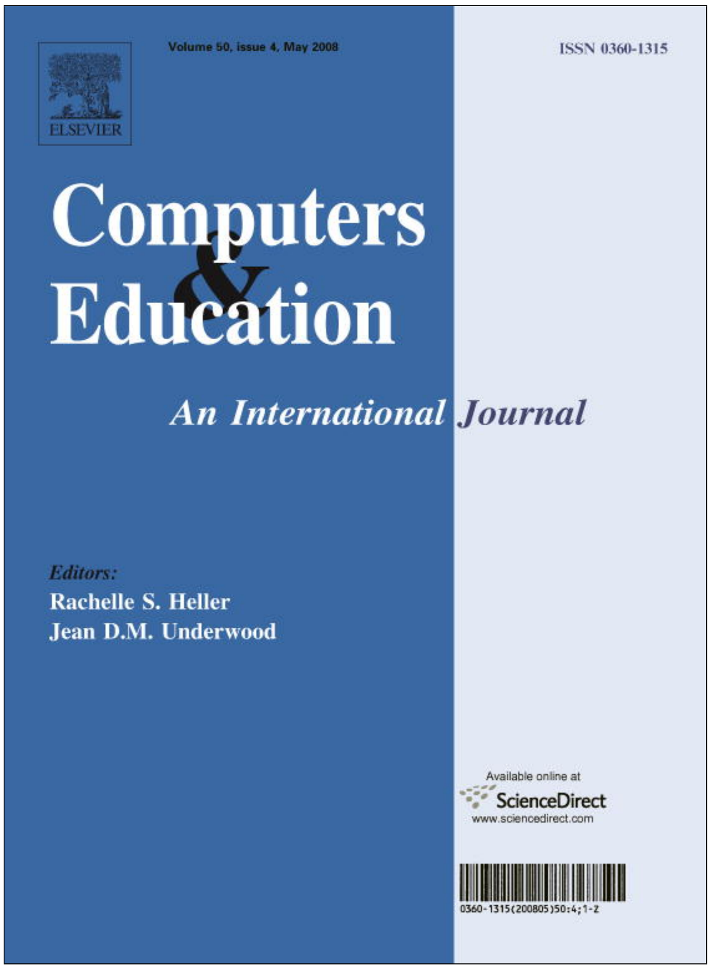

Volume 50, issue 4, May 2008
ISSN 0360-1315
ELSEVIER
Computers & Education
An International Journal
Editors:
Rachelle S. Heller
Jean D.M. Underwood
Available online at
ScienceDirect
www.sciencedirect.com
0360-1315(200805)50:4;1-2

Available online at www.sciencedirect.com

ScienceDirect

Computers & Education 50 (2008) 1331–1338

COMPUTERS & EDUCATION

www.elsevier.com/locate/compedu

Version control in project-based learning

Ivan Milentijevic *, Vladimir Ciric, Oliver Vojinovic

Computer Science Department, Faculty of Electronic Engineering, University of Nis, Aleksandra Medvedeva 14, P.O. Box 73, 18000 Nis, Serbia

Received 4 September 2006; received in revised form 28 November 2006; accepted 11 December 2006

Abstract

This paper deals with the development of a generalized model for version control systems application as a support in a range of project-based learning methods. The model is given as UML sequence diagram and described in detail. The proposed model encompasses a wide range of different project-based learning approaches by assigning a supervisory role either to instructor or students in different project stages. Different strategies for supervisor role assignment are given. Project duration, project milestones, as well as a number of team members are discussed in respect to project-based learning method that the proposed model supports. Possible implementations of different project-based learning approaches on the proposed model are demonstrated by setting the model parameters. Version control server security issues are discussed in the manner of implementation aspects of the proposed model. One of possible model implementations is evaluated in respect of cooperation on the test group of 21 students. Implementation details are presented and compared with other approaches. Mentoring and monitoring students efforts during the development by implementing proposed model with specific model settings introduces controlled cooperation with high clarity in evaluation of individual students work. Using open source version control software on Linux platform, with web interface package, we implemented a low-cost support for project-based learning.
© 2007 Elsevier Ltd. All rights reserved.

Keywords: Education; Version control systems; Hardware design; Software engineering; Project-based learning

1. Introduction

Project-based learning (PBL) is a constructivist pedagogy that intends to bring about deep learning by allowing learners to use an inquiry based approach to engage with issues and questions that are rich, real and relevant to the topic being studied. It is designed to be used for complex issues that require students to investigate in order to understand (Barron, 1998). PBL is more than just a web-quest or internet research task. Within this type of learning, students are expected to use technology in meaningful ways to help them investigate or present their knowledge. Technology is infused throughout the project to reflect the emphasis on technological and "soft" skills, as well as academic content (Blumenfeld, 1991).

* Corresponding author. Tel.: +381 18 529 603; fax: +381 18 588 399.
E-mail address: milentijevic@elfak.ni.ac.yu (I. Milentijevic).

0360-1315/$ - see front matter © 2007 Elsevier Ltd. All rights reserved.
doi:10.1016/j.compedu.2006.12.010

PBL relies on learning groups that take full responsibility for their learning, and PBL framework differs from inquiry-based activity in its emphasis on cooperation and collaboration between team members. This is what makes PBL constructivist (Barron, 1998). Cooperation, as a term, refers to the practice of working in line with commonly agreed goals and possible methods, instead of working separately in competition. Cooperation is the antithesis of competition, or in other words, an altruistic sharing, while term collaboration refers abstractly to all processes wherein people actually work together at the same time (Barron, 1998; Blumenfeld, 1991). There are a number of different approaches in PBL (Atkinson, 2001; Breiter, Fey, & Drechsler, 2005; Glassy, 2006; Janneck & Bleek, 2002; Losoncy, 1996; Reid & Wilson, 2005), which differ in project duration, number of team members, as well as in the way the students collaborate and/or cooperate.

PBL is generally a less structured approach than traditional, teacher-led classroom learning. However, working in non- or low-structured environment can introduce significant side effects. In such an environment it is difficult for students to clearly identify project design-flow phases. Furthermore, cooperation and collaboration levels are difficult to control, which leads to the lack in clarity of individual work mentoring and evaluation. Organizing PBL courses in engineering is always a challenging task for instructors due to the fact that projects can be complex and demanding for supervision. Thus, instructors often engage software support for managing and communication, using general purpose groupware tools (synchronous or asynchronous) (Glassy, 2006; Reid & Wilson, 2005), or creating specialized tools for learning purposes (Breiter et al., 2005; Janneck & Bleek, 2002).

With the aim to build single environment that can support various different project-based learning methods, we transfer the idea of project management based on version control to project-based learning, and implement a low-cost PBL support. Version control allows multiple developers to share files in a safe, controlled environment, while automatically creating a history of the project's evolution (Sussman, Fitzpatrick, & Pilato, 2004; Vesperman, 2003). This paper deals with the development of a generalized model for version control systems application as a support in a range of project-based learning methods. The model will be given as a UML sequence diagram, and it will be described in detail. The proposed model encompasses a wide range of different project-based learning approaches by assigning a supervisor role either to instructor or students in different project stages. Different strategies for supervisor role assignment will be given. Project duration, project milestones, as well as a number of team members will be discussed with respect to project-based learning method, and cooperation and collaboration level that the model supports. Possible implementations of different project-based learning approaches on the proposed model will be demonstrated by setting the model parameters. Version control server security issues will be discussed in the manner of implementation aspects of the proposed model. Evaluation of one possible implementation, in respect of cooperation, on the test group of 21 students will be given. Model implementation details in the manner of model parameters will be presented and compared with other approaches. Mentoring and monitoring students' efforts during the development by implementing the proposed model with specific model settings introduces controlled cooperation with high clarity in evaluation of individual students work.

The paper is organized as follows: Section 2 gives background of version control systems; Section 3 is the main section and presents the generalized model of version control application in course organization; Section 4 is devoted to implementation results and comparisons, while in Section 5 concluding remarks are given.

2. Version control systems

With the aim to clarify the role of version control systems (VCS) as a potential support to project-based courses, and to highlight the terminology, we give the basics of VCS.

Version control (also known as revision control or source control) is the management of multiple revisions of the same unit of information. It is most commonly used to manage ongoing development of digital documents like source code, art resources or electronic models and other critical information that may be worked on by a team of people. Changes to these documents are identified by incrementing an associated number or a letter code, termed the "revision number" or simply "revision", and associated historically with the person making the change. VCS is the system that provides version control (Sussman et al., 2004; Vesperman, 2003).

The electronic documents are stored in, so called, modules. A term module stands for a single project (set of related electronic documents) managed by version control (VC) server. Fig. 1 shows a typical VCS topology.

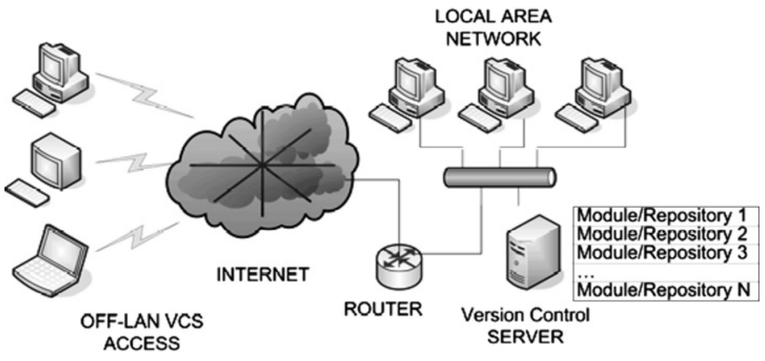


Fig. 1. Topology of Version Control System.

The VCS is established by adding a VC server to a local area network (Fig. 1). Modules stored and managed by VC server are called repositories (Fig. 1). In VCS terminology, acquiring a copy of a module is called *checking out* (the checked out files serve as a local working copy). The VCS server is usually equipped with security management software. Depending on privileges, LAN users are either able or not, to check out specific modules (Fig. 1). Changes on the working copy are reflected in the repository by *committing*. Acquiring the latest changes from the repository in the working copy is called *updating* (Sussman et al., 2004; Vesperman, 2003).

Mostly, two or more developers are working on the project at the same time. If two developers try to change the same file at the same time, without any method of managing access, they may well end up overwriting each other's work. Some systems prevent ''concurrent access'' problems by simply locking files so that only one developer at a time has write access to the central ''repository'' copies of those files. Others, such as CVS, allow multiple developers to edit the same file at the same time, and provide facilities for merging changes later. In the latter type, the concept of a reserved edit can provide optional means to explicitly lock a file for exclusive write access, despite the merging capability (Sussman et al., 2004; Vesperman, 2003).

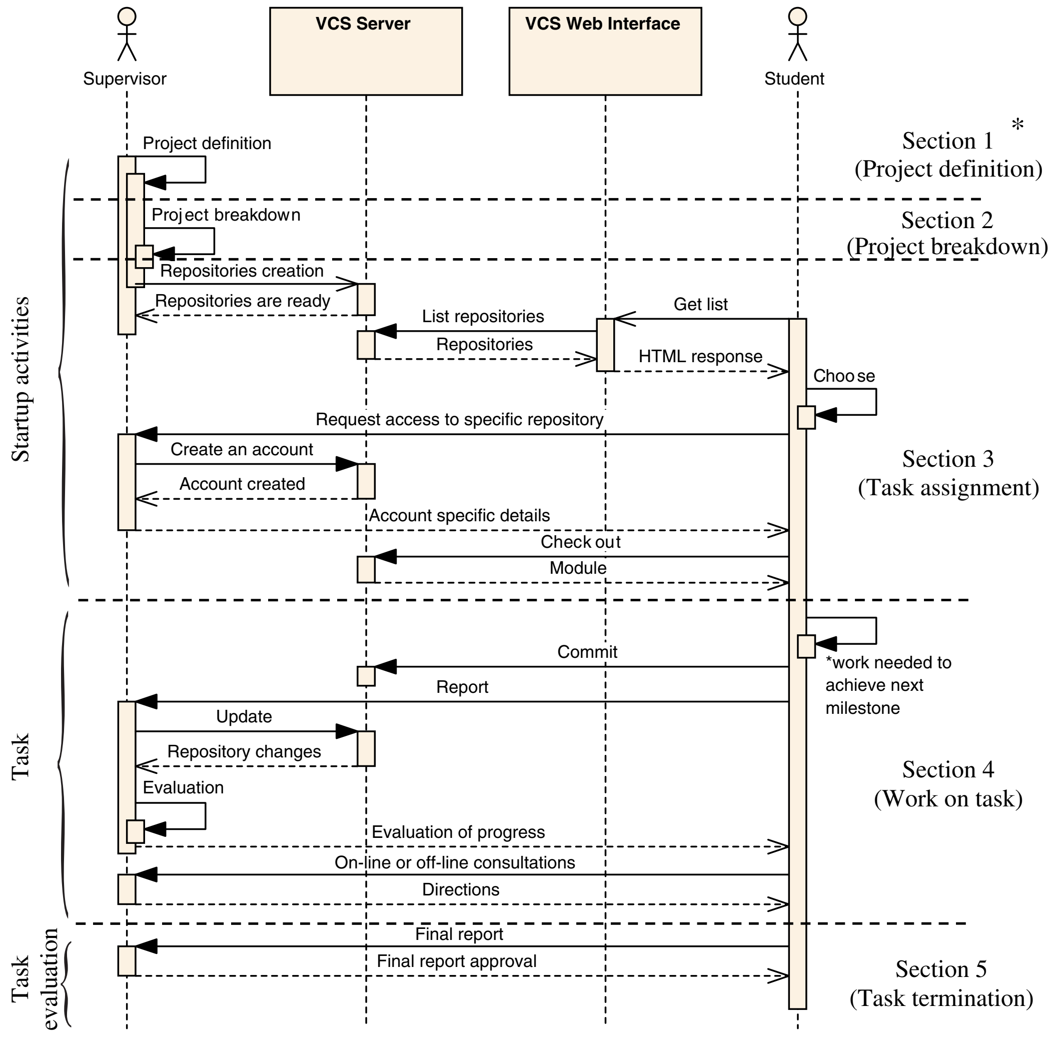
The VC server is a good core for development of project-based learning environment as long as it has both access and security set up. Modules could be accessed from the same local network, or from a computer connected through internet or another wide-area network (Fig. 1).

3. The role of version control system in course organization

The role of VCS from Fig. 1, along with associated web interface, during the project lifetime, is shown by UML sequence diagram (SD) in Fig. 2. The SD consists of four timelines (vertical dashed lines). Two timelines stand for actors (supervisor and student), while the other two are for system components. The role of supervisor can be associated either to instructor or student, depending on particular learning method applied in the course. The diagram in Fig. 2 is divided by horizontal dashed lines into five sections. The first two sections stand for project definition and project breakdown activities. The third section delineates a process that begins with the creation of the repository by the supervisor and that lasts until student checks out the repository; the forth section is devoted to the process where student works on the task, while the last section stands for final report approval.

In project management, a project breakdown is an activity with the outcome in a form of project breakdown structure (PBS). PBS is an exhaustive, hierarchical (from general to specific) tree structure of deliverables and tasks that need to be performed to complete a project. The PBS is very common and is considered a key part of project management. Its hierarchical arrangement allows easy identification of the terminal elements (the actual items to be done in a project). Being an exhaustive document of the project scope, the PBS serves as the basis (indeed the backbone) for much of project planning. All the work to be done in a project must trace its origin from one or more PBS entries (Dinsmore & Cabanis-Brewin, 2006). The first two sections in Fig. 2 are devoted to project definition and breakdown, and performed by project supervisor.

The third section within SD relates to task assignment. The section begins with creation of repositories. One repository can contain one task or group of few related tasks from PBS. The number of tasks associated with the repository, as well as the number of students that have access to the repository, are model implementation



* A group of tasks that belong to a single project has common sections 1 and 2, and each task has its own sections 3, 4, and 5.

Fig. 2. UML sequence diagram of a model for application of version control tools in project-based learning.

decisions, and define the course learning method. Implementation solutions, which are of interest for project-based learning, are covered by the proposed model as follows: one student—one task—one repository (1-1-1), one student—multiple tasks—one repository (1-T-1), group of students—one task—one repository (S-1-1), group of students—multiple tasks—one repository (S-T-1), and group of students—group of tasks—group of repositories (S-T-R), where S,T,R are integers grater then 1.

The goal of the third section in SD is application for task assignment, or, in other words, request for access to the repository (Fig. 2), which contains files related to the terminal element(s) of PBS. It is mentioned that VC server should be equipped with web interface, which dynamically creates HTML response containing a list of repositories. In order to highlight an overall project scope, the response should involve all repositories related to the project which in turn are clearly marked as “done”, “work in progress”, or “available” (Faculty of Electronic Engineering, 2006). Request for available repository is posted by a student (Fig. 2). Upon requesting the access to repository, project supervisor creates an account and gives the student account specific details (server address, protocol, user name, password, etc.). Using the received account parameters, student is able to access VC server and check out files from the permitted repository. While checking out the files for the first time, a student is provided with starter code, background material and templates for reporting documents, which in turn makes student a team member.

The fourth SD section is devoted to development. Once the project files are checked out, student is able to begin work on the assigned task. In order to enable supervisor to monitor project work as a process, all work should be done in increments (Fig. 2). The duration of incremental work is the model implementation parameter, and depends on used learning method. The end of each increment is considered a milestone. In each milestone student should write a short report that contains a list of performed actions, short plan for the next milestone achievement, and a list of problems that have occurred during the previous increment. Changed files should be committed at the same time with report delivery (Fig. 2), and it is a key point in the model that enables project monitoring. Supervisor is notified by the received report to update the repository (Fig. 2). The decision about achievement of the task is made and using either on-line or off-line consultations student is provided with suggestions and directions for further work. A crucial decision within this section is the duration of the work per increment. If the work in increment lasts too short, then students do not have enough time to work on the task and to point out their skills. Thus, it can cause difficulties for both students and supervisor. For students, it is difficult to work on the assignment keeping in mind milestones and reports that should be written. For supervisor, it is hard to evaluate the goal achievement because of the shortness of working interval. On the other hand, using a too long increment period, the idea of project monitoring and management could be lost. Group of activities, marked as Section 4 in SD (Fig. 2), represents repetitive procedure followed by evaluation of goal achievement in Section 5.

There are several aspects of proposed model that are left as implementation parameters: duration of project, the role of supervisor, the number of students and tasks per repository, and duration of work per one increment. These aspects are implementation decisions, and should be defined according to learning approach that the model supports. Proposed model gives the possibility either for instructor or student to play the role of supervisor in different sections of the model. It is up to instructor to decide where the supervisor role will be passed to the students. The possible settings for supervisor role per model sections are shown in Fig. 3, and marked as a1–a4.

Thus, in the a1 setting, supervisor role is never passed to student (Fig. 3). Such a setting is suitable for small projects or even for demanding homeworks (Reid & Wilson, 2005). However, setting a1 does not provide a team work. Allowing a student to be a supervisor in Section 4, well controlled team work is employed. Giving the freedom to students to assign tasks and create repositories by passing them supervisor role in Section 3, team work is moved to higher level in which more managing activities are performed by students (Glassy, 2006). Final issue is when instructor decides to give students supervisor role right after project definition (Fig. 2), and to let them manage even project breakdown activities (Fig. 3). Such a setting is very close to real life projects, where students have almost all responsibilities in project management. This approach requires project duration of at least one semester and should be foreseen by curriculum designer (Breiter et al., 2005). Different settings of supervisor role (al–a4) are followed by different model implementation parameters such as number of students and tasks per repository, duration of one increment, as well as duration of project. The parameters should be set by instructor according to the chosen setting for the supervisor role. Suitable parameters settings are shown in Table 1.

Settings a1–a3 are intended for course organization and can be used in different courses during studies, while a4 is applicable for bigger projects that can involve several courses from different modules which are important for student specialization. The a4 setting is applicable for senior level students, usually no more then once during the studies.

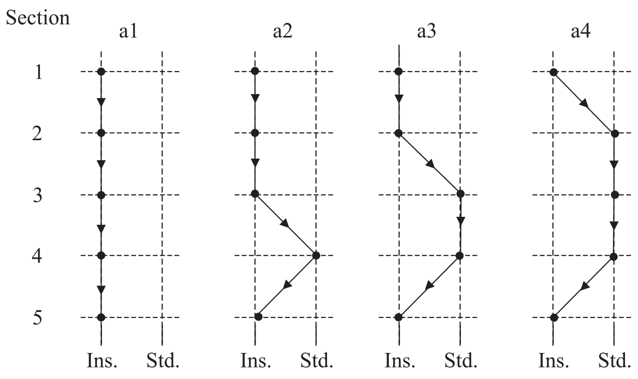


Fig. 3. Possible settings of supervisor role on the proposed model.

Table 1
Recommendations for model parameters setting with respect to applied learning method

	Project duration	Increment duration (weeks)	Number of students-number of tasks-number of repositories per project
a1	1 week for homeworks, few weeks per project task	1	1-1-1
a2	Up to one semester	1 or 2	1-1-1 or 1-T-1
a3	1 semester	1 or 2	S-1-1 or S-T-1
a4	1–4 semester	2–4	S-T-1 or S-T-R

Security issues related to VCS should be considered depending on the chosen model setting (Table 1). There are two aspects of security: internal and external security. Internal security relates to security issues between project members (VCS actors) and depends on the collaboration degree of the chosen model setting approved by instructor. External security is the security policy towards the outside world. Discussion on external security makes sense only in case of low internal security level. A low level of external security can provide an open-source status to the project.

Internal security policy should be very rigorous in a1 setting. Students should not be able to see each other's work. It means that neither collaboration nor cooperation between students is involved. Setting a2 is designed to avoid this problem by letting students see (read-only) each other's work, but still keeping one repository per student. Setting a2 allows cooperation maintaining highly-controlled environment. Unlike a2, a3 setting does not hide VC maintaining activities (Fig. 3). If a3 is set as S-l-1, then a3 is more like a1 setting, while in case S-T-1 it is more like a2. In the S-T-1 case, a3 setting requires lower internal security, gives more control to students, and allows more students per project. In order to develop professional and managing skills of students, instructor can decide to apply a4 setting with lowest internal security, which allows high degree of collaboration between team members.

4. Implementation results

We implemented version control as project-based learning environment in senior-level course "Algorithms and architectures for dedicated architectures" at the Faculty of Electronic Engineering, University of Nis, Serbia. The course, which follows Parhi's book (Parhi, 2000), is devoted to VLSI digital signal processing systems with emphasis on design and implementation. Having in mind that this course is an advanced one, and that prerequisite is knowledge of computer architecture area, VLSI design, and basic DSP algorithms, we employed project-based learning as the most suitable method that brings together algorithms and circuit designs for special purpose DSP applications. Stages, which each candidate has to pass working on the project, are shown in Fig. 4.

Model proposed in Fig. 2 is general and covers different implementations of project-based learning. In order to force the students to pass all stages in hardware design flow from Fig. 4, and allow them to use the components designed by other students, our implementation is based on a2 model setting that allows cooperation maintaining highly-controlled environment.

Proposed model (Fig. 2) is applicable for any version control system that can provide web interface. In order to implement the system from Fig. 1, for version control we decided to use CVS (Vesperman, 2003) on Linux platform, because it is an open source VC software, well documented, with several web interface packages, such as cvsview (Vesperman, 2003). Subversion (Sussman et al., 2004) is also a possible solution.

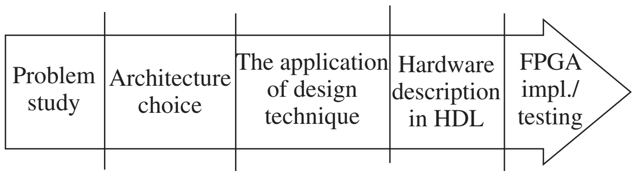

Problem study
Architecture choice
The application of design technique
Hardware description in HDL
FPGA impl./ testing

Fig. 4. Design flow.

When the system from Fig. 1 is used according to the proposed model (Fig. 2) with a2 setting (Fig. 3), there is no need to consider security issues further from preventing a student modifying or accidentally destroying other students work. Furthermore, the problem of cheating preclusion, which relates to modification of time-stamps on committed files by students, with the aim to show better time performance (Reid & Wilson, 2005), is avoided by involving written weekly reports. The reports are useful not only to TS to get an overall picture of the previous work and possible problems, but also to students as they force them think about and summarize results and achievements, thus giving a clue for the better start of the next incremental period.

We had a test group of 21 students in the duration of one year. The course web site, CVS web interface, and course forum, have low-level external security. In other words, they are public available at (Faculty of Electronic Engineering, 2006). For the test group, we made a record of raised problems posted by weekly reports and forum. Table 2 gives the number of students that faced typical problems (rows in Table 1) with respect to task phases shown in Fig. 4 (columns). The table values are written as y/x, where x is the total number of students that faced the problem, while y stands for students that are recorded in CVS log files while mining other students work.

From Table 2 it can be seen that there is a significant cooperation between students, which is enabled by the applied model setting. Having in mind project flow stages, we notice the following:

- Carefully selected scientific and technical papers by teacher, as well as recommendations for book chapters are provided by the system, right after the first login (Fig. 2). This enables good and fast start-up for students tasks.
- Model setting a2 allows students to explore source codes and to use the components designed by other students. These activities support critical stages of the project such as *architecture choice* and *application of design technique* by allowing studying of other students work (Table 2). By leaving the completed assignments on CVS, we enable students to study the examples of different architecture design styles, to notice the differences in arithmetic, and to become aware of area-time-power compromising techniques in hardware design.
- Mentoring and monitoring students efforts during the development by a2 model setting, rather then evaluating task results, introduces high clarity in evaluation of individual students work in highly controlled environment.
- Students get more familiar with version control, and accept it as a very helpful tool.

At the end, let us identify where our approach is among the others. Settings a1–a4 of the proposed model (Fig. 2) covers a wide range of different project-based teaching methods. The solution proposed by authors in (Reid & Wilson, 2005) is assumed as setting a1 of the proposed model, while the teaching method used in (Glassy, 2006) can be accomplished by implementing the proposed model with setting a3. Ambitious projects described in (Breiter et al., 2005) could be well-supported by the proposed model by setting the track a4 from

Table 2
Typical problems occurrence and CVS log-records per design flow stage

	Problem				
Task phase	Problem study	Architecture choice	Design technique	Hardware description	FPGA
Background	8/12	2/3	–	–	–
Design choices	–	7/9	12/18	4/12	–
Code-related	–	–	8/12	2/8	0/6

Table 3
Comparison of different project-based learning methods

Model setting	Professional skills	Soft skills	Cooperation	Collaboration	Clarity of work evaluation
al- Reid and Wilson (2005)	High	No	No	No	High
a2- our impl.	High	Low–Medium	Low	No	High
a3- Glassy (2006)	High	Medium	Medium	Low–Medium	Medium–Low
a4- Breiter et al. (2005)	High–Medium	High	High	High	Low

Fig. 4. Different teaching methods can be supported by the proposed model depending on what type of skills student has to acquire during course. Additionally, methods differ in duration of project, degree of cooperation and collaboration, as well as in clarity of individual work evaluation. The mentioned aspects of a1–a4 model settings are summarized in Table 3.

5. Concluding remarks

In this paper, the development of a generalized model for version control systems application as a support in a range of project-based learning methods is presented. The proposed model encompasses a wide range of different project-based learning approaches by assigning a supervisor role either to instructor or students in different project stages. Different strategies for supervisor role assignment are given. Project duration, project milestones, as well as a number of team members are discussed in respect of project-based learning method that the proposed model supports. The model is given as UML sequence diagram and described in detail. Possible implementations of different project-based learning approaches on the proposed model are demonstrated by setting the model parameters. Version control server security issues are discussed in the manner of implementation aspects of the proposed model. One of possible model implementations is evaluated in respect of cooperation on the test group of 21 students. Implementation details are presented and compared with other approaches. Mentoring and monitoring students efforts during the development by implementing the proposed model and defining model specific settings, introduces controlled cooperation with high clarity in evaluation of individual students work. Using open source version control software on Linux platform, with web interface package, we implemented a low-cost support for organization of project-based learning.

References

Atkinson, J. (2001). *Developing teams through project-based learning*. Hampshire, UK: Gower Publishing.

Barron, B. (1998). Doing with understanding: Lessons from research on problem- and project-based learning. *Journal of the Learning Sciences, 7*(3, 4), 271–311.

Blumenfeld, P. C. et al. (1991). Motivating project-based learning: sustaining the doing, supporting the learning. *Educational Psychologist, 26*, 369–398.

Breiter, A., Fey, G., & Drechsler, R. (2005) Project-based learning in student teams in computer science education, Facta Universitatis. In: M. Stojcev & I. Milentijevic. (Eds.), Electronics and energetics, Special issue on computer science education (pp. 165–180). Serbia: Nis,vol. 18, No. 2, April 2005.

Dinsmore, P., & Cabanis-Brewin, J. (2006). *The AMA handbook of project management* (2nd ed.). USA: Amacom, American Management Association.

http://L3.elfak.ni.ac.yu/algarh/, Faculty of Electronic Engineering. (2006) University of Nis, Serbia.

Glassy, L. (2006). Using version control to observe student software development processes. *Journal of Computing Sciences in Colleges, 21*(3), 99–106, February.

Janneck, M., Bleek, W-G., (2002). Project-based learning with commsy, *Proceedings of CSCL 2002* pp. 509–510. Colorado, USA: Januar.

Losoncy, L. (1996). *Best team skills*. London, UK: CRC Press.

Parhi, K. (2000). *VLSI digital signal processing systems (Design and implementation)*. New York: John Wiley & Sons.

Reid, K., & Wilson, G. (2005). Learning by doing: introducing version control as a way to manage student assignments. In *Proceedings of the 36th SIGCSE technical symposium on Computer science education* (pp. 272–276). St. Louis, Missouri, USA, February 23–27.

Sussman, B., Fitzpatrick, B., & Pilato, M. (2004). *Version control with subversion*. USA: O'Reilly.

Vesperman, J. (2003). *Essential CVS*. USA: O'Reilly.