Methods, Knowledge and Tools for Distributed Education in Software Engineering

iProject by the Ministry of Science, Education and Sports, project leader professor dr. Mario Žagar. Project code: 2004-125

Development of information and communication technologies (ICT) is based on the concentration of knowledge and skills, and as such must not be limited by time or geographical distance. The term of global software development has been present in the field for the last ten years and is based on distributed software development, regardless of geographical (kilometres, countries) and time distances (time zones) of the teams involved in the development of the same product. Apart from the familiar problems characteristic of «centralised» program engineering, in distributed software development there are additional problems in the establishment of communication between the distributed teams, caused by differences in professional cultures, languages etc. In cooperation with **Mälardalen University (MdH), Västerås, Sweden**, a pilot project of **DSD (Distributed Software Development)** course was conducted during the academic year 2003/04, in which the teaching staff and students from both faculties were included in joint instruction delivery and development of student projects. The goals of the project were threefold.

- Examine the possibilities and evaluate the risks of maintaining distributed instruction (connecting two locations in real time through Internet streaming of image and sound) by using available communication resources
- Evaluate the feasibility and risks related to the development of student projects which involve simultaneous participation of teams from both faculties
- Evaluate the advantages and disadvantages of the existing virtual learning environment, suggest improvements through project tasks for the next generation of students.

As a result of successful implementation of the pilot project, basic knowledge and experience in the development and maintenance of this type of instruction was gathered. However, success was largely based on the enthusiasm of the faculty and students, which compensated primarily for technical problems such as lack of specialised software and communication equipment. Overcoming these problems would ensure a stabilisation and long-term delivery of the course, as well as expansion of knowledge and acquisition of experience in the studied subject. Apart from supporting the delivery of instruction, the gathered knowledge and experience, as well as the results of student projects, would be made available to a wider range of potential users on higher education institutions as well as to interested parties in the economic sector.

The goal of this project is to establish long-term joint delivery of an undergraduate course in program engineering on two faculties, Faculty of Electrical Engineering and Computing (FER), Zagreb and Mälardalen University (MdH), Västerås, Sweden. Within the course simultaneous delivery of instruction must be ensured, regardless of the location of the students and instructor, by using image and sound Internet streaming. In addition, the project had to enable the students from both locations to work jointly on the software development projects by using synchronous and asynchronous communication technologies and distributed development tools.

The goal of student projects is the development of a tool that would support this type of instruction. One of the desired goals is also to actively include partners from industry (from both countries) with suitable roles in the process of experience and knowledge transfer on distributed implementation of projects and participation in the development of concrete student projects. By participating in this course the students will be acquainted with the technical, social, language and cultural aspects of distance learning in an international environment. Gathered experience and knowledge will be stored in a knowledge base on development and delivery of this type of courses as a support for other similar future projects in higher education, and will be made publicly available through web pages.

The research results show that students are satisfied with the course in general and that they have acquired enough knowledge and experience that will enable them to successfully participate in distributed projects or manage such projects in the future. Instructors on both sides acquired worthy experience in designing and delivering distance courses, as well as learned how to overcome cultural and language barriers.

Course objectives

When planning the course from the design point of view, both sides first agreed upon course objectives. Initial survey amongst potential students showed global interest for the topic, but since the general student awareness for the issues concerning the DSD was relatively low, lecturers and other teaching assistants had to primarily rely on professional body of knowledge in the field of DSD and key issues documented in various books and papers, as well on their own experiences in past collaboration on joint IT projects with different European firms.

It was concluded that by the end of this course, students should be able to:

- identify the differences between classical software development and DSD
- identify centrifugal and centripetal forces found in DSD teams
- assess and evaluate technologies that are used to facilitate DSD
- identify reasons behind corporate decisions to outsource software development to other countries
- understand importance of cultural differences on the DSD project management, communication and collaboration
- recognize key obstacles in the process of DSD
- apply best practices of DSD in their professional work

To achieve these goals it was necessary to blend theory and practice, utilizing constructivist approach to teaching when possible. In the second part of the course, problem based learning through student software engineering projects begins to dominate, giving students opportunity to map experiences with theoretical knowledge acquired in the first part of the course.

Course objectives listed previously were achieved by following means:

- presentation of theoretical background of the matter, so that during practical part students become familiar and develop best practices as well as workarounds for common pitfalls. The theoretical part includes:
    - reasons behind necessity to collaborate with distributed teams in the global IT economy
    - project management issues in DSD
    - software development in pervasive computing
    - Unified modeling language (UML) as an OO software design tool

- bringing in guest lecturers:
    - a software professional or IT manager from corporate software development world to discuss his/hers experiences with students
    - professional psychologist with expertise in cultural issues to lead a fruitful discussion on defining a sensitive borderline between common ethnical and national stereotypes and real characteristics of a certain ethnic society, including understanding of typical high and low context cultures, processes of building trust and decision making, common sets of values and beliefs, etc.
- design of meaningful set of student software development projects that would, at the same time, facilitate:
    - gaining experiences in DSD project management, communication and collaboration;
    - producing software that helps DSD
    - producing high quality project documentation using prepared templates
    - adapting to the level of knowledge and language specifics of another distributed team
    - role-playing as a powerful tool for students to experience being a software development manager, with real-life set of duties and responsibilities**.**

Course design

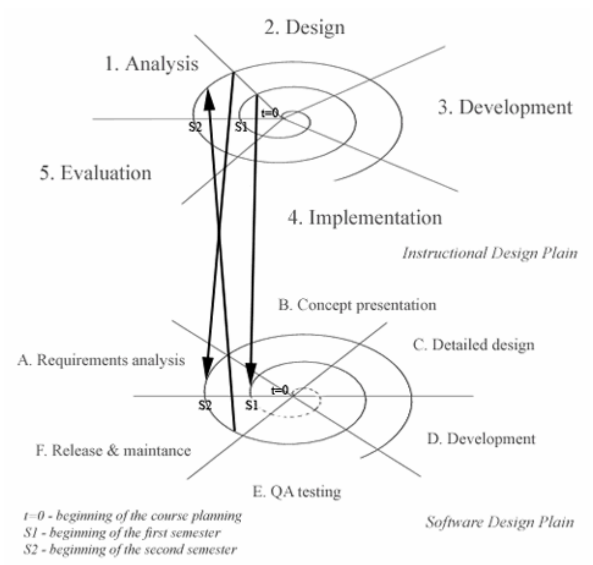
Concerning the course objectives, when it become obvious that producing software that helps DSD would affect future student projects, course content, and potentially also the ways how e-learning is used to deliver courses in virtually joined classrooms, a time-based model with two plains was created. The idea was to describe the way instructional and course design of the DSD course and results of student projects (especially in cases when student teams achieved greater progress) correlate.

*Instructional design phases (**ADDIE** model):*

1. Analysis
2. Design
3. Development
4. Implementation
5. Evaluation

Software development phases of student projects:

A. general requirements analysis, team forming
B. concept presentation and acquirements of other teams' buy-in
C. detailed design[1]
D. development[2]
E. QA testing
F. release and maintenance[3]



On the upper figure, on both plains, one cycle represents one delivery of the course, i.e. one semester. Initial instructional design was done a semester prior to the first delivery. One of the designers of the DSD course, a graduate student from Sweden, spent a semester in Zagreb during that period, planning and creating the course with his Croatian counterparts.

[1] design is presented to other teams
[2] current project state is presented to other teams
[3] the results and product demo are presented to other teams

When the instructional design cycle reaches the “Analysis” phase, before considering changes for the next semester, the results of the students' software development projects are examined and evaluated in order to determine how they can make a meaningful contribution to the instructional and course design for the next semester. Later, during phases “A” and “B”, while deciding on future student projects during course delivery, instructional design cycle begins affecting software development cycle, setting priorities for future software development projects, team formations, inventive role-playing, etc.

Distributed lectures

For delivery of distributed lectures both sides used specially equipped teleconferencing rooms. Both locations have the same videoconferencing equipment (PolyCom 4000) and the reason of many problems was not the equipment itself but the inexperience of the teaching staff. Picture and sound were transferred through available teleconferencing system. However, NetMeeting was used for presentations. It proved to be far more suitable than the transfer of static images which is what the basic system provided.

Each location has two projection screens. On the location on which the lecturer of the current class was, a picture of the remote location was displayed, while on the remote location a picture of the lecturer was displayed. Both sides displayed the same presentation (or some other content) simultaneously, where NetMeeting was used for transfer of content to the remote location. The advantage of the NetMeeting system is its ability to enable shared display of any application where both parties can participate in the control of the current presentation. This is very useful if team members are at both locations.

Guest lecturers

During the three years the course has been held, apart from regular professors (**dr. Ivica Crnković, MdH**, and **dr. Mario Žagar, FER**), there have been several guest lecturers. The objective of such lectures was to provide the students with an insight into how things are done in the business world, as well as into real problems which they might face in program engineering, particularly distributed program engineering. In addition a lot of time is dedicated to project management experience in order to provide support for the students chosen to act as project team leaders. The students are also given information about cultural differences between different countries, which is of particular importance, as for most of the students in this course this was the first time that they had to work with people from different countries.

Our guests were:

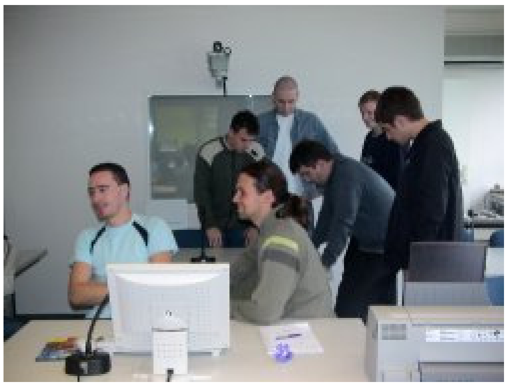
Stig Larsson, ABB: Experiences in Distributed Development
Igor Borojević, IBM: Selected Chapters from Project Environment
Claes Berlin, SAAB: To make business with foreign customers
Anders Berglund, Uppsala University: Distributed course Sweden – USA

Distributed students' projects

Development of concrete project tasks in distributed international environment enables students to both apply the knowledge acquired at the lectures during the preparation part of the course and to gather experience which is only possible through encountering and overcoming challenges in a real project. **Simulation of work environment** similar to that in real distributed projects is not a simple task, primarily because of limitations imposed by the academic environment. **Limited time period** in which the instruction is to be delivered (and harmonised between different academic environments), **limited resources** (finances, equipment) impose some solutions which do not exist in real practice.

Successful instruction, therefore, depends on careful preparation of both the students and organisation and the projects the students will work on. The initial phases of the project (gathering requests, analysis and definition of the project) have to be facilitated for the students either by providing solutions or the structure of the tasks focussed on solutions designed in advance. The task of the teaching staff in the initial phases is to direct the students towards those designed solutions in order for the other phases of the project to be accomplished in the available time: in the time period until the end of the course and the amount of time that the students have at their disposal. Badly managed projects ended either in incomplete implementation or exhaustion of the students because of too much work invested into a badly defined and managed project.

Students' projects



With the beginning of joint distributed lectures student projects are presented and project teams formed most often on voluntary selection basis. An attempt is made to engage an equal number of students from both locations on individual projects, in order to emphasise the distance cooperation component. Immediately after forming the project teams, the team leaders are selected. It is with team leaders that the teaching staff works on:

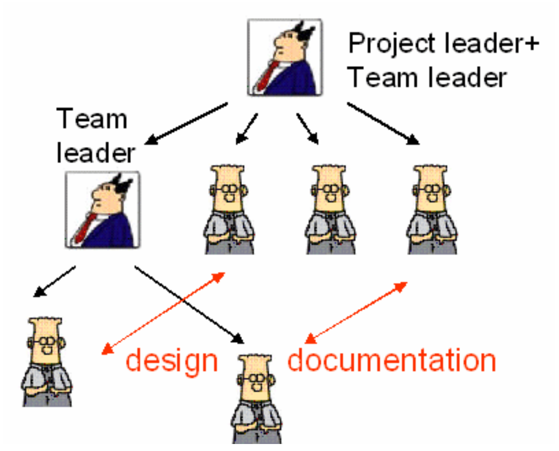
- Assigning project roles to individual team members
- Familiarising them with the project structure, project development process, documentation and available resources
- Submitting the project task
- Development of the initial phases of the project (gathering the requests, analysis and project definition)

Project structure

Management roles

The role of the product contractor is assumed by a member of the teaching staff. Their role is to define the characteristics of the products on which the project team will work. The project contractor communicates only with the project leaders with the goal to:

- Define the demands of the project
- Determine the project plan
- Monitor the project
- Provide final evaluation of the product



The **project leader** role is assumed by one of the students. Their task is to coordinate the work of the project team members, communicate with the leader of the remote team and communicate with the project contractor. In order to achieve a more efficient communication of the project leader with the contractor, the leader is selected from one of the students on the same location as the project contractor. The responsibility of the project leader is to manage the project documentation, send weekly reports to the contractor and coordinate the preparation of the project presentation. It is recommended that the project leader does not participate in the development of the project itself (programming), but they can participate in the writing of the project documentation, testing of the product and other project activities.

The role of the **team leader** has been introduced with the goal to better coordinate the communication between the project team members on different locations. Their task is to coordinate the team members on different locations and ensure efficient information exchange with the remote part of the team with the help of the project leader.

Other roles

After the phases of gathering requests, analysis and determining the system are implemented, the students determine the components of the project, form development teams according to the knowledge of students in each location and assign the roles within the project. The right identification of the components of the system that is being developed, good definition of the component interface and the disitribution of components to specific teams are of extreme importance. In this phase of the project the most prominent is the need for help of teaching staff, in order to avoid wrong identification of components due to lack of experience on the part of the students, which would cause the project to fail or would cause severe problems in the coordination of the team during the development of the project. Intervention of the teaching staff is two-fold: in the project preparation phase the project task is defined by establishing clearly expressed and relatively independent components with clear interfaces, while in the planning phase students are helped by providing advice on how to distribute the components and by clear definition of interfaces and accompanying documentation.
Development team members are responsible for the development of the program product. Each team is in charge of the development of one or more components of the final product and their integration into the final product. Cooperation of the development team members with the other roles within a team is necessary for coordination with the other distance teams as well as with the other roles within their own team. Coordination in the use of resources shared between several

teams, such as a system for management of the database and different versions, is of particular importance.

There are two approaches to distribution of work on the project components:

- Distributed approach presupposes including students from several locations on the development of a single component of the system.
- Centralised approach limits the participation of students to those from only one location when developing individual components of the system.

In addition, distributed approach presupposes a far greater need for distance communication between the team members, which prolongs the work and includes great risk of the final success of the development.

Project monitoring

Monitoring the project is based on two mechanisms: weekly reports by the project leader and presentation of the state of the project at roughly two thirds into the time available for the project.

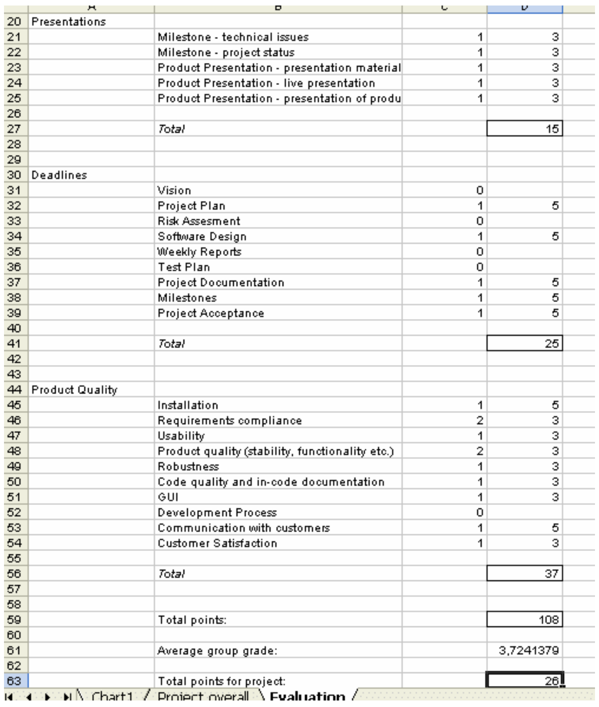
By developing a **project plan**, after having successfully determined the phases of analysis and definition, students provide a rough project implementation plan by activities. Weekly reports, which are the responsibility of the project leader, provide the contractor (as well as the team members) with an insight into the progress of the project, problems that were identified and how these problems could be solved. The final analysis of the project progress is given in the final project report, in which the students analyse the flow of the project, the errors that were identified and evaluate the quality of the risk assessment.

Presentation of the state of the project puts the students in a position in which they have to defend the current state of their project in front of the project contractor. In addition, all the students are requested to participate in the presentation, in order to acquire a more accurate picture of their specific roles and their degree of involvement in the project.

Project evaluation

The project grade is formed based on various parameters distributed into three basic groups:

- Quality of the project documentation
- Adhering to the deadlines
- Quality of the final product



	A	B	C	D
20	Presentations			
21		Milestone - technical issues	1	3
22		Milestone - project status	1	3
23		Product Presentation - presentation material	1	3
24		Product Presentation - live presentation	1	3
25		Product Presentation - presentation of produ	1	3
26				
27		*Total*		15
28				
29				
30	Deadlines			
31		Vision	0	
32		Project Plan	1	5
33		Risk Assesment	0	
34		Software Design	1	5
35		Weekly Reports	0	
36		Test Plan	0	
37		Project Documentation	1	5
38		Milestones	1	5
39		Project Acceptance	1	5
40				
41		*Total*		25
42				
43				
44	Product Quality			
45		Installation	1	5
46		Requirements compliance	2	3
47		Usability	1	3
48		Product quality (stability, functionality etc.)	2	3
49		Robustness	1	3
50		Code quality and in-code documentation	1	3
51		GUI	1	3
52		Development Process	0	
53		Communication with customers	1	5
54		Customer Satisfaction	1	3
55				
56		*Total*		37
57				
58				
59		Total points:		108
60				
61		Average group grade:		3,7241379
62				
63		Total points for project:		26

This puts an emphasis on the comprehensiveness of the quality of project development and implementation, and not only on the quality of the product, which is usually the only measurement of success used in project organisation of instruction.
The result of the project grade is a number of points, from 0 to (the number of project team members x 5). It is the project leader's task to distribute the number of assigned points to different team members in order to get the grades for the

individual team members. In order to prevent conflict of interests, a rule has been implemented by which the project leader is graded with the average grade of the project.
The final grade of a student is a combination of their participation in the project and the grade of the final paper in which a description and analysis of working in a distributed project is provided.

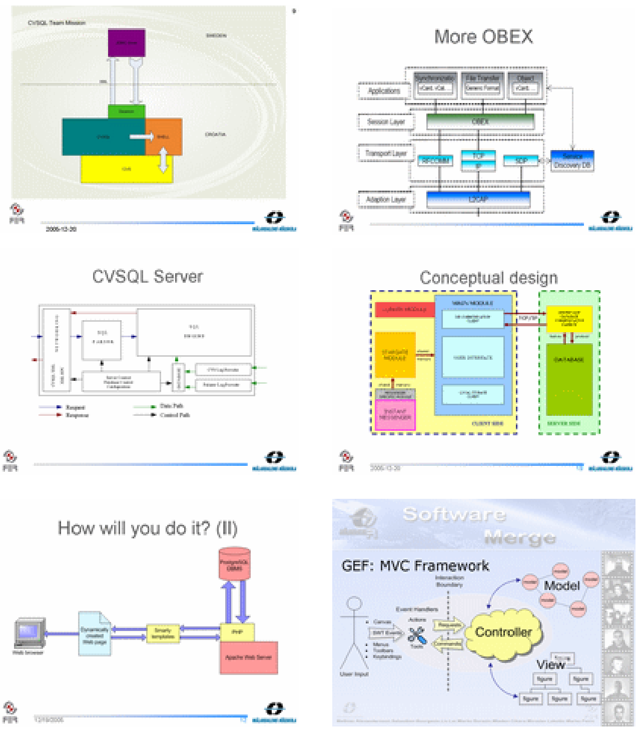
Finished projects themes

In the three years of organizing the Distributed Software Development course, 14 projects have been made. The goal of most of them was to develop or update the tools which would be used as a help in the course. Some projects were continuation of the projects from the previous generations. Project themes were:

2003./2004.
- CVSQL
- CVS Reporting
- Chat
- WebProject
- Bluetooth stack

2004./2005.
- CVSQL
- CVS Reporting
- Chat

2005./2006.
- VLab
- WebProject
- WebProject2
- LoCo
- Merge
- SoMerTime

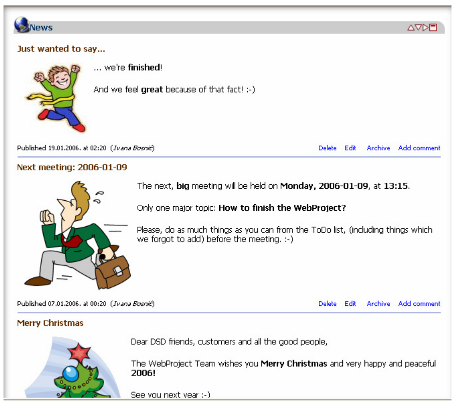


More information about the finished students' projects can be found on iProject Web page.

Communication and Tools

Resulting from instructional and course design, following technical infrastructure needed to fulfil the requirements:

1. synchronous virtual environment to facilitate 2-way course delivery in which classrooms in both Sweden and Croatia are virtually joined
2. asynchronous Web-based environment for the project coordination, collaboration and communication of distributed teams
3. asynchronous Web environment for conducting student surveys as part of course evaluation at the end of the instructional design cycle
4. specialized software engineering tools primarily focused on code development and maintenance, commonly known as (**Software Configuration Management - SCM**) tools


News

Just wanted to say...

... we're **finished**!

And we feel **great** because of that fact! :-)

Published 19.01.2006. at 02:20 (*Ivana Bosnić*) Delete Edit Archive Add comment

Next meeting: 2006-01-09

The next, **big** meeting will be held on **Monday, 2006-01-09**, at **13:15**.

Only one major topic: **How to finish the WebProject?**

Please, do as much things as you can from the ToDo list, (including things which we forgot to add) before the meeting. :-)

Published 07.01.2006. at 00:20 (*Ivana Bosnić*) Delete Edit Archive Add comment

Merry Christmas

Dear DSD friends, customers and all the good people,

The WebProject Team wishes you **Merry Christmas** and very happy and peaceful **2006!**

See you next year :-)

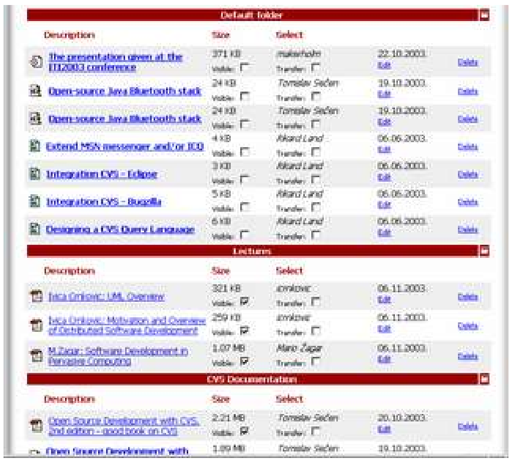
During the cooperation on the project development, students have both **synchronous** and **asynchronous** mechanisms of communication at their disposal. Synchronous mechanisms that are used are videoconferences, teleconferences and chat. The asynchronous mechanism are far more numerous and include e-mail, mailing lists, forum, file repositories, version management system etc. It should be noted that, due to the specifics of student time management, asynchronous communication systems were far more used than the synchronous.

Synchronous Communication

Students can use the videoconferencing system during the joint lectures, as well as NetMeeting videoconferencing at any time they set up for themselves. The teaching staff provides the required videoconferencing equipment (computers with fast Internet connection, microphones, cameras). Apart from critical situations, telephone calls and teleconferences were limited to using NetMeeting and Skype systems, due to high telephone prices.

MSN Messenger, ICQ and other chat systems were mostly used for shorter exchange of ideas.

Asynchronous Communication



Default folder

Description	Size	Select		
The presentation given at the ITI2003 conference	371 KB Visible: ☐	makerholm Transfer: ☐	22.10.2003. Edit	Delete
Open-source Java Bluetooth stack	24 KB Visible: ☐	Tomislav Sečen Transfer: ☐	19.10.2003. Edit	Delete
Open-source Java Bluetooth stack	24 KB Visible: ☐	Tomislav Sečen Transfer: ☐	19.10.2003. Edit	Delete
Extend MSN messenger and/or ICQ	4 KB Visible: ☐	Rikard Land Transfer: ☐	06.06.2003. Edit	Delete
Integration CVS - Eclipse	3 KB Visible: ☐	Rikard Land Transfer: ☐	06.06.2003. Edit	Delete
Integration CVS - Bugzilla	5 KB Visible: ☐	Rikard Land Transfer: ☐	06.06.2003. Edit	Delete
Designing a CVS Query Language	6 KB Visible: ☐	Rikard Land Transfer: ☐	06.06.2003. Edit	Delete

Lectures

Description	Size	Select		
Ivica Crnković: UML Overview	321 KB Visible: ☑	icrnkovic Transfer: ☐	06.11.2003. Edit	Delete
Ivica Crnković: Motivation and Overview of Distributed Software Development	259 KB Visible: ☑	icrnkovic Transfer: ☐	06.11.2003. Edit	Delete
M.Žagar: Software Development in Pervasive Computing	1.07 MB Visible: ☑	Mario Žagar Transfer: ☐	06.11.2003. Edit	Delete

CVS Documentation

Description	Size	Select		
Open Source Development with CVS, 2nd edition - good book on CVS	2.21 MB Visible: ☑	Tomislav Sečen Transfer: ☐	20.10.2003. Edit	Delete
Open Source Development with	1.09 MB	Tomislav Sečen	19.10.2003.	

Within the course, a web page was opened on the **FER Content management system** containing the information on the course itself (communication: teaching staff - students) as well as web pages of individual projects (communication student - student). These pages enable the publishing of news, creating discussions and saving files and links.

E-mail and mailing lists are the most common type of communication between team members, but the problem with this type of communication is that it is open and can be seen by the teaching staff.

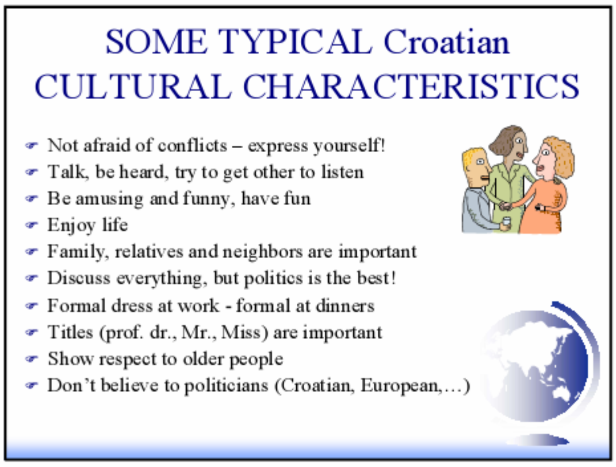
The source code of the projects was to be maintained and coordinated by using the **CVS** system, however, its successful use presupposes that the students' have foreknowledge on configuration management.

Cultural and institutional issues and workarounds based on real-life project experience

Cultural differences

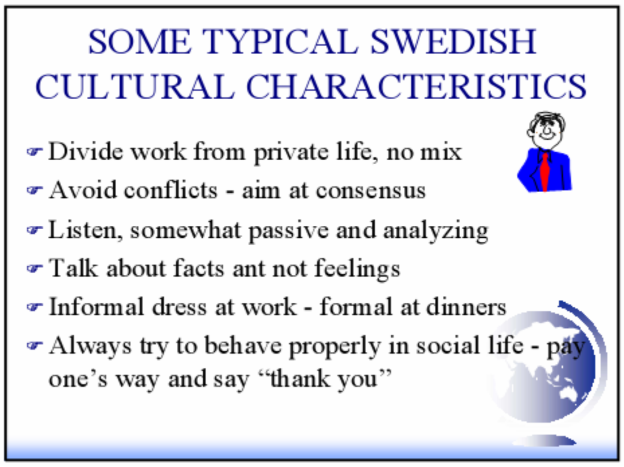
In this type of a course, the communication is one of the most important factors for success of the project. Since all students at the beginning of the course didn't know each other at all, it was important that they got to know each other very soon and learned each other's advantages, but also disadvantages. When they are cooperating with other students from the same environment they are able to evaluate other person much faster than in the case of cultural difference. The way students work, differs in Croatia and Sweden, which was the source of a large number of misunderstandings at the beginning of each team's project.

This is in no way surprising. Swedish culture is a typical "Low-context" (explicit) culture, while Croatia fits well in the "High-context" (implicit) culture model, typical for the Mediterranean region. Differences include time needed for building trust and creating relationships, notion of time (deadlines), space and the way verbal and written agreements are understood. Swedish culture is typical "monochronic", taking agreed schedules very seriously, while Croatian is more "polychronic", where commitments are achieved only if possible and are understood more as guidelines then a strict obligation. This problem was tackled at the beginning of the DSD course and students were warned to take this into consideration when communicating with each other.



SOME TYPICAL Croatian CULTURAL CHARACTERISTICS

- Not afraid of conflicts – express yourself!
- Talk, be heard, try to get other to listen
- Be amusing and funny, have fun
- Enjoy life
- Family, relatives and neighbors are important
- Discuss everything, but politics is the best!
- Formal dress at work - formal at dinners
- Titles (prof. dr., Mr., Miss) are important
- Show respect to older people
- Don't believe to politicians (Croatian, European,…)



SOME TYPICAL SWEDISH CULTURAL CHARACTERISTICS

- Divide work from private life, no mix
- Avoid conflicts - aim at consensus
- Listen, somewhat passive and analyzing
- Talk about facts ant not feelings
- Informal dress at work - formal at dinners
- Always try to behave properly in social life - pay one's way and say "thank you"

In order to successfully finish the project, students very soon realized the importance of understanding. At the very beginning all students had to give their own evaluation of their capabilities to their project leader. That evaluation had to be realistic since the success of the complete project depended on it. Students have a lot of difficulties to acknowledge their weaknesses especially when they are in an environment that is unknown to them. Additional problem arose when they didn't know what kind of knowledge is expected. In development of software products, students didn't know what technologies were covered in the curriculum in the other country and were very worried whether they would be able to do all that was expected from them.

The teams in which all members knew all team weaknesses, but also its strengths, achieved much better results, since they tried to adapt the project that was given to them to their abilities. Teams which didn't built enough trust experienced a lot of problems even at the very beginning of the project. Due to the cultural differences students had a hard time distinguishing other student's cultural differences from their weaknesses, which is a big problem in building trust.

One way of establishing trust is to introduce discipline in the communication. Another way is to request regular progress report from all teams. These requests force students to give real estimation of their own capabilities to the other students, which builds trust. In order to ensure success of the project it is important to define all milestones at the beginning of the project and the expected results for each of them. It is also important to clearly define all vaguely defined areas of the project and responsibilities of each team member.

Teams that have managed to follow these rules achieved much higher success than those where the role of each member was not precisely defined or some parts of the project remained unclear.

Institutional differences

Another "cultural" difference, this time institutional, concerned the ECTS points.

Students who enrolled into the DSD course in Croatia (last semester, the count was 19) were enabled by the Swedish partner to officially enroll to the Mälardalen University (MdH). They were given 7.5 ECTS points for participating in the DSD course. Croatian students who enrolled in MdH were also credited with 7.5 ECTS. However, Croatian students who took part in the course got only 4 ECTS points for the same course from the Croatian side (FER), which was quite confusing and non-stimulating. The Bologna process will have to mature in Croatia before such obvious injustices are properly addressed.

Students' surveys

Results of the anonymous surveys about the success of education in Distributed Software Development course, for the past three years, can be found at:

http://www.idt.mdh.se/kurser/cd5610/2003/evaluator/html/summaries/dsd_2003Summary.html
http://www.idt.mdh.se/kurser/cd5610/2004/evaluator/html/summaries/dsd_2004Summary.html
http://www.idt.mdh.se/kurser/cd5610/2005/evaluator/html/summaries/dsd_2005Summary.html

Overall grade of the course from the student perspective was **4.23** (out of 5) in 2003, **4.53** (out of 5) in 2004. and **4.75** (out of 5) in 2005.

Students' comments

Here's what students said about their course experience:

The course in general:
"This course was one of the best that I had on faculty."
"I was scared at beginning, but now I feel lucky I had been a part of this course."
"With respect to that it was the first time for the course, it was excellent!"
"This course was a very good idea. Hope the next generations of students will enjoy it, too."

About the workload:
"Another week and I would have died."
"Great new experience!"
"...I found out that it takes a lot of work to make something work as you want it to work."
" I learned much more than I expected. I learn lot of team working, good organization of work, time and documentation importance."

The most liked things about the course were:
"meeting students from a different country."
"experience gained from working distributed."
"team work...You know, during the study we hadn't course like this, and I love it."

Wish list for the future:
"Easier projects, better equipment for distributed work, final grade must not depend only on project results."
"Real face-to-face contact for at least one day (~8-10 hours) in ~2-3 week of project. It would provide us with a lot of information about the other side that could be helpful in future contact and way of communication. We still don't know very much about each other."

"More guest lecturing..."
"Start the project earlier so we don't have to spend the Christmas Eve glued to the computer screen."
"More lectures about cultural differences and more about project management and team leadership."
"Availability of technical resources."
" ...more video conferences."

"All in all, DSD course in 2003/2004 was nothing short of a revelatory experience! It was radically different from all other courses undertaken in usual student "lifetime" and closer to the real world even more than everybody wanted to go."

iProject Distributed Software Development Web page

More information about this iProject can be found at:

http://www.fer.hr/rasip/projekti/rpi

Contact:
Prof.dr.sc. Mario Žagar
E-mail: mario.zagar@fer.hr
Tel./Fax: +385-1-6129617

Project Team

Participants on the project were:

- Prof.dr.sc. **Mario Žagar**
- **Kristijan Zimmer,** dipl.ing.
- Mr.sc. **Igor Čavrak**
- **Ivana Bosnić**
- **Siniša Tomić**, dipl.ing.
- **Vlatka Paunović**, dipl.ing.
- **Marin Orlić**, dipl.ing.
- Mr.sc. **Branko Mihaljević**

- Translator: **Branka Vuk - Koračak**

Prof.dr.sc. Ivica Crnković and **Mr.sc. Rikard Land,** both from the Mälardalen University (MdH), Department of Computer Science and Electronics, Västerås, Sweden are responsible for half of the work done in preparing and establishing the described DSD course. We express our gratitude to them for all they have done in the past and will do in the future.