Int. J. Engng Ed. Vol. 26, No. 6, pp. 1472–1483, 2010
Printed in Great Britain.

0949-149X/91 $3.00+0.00
© 2010 TEMPUS Publications.

Designing Senior Graduation Project Course for Computing Curricula: An Active Learning Approach*

HUREVREN KILIC
Department of Computer Engineering, Faculty of Engineering, Atilim University, Ankara, Turkey. E-mail: hurevren@atilim.edu.tr
MURAT KOYUNCU
Department of Information Systems Engineering, Faculty of Engineering, Atilim University, Ankara, Turkey. E-mail: mkoyuncu@atilim.edu.tr
MOHAMMAD REHAN
Department of Computer Engineering, Faculty of Engineering, Atilim University, Ankara, Turkey. E-mail: mrehan@atilim.edu.tr

This paper proposes an active learning-based design approach to senior graduation project courses for computing curricula. The proposed approach focuses mainly on course requirements including increasing the interaction between instructor and project team members; providing better and fair student/team performance assessment; encouraging students to practise life-cycle driven development; preparing students for role-based team-working; motivating students to communicate with experts from industry and supporting cooperation between students. It is observed that implementation of the proposed approach increases the student course satisfaction level while higher quality student projects are achieved.

Keywords: senior graduation project; active learning; project-based learning; computing curriculum; undergraduate education.

1. INTRODUCTION

WHETHER FROM THE HUMAN OR MACHINE LEARNING RESEARCH perspectives, the method of learning realized by any entity can simply be studied in two broad categories: active learning and passive learning. From the mathematical point of view, active learning is the problem of optimally designing the location of training input points in supervised learning scenarios [1]. From the machine learning viewpoint, active learning involves sequential sampling procedures that use information gleaned from previous samples in order to focus the sampling and accelerate the learning process relative to passive learning [2]. Clearly, although we can set up analogies between designing learning environments to educate people better and designing a successful machine (or agent) that aims to achieve its design goals in some complex environment, the distinct input of a human instructor or a machine designer are required. In fact, there are many different intelligent machine design approaches in which the active machine learning process models that of those already inspired by human behaviour or nature and uses them as suitable metaphors for high quality adaptive design solutions [3]. On the other hand, the sophistication of behaviour shown by a student during his or her learning process cannot be comparable to that exhibited by a machine.

In order to illustrate the difficulty in managing the human learning process, suppose that in some scenario, instructors at a college or university are responsible only for presenting ideas and information to the students and it is the student's responsibility to learn. Students are often conditioned by different types of sources of data and information (i.e. magazines, television and movies) to be passive learners, and therefore they expect to be entertained by their textbooks and instructors. The only instructors to which these students pay attention are those who catch their attention just as professional entertainers do. Consider, on the other hand, developing an attitude that the student is going to become (or be made) an active learner. Being an active learner, a student should be determined to learn everything from each of his or her teachers and textbooks. Sancho-Thomas *et al.* [4] have found that traditional courses appear not to succeed in helping students to acquire the required skills. Most of the traditional courses focus mainly on teaching technical content and they are usually organized according to teacher-centred approaches, where the teacher plays the role of information dispenser while the students act as

* Accepted 5 July 2010.

passive receptors. Just listening to instructors and taking notes is not an adequate method and currently many teachers are increasingly demanding that their students work in group assignments or projects. An active strategy, on the other hand, can enable students to achieve academic success. According to McManus [5], active learning is one in which 'The instructor strives to create a learning environment in which the student can learn to restructure the new information and their prior knowledge into new knowledge about the content and to practice using it', however, in passive learning 'Students are assumed to enter the course with minds like empty vessels or sponges to be filled with knowledge'. From this perspective, the senior graduation project (SGP) courses provide such a useful basis that can suitably be structured to create active learning environments, where the students can learn from each other and restructure their acquired and a priori knowledge via product development practices. When it is well designed, the SGP course provides a good framework for active learning that is considered to be one of the best learning methods [6, 7]. In fact, project-based learning (PBL) by itself is known to be one of the basic settings for suitable university course teaching [8–11]. Project-led education is a successful way of improving the motivation and progression rate of students through a given academic programme and is being adopted in many European countries [12]. In order to accomplish their future engineering missions efficiently in professional situations, emerging engineers need to rely not only on a whole body of scientific and technical knowledge, but also on a wide set of individual and group skills [13]. Furthermore, by taking both cognitive and affective learning into consideration, one can establish a PBL model to create an active learning context and to provide an opportunity for the development of collaboration, communication, co-operation and management of the learners [14].

In any project, teams are the basis for the organization of development because the increasing complexity of projects has made this unachievable by individuals. Development teams commonly distribute the work between their members by following well-defined structures of interdependent responsibilities, with typical roles such as designers, testers, architects and project managers [15, 16]. While at university, computing curriculum undergraduates should be prepared for teamwork. This can be achieved by taking either a capstone [17] or a focused core [18] software development course. As an alternative, student development groups can be established to work on real projects [19]. Senior graduation projects are known to provide students with an opportunity to learn about working in groups and applying theory to practice [20]. As a consequence, SGP course can be designed as a capstone course in which interaction and working as a team are facilitated in an active learning environment.

In this paper, we propose an SGP course designed for a typical computing curriculum in which an active learning environment establishment is the main concern. The course can be implemented as part of the curriculum of any computing related university undergraduate department including computer science, computer engineering, information systems, information technology and software engineering. Note that for each of these five disciplines, software production through, for example, efficient algorithm design and development, embedded system development or component integration can be the main concern of their SGP course implementation [21]. In practice, the designed course has been implemented and executed as a two-semester must-course in the curricula of the Computer Engineering and Software Engineering departments of Atilim University, Ankara, Turkey. Based on our experience gained over the last three years, the course satisfaction of the students has been observed to increase as indicated by their responses to the prepared questionnaire. Also, based on the evaluations and critiques performed by an unbiased group of academicians, we conclude that the realized products are high quality when compared with typical student products developed in a passive learning mode.

Our study differs from previous studies in a number of aspects:

1. This paper focuses on developing an active learning environment for SGP courses in computing curricula. The main discussion is not the active learning, but its effective implementation in a specific discipline. Therefore, this study is different from the studies [2, 6–10, 22, 23] that focus on different active learning issues.
2. There are some studies that discuss SGP course implementation for different disciplines [24, 25]. However, it is impossible to implement the same techniques in computing-related departments. Although it is possible to benefit from those studies, a new design is needed for computing disciplines.
3. There are also a few studies [17, 20, 26] related to SGP course (or project course) implementation of computing disciplines. However, this study proposes a different approach to improve the interaction between supervisors and students, team-working performance, grading fairness, and the preparation of students for professional life. To the best of our knowledge, there is no such study that has fully implemented the same techniques for SGP courses for the computing curricula.

2. MOTIVATION AND REQUIREMENTS

Looking at modern engineering education, the required skills base is no longer just technological; it now includes a demand for the graduate to be proficient in open-ended problem-solving and applications [27]. SGP courses are known to be important components of any computing curricula because of the expected role-/team-based student activities and the need to handle relatively large scale open-ended computing problems that are different from other small-sized regular course projects. In many universities, the SGP course is one of the last and best chances for students to prepare themselves for professional life. The entire course is a project in which various topics learned from other courses or learned during the course itself are practised via student-level projects. Based on our experience of previous SGP course implementations, we identified the following requirements that constitute the basis for the motivation behind the establishment of an efficient active learning environment for the course:

2.1 Two-way Communication

Two-way communication is already known to be a critical component of any active learning process. According to Modell and Michael [28], in an active learning process, faculty become facilitators of learning and students become active participants, engaging in a dialogue with their colleagues and the instructor. In a typical SGP course, the interaction between supervisor and students is high at the beginning of semester, but slows down after several weeks and starts to increase towards the end of the semester. This interaction should be kept high during the whole semester in order to establish an efficient active learning environment.

2.2 Higher Student Involvement

One of the basic problems with attaining project success was related to the establishment of sustainable team performance throughout the project. Team performance degradation is thought to be due to decreased individual student involvement during a project's lifetime. Generally, some students lead in projects and the others give marginal support, but they all get the same grade. Therefore, we need to develop methods to increase individual student involvement to the projects.

2.3 Mentoring

For most student projects, it is observed that development tools (or platforms) are necessary for high quality product development. However, students may not have a basic knowledge of such tools or platforms. Mentoring, on the other hand, is known to be a necessary component for setting active learning environments. It is necessary to guide students and give some technical information. However, this help should not be directly related to the solution of the given problem. It is the students' project team that will develop a solution for the given problem. We need to take great care to avoid the mentoring effort be turned into a typical passive instruction, since active learners are expected to strive energetically to take greater responsibility for their own learning [30].

2.4 Becoming a Professional

Students generally do not have a clear idea about the professional life that they will soon encounter. In general, they can only guess the expectations of industry based on a model that they have constructed throughout their university education. So, 'turning students into apprentices on the way to becoming professional (or colleagues) [5]' should be one of the main aims of the SGP course design.

2.5 More Cooperation

Lack of cooperation between team members throughout a project has been observed to be one of the main reasons behind failure. 'Collaboration with other students to discover and construct a framework of knowledge that can be applied to new situations' constitutes an example expectation from students in an active learning environment [5]. We know that active learning course settings affect the development of team competence at the individual level [31].

2.6 Grading Fairness

One of the important complaints from students is the unfairness of grading in a typical SGP course. Because many individual faculty members give projects to the students and different instructor groups evaluate the results. Students' teams work on different projects and different juries assess the results. It is impossible to establish a fair grading system in such an environment. Therefore, the grading system needs to be improved.

3. COURSE STRUCTURE AND PROCESS

The SGP course is aimed at including as many professional project activities as possible while adopting the steps of typical engineering product development life-cycle. The course process model is given in Fig. 1. The course starts with team formation and project assignments. Students work on a project throughout the whole year (two semesters) to develop a software product. Product development is achieved according to known development methodologies. Students produce reports consistent with real-world projects and make presentations and demos during the development process. At the end of the year, they present their final product. They are assessed by internal project supervisors, who take into account their reports, presentations, demos and the final product. In addition, their projects are cross-evaluated by examiners from university and industry at the end of the year, which is shown as external grading.

During the execution of the SGP course, a project-based active learning model is adopted.

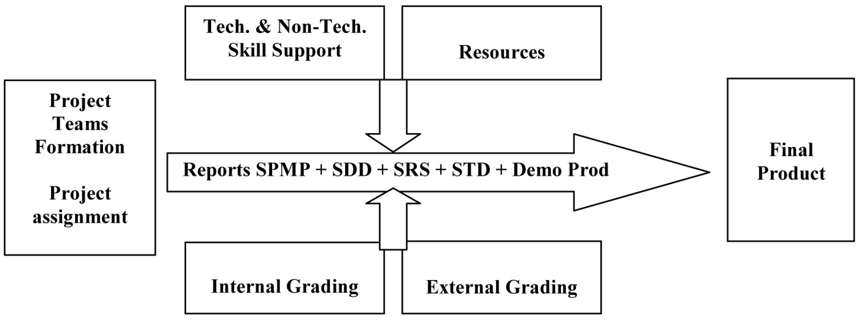

Fig. 1. Course process model.

In this way, student learning is organized around problems and carried out in projects. A different problem is assigned to each project team and they are supposed to solve it by developing a programme that applies a software development process model. The problem provides the starting point for the project development process, places learning in a context, and bases learning on the learner's experience. Learning is also project-based in such a way that students have to work on a unique task involving complex and situated problem analyses and problem-solving strategies.

The adoption of role-based development throughout the SGP course contributes to the students' professional attitude development. The roles may include user/customer, project manager, system architect, developer, tester, quality manager and configuration manager. The students can play more than one role simultaneously.

In the SGP course there are two presentations made at the end of each semester. In presentations, students are expected to introduce and give information about their project content and its current status. Getting feedback from other students and supervisors is an important part of the presentations. We recommend that the presentations be completed in a block of three or four days made one week before the end of semester examinations.

Each team deploys a website for their projects. The minimum published material requirements for project websites are: project documentation, content of presentations and prototypes. Other than these, groups can either deploy their applications directly into their website or an installable version of their products can be uploaded. Furthermore, information about group members and the application domain of the project can be useful. The students should consider the SGP course project websites to be good references that can be useful in professional job applications after graduation.

Project documentation is known to be an important project component since it contributes to project memory and establishes a common view for involved shareholders. It also forms a basis for a standard and fair evaluation and grading of the project outputs. Supervisors can partly monitor project development by examining versions of Software Project Management Plan (SPMP), Software Requirement Specification (SRS), Software Design Document (SDD) and Software Test Document (STD). These four living documents are required to be updated by students and published on the student project website for supervisors' evaluations. In the SGP course implementation, we adopted a documentation firmly based on IEEE standards [32–36]. The standards provide guidance for both the required format and content of documentation deliverables. They also constitute a framework for the evaluation of the projects based on deliverables. Since the documents are living documents, their grading needs to be done incrementally rather than once by the end of each semester. The document evaluation period is announced at the beginning of the semester. At each evaluation the latest versions of whole documentation are considered.

Five or six weeks of the first semester class hours are reserved for providing technological brush-up sessions for the students. An assistant is assigned and he or she is expected to guide the students throughout the sessions. The course assistant is also responsible from the establishment of IT related infrastructure setup for the course including making software installations, administrating and backing up for the course website and course laboratory environment.

Course design requires a prototype development and its presentation by the end of the first semester. The prototype improves the communication between students and their supervisor. Depending on the project, the prototypes can evolve into fully-featured products. However, the main expectation from prototyping is to motivate the students in implementation and coding by means of Graphical User Interface (GUI) development, if applicable. The prototypes are demonstrated at the end of first semester project presentations. In their grading, the coverage of the user requirements defined in SRS shown through user interface, rather than the implemented functionalities, is the main concern.

Intermediate product code reviews and the code review for the final product support communications between students and the instructor.

They are also useful for fair grading and better evaluation of both individual and team performances. Although the product quality is an important concern in code reviews, the main focus is on the measurement of students' performances. For this purpose, we developed a form for student coding/testing effort evaluation. During product code reviews, we look at: the timely appearance of product demonstration; whether the team is able to set up the product review environment or not; the amount user requirements that are satisfied; the development and/or testing of the maturity level and success of individual students in product modification tasks. Note that a student can play simultaneously more than one role (e.g. developer and tester).

For promoting competition between project teams, the best product contest is organized at the end of the second semester. Any volunteer team with its finalized product can join the contest. The products are evaluated against some predefined and announced criteria. The evaluation of the candidate products is made by an unbiased group formed of senior industry experts and academicians who do not have any direct involvement in the execution of the SGP course. Their evaluations are considered only in the context of the best product contest at which teams present and demonstrate their product to the jury.

The promotion of teamwork and evaluating team performances is an important aspect of the course. Therefore, an award for best teamwork is announced, to encourage the teams to perform better, in addition to best product awards. For this award, each supervisor can decide on the best teamwork performance shown in the groups that he/she supervised throughout the academic year. Notice that the main focus in the selection of best teamwork is the teamwork quality rather than the product quality and/or possible technical achievement(s). The evaluation criteria for this award may include the team synchronization and harmony shown throughout the year, the individual success of members in achieving tasks and in role playing, the management of the degree of project load balancing between team members, the successful completion rate of the project in terms of user requirements, and the problem solving capability of teams.

4. COURSE MANAGEMENT

The participants in the SGP course are depicted in Fig. 2. The lines between the actors show the existence of two-way interaction. The supervision of the SGP course needs a well organized supervision team comprising three or four instructors,

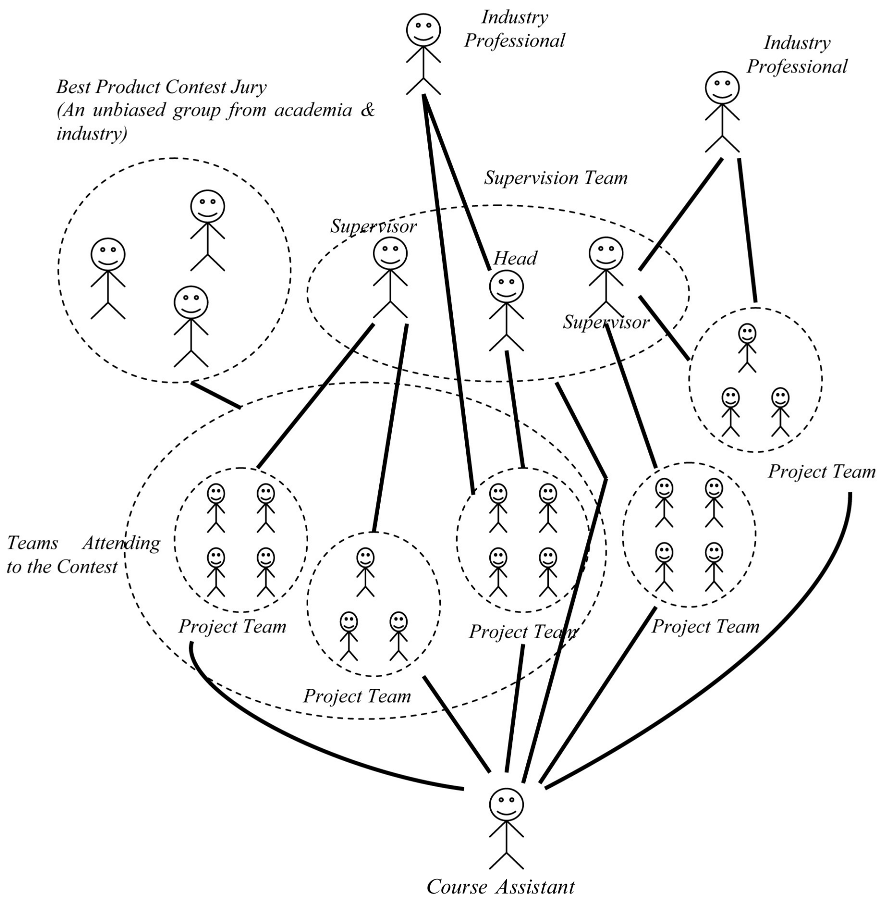

Fig. 2. SGP course actors.

depending on the number of student teams. One of the supervisors is elected as head of the supervision team. The team head is the coordinator of any activity related to the course. The goal of such a supervision team is to facilitate the unity, synchronization, timeliness and to establish fairness in supervision, assessment and evaluation throughout the course. Notice that the cooperation, synchronization and harmony of the supervision team are expected to result in a good professional team model for the students as well. It needs special consideration and care from pedagogical perspectives.

After its establishment, the supervision team starts a course execution process realized via packaged tasks. The tasks to be done before official kick-off of the student projects include the following.

- *Construction of official course website*: It is an important official medium of communication between students and their supervisors.
- *Deciding on milestones and deadlines of the course and preparing a course execution schedule*: The milestones and deadlines are the critical part of the course execution. Their official announcement is especially important to set up synchronization between members of the project and supervision team.
- *Developing a team/individual performance assessment policy and setting up/announcing a detailed course grading criteria.*
- *Deciding on project topics and their short contents in the form of an initial project contract*: The project topics can either be announced by supervisors or proposed by students. The proposals are nothing but a description of initial/rough user requirements.
- *Matching project topics to the student formed project teams*: Fairness and transparency in matching the topics (projects offered by supervisors) to the project teams is very critical. For this reason, the procedure for the matching purpose should be defined clearly as early as the first official course class hour.

The project topic and its initial content described in the Senior Project Proposal Document are the basis for the later user requirements specification document. The form includes the title of the project; a short user requirement description, keywords and subjects describing the project, tools, methods and techniques that are expected to be used throughout the project, and the output of the project. If it is proposed by a supervisor, the prerequisite courses to be taken (if any) should be indicated. Otherwise, if it is proposed by a student team, the names of the project team members and a paragraph justifying the proposal should also be written in the form. The proposed projects can be small-sized industrial projects or a part of a large industrial project. It is the supervision team's responsibility to determine and to announce sufficient project topics for the student teams who do not have their own project proposal. The projects proposed by students are evaluated by the supervising team. The proposed project can either be accepted as is, sent back with some recommendations for modification or directly rejected with the reason for rejection. If rejected, the team may select one of the projects proposed by the supervisors.

The main tasks to be performed by project team members after official kick-off include the following.

- Preparation of Software Project Management Plan (SPMP)
- Preparation of Software Requirement Specification (SRS) Document
- Preparation of Software Design Document (SDD)
- Preparation of Software Test Document (STD)
- Prototype development and demonstration
- Project presentation (at end of both semesters)
- Implementation (code writing)
- Testing
- Product presentation and demonstration (at the end of second semester).

Face to face communication between supervision team and student project teams is important. Throughout the course, interactions between students and supervisors occur in three different forms:

1. Getting together during regular weekly class hours
2. Students have an open door policy with instructors concerning any matter
3. Communicating via project websites/e-mailing.

For example, when project teams load their documents onto their website, supervisors can see and download the documents, and evaluate them and give the necessary feedback to improve them. Two class hours per week are allocated for student–supervisor interactions. During class hours, students are informed about course requirements (e.g. things to be considered in project team formation, preparation of project documentation, announcements about upcoming activities and deadlines, feedbacks from evaluations). Class hours are also used to answer questions from the students that have arisen during project steps. The course assistant helps the supervisors to manage the course. His/her main responsibility is mentoring technological skill-brushing sessions. Note that supervisors and course assistants are neither problem-solvers nor cooperative partners of the student teams. Supervisors are expected to play the user/customer role, give necessary educational feedback and to make their assessments/evaluations through grading. Critical project decisions, for example, methodology/method selection, finalizing the system architecture, deciding on the tools to be used, and project role assignments are all expected to be initiated by the students themselves.

Table 1. First semester assessment

Evaluation parameters	Percentage to the whole grading (%)	Evaluation type (Team / Individual)	Evaluator
SPMP (1st evaluation)	10	Team	Supervisor
SRS (1st evaluation)	10	Team	Supervisor
SDD (1st evaluation)	15	Team	Supervisor
STD (1st evaluation)	10	Team	Supervisor
SPMP (2nd evaluation)	5	Team	Supervisor
SRS (2nd evaluation)	5	Team	Supervisor
SDD (2nd evaluation)	5	Team	Supervisor
Prototype	15	Team	Supervision team
Presentation	15	Team	Supervision team
Project website	5	Team	Supervisor
Attendance	5	Individual	Coordinator
Questions asked during presentations (bonus)	For each question: Good: 0.50 pt Moderate: 0.25 pt Unnecessary: 0.00 pt	Individual	Supervision team
Supervisor bonus	4 pt	Individual	Supervisor
Total	**100 +**		

5. EVALUATION AND GRADING

The supervision team is the main responsible authority for both individual and team performance evaluations made throughout two consecutive semesters. Evaluation of the first semester is mainly based on the evaluation of documents, prototype and presentation. The documents (SPMP, SRS, SDD and STD) include the outputs of the project management, requirement analysis, system design and test planning activities. A prototype development is also included to help project teams identify their missing requirements and design mistakes. Project website quality and class attendance are two other assessment criteria. Details of the first semester grading are given in Table 1. The third column shows evaluation type either as team or individual, since some parameters are appropriate for team-wide evaluation and some are more appropriate for individual evaluation. Some parameters are evaluated only by the supervisor while some others are evaluated by the supervision team as shown in the last column of the table.

In Table 1, the first four rows represent the project documents delivered consecutively on different dates during the semester. After each document delivery, supervisors evaluate the submitted document and grade. In addition, supervisors indicate the mistakes or missing parts of the documents and give them to students as feedback. Students are expected to go on improving their documents until a given date. The second evaluations are shown on the 5–7 rows of Table 1. Since STD is delivered only at the end of the semester there is no second evaluation for this document. Presentation of the project, including the prototype, is done at the end of the semester and it is graded immediately by the supervisors. Each team member has to explain his/her role in the presentation actively. Prototypes are allowed to be either screen designs or draft screens of the potential product obtained using any programming tool. After the presentations, a question–answer period is given to the students for asking questions related to the presented project and they can get bonus points depending on the validity of the questions. The aim of this grading is to increase student involvement and to motivate them to focus on the presentations. The supervisor bonus is awarded by each supervisor to distinguish the students who have made high contributions and it constitutes a subjective part of the grading.

The assessment of the second semester is completely different from that of the first, because the main activities of the second semester are code writing and testing. Therefore, the evaluation of the student projects focuses on these activities and the product. The evaluation criteria for the second semester are given in Table 2. The first three grading, i.e., code reviews, is done in the last week of every four-week period. In this way, supervisors are able to follow the developments on the projects.

At the end of the second semester, teams make a presentation and a technical demo for their completed project. The expectation is that all the user requirements are satisfied and a fully functional software program is developed and tested. For the technical demo, the contribution of each student is evaluated similarly to the code review assessment. Presentation of the project contains not only second semester activities, but also the first. Another assessment criterion is the product and document compatibility. The assessment must be done according to the documents prepared during first semester. There may be some changes in the user requirements or in the design. In such cases, the project teams have to update their documents and submit them before the technical demo. After the technical demo, supervisors evaluate compatibility of the documents with the product. The quality of the product is another parameter for evaluation. Although all of the

Table 2. Second semester assessment

Evaluation parameters	**Percentage to the whole grading (%)**	**Evaluation type (Team / Individual)**	**Evaluator**
Code review (1st evaluation)	15	Individual	Supervisor
Code review (2nd evaluation)	15	Individual	Supervisor
Code review (3rd evaluation)	15	Individual	Supervisor
Presentation	10	Team	Supervision team
Technical demo.	20	Individual	Supervisor
Compatibility between documents and product	10	Team	Supervisor
Quality of the product	10	Team	Supervisor
Attendance	5	Individual	Coordinator
Supervisor bonus	4	Individual	Supervisor
Total	**100**		

user requirements are satisfied, the quality of the product may still be poor. For example, it may not be user-friendly, efficient, appealing, etc. Attendance taken from class activities, seminars and final presentations is converted to a score out of 5. A supervisor bonus is given similar to the first semester.

6. DISCUSSIONS

The proposed approach enhances active learning by giving more responsibility to the students about their learning, working in a well-formed team, writing reports, implementing and testing a part of a project, presenting and demonstrating their work in the project. Here, giving more responsibility implies that supervisors are not directly involved in the project developments of the student teams. Instead, they act like customers for the product and provide user requirements to the students. SRS documents are accepted as a contract between customer (i.e. supervisors) and developers (i.e. students). During the execution of the course, although some advice about the projects is given by supervisors, the final decision is always made by the project team. In order to assess the proposed model, it is best to look at how the requirements given in Section 2 were satisfied. The requirements and the related actions are summarized briefly below:

6.1 Two-way Communication

In our design, in order to produce better products, user/customer roles are taken by instructors, which resulted in the requirement of increasing two-way communication/interaction between them and their supervisors. One step further, in our SGP course design the user/customer role is also played by true end-users or by professionals from an industry for some projects. (This was optional for students.) Life-cycle driven development which forces students to work continuously has a positive effect in increasing the interaction between supervisors and students. Organizing weekly class hours, submitting reports and giving prompt feedback, and making presentations and demos are other methods of keeping the two-way communication level high. In addition, the established two-way communication environment is also supported by seminars given by experts from industry to facilitate the interaction between students and professionals.

6.2 Higher Student Involvement

In order to provide higher student involvement in the projects, we wanted the students to practice one process model strictly from the beginning to the end of the project. This was the critical motivating factor behind the product *life-cycle driven* development requirement for the student projects. This leads to higher student involvement because of the more systematic documentation required by the selected development methodology. Note that the approach also contributes to the students becoming professionals. As a consequence of this solution, student knowledge is directly experienced, constructed, acted upon, tested and revised by the students themselves [29].

6.3 Mentoring

In the proposed implementation, the supervisors guide the student teams when they require help. In addition, technological skill-brushing sessions mentored by a course assistant in a lab environment is another support given to the students. However, supervisors and course assistants never become members of the project teams. They do not intervene in the solutions of the problems.

6.4 Becoming a Professional

Our related action to satisfy this requirement is to enforce students to adopt a *role-based* project development approach that contributes to early career shaping for the students. Playing roles such as system architect, developer and tester and taking related responsibilities are thought to be the first step in becoming a true professional. In addition, in some projects the customer role is played by real customers or by experts from the industry. This decision also contributes to the development of the students' professional perspective. As a consequence, he/she becomes an adaptive self-learner in a semi-professional project development environment. Note that good professionals are adaptive self-learners. The seminars given by professionals are orientation programmes to prepare students for professional life.

6.5 More Cooperation

In our proposal, we tried to motivate teamwork by applying a role-based approach and assessing the contributions of team members individually. With the role-based approach each student has specific tasks to achieve in the project. In the grading, some individual assessments are done to evaluate the individual contributions of team members. If the role-based approach is not applied, some students may lead the project and others can obtain the same grade by standing by. Note that we also evaluate team performance independently of individual performances. In this way, students cooperate in favour of their teams for their relative performance compared with the other teams, indirectly. In addition, in order to promote the cooperation between team members, we set up a best product contest between teams and observed some positive effects of inter-team competence over intra-team cooperation.

6.6 Grading Fairness

When there are too many supervisors involving to the SGP course, it is difficult to obtain a fair grading. In this approach, since the course is managed by a limited number of instructors, it is easier to coordinate the grading policies. Some of the activities of all the teams are already evaluated by the same supervisor team, which provides a fairer grading.

In order to assess our proposal further, we can also compare our proposal with well known studies in the literature. For example, the seven principles proposed by Chickering and Gamson [37] can be used for this purpose. Here, we comment on whether our proposal is compatible with each of the good practice principles. Note that the first principle is already about '*encouraging active learning*', which is the main goal of our work. The next two items namely, '*encouraging student-faculty contact*' and '*encouraging cooperation among students*' are also directly compatible with our main requirements. The last four principles, on the other hand (*Giving prompt feedback, Emphasizing time on task, Communicating high expectations, Respecting diverse talents and ways of learning)*, are not primarily/directly related to the basic requirements of the proposed model. However, the implementation mainly covers these principles. In our approach, giving feedback becomes a continuous effort throughout the whole year. In the first semester, the submitted student reports are evaluated and timely feedback is given for improving them. In the second semester, code reviews are performed to assess the progress of the students and on time feedback is given to improve the quality of their implementations. In regular class hours, feedback related to general mistakes or missing points is provided. In our approach, a schedule including all the activities and due dates is declared at the beginning of each semester. Students must strictly obey the declared schedule. On late delivery of the reports, the given grade is reduced proportionally to the delay, which is a kind of penalty for inefficient time usage.

During execution of the course, we tried to motivate students by increasing their self confidence. Since they work in teams, some well motivated students lead the others to perform better. A competitive environment is also created between teams by organizing a best product contest. The organized seminars that are given by experts from industry had the positive effect of increasing the motivation of the students. Finally, students need the opportunity to show their talents and learn in ways that work for them. The SGP course provides students with a different way to learn and develop themselves for real life. Many students with low CGPA perform better in this course, because they want to show their talents and achieve high performance. A role-based approach gives students an opportunity to show their talents in different project tasks.

In our discussion, we also need to consider the feedback from the students about the course. Although the proposed approach places a heavy burden on both instructors and students, performance and course satisfaction have been observed to increase based on students' responses to a prepared questionnaire. At the end of each semester, students fill out a course evaluation form that included 19 different evaluation criteria. Each criterion is evaluated on a scale of between 1 and 5, where 1 shows the lowest satisfaction level while 5 represents the highest satisfaction level. The last criterion in the evaluation form is the 'overall satisfaction level'. Figure 3 compares the students' overall satisfaction level of the SGP course with the overall satisfaction level of the same year. As seen in the figure, at level 4 and 5, which represents the two highest levels, students are more satisfied than the average satisfaction level of the departments. The percentage of the students at level 5 is 54% of the students enrolled to the SGP course. Another important indicator that shows student satisfaction is the number of complaints raised to the management of the department. Although there were at least several complaints about different issues related to the SGP course in previous years, this implementation cleared all the complaints and management did not receive any complaints after implementation of the proposed approach.

Furthermore, in order to assess and work out the product and project qualities of the classical

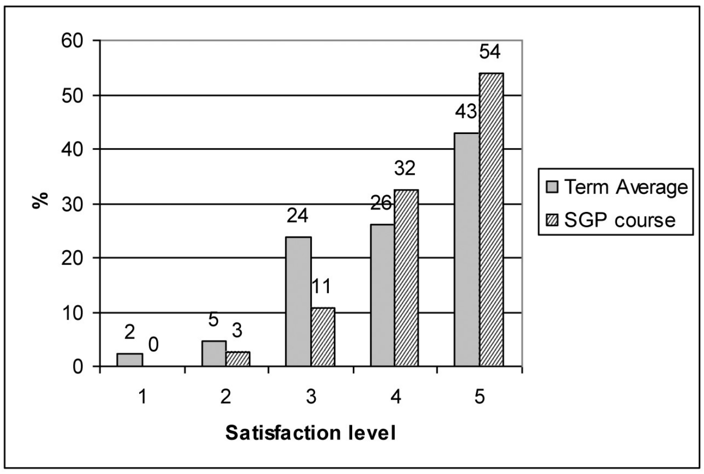

Fig. 3. Student overall satisfaction level.

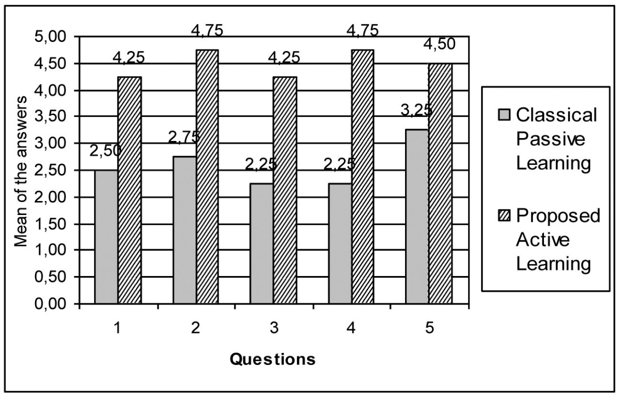

Fig. 4. Unbiased academician group evaluation results for classical passive and proposed active learning settings.

passive and the proposed active learning based SGP approaches, we prepared and applied a questionnaire to an unbiased group of academicians from Computer Engineering, Software Engineering and Information Systems Engineering departments of the Atilim University. All the group members had experience of both classical and the new SGP course settings. We asked five questions of a typical student product, measuring:

1. the potential market/industry value;
2. the degree of completeness;
3. being well-documented;
4. the required 'supervision effort';
5. the frequency of supervisor–student team interactions throughout the projects.

Also, they were asked to write down the observed advantages and disadvantages of the new proposal.

Figure 4 depicts the answers for each question. Based on the calculated results, we can say that the proposed active learning set-up leads to student products having 'higher market value', being 'much more complete' and 'better documented'. Also, it enables more 'supervisor'/'student team' interactions. However, the required supervision effort is higher than in the classical set-up. The responses to the advantages and disadvantages of the proposal are given in Table 3.

The indicated advantages are clear. However, it is also clear that the new SGP set-up is much more bureaucratic than the old one. This may cause real trouble, especially through the end of semesters. Another problem is to find dedicated supervision staff to execute the set-up. If the number of academic staff in the supervision team is small, the competition between project teams (and indirectly between supervisors) is lowered. The last disadvantage is due to the inherent difference (or incompatibility) between scientific research methodology and computing system design and development methodology. In the new set-up, those students that had greater theoretical involvement and focus cannot develop and show their academic capabilities via the SGP course. We believe that any SGP course set-up should include this perspective and we are planning to make the necessary changes and arrangements in the SGP course set-up in the following semesters.

In spite of the difficulties pointed out by the unbiased group of academicians, the SGP proposal provides a smooth transition from student to professional life. The contribution of the proposal

Table 3. Indicated advantages and disadvantages of the SGP proposal by an unbiased group of academicians

Advantages	Disadvantages
1. Very valuable in student's preparation for real-life projects	1. Much more bureaucratic
2. Much more product oriented	2. Requires dedicated staff
3. Systematic, methodological and complete development and well-documented	3. Limited number of involved academic staff
4. Increased communication between supervisor and student teams	4. Lowered team competition
	5. Does not enable the students to perform any academic research activity

to the students' self-development is much more valuable and greater than in the old SGP course management.

6. CONCLUSIONS

The focus of engineering education should be on solving the increasingly complex problems of rapidly evolving contexts (e.g. human resources, quality requirements, standards, time and cost constraints). As a consequence, the establishment of active learning in the SGP course environment for computing curricula becomes a necessity. We have presented an approach to effectively manage an SGP course in any computing curricula via active learning experience. It is observed that the implementation of the proposed approach increases student course satisfaction level, while higher quality student projects are achieved. Furthermore, an assessment made by an unbiased group of academicians of the product and project qualities of the experienced classical passive and the proposed active learning based SGP approaches showed that both qualities are higher in the proposed SGP set-up. Note that the project, which was selected as the best product in the contest in 2007–2008 academic year was awarded $70 000 support from the government to convert it to a commercial product. Also, one of the student projects successfully reached the finals of the Microsoft Imagine Cup 2008 University Software Project Competition in Turkey. It is clear that the unit of success of the student can not be CGPA or USD but it is his/her contribution to the charity, welfare and peace of the world via his/her later developed processes and products.

Acknowledgements—The authors would like to thank Mr. Elzie Cartwright (late) from Atilim University Academic Writing & Advisory Center for his manuscript polishing effort.

REFERENCES

1. V. V. Fedorov, *Theory of Optimal Experiments*, Academic Press, San Diego, 1972.
2. R. M. Castro and R. D. Nowak, Minimax bounds for active learning, *IEEE Transactions on Information Theory*, **54**(5), 2008, pp. 2339–2353.
3. L. N. Castro, *Fundamentals of Natural Computing: Basic Concepts, Algorithms and Applications*, Chapman & Hall/CRC, Taylor & Francis, Boca Raton, FL, 2006.
4. P. Sancho-Thomas, R. Fuentes-Fernández and B. Fernández-Manjóna, Learning teamwork skills in university programming courses, *Computers & Education,* **53**(2), 2009, pp. 517–531.
5. D. A. McManus, The two paradigms of education and the peer review of teaching, *Journal of Geoscience Education*, **49**(5), 2001, pp. 423–434.
6. C. Meyers, and T. B. Jones, *Promoting Active Learning, Strategies for College Classroom*, Jossey-Bass Publishers, San Francisco, 1993.
7. C. C. Bonwell and J. A. Eison, *Active learning: Creating Excitement in the Classroom*, ASHE-ERIC Higher Education Report No. 1, The George Washington University, School of Education and Human Development, Washington, DC, 1991.
8. J. W. Thomas, A review of research on project-based learning, http://www.autodesk.com, (Accessed 27 October 2009).
9. BIE, Project Based Learning, http://www.bie.org/index.php/site/PBL/ overview_pbl/, Buck Institute of Education (BIE), CA, USA, (Accessed 20 May 2010).
10. P. K. Hansen, Does productivity apply to PBL methods in engineering education?, *International Journal of Engineering Education*, **19**(1), 2003, pp. 177–182.
11. M. Lehmann, P. Christensen, X. Du and M. Thrane, Problem-oriented and project-based learning (POPBL) as an innovative learning strategy for sustainable development in engineering education, *European Journal of Engineering Education*, **33**(3), 2008, pp. 283–295.
12. P. C. Powell, Assessment of team-based projects in project-led education, *European Journal of Engineering Education*, **29**(2), 2004, pp. 221–230.
13. S. Rouvrais, J. Ormrod, G. Landrac, J. Mallet, J. M. Gilliot, A.Thepaut and P.Tremenbert, A mixed project-based learning framework: preparing and developing student competencies in a French Grande Ecole, *European Journal of Engineering Education*, **31**(1), 2006, pp. 83–93.
14. E. D. Graff and A. Kolmos, Characteristics of problem-based learning, *International Journal of Engineering Education*, **19**(5), 2003, pp. 657–662.
15. G. Benarek, W. Zuser and T. Grechenig, Functional group roles in software engineering teams, *Proc. of the ACM 2005 Workshop on Human and Social Factors of Software Engineering (HSSE'05),* 2005, pp. 1–6.

16. M. Barak, T. Maymon and G. Harel, Teamwork in modern organizations: Implications for technology education, *International Journal of Technology and Design Education,* **9**(1), 1999, pp. 85–101.
17. M. Mitri, A software development capstone course and project for CIS majors, *Journal of Computer Information Systems*, **48**(3), 2008, pp. 1–14.
18. L. Layman, L. Williams, K. Slaten, S. Berenson and M. Vouk, Addressing diverse needs through a balance of agile and plan-driven software development methodologies in the core software engineering course, *International Journal of Engineering Education,* **24**(4), 2008, pp. 659–670**.**
19. R. Casallas and N. Lopez, An environment to help develop professional software engineering skills for undergraduate students, *International Journal of Engineering Education*, **24**(4), 2008, pp. 648–658.
20. M. Shaw and J. E. Tomayko, *Models for Undergraduate Projects Courses in Software Engineering*, Technical Report, Carnegie Mellon University, 1991.
21. ACM, AIS and IEEE-CS Task Force, (2005). Computing Curricula Overview Report. http://www.acm.org/education/curric_vols/CC2005-March06Final.pdf, (Accessed 25 May 2009.
22. J. Marcos-Acevedo, S. Pérez-López, J. Sánchez-Real, R. Alvarez-Santos and M. Suárez Alvarez, Active learning approach for engineering in collaboration with the corporate world, *International Journal of Engineering Education*, **25**(4), 2009, pp. 777–787.
23. A. McKay and D. Raffo, Project-based learning: a case study in sustainable design, *International Journal of Engineering Education*, **23**(6), 2007, pp. 1096–1115.
24. I. M. Verner and E. Hershko, School graduation project in robot design: a case study of team learning experiences and outcomes, *Journal of Technology Education,* **14**(2), 2003, pp. 40–55.
25. A. A. Polat, Importance of graduation project in the education of agricultural engineering: case studies from Turkey, *IV Int.Symposium on Horticultural Education, Extension and Training*, Perth, Australia, 2005.
26. E. Scott, Systems development group project: A real-world experience, *Information Systems Education Journal*, **4**(23), 2006, pp. 3–10.
27. A. Andersen, Preparing engineering students to work in a global environment to co-operate, to communicate and to compete, *European Journal of Engineering Education*, **29**(4), 2004, pp. 549–558.
28. H. I. Modell and J. A. Michael, *Promoting Active Learning in the Life Science Classroom*, New York Academy of Sciences, 1993.
29. J. G. Thompson and S. Jorgensen, How interactive is instructional technology? Alternative models for looking at interactions between learners and media, *Educational Technology*, **29**(2), 1989, pp. 24–26.
30. N. A. Glasgow, *Doing Science: Innovative Curriculum for the Life Sciences*, Corwin Press, 1996.
31. R. Motschnig-Pitrik and K. Figl, Developing team competence as part of a person centered learning course on communication and soft skills in project management, *37th ASEE/IEEE Frontiers in Education Conference*, Milwaukee, USA, 2007, pp. 15–21.
32. IEEE Std. 1016, IEEE Recommended practice for software design descriptions, International Telecommunications Union, Geneva, Switzerland, 1998.
33. IEEE Std. 1058.1, IEEE standard for software project management plan, International Telecommunications Union, Geneva, Switzerland, 1987.
34. IEEE Std. 829, IEEE Standard for software test documentation, International Telecommunications Union, Geneva, Switzerland, 1998.
35. IEEE Std. 830, IEEE Recommended practice for software requirement specifications, International Telecommunications Union, Geneva, Switzerland, 1998.
36. D. Delaney and S. Brown, Document templates for student projects in software engineering, *Technical Report NUIM-CS-TR2002-05*, 2002.
37. A. W. Chickering and Z. F. Gamson, Seven principles for good practice in undergraduate education, *American Association for Higher Education (AAHE) Bulletin 39*, 1987, pp. 3–7.

Hurevren Kilic is an Assistant Professor Dr in the Computer Engineering Department of Atilim University, Ankara, Turkey. He has B.S., M.S. and PhD degrees from the Middle East Technical University, Ankara, Turkey obtained in 1989, 1992 and 1998, respectively. He has eight years of academic research and teaching experience and twelve years of industry experience in government and university software system development. His current academic research interests include: intelligent agent technologies, multi-agent systems, industrial informatics, ecological informatics, nature inspired computation and graduation project design for computer/software engineering education.

Murat Koyuncu is currently an Assistant Professor in the Department of Information Systems Engineering at Atilim University, Ankara. He received his Ph.D. degree in computer engineering from the Middle East Technical University, Ankara, Turkey, in 2001. His research interests include fuzzy logic, object-oriented databases, knowledge-based systems, multimedia databases and computer networks.

Mohammad Rehan is an Associate Professor of Computer Engineering at Atilim University, Ankara Turkey. His areas of interest are in MIS, eSCM, eCommerce, eGovernment, and public eProcurement. He has extensive experience in curriculum development, distance education related to computers and management courses. He is author of the book, "*eProcurement: Supply Chain Management*". His publications are in refereed journals and conferences.