LEARNING FROM MASTERS: ACADEMIC APPRENTICESHIP MODEL FOR COMPUTATIONAL DESIGN EDUCATION

HALIL ERHAN
College of Information Technology
UAE University, UAE Al-Ain
hierhan@uaeu.ac.ae

Abstract. Education is taking a new shape for the emerging needs of society. A considerable number of schools and disciplines are adapting active and cooperative learning to foster critical thinking and cognitive skill gaining. Design computation discipline also has to search for new models in education and experiment with these to evolve. This paper presents the current status in other disciplines. The lessons learned are used to develop the Academic Apprenticeship Model for teaching design computation courses.

1. Introduction

Teaching is generally defined as "imparting" or "providing" knowledge or skill that implies an educational activity involving knowledge transfer from one individual to others. The bottlenecks of traditional teaching approaches based on this definition mainly take place as the person moderating learning is called "teacher"—knowledge provider—and "students" are knowledge receivers. This relationship hinders, if not eliminates, the feedback from the students to the instructors and only addresses short-term learning objectives. In addition, most of the opportunities through pair-, group-, or corporative-learning are missed due to the limited interaction between the students; it does not count learning from peers. Critical thinking opportunities are also missed due to "authoritative" environment. However, our definition of teaching should be more comprehensive and evolving; teaching is a social activity requiring complex yet organized interaction between both the instructors and the students.

In this paper, we investigate recent and proven-to-be-effective educational models replacing traditional teaching for general science and engineering programs. We discuss possible rationales for success of these from a

theoretical perspective. This is followed by the proposed Academic Apprenticeship Model (AAM) for design computation courses.

2. Design computation and Characteristic Challenges in Education

Design computation is a discipline integrating knowledge, experience, and skills from science, technologies, engineering, and humanities. The students of this discipline often find themselves studying a mixture of subjects at a time to address a given problem through designing and implementing computational design tools. They are expected to be capable of adjusting rapidly changing situations and volatile computation techniques by continuously equipping with new knowledge and developing new skills.

The main challenge in design computation education is, therefore, to teach students independent and interdependent learning, thinking, and metacognitive skills. Obviously, this is not trivial task for the students and the instructors; and requires shifts from the traditional education models to novel ones. In order to formulate new models we investigate similar disciplines and how they address education-related problems.

3. Education approaches in various disciplines

Similar challenges mentioned above are addressed by extensive pedagogical and instructional design studies in the literature that can be discussed in two groups. The first group focuses on course-based instructional methods. A study by Smith et al. (2005) proposes methods on classroom-based cooperative and problem-based learning. The authors present substantial experimental data in a historical perspective and discuss the effectiveness of these methods in different engineering courses. In line with this study, Felder and Brent (2003) advocates active learning to engage student in the learning engineering design rather than traditional in-class teaching. In this group also, Lawson (2001) proposes an instructional method to promote creative and critical thinking through analogical reasoning in biology education. For interdisciplinary economics courses, a team-based learning approach is suggested by Borg and Borg (2001). The approach aims at educating students with thinking skills to solve competing assumptions and values between economics and an applied domain.

In the second group, comprehensive curriculum-changes are suggested to enhance college education. Dym et al. (2005) proposes ‘cornerstone’ courses adapting project-based learning in engineering design curriculum. These courses give opportunities to students for developing intuitions for engineering design, using knowledge acquired from different courses, and applying their

knowledge in different situations. In software engineering education, Carnegie Mellon's Master of Software Engineering Program moves forward by successfully implementing a curriculum similar to discussed by Dym et al (2005) (Garlan, Gluch, and Tomayko, 1997; Hazzan and Tomayko, 2005). However, the program itself goes beyond having project-based courses and becomes a project-based curriculum: it is centered on long-term projects and courses to help students develop broad-based problem-solving skills. Felder (2004) reports that problem- and project-based learning is rapidly becoming part of departmental curricula in different universities particularly the ones that are member of Foundation Coalition (2006). These universities are attempting to integrate sequence of fundamental courses—which are traditionally though in isolation—in order to establish interrelationships and applications to specific disciplines, like engineering and sciences.

4. Design Computation, Teaching, and Learning

Although the examples above are not exhaustive, they give a general idea about changing education models for the better in different disciplines. I have not come across much recent research on education theories and methods in design computation curriculums. However there have been some individual and isolated efforts that discuss an educational model and pedagogical method. The list of studies relevant to this discussion includes, but not limited to the following.

Bridges (1992) and Akin (1990) introduces alternative strategies for integrating computational design-support tools in design education. There are two computational courses that are designed with a teaching strategy in mind. The first course is Object-oriented Application Development in CAD (Fleming, Erhan, and Özkaya, 2004). Even though it does not explicitly mention a teaching and learning theory-based approach, project-based learning can be found in the core of the course and involves pair-programming (Beck and Andres, 2004), which is conducive to cooperative learning. An advanced level course for design requirements offered by Özkaya, Akin, and Tomayko (2005) is an effort bringing critical thinking and strategic knowledge building concepts in design computation education. The course, induced by software engineering methods, combines dynamic nature of software and computing technologies with architecture and engineering domains. The project-based course acknowledges and prepares the students for volatility of knowledge with hands on experience that cultivate generalizable concepts. The common objective of these courses is to equip students with experience to improve their problem solving skills that they can transform themselves.

5. Active and Cooperative Learning

Although traditional teaching models are still effective for some disciplines, there is a trend towards developing and adapting active and cooperative learning models (Chickering and Gamson, 1987; 1999). In this trend the main goal is to overcome the challenge of educating future professionals as lifelong learners with analytical, evaluative, and creative thinking skills.

Active learning aims at encouraging students engage and involve in improving their knowledge, thinking, and learning skills. Students solve problems, answer questions, and formulate questions of their own. The instructor serves as 'experts' to facilitate learning. Active learning is coupled by *cooperative learning* (Bonwell and Eison, 1991; Johnson, Johnson, and Smith, 1998; Springer, Stanne, and Donovan, 1999). The students work together towards common goals in teams on problems and projects and are expected to develop a sense of both positive interdependence and individual accountability.

The success of cooperative learning emerges from team forming where the students must acknowledge each others part in the project and individual accountability. Therefore, not all project-based courses apply cooperative learning unless they are managed by the instructors following 'heuristics' of cooperative learning (Silberman 1996; Felder and Brent, 2001).

5.1. ROOTS OF ACTIVE AND COOPERATIVE LEARNING

Rationale for active and cooperative learning can be found in two different sources. The first one is attributed to Confucius: "*I see and I forget, I hear and I remember, I do and I understand.*" The second is more descriptive and shows the relation between participation and information retention (Dale, 1969). The highest amount of information retention occurs when we actively involve in learning by doing (Figure 1). On the other hand, as our involvement becomes less, our retention drops dramatically.

However, this does not entirely describe why active and cooperative learning are successful models in general. Retention of information, I believe, is not only indicator for assessing how we learn. In addition, we must tackle transformation of cognitive skills and development of metacognition. For this, we need higher-level (cognitive and social) paradigms and theories.

Figure 1. Edgar Dale's Cone of Learning (Dale, 1969).

5.2. SOCIAL LEARNING AND COGNITIVE APPRENTICESHIP

According to Vigotsky's Social Learning theory, social interaction is an important component of cognitive development: we can develop more skills with guidance or peer collaboration in a social context than alone (Vigotsky, 1978). Influenced from Vigotsky's theory, Situated Learning theory by (Lave and Wenger, 1991) suggests that knowledge needs to be presented and acquired in an authentic context, i.e. settings and applications that would normally involve that knowledge.

On the other hand, the roles of the students and instructors must be defined in a social learning context. Cognitive Apprenticeship theory, a synthesis of formal teaching and traditional apprenticeship, can be a candidate for this (Brown, Collins and Duguid, 2006). In traditional apprenticeship model the master helps the apprentice in skill development; in adaptation of this model to education, educators coach students at the skill level to facilitate their cognitive development. The students acquire, develop, and use cognitive tools in an authentic and social context. As students master needed skills by practice at 'a close approximation to the master's level', instructors reduce controlling by deliberately turning some control over to students (Johnson, 1992). In the reinterpreted version of apprenticeship, instructors become part

of the class activities and students can learn by observing, imitating, discovering, and improving the presented skills by the instructors or peers.

It is obvious that the principles of active and cooperative learning can be derived from these theories and can form a framework for a social interaction between students and educators. I believe that if the framework is supported by effective strategies, potentially a model can emerge where knowledge acquisition and retention, and cognitive skill development becomes tightly intertwined.

6. Academic Apprenticeship Model for Design Computation Courses

I propose a model to potentially overcome educational challenges that we face in current design computation education. This model is called Academic Apprenticeship Model (AAM). According to my experiences in design computation education and literature survey, the active and cooperative learning models, Social and Situated Learning theories and Cognitive Apprenticeship theory—which are based on Social-Constructivist paradigm—can be used as components of this model. Thus, these leave us with a missing component: *instructional methods*. We don't have to look far for it because traditional methods in *design studios* can perfectly fit in this framework. Figure 2 shows the components of the AAM.

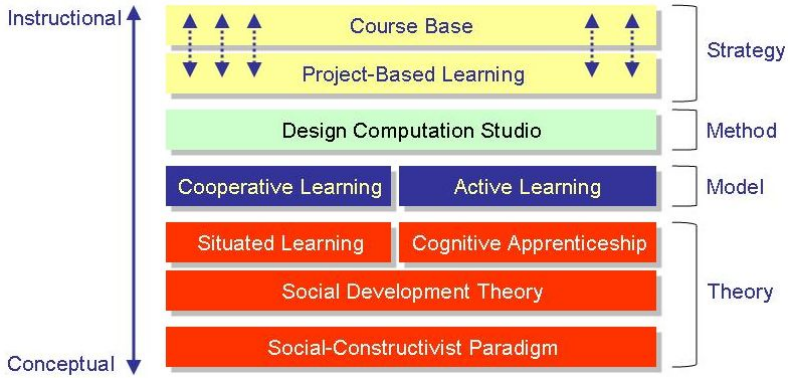

Figure 2. Building blocks of Academic Apprenticeship Model

The model proposes that students are provided with 'short-cuts' in learning 'tools', 'languages', and 'syntax' in order to keep the students focus on concept attainment, knowledge acquisition, and cognitive skill development in a social context. This model involves a small group of students interacting with the instructor. The approach is consistent with the principles of active and cooperative learning. The purpose is to provide guidance for the students as it happens in traditional skill gaining by artisans and proposed in cognitive

apprenticeship model. It also catalyzes active learning with technology-related skill gaining.

The inspiration for the model came from a graduate level course called Object-Oriented Application Development in CAD (Fleming, Erhan, and Özkaya, 2004). The approach followed in this course caries main characteristics of the model. I followed a more structured version of the AAM in another course, Web-based Software Development as part of Software Engineering at UAE University. While these courses initially seem different, they share a common goal: designing computational tools to solve real world problems. The activities in both courses overlap: definition of requirements, formulation of initial design solutions, implementation, and testing. In both courses the students are expected to gain knowledge in object-oriented paradigm and application development. The participants of these two courses formed 'teams' to design and implement an application as part of the class activities. In this way, students gain experience as members of a software development team. In the first course the students are given a project topic that must be addressed by implementing an application running on a commercial CAD system. Each time the course offered the teams successfully managed to develop running prototypes of different applications[1]. I taught the second course one semester in which the students successfully developed ‘a synchronized meeting scheduler’ application with minimum assistant from the instructor.

7. Conclusions

The studies in design computation education mainly centers on two concerns: use of computational design-support tools to enhance design learning and apply various instructional styles to teach students designing and implementing design-support tools. In both groups, it is difficult to find explicit concerns addressing educational theories and models behind. This study presents a model, the AAM, for design computation education based on proven theories and models. Although the model is not matured, it presents a position to start debate on the subject.

The AAM inherits the same challenges of other active and collaborative learning-based models: it requires a considerable course preparation time and effort. Added to the interaction time with the students the instructors should plan the course flow carefully. The teams in the class may have a sweet competition, but also the usual team conflicts have been observed. Assessing the performance of individual students becomes another challenge. Despite of these, when the remedies for these challenges, given in the active and

[1] http://www.andrew.cmu.edu/course/48-756/

collaborative learning literature are followed the benefits of adapting AAM can be quite rewarding particularly in design computation courses. However, I am aware that the effectiveness of the AAM must be empirically studied. I initiated a research to improve the model and test its further applicability in design computation which funded by the UAE University.

References

Akin, Ö.: 1990, Computational Design Instruction: Toward a Pedagogy, The Electronic Design Studio: Architectural Knowledge and Media in the Computer Era, CAAD Futures '89 Conference Proceedings, Massachusetts, 1989, pp. 302-316.

Beck, K. and Andres, C.: 2004 Extreme Programming Explained: Embrace Change, Second Edition, Addison-Wesley.

Bonwell, C. and Eison, J.: 1991, Active Learning: Creating Excitement in the Classroom, *ASHE-ERIC Higher Education Report* No. 1, 1991. (http://www.ntlf.com/html/lib/bib/91-9dig.htm) Visited on 01.19.2006

Borg, J. R. and Borg, M. O.: 2001, Teaching Critical Thinking in Interdisciplinary Economics Courses, *College Teaching*, **49**(1), pp. 20-25.

Bridges, A.H.: 1992, Computing and Problem Based Learning at Delft University of Technology Faculty of Architecture, CAAD Instruction: The New Teaching of an Architect, *eCAADe Conference Proceedings*, 12-14 November 1992, pp. 289-294

Brown, J.S., Collins, A. and Duguid, P.: Situated Learning and the Culture of Learning, http://www.sociallifeofinformation.com/Situated_Learning.htm, Visited on 01.22. 2006.

Chickering, A.W. and Gamson, Z.F.: 1987, Seven Principles for Good Practice in Undergraduate Education, *In* Poulsen, S. (ed.) *The Wingspread Journal* **9**(2)

Chickering, A.W. and Gamson, Z.F.: 1999, Development and adaptations of the seven principles for seven principles for good practice in undergraduate education. New Directions for Teaching and Learning, (80) 75-81.

Dym, C., Agogino, A., Eris, O., Frey, D., and Leifer, L.: 2005, Engineering Design Thinking, Teaching, and Learning, *Journal of Engineering Education*, pp. 103-120.

Felder, R.M. and Brent, R.: 2001, Effective Strategies for Cooperative Learning, *J. Cooperation & Collaboration in College Teaching*, 10(2), pp. 69-75.

Felder, R.M. and Brent, R.: 2003, Learning by Doing, *Chemical Engineering Education*, **37**(4), pp. 282--283.

Felder, R.M.: 2004, Changing Times and Paradigms, *Chemical Engineering Education*, **38**(1), pp. 32-33.

Flemming, U., Erhan, H.I., and Özkaya, I.: 2004, Object-Oriented Application Development in CAD, *Automation in Construction, Special Edition*, Elsevier Science, NY.

Garlan, D., Gluch, D.P., and Tomayko, J.E.: 1997, Agents of Change: Educating Software Engineering Leaders, *IEEE Computer* **30**(11): pp. 59-65

Hazzan, O. and Tomayko, J.: 2005, Reflection and abstraction processes in the learning of the human aspects of Software Engineering, *IEEE Computer*, **38**(6), pp. 39-45.

Johnson, S.D.:1992, A framework for technology education curricula which emphasizes intellectual processes, *Journal of Technology Education*, 3; pp. 1-11.

Johnson, D.W., Johnson, R.T., and Smith, K.A.: 1998, Active learning: Cooperation in the college classroom (2nd ed.). Edina, MN: Interaction Book Co.

Lave, J. and Wenger, E.: 1991, Situated Learning: Legitimate Peripheral Participation, New York: Cambridge University Press.

Lawson, A.E.: 2001, Promoting Creative and Critical Thinking Skills in College Biology, Critical Thinking Skills Bioscience, **27**(1) pp. 13-24.

Özkaya, I., Akin, Ö., and Tomayko, J.E.: 2005, Teaching to Think in Software Terms: An Interdisciplinary Graduate Software Requirement Engineering Course for AEC Students, in Soibelman, L. and Pena-Mora, F. (ed) Computing in Civil Engineering, Proceedings of the 2005 ASCE International Conference on Computing in Civil Engineering, pp. 1-10.

Silberman, M.: 1996, Active Learning: 101 Strategies to Teach Any Subject, A&B, Boston.

Smith, K.A., Sheppard, S. D., Johnson, D.W., and Johnson, R.T.: 2005, Pedagogies of engagement: Classroom-based practices, *Journal of Engineering Education Special Issue on the State of the Art and Practice of Engineering Education Research*, 94(1): 87-102.

Springer, L., Stanne, M. E., and Donovan, S. S.: 1999, Effects of small-group learning on undergraduates in science, mathematics, engineering, and technology: A meta-analysis. *Review of Educational Research*, **69**:1, pp. 21-51.

Vygotsky, L.S.: 1978, Mind in Society. Cambridge, MA: Harvard University Press.