on issues that had arisen in the performance of the projects. We discuss some of these issues in the next section.

5 PRELIMINARY RESULTS

This section presents some preliminary findings from using the project-based learning approach to teach database analysis and design in the FDBS module. A quantitative analysis of students' performance in the FDBS module is presented in Connolly *et al*. (2006). The paper compares the performance of 977 students divided into three groups, one of which used the constructivist project-based approach albeit through online delivery. The evidence supports our view that the constructivist approach can improve student learning. The results were not fully conclusive because the effect could have been entirely attributable to online delivery rather than the project-based approach and further quantitative research is required. However, the qualitative analysis of student and faculty feedback from the FDBS module that we undertook in parallel provides some interesting results to further support our view as we now discuss.

Finally, a qualitative analysis of student and faculty feedback provide further insight into this approach. Generally, student feedback was extremely positive, all students reporting that they had enjoyed the experience. They were able to compare this approach with the more traditional case study approach that they had encountered in their previous studies and had felt that the project-based approach with learning *in situ* had provided a better, more motivating, more engaging method to learn about database analysis and design. They also appreciated that this approach gave them relevant work experience that could help their employment prospects on completion of the course. The students were also very receptive to the concept of a reflective journal and, while it was sometimes difficult to find the time to maintain it, many reported that they had benefited from this approach and would keep a reflective journal for the remainder of their studies and into employment. On the negative side, most students reported that the workload was significantly higher than in other modules. They also found time-management was an issue, particularly as they had no real feeling at the outset for scope and complexity of the projects they had selected (many were led by their enthusiasm for working as a professional consultant). All were in agreement that the approach should be extended to other modules, but rather than having a project per module, they suggested that one assessment-based integrative project that extended over a number of modules would be an extremely powerful approach to teaching and learning.

Faculty were also enthusiastic of this approach and felt the students had learned more than with the case study approach, particularly in areas not traditionally covered in the database modules (use of fact-finding techniques, and people- and business-oriented skills). It was important that sufficient guidance was given during the project, particularly in the early stages when the groups were selecting projects (as noted above, student enthusiasm had to be tempered with realistic expectations). At the same time, as students were now working in an environment that had not been purpose-built for their effective learning, care had to be taken to ensure students were not overwhelmed with all the complexities that a real-world project can present, otherwise their initial enthusiasm quickly dissipated. The students needed guidance with both group and personal reflection initially until they found tools they were comfortable with (eg. wikis, blogs).

Typically each faculty member handled between 4-6 project groups compared to sometimes as many as 20 groups with the case study approach. Nevertheless, faculty found that their workload was significantly higher than with traditional approaches and that it was necessary to develop in-depth knowledge of each project to be able to support the students effectively. This gave rise to grave concerns over scalability and faculty felt that they could not have coped with any further project groups.

Faculty observed that students generally underestimated the time required to undertake the project and the facilitator needed to discuss the similarities and differences between case study assessments and project-based assessments. For example, some students underestimated the time spent securing a company's involvement in their project and establishing that relationship cannot always be rushed to fit a timescale that suits the students and meets the demands of faculty. It was also important for the facilitator to identify the gaps in the students' knowledge and skills and direct them to appropriate sources to enable them to undertake the project effectively. Failing to do this in a timely manner, led some students to lose confidence and meant they simplified and converted the project into a form of case study that they could cope with. However, this should and can be avoided with sufficient support from the facilitator to encourage students to accept the realities and complexities of PBL as a positive aspect of their work. It is the students' ability to cope with and manage the project that is being assessed and therefore it is necessary that they do not ignore or smooth over the problems of working with a real company.

While assessments based on case studies for database analysis and design usually present a simplified and contrived set of requirements that the students then analyse and solve, our PBL approach requires that the students must first capture the requirements for the new database. Capturing requirements require that students use fact-finding techniques that may be known in theory but not practised. Therefore, while case study assessments cover requirements analysis through to physical database design and possibly thereafter to implementation, PBL extends the coverage of the database system development lifecycle from the systems definition stage through to implementation. It is therefore clear that the skills required to undertake PBL differs to that of the case study approach.

As the success of the PBL approach is dependent on the support of industry, faculty emphasized that the facilitator must carefully guide students in their relationship with the company while ensuring that students achieve the specified learning outcomes. This sometimes required significant diplomacy from the facilitator when the academic objectives did not fully match the commercial objectives. It is important that faculty explain to companies at the outset what constitutes reasonable expectations for parameters such as project size, project complexity, and overall timescales. However, in most cases, both students and companies benefited from the relationship and this is why PBL has been well supported by companies over the last few years.

Occasionally, faculty encountered problems with group dynamics, for example, autonomous students tend to prefer to work individually, there can be lack of group cohesion, dominant group members, insecure group members, and free-riders (referred to in group dynamics research as 'diffusion of responsibility'). To highlight that these can occur in industry and need to be overcome, students were encouraged to tackle these problems as a group and only in extreme cases did faculty intervene to facilitate a solution acceptable to all.

There was agreement among faculty that the PBL approach was pedagogically sound for postgraduate courses and for third/fourth years of undergraduate courses, but were reluctant to use this approach in first or second year, on the grounds that students may not be sufficiently mature learners and may not have developed the necessary discipline and time-management skills required. Further, it was generally felt that rather than moving from being a 'sage on the stage' to a 'guide on the side', the facilitator had to be more of a 'fount of all knowledge' with project-based learning.

6 CONCLUSIONS

This paper has examined some of the issues surrounding the teaching of database analysis and design and has described a teaching approach motivated by principles found in the constructivist epistemology, based on the cognitive apprenticeship model and PBL. The approach used points toward learning about design by *doing* design, and relying less on overt lecturing and traditional teaching. Design is learned by becoming a practitioner, albeit for the duration of the module, not merely by learning about practice. In brief, students should engage in challenging problems that reflect real-world complexity. The problems should be authentic and ill-structured; that is, they should not have one predetermined, foregone solution but rather be open to multiple interpretations and multiple 'right answers'. Students should engage in actively working on solving problems in collaborative groups to reflect the social nature of learning.

This approach requires a shift in the roles of both students and faculty. The student becomes a cognitive apprentice, exploring and learning about the problem in the presence of peers. Faculty shifts from being the 'sage on the stage' to the 'guide on the side' (possibly, in the extreme, the 'fount of all knowledge'), becoming a facilitator who assists students in developing an understanding of the professional practice of database analysis and design.

The paper presents some preliminary results of this work that shows the approach can be used successfully. The preliminary qualitative findings show that students and faculty reacted extremely positively to the approach and found it more motivating and engaging than the more traditional case study approach. However, both students and faculty found the workload higher than with more traditional teaching methods and that scalability was an issue. Faculty also felt that this approach required mature learners and may not be entirely appropriate for first and second year undergraduates.

REFERENCES

Armarego, J. (2002). Advanced Software Design: A Case in Problem-Based Learning. In *Proceedings of the 15th Conference on Software Engineering Education and Training* (pp. 44–54). 25–27 February 2002, Covington, Kentucky, USA.

Ben-Ari, M. (2001). Constructivism in Computer Science Education, *Journal of Computers in Mathematics and Science Teaching*, *20*(1), 45–73.

Brown, J.S., Collins, A., & Duguid, P. (1989). Situated cognition and the culture of learning, *Educational Researcher*, *18*(1), 32–42.

Bubenko, J. (1995). Challenges in Requirements Engineering. Keynote address. In *Second IEEE International Symposium on Requirements Engineering*. York, England.

Budgen, D. (1995). Is teaching software design a 'wicked' problem too? In *Proceedings of the 8th SEI Conference on Software Engineering Education* (pp. 239-254). New Orleans (La).

Collins, A., Brown, J.S., & Newman, S.E. (1989). Cognitive Apprenticeship: Teaching the Craft of reading, writing, and mathematics. In L. Resnick (Ed.). *Knowing, Learning, and Instruction: Essays in Honor of Robert Glaser*. Hillsdale, NJ: Lawrence Erlbaum.

Connolly, T.M. & Begg, C.E. (2006). A Constructivist-Based Approach to Teaching Database Analysis and Design, *Journal of Information Systems Education* (accepted for publication).

Connolly, T.M. & Stansfield, M.H. (2006). From eLearning to Games-based eLearning: Using Interactive Technologies in Teaching Information Systems, *International Journal of Information Technology Management* (submitted).

Connolly, T.M., MacArthur, E., Stansfield, M.H., & McLellan, E. (2006). A Quasi-Experimental Study of Three Online Learning Courses in Computing, *Journal of Computers and Education* (in print).

Connolly, T.M., Stansfield, M.H, & McLellan, E. (2005). An Online Games-Based Collaborative Learning Environment to Teach Database Analysis and Design, In *Proceedings of 4th IASTED International Conference on Web-Based Education*. Grindelwald, Switzerland, February 2005.

DeGrace, P. & Hulet Stahl, L. (1998). *Wicked Problems, Righteous Solutions: A Catalog of Modern Engineering Paradigms*, Prentice Hall.

Forman, E. & McPhail, J. (1993). Vygotskian perspectives on children's collaborative problem-solving activities, In E.A. Forman, N. Minick, and C. Addison Stone (Eds.). *Contexts for learning. Sociocultural dynamics in children's development*. Oxford University Press.

Gance, S. (2002). Are constructivism and computer-based learning environments incompatible?, *Journal of the Association for History and Computing*, *V*(1).

Gruba, P. & Søndergaard, H. (2001). A constructivist approach to communication skills instruction in computer science, *Computer Science Education*, *11*(3), 203–219.

Hadjerrouit, S. (1999). A Constructivist Approach to Object-Oriented Design and Programming. In *Proceedings of 4th Annual SIGCSE/SIGCUE Conference on Innovation and Technology in Computer Science Education (ITiCSE'99)*, 171–174.

Koehler, M.J. & Mishra, P. (2005). Teachers Learning Technology by Design, *Journal of Computing in Teacher Education*, *21*(3), Spring 2005.

Mohtashami, M. & Scher, J.M. (2000). Application of Bloom's Cognitive Domain Taxonomy to Database Design. In *Proceedings of ISECON (Information Systems Educators Conference) 2000*. Philadelphia.

Papert, S. (1980). *Mindstorms: children, computers and powerful ideas*, Basic Books.

Pullen, M. (2001). The Network Workbench and Constructivism: Learning Protocols by Programming, *Computer Science Education*, *11*(3), 189–202.

Schön, D.A. (1983). *The Reflective Practitioner: How Professionals Think in Action*. Basic Books: New York.

Taxén, G. (2003). Teaching Computer Graphics Constructively. In *Proceedings of International Conference on Computer Graphics and Interactive Techniques* (pp. 1–4). San Diego, California.

Wilson, B. (1996). *Constructivist learning environments: Case studies in instructional design*. Educational Technology Publications: New Jersey.