# Experiences with Designing a Team Project Module for Teaching Teamwork to Students

Mária Bieliková[1], and Pavol Návrat[1*]

[1]Institute of Informatics and Software Engineering, Faculty of Informatics and Information Technologies, Slovak University of Technology in Bratislava

Team projects play an important role in the education of engineers. This paper describes a team project module that is part of a postgraduate course in Informatics. Its main objective is to give students a hands-on experience with different aspects of working in team on a problem. We discuss several issues that should be considered in designing such module as a part of a curriculum: team formation, team communication, team assessment, problem statement and assignment, development process and team supervision. We outline several different alternatives, analyse their pros and cons and describe our approach and related experiences.

*Keywords:* teaching, teamwork, team formation, team communication, team assessment, software development process

## 1. Introduction

The main objective of a good curriculum is to prepare students for their professional career so that they will be able to cope the changing nature of the world and to solve demanding and complex tasks as required by practice. There is a growing recognition that work in teams is critical in managing the complexity of solved tasks (e.g., developing software systems).

Team projects play an important role in the education of engineers. The students should acquire and display abilities to communicate within a team, to plan a relatively large project, and to cooperate in order to develop the required output of such a project. One question is how this requirement is reflected in the existing curricula. One of perhaps the most distinguished features related to teamwork of the CMU course is the software development studio [17], which runs throughout the whole four-semester course. Students work closely with instructors and other students to carry out a significant development project. Another possibility of incorporating teamwork into software engineering curriculum relates to the module entitled "Setting up and Running your own I.T. Company" at the University of Sheffield [8]. Diversity of course methods are apparent from the Forum for Advancing Software Engineering Education June 2001 issue where academic project courses taught at thirteen universities are described [10].

While many aspects of existing approaches are attractive, it may be difficult to adopt them because each is designed within a specific context. We deal with aspects, which should be considered in the design of a team project module aimed at software projects. We discuss variability of different approaches to implement such a module and present the team project that is part of a postgraduate course in Software Engineering (formerly Informatics) at the Slovak University of Technology. Module named Team project was introduced in the academic year 1997/1998. The Team project module was strongly influenced by the Team Software Process [9, 7]. Its intake is each year 15-17 teams of 5-6 students. The main objective of the module is to give students a hands-on experience with different aspects of working in team on a relative large task.

## 2. Teamwork dynamics

An objective of any good curriculum design is to prepare the graduates for their envisaged professional careers by providing them an appropriate education. In software engineering, the essential feature is that the software developers develop and maintain soft-

*This work has been supported by the Grant Agency of Slovak Republic grant No. VG1/ 0162/03.

ware of such a complexity that these tasks cannot be handled at an individual level. Such tasks can only be accomplished in groups or teams. The engineers need to join their efforts to be able to complete the tasks in a realistic time. Moreover, they usually need to join with experts in other fields to be able to understand and analyze the problem domain and to design a system solution for it.

One comment is appropriate to explain the difference that is sometimes made between a group and a team. Both groups and teams comprise individuals who have a common task and who collaborate to accomplish it. A team is distinguished by complementary efforts of its individuals that induce a synergic effect. Groups are often used in educational modules, which concern acquiring knowledge, skills and experiences related to the particular subject (software analysis, programming, etc.). One approach to practice the real teamwork within an educational module is to design it in such a way that *teamwork is one of the main goals of the module* (or the only goal of the module). We describe experiences with such approach.

In designing a team project as a part of a curriculum, several aspects should be considered and decided. We focus on the following ones: team formation, team communication, team assessment, problem statement and assignment, development process and team supervision. Design of these aspects is influenced by several factors (see Figure 1). Note that the team process (as defined for example in [9]) is only one aspect of the organization of a team project.

Organization of a team project is influenced mainly by a class size (very focused module for 10-25 chosen students [3] versus large module of more than 300 students in 50 teams [2]). In fact, size of the class determines mainly choice and organization of supervisors. For example, at the Vienna University of Technology 5 professors, together with 20 senior student supervisors, guide 50 teams of 5-7 students. A professor takes the role of a middle management, a supervisor acts as a quality assurance person, and team members are to work as project managers and software engineers.

Type of the study (either postgraduate or undergraduate) influences mainly complexity of the assigned problem and requirements on quality of the solution. Performed activities, their extent and the number of cycles of the development process depend strongly on duration of the team project and number of hours allocated per week. Our experience with such projects is that a satisfiable solution (in terms of the team project objectives, i.e. experience with different aspects of working in team on a large problem) requires time longer than one term. Supervisors who are available (either academic staff or an industry partner) determine problems being solved. Cultural factors and a context of whole curriculum should be also considered.

Design of a team project module requires deciding on all the mentioned aspects. We shall discuss them in the subsequent paragraphs. We outline different alternatives, analyze their pros and cons and describe our approach and experiences.

## 3. Team formation

The way teams are formed can have a great influence on a project. There are two basic methods of forming teams:

1. students create teams by themselves or
2. students are assigned to the teams.

Several variations of these methods can be adopted for team projects. For example, assigning students to teams can follow different goals: a) to equalize the differing project specific knowledge and skills of teams, b) to balance team membership according to differing psychological profiles (using the results from known psychological studies [15]).

Our approach in the first two years of running the Team project was to let students assemble their teams. The advantage is simplicity (from the managerial point of view). If the students can sign up for teams, there is a tendency to base the choice on a personal acquaintance. The advantage of such a behavior is that a time periods needed for a team "storming" and "norming" activities are minimized. Team efficiency in such cases is usually high in some teams. Acquaintance based teams often overcome, or at least tend to minimize many of the problems which can arise in a team life cycle: little effort is needed to get teams started; scheduling and arrangements of out-of-class meetings is easier as the students know each other well and often have similar preferences. Moreover, in many cases, there is more pressure on each member of a team "not to let down one's friends". On the other hand, there is a risk that team quality can vary quite significantly across teams. This risk was in our team project become a reality.

One possible solution is forming teams in two steps. First, students express their preferences according team compositions. Then, supervisors rearrange those teams where an undesirable mixture of skills can be recognized [2]. A justification for this procedure is that to work with people one already knows and is a friend with is only one of the possible scenarios encountered in practice. It may not be a prevailing characteristic of teams formed ad hoc in industrial conditions for a particular project [1].

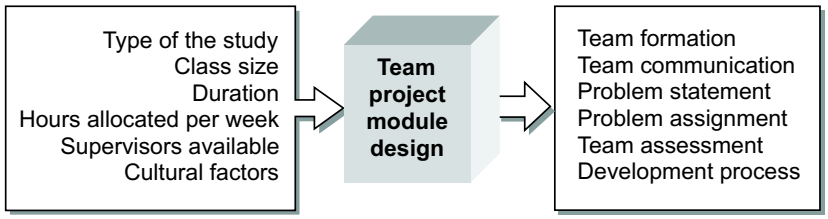


Figure 1: Aspects of a team project module design.

First arrangement based on freedom in team creation for students is useful when the main goal of the module is to gain knowledge and skills in particular technical topic of engineering and practicing teamwork is only minor goal (e.g., in the case of projects, which are a capstone experience, intended to reinforce what has been taught previously in the curriculum). After gaining some experience with teaching team skills, we recommend to made rearrangements when big differences between teams became apparent.

At the moment we provide active control in the process of teams formation. A team composition is based on students' previous experience and the potential role of each individual in the team. We also consider their preferences (a student can specify one student to become a member of the same team). Using such arrangement we observed higher number of conflicting situations within teams. We believe that our students are thanks to having experienced such situations better prepared for their professional career.

One particular aspect concerns the size of a team. The bigger is a team, the more difficult it is for its members to participate fully in its activities. In a big team, there are formed subgroups that start to work independently. Individual roles in a team become fuzzy and they overlap, which can be a source of possible conflicts. It is more difficult for a team leader to perceive individuals and atmosphere in the team.

Our experience endorses the frequent suggestion to have the size of a team between 4 and 8 (with 5 or 6 as optimum). If the size of the team is smaller, its creativity and flexibility decreases. If the size is bigger, number of interactions among its members rapidly increases and the team becomes less manageable.

# 4. Problems to be solved

## 4.1. Problem statement

Problem complexity, its breadth and depth are interrelated above all with the expected workload attributed to the team project module. Problems to be solved should meet the criterion that they are too complex for one or even two excellent students to complete all the work needed for the team's assignment.

Information systems development is a frequently used theme for software projects. We found the theme very suitable when introducing a team project module. Later, we have been introducing also very complex, 'wicked' problems [16]. When introducing the Team project in the academic year 1997/1998, we offered three kinds of information systems development projects and two kinds of projects of a simulation software system development [11]. Later (1999/2000) we offered besides software development projects also two kinds of software reengineering projects.

Now we offer a mixture of different topics, which depends on our teaching staff capacity. Our students were involved within the Team project also in an international research project in which two teams of students worked with physicians (customers) and developed software for collection and management of electromyography data. Other themes include

- a RoboCup simulation league soccer team,
- human walking animation,
- on-line university admission application,
- computer support of a programming contest,
- on-line homework assignment submission and evaluation,
- multimedia presentation of history of computing at the university,
- scientific journal submissions editorial processing support.

## 4.2. Problem assignment

A problem assignment can be such that each team works on different projects during each standard project completion phase or cycle. For example, a team designs a system that was defined in another team's specifications. It implements another team's

design, and tests yet another team's implementation. This approach causes students to participate seriously in all phases. Students realize the importance of a well developed documentation, possibility of incorporating changes, etc. Such an approach can be effectively adopted only if the module is spread across at least two semesters. One difficulty with such a pipelined approach is that it is very demanding. Students must swiftly become knowledgeable of several different domains.

We assign one problem to more than one team (usually two teams work on the same assignment). The advantage is increase of competition, more competent mutual reviewing, more intensive and focused class discussions among several teams, and more transparent evaluation and assessment. We have positive experience with such an approach.

Similar methods as for assembling teams can be adopted for assigning problems to them. The basic possibilities are:

- assign randomly;
- get teams to bid for problems;
- let teams indicate their preferred choices and assign so that the maximum number of teams will be assigned their preferred problems;
- use a first come first served policy.

We use the second method because this is probably the most realistic approach. At least some of the teams will work on the problem in which they are interested and often have previous experience. Moreover healthy competition between teams can be established and students have opportunity to exercise writing and presenting the bid. The students bid with their knowledge, skills and achievements related to the selected problem, and with a preliminary sketch of solution based on the open question-answer session with a customer.

Things become a little bit complicated by the fact that the bidding period causes a delay in the actual start of the project work. On the other hand, this method also helps students become aware of the fact that they cannot, and should not try to solve all problems. They should set goals that are reachable within the amount of time available. In our opinion, students underestimate the complexity of problem to be solved and overestimate their own capacity. We observed that activities towards giving to students more detailed instructions and advices in order to make accurate estimations work only partially. In most cases students have to make this mistake by themselves (i.e., to offer a grandiose solution) before they realize they have underestimated the problem.

## 5. Team communication

Students must decide how to mesh their various work patterns to develop the required result. Basic skills which should be exercised and monitored include time planning, product presentation, decision making, keeping personal log on work hours, negotiating and keeping work contracts, managing priorities and deadlines, acquiring information (e.g., by interviewing people) and accessing information (e.g., by searching documents). The communication plays crucial role especially in cooperation projects where students work in teams consisting of team members from remote places [3, 5].

To facilitate communications among team members (and also among teams) there is more and more often used Internet or an intranet. We encourage this practice, not to speak about such elementary means of electronic communication as e-mail, which is indispensable. Moreover, each team should create and regularly maintain its web page that includes also the team "official" presentation of the project. Communication through the web works in both directions: students inform about project progress, and students are informed about details of the organization of the module, time schedule, frequently asked questions, etc.

Team documents are another form of communication among team members and more importantly among teams during reviewing results of work. Our students have also access to a documentation of past team projects.

Students within a team should meet on a regular basis. Our teams meet with their project supervisor (who takes a role of facilitator instead of manager) at least once a week. There are also regular private meetings of the team. All major decisions should be recorded in a project log.

The university is important as a meeting place for team discussions for the whole team. It is important to have meeting room designed and furnished in such a way that it supports the teamwork. Figure 2 depicts arrangement of our software studio. Some part of the overall project work is accomplished in small subteams of two or three students, and some time is spent as an individual work (see average distribution during autumn semester 2002/2003 on Figure 3, based on questionnaire evaluation of 68 students). This work is carried out at home most of the time. Individual work increases as the project proceeds. In software projects most of the implementation phase is done individually. However, at the end of the project, during integration and testing, students spend most of the time in teamwork.

We adopted an approach where students should experience various responsibilities for different parts of the system that is being developed. Distributing

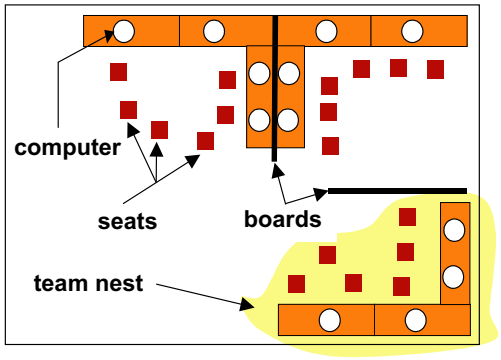


Figure 2: Outline of our software studio.

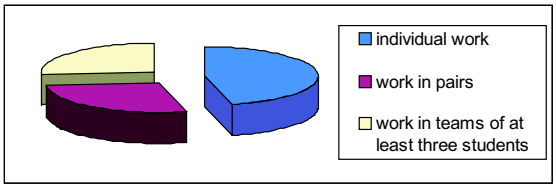


Figure 3: Teamwork distribution.

the work evenly among the team members is a precaution against a possible excessive individual workload and also against problems in case when a team member quits for whatever reasons.

## 6. Assessment

Methods of team assessment should be based on some objective criteria. Based on them, a supervisor is usually responsible for determining students' grades. There are two common approaches and their combinations [18]:

1. to give each team member the same grade and
2. to give each team member a grade based on his individual contribution.

Further, assessment can be based solely on the "product" developed or can combine evaluation of the quality of product and the management issues. Let us note that in our last survey (which we perform regularly each term) none of our students selected the option to evaluate all team members equally. 30% of students reported that efforts of individual team members were rather different. Only 2 students (from 74) reported that the effort of all team members was equal. In first years of offering the Team project with teams formed by students, "equality syndrome" was quite common.

Assessment in our Team project comprises the following components: product quality, documentation, quality and topicality of a web page, reviewing, individual team members activity, presentation and answers to questions on product and software engineering methods used, active participation in the presentation of competing team(s).

A supervisor evaluates each component on a predefined 8 points scale (excellent, very good, good, adequate, satisfactory, sufficient, poor, very poor, unacceptable). Particular component is either related to an individual or to the team as a whole. In the latter case, part of the work performed by each team member (in %) is determined (using peer evaluations as a help). We created Excel sheet for assessment evaluation – an example is depicted on Figure 4.

The supervisor fills colored cells: orange cells are for evaluating particular outcome (numbers from 0 to 8 correspond to 8 points scale), and blue cells are for evaluating a contribution of each team member. Note that the contribution of the team member is not assigned for each particular outcome but for a group of outcomes related to the particular milestone (see next section Development process). Teamwork is assessed both semesters individually for each team member and constitutes 10+10% of overall assessment. Assessment is recommended automatically (columns Autumn and Final Suggestion) based on evaluated outcomes for whole team, contribution of a member, individual assessments (teamwork and project presentation) and their importance. Finally, the supervisor considering all factors (not only those expressed by numbers in the table) determines final mark (last column).

We use several methods for students peer evaluation: students are asked to order their team mates; students assess each of their team mates on a predefined point scale; students determine explicitly part of work performed by each team mate. The last method seems to be most effective.

## 7. Development process

Project phases and number of cycles used for a particular project is mainly influenced by the problem domain and project resources. Usually life cycle ends with implementation, integration and system testing phase. There are reported also approaches where a maintenance phase is involved. For example, a module can consist of two parts: the first part is the development of a new system, and in the second part the teams are required to make certain modifications to those systems [14].

Our experience is that the most important factor is the time (how long the project lasts and how many hours per week it is allocated). If a team project module is just one out of several modules taken concurrently, covering all phases of the software life cycle together with several iterations in 2

**Team project assessment 2003/2004**

| Number of team members | Bid | Specification document | Specification review | WWW project presentation | Prototype and its presentation | Prototype review | WWW project presentation | Teamwork | Autumn | | Documentation | El. medium | Documentation review | WWW project presentation | Product | Product review | Project presentation | Teamwork | Final Suggestion | | Mark |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Specification | | | Prototype | | | | | | Final Documentation | | | | Final Product | | | | | | |
| 5 | 6 | 10 | 2 | 2 | 10 | 2 | 2 | 10 | 44 | | 8 | 4 | 2 | 4 | 14 | 2 | 12 | 10 | | | |
| **Team evaluation** | **3** | **3** | **5** | **6** | **3** | **5** | **6** | | | | **5** | **5** | **4** | **6** | **6** | **5** | | | | | **Mark** |
| Team member 1 | 21 | 12 % | | | 14 % | | | 3 | poor | 2,6 | 16 % | | | | 10 % | | 2 | 5 | D | 3,1 | D |
| Team member 2 | 16 | 24 % | | | 16 % | | | 7 | adequate | 4,3 | 24 % | | | | 26 % | | 8 | 7 | B | 5,9 | A |
| Team member 3 | 21 | 20 % | | | 16 % | | | 3 | sufficient | 3,2 | 18 % | | | | 20 % | | 4 | 5 | C | 4,2 | C |
| Team member 4 | 21 | 24 % | | | 29 % | | | 7 | adequate | 5,2 | 24 % | | | | 24 % | | 7 | 7 | B | 6,0 | B |
| Team member 5 | 21 | 20 % | | | 25 % | | | 5 | satisfactory | 4,2 | 18 % | | | | 20 % | | 2 | 4 | C | 4,3 | C |
| | | sufficient | | | sufficient | | | | satisfactory | 3,9 | adequate | | | | good | | | | adequate | 4,7 | |

valuation

| | | | |
|---|---|---|---|
| 8 | *excellent* | | |
| 7 | *very good* | 3 | *sufficient* |
| 6 | *good* | 2 | *poor* |
| 5 | *adequate* | 1 | *very ppor* |
| 4 | *satisfactory* | 0 | *unacceptable* |

Team members contribution

Figure 4: An example of the assessment sheet.

or even 3 semesters is hardly possible. From the point of design of the educational module, stress should be put on the team process, communication, etc. rather than on concentrating on the particular phases of software development. However, this is an open issue. If the module objective is formed as an aggregation of effective teamwork and software engineering topics (such as software processes, software architecture or methods of software analysis), one should balance these objectives (often with a priority to software engineering topics). Providing the *effective teamwork as the primary goal*, we warn against expecting too much from the team project in the sense of teaching students new methods, tools or technologies.

Although the quality of the final result (e.g., a software system) is an important measure of a success of a team, we concentrate on the process applied. Figure 5 illustrates main phases accomplished in our Team project as adopted for a project of developing a software system from the Team Software Process [9]. Bid period together with first iteration (Cycle 1) is performed during first semester and the second iteration is scope of second semester. We believe that classic process oriented development methods are for educational purposes more suitable than agile approaches such as eXtreme Programming, Lean Programming, Scrum or Crystal [4]. However, this statement should be more elaborated and real experience with agile methods should be acquired.

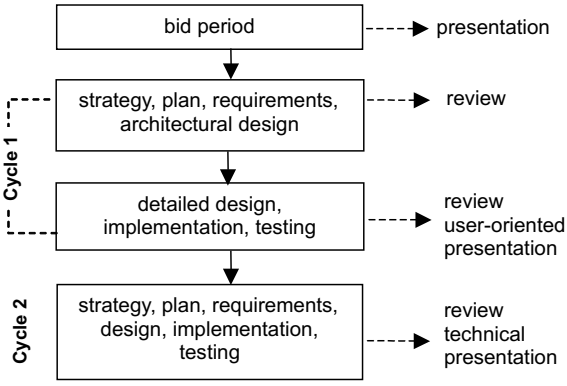


Figure 5: Main phases of the software process.

Each team of students must present the project in a seminar, to an audience of department staff and students (mainly those who worked on similar assignments). An important part of the development process is reviewing the work of another team. For each team a competing team is assigned. Students should write a review document with its structure specified beforehand. Results are then discussed in a joint meeting where students of one team are supposed to present their review and respond to another team's review of their work. Different kinds of review can be adopted with different frequency. The main constraint here is duration of the module. We adopted approach where the project is reviewed by another team after completing first specification,

first worked system (Cycle 1), and final software product (including review of a source code).

## 8. Team supervision

The amount of freedom and supervision should be balanced in order to create a true learning experience for students. To simulate the reality, students should have a considerable amount of freedom. On the other hand, since students usually have no project experience, some amount of supervision, monitoring and guidance is needed to ensure sufficient progress and a successful result.

Gordijn and Niessink [6] report on an approach to reach equilibrium between freedom for students to make their own decision on the one hand and supervision on the other hand. They distinguish four different roles in the practicum: a project group of 6 to 10 students, a customer from an existing company, a steering committee formed by university staff, and mentors. The steering committee plays a role of project supervisor, provides students with advice, and monitors progress of the project. Mentors contact with students on a daily basis, review documents produced by the students and coordinate practicum from a didactic point of view.

Figure 6 shows roles and numbers of people involved in our module during the academic year (2002/2003). To discuss issues that concern the whole module, all supervisors meet with a chief module supervisor about once a month or as the situation demands. Our experience shows that is it more effective for students to have different supervisor for each team. From the organizational point of view such approach makes the level of supervisors and their coordination not an easy task. On the other hand, it is not easy to supervise several teams in such large project without influencing one team with ideas devised during sessions of the other team.

In order to reach balance between freedom of students and supervision we specify in advance certain requirements on the content of documentation to be produced. Students have to prepare and follow a detailed project plan. We prescribe certain parts of the project plan, such as list of activities, milestones, dependencies, and responsibilities according to established team process [9]. Students are free to define the activities that are necessary to successful accomplishing of the project. We accompany the Team project by lectures on project management, teamwork, and quality assurance.

## 9. Conclusions

From the above it follows that the idea of incorporating a teamwork practice usually in a form of some kind of a team project has been adopted by and large. However, there are many issues that remain open. To name a few, organizing projects as truly real-life ones mirroring the industrial conditions including the problem tasks defined by an industrial partner is discussed as clearly a desirable, but hardly a completely realistic goal. On the other hand, curricula designers should be careful when relying solely on academic environment and should seek at least some kind of an industrial involvement.

In design of our curriculum [13, 12] we followed an approach where students first learn the required knowledge and skills individually or in small groups (2 or 3 students), and then make use of it in a team. The Team project is running now (i.e., in the academic year 2004/2005) for the eighth time. We have 18 teams (102 students). The Team project is a compulsory subject to all students enrolled in the Master course in Software Engineering and Computer Engineering. Much stress is put on the supervisors who should manage the teams and help them change from a supervisor-managed mode to a self-managed mode. Quite often students voice the desire of having a wider diversity of themes, including problems, which would be defined by them. Some students questioned the value of work that went beyond programming, especially of writing documentation (similar experience is reported in [7]). This is a broader problem with underestimating the documentation in software projects. In line with others' experience [2], most teams complained that the module demanded significantly more work than they were credited for in their curriculum. In spite of initial problems the students responded well to the challenge and several commented that this was one of the most exciting activities of their student career. Most of them valued experience with system integration and reengineering.

## References

[1] S. Becker and A. La Salle. Translating software engineering principles into practice: team frameworks. In P. Klint and J.R. Nawrocki, editors, *Proc. Software Eng. Education Symposium*, pages 54–61, Poznań, Poland, 1998. Scientific Publishers OWN.

[2] S. Biffl and G. Thomas. Preparing students for industrial teamwork: a seasoned software engineering curriculum. *IEE Proc. Softw.*, 145(1):1–11, 1998.

[3] O.P. Brereton et. al. Student group working across universities: a case study in software engineering. *IEEE Trans. on Education*, 43(4):394–399, November 2000.

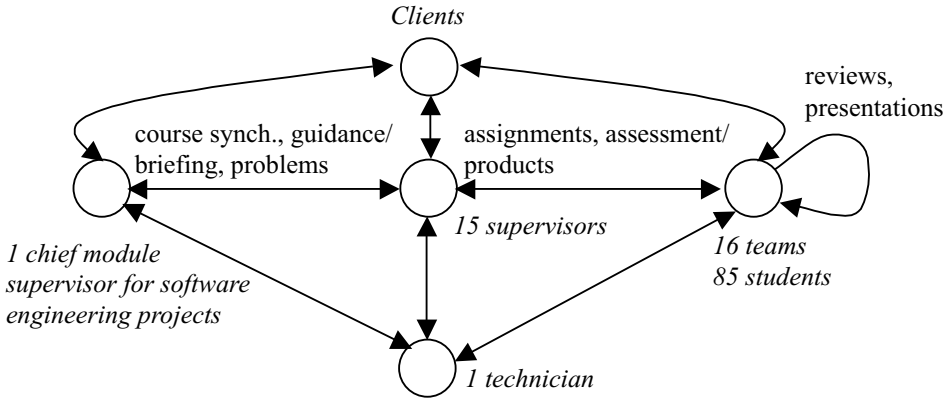


Figure 6: Roles and numbers of people involved in the 2002/2003 Team project.

[4] A. Cockburn and J. Highsmith. Agile software development: The people factor. *Computer*, 34(11):131–133, November 2001.

[5] J. Favela and F. Pena-Mora. An experience in collaborative software engineering education. *IEEE Software*, 18(2):47–53, March/April 2001.

[6] J. Gordijn and F. Niessink. Balancing freedom and supervision in a real-life educational project. In P. Klint and J.R. Nawrocki, editors, *Proc. Software Eng. Education Symposium*, pages 77–84, Poznań, Poland, 1998. Scientific Publishers OWN.

[7] T.B. Hilbourn and W.S. Humphrey. Teaching teamwork. *IEEE Software*, 19(5):72–77, September/October 2002.

[8] M. Holcombe and A. Stratton. Vici: Experiences in introducing student run software companies into curriculum. In M. Holcombe et. al., editor, *Proc. Project 98 Workshop, Projects in the Computing Curriculum*, pages 103–116. Springer, 1998.

[9] W.S. Humphrey. *Introduction to the Team Software Process.* Addison Wesley. Reading, Massachusetts, 2000.

[10] S. Jarzabek (Guest Editor). Teaching software project courses. *Forum for Advancing Software engineering Education (FASE)*, 11(6), 2001. Available at http://www.cs.ttu.edu/fase.

[11] P. Návrat and M. Bieliková. The context and the contents of software engineering education in Slovakia. In P. Klint and J.R. Nawrocki, editors, *Proc. Software Eng. Education Symposium*, pages 141–148, Poznań, Poland, 1998. Scientific Publishers OWN.

[12] P. Návrat and M. Bieliková. Software engineering education: different contexts, similar contents. *SIGSCE Bull.*, 31(2):55–59, 1999.

[13] P. Návrat and L. Molnár. Curricula transformation in the countries in transition: an experience from Slovakia. *IEEE Trans. on Education*, 41(2):88–91, 1998.

[14] L. Ohlsson and C. Johansson. A practice driven approach to software engineering education. *IEEE Trans. on Education*, 38(3):291–295, 1995.

[15] R.H. Rutherfoord. Using personality inventories to help form teams for software engineering class projects. *SIGCSE Bull.*, 33(3):73–76, 2001.

[16] I. Sommerville. Learning from the impossible: the case for wicked problems in teaching systems engineering. In P. Klint and J.R. Nawrocki, editors, *Proc. Software Eng. Education Symposium*, pages 275–283, Poznań, Poland, 1998. Scientific Publishers OWN.

[17] J.E. Tomayko. Carnegie Mellon's software development studio: a five year retrospective. In *Proc. 9th Conf. on Software Eng. Education*, pages 119–129, Florida, 1996. IEEE Computer Society Press.

[18] D.E. Wilkins and P.B. Lawhead. Evaluating individuals in team projects. *SIGCSE Bull.*, 32(1):172–175, March 2000.

*Received:* January, 2003
*Revised:*
*Accepted:* September, 2004

*Contact address:*

Mária Bieliková
Pavol Návrat
Institute of Informatics and Software Engineering
Faculty of Informatics and Information Technologies
Slovak University of Technology in Bratislava
Ilkovičova 3, 842 16 Bratislava 4, Slovakia
e-mail: bielik@fiit.stuba.sk
navrat@fiit.stuba.sk

---

Mária Bieliková received her Ing. (Master) summa cum laude in 1989 from Slovak University of Technology in Bratislava, and her PhD. degree in 1995 from the same university. Since 1998, she is an associate professor presently at the Institute of Informatics and Software Engineering at Slovak University of Technology Bratislava. She (co-)authored one book, several teaching materials and more than 70 scientific papers. She is a member of Editorial Board of Journal of Applied Mathematics & Computing and an editor of five proceedings of the international conferences. Her research interests include knowledge software engineering, information systems modeling and development, especially adaptive hypermedia educational systems.

---

Pavol Návrat received his Ing. (Master) summa cum laude in 1975, and his PhD. degree in computing machinery in 1984 both from Slovak University of Technology. He is currently a professor of informatics at the Slovak University of Technology. During his career, he was also with other universities over-seas. His research interests include related areas from software engineering, artificial intelligence, and information systems. He published numerous research articles and several books. He co-edited and co-authored monographs: Knowledge-Based Software Engineering (Amsterdam, IOS-Press, 1998) and Advances in Databases and Information Systems (Berlin, LNCS Springer, 2002). He was editor of a special issue on Knowledge-Based Software Engineering of the journal Informatica in 2001.