

Student Collaboration across Universities: A Case Study in Software Engineering

O. P. Brereton, S. Lees and R. Bedson
*Department of Computer Science
Keele University
Keele
Staffordshire ST5 5BG, UK
email: o.p.brereton@cs.keele.ac.uk*

C. Boldyreff and S. Drummond
*Department of Computer Science
University of Durham
Durham DH1 3LE, UK*

P. Layzell, L. Macaulay and R. Young
*Department of Computation
UMIST
PO Box 88
Manchester M60 1QD, UK*

Abstract

Distributed group working amongst teams of software engineers is increasingly evident in the 'real world'. Tools to support such working are at present limited to general-purpose groupware involving video, audio, chat, shared whiteboards and shared workspaces. Within software engineering education, group tasks have an established role in the curriculum. However, in general, groups are local to a particular university or institution and are composed of students who have a significant shared history (in terms of technical background and social interaction) and who are able to meet face-to-face on a regular basis. This paper reports on work undertaken by three UK universities to provide Computer Science students with the opportunity to experience group working across universities using low-cost tools to support distributed co-operative working.

1. Introduction

The benefits of collaborative working in software engineering education have been widely recognised [1,2,3,4] and stem from its role in reinforcing the more formal teaching through simulating real-world situations. Collaborative group projects highlight many of the difficulties associated with management activities such as scheduling meetings, task allocation and negotiation as well as addressing technical issues (e.g. agreeing requirements, system partitioning and integration).

It is clear that industrial software engineering is increasingly being carried out by teams that are geographically distributed across a number of sites [5,6]. The ever-increasing presence of the Internet together with improved groupware is likely to provide further support for distributed co-operative working. It is against this background that we have sought to provide students with the opportunity to undertake distributed group working in software engineering.

This paper focuses on distributed group 'sub-projects', which formed a part of individual project work carried out by final year Computer Science undergraduates. The sub-projects involved groups of 3 students (one from each of Keele University, UMIST and the University of Durham) which were required to specify, design, develop, test and document a software application. The experience gained from the task fed into the individual student projects. Section two of the paper gives an overview of the collaborative 'super project' to develop a 'Virtual Community for Student Groupwork' which provided the framework for the sub-projects. Section three presents the details of the sub-projects, section four summarises the results of our evaluation activities, and section five focuses on the lessons learned.

2. Background

The collaborative sub-projects described in this paper were carried out as part of work funded within the UK's Joint Information Systems Committee's Technology Applications Programme which aims to further the development and exploitation of communications and information technologies for the benefit of the UK higher education and research communities. The overall (super) project, 'Developing a Virtual Community for Student Groupwork (JTAP-2/140)', was undertaken by members of the Department of Computation at UMIST (Manchester) and the Departments of Computer Science at Keele and Durham Universities. The consortium investigated the development and execution of distributed software engineering group activities supported by co-operative working tools.

The main objectives of JTAP-2/140 were:

- to give students experience of working collaboratively in a geographically distributed team using modern groupware technology (including video conferencing);
- to develop staff experience of operating distributed, collaborative projects;
- to produce documentation and experience reports which will enable such projects to be successfully implemented in other institutions.

These objectives have been addressed in three stages. The initial set-up stage involved the selection, acquisition and setting up, at each site, of a low-cost PC-based co-operative environment including video conferencing, shared drawing and shared repository facilities [7]. The second stage was made up of a series of small case studies involving software engineering tasks carried out by distributed groups of students using selected synchronous group working tools [1,8,9]. The final stage, which is the subject of this paper, involved larger-scale collaboration sustained over a number of weeks and encompassing both synchronous and asynchronous working.

The organising team at each university was composed of one or two academic staff plus a research assistant. The three teams occasionally met face-to-face (approximately 3 monthly) at one of the sites. Most of the organisation of the sub-projects was undertaken

by the three research assistants using Computer Supported Co-operative Working (CSCW) technology. This, of course, provided a valuable learning opportunity in itself.

The planning of such collaborative working needed to take into account a number of factors including:

- different teaching periods at the three universities (those at UMIST and Keele are based on a two-semester year whilst Durham has a three-term academic year);
- different degree structures and technical backgrounds of participating students (Durham and UMIST have traditional 3-year full time undergraduates whilst Keele has a joint degree system within which most Computer Science students spend slightly less than half their time studying the subject);
- different final year project requirements, especially in terms of deliverables and assessment criteria, at each university.

3. Student collaboration

Student collaboration took the form of sub-projects carried out as part of each student's final (third) year project. At each participating university, students undertake substantial individual third year projects. However, the start dates, end dates, duration and weightings of these all differ across the universities. Within these constraints, two periods, totalling seven weeks were available for collaboration. Students were expected to devote approximately six hours a week to the sub-projects. The role of the sub-project in relation to the student's individual project (especially in terms of its contribution towards assessment) differed across the universities. The main differences derived from the fact that all final year projects required an implementation of some artefact. For some students, this was provided by the implementation component of the sub-project whilst others undertook an implementation in addition to the sub-project.

Each organising team (i.e. one at each university) provided and managed a software engineering task that constituted a sub-project. Each task was undertaken by two groups of three students (one from each site). Therefore, overall, 18 students took part. The research assistants provided additional supervision and technical support.

Based on previous work, we considered it to be beneficial for the students to meet each other prior to the collaboration. The first meeting was held immediately before the first period of collaboration and took the form of an overnight stay in a hotel in Cumbria. At this meeting students were told about the aims and objectives of the JTAP project as a whole and about the role of the sub-projects that they were to undertake. They were also given outline system requirements and an opportunity to discuss these with the systems' customers. Perhaps most importantly, they were able to get to know each other a little before starting on what was a quite intensive period of distributed collaboration.

Just before the second period of collaboration students again met face-to-face (at the site of the organisers of their sub-project) to discuss any problems which had arisen, to get further clarification of user requirements and to plan and co-ordinate the work ahead.

In the remainder of this section, we outline the technology used and the tasks undertaken by the distributed groups of students.

3.1 Technology

Students used personal computers running a range of tools for synchronous and asynchronous working. A typical configuration is shown in Table 1.

Further details of the environment used for synchronous working, including analysis of requirements, procurement, implementation and life-cycle issues (such as logging and printing) are available in a guide to 'Procurement and set-up of low cost Desktop Video Conferencing (DTVC) in a student environment' [7]. Further information on 'Adoption and Evaluation of the use of low-cost DTVC to support distributed student groupware' [8] and 'Setting up Video taping of DTVC - screen and audio' [9] are also available.

Access to the equipment varied across the three universities both in terms of access to the building and availability of the room(s) housing the equipment. Other variations included whether there was access to a telephone near the DTVC equipment so that problems could be sorted out in the case of failure of any element of the hardware, software or of the organisation of the synchronous working session.

3.2 Tasks

Each of the sub-projects involved the specification, design, implementation and documentation of a database application. The initial information provided to the students contained an informal description of the user domain and outline system requirements. Students were also given instructions about the phases to be undertaken and the deliverables to be produced.

Comparative shopping monitor: This sub-project was organised by the Durham team. The customer, who provided an outline system specification, was an IT specialist at a well-known high street store.

The comparative shopping monitor was to be used by the customer to build a database of company and competitor products. The monitor had to allow products to be entered into the database along with information such as price and weight. It had to allow products to be grouped into categories, e.g. wine or yoghurt. These categories had in turn to be grouped into departments such as dairy and international foods. Each department within a store would have its own database and would operate essentially as a separate company. It had to be possible to move product information from one category to another (within the same database) and from one department to another (by means of a floppy disk).

The system had to be able to calculate and display percentage price differences, price per unit and outright price differential for a given product, product group or department.

Room booking system: This sub-project was organised by the Keele University team and the customer was a Computing Officer within the Computer Science department at Keele. The room booking system was intended to automate and improve the manual departmental room booking system previously in use.

The system had to support the placement and cancellation of lecture room and laboratory bookings by approved staff both from within and outside the department. Bookings were to be on a per hour basis and could be either for one week only or across a block of weeks (e.g. for a short course, a semester or an academic year). Rooms had to be described in terms of maximum occupancy and available resources. It had to be possible to add and remove rooms and to modify the information held about rooms. Reports were to include a weekly timetable for each room.

Student hall of residence pastoral care system: The pastoral care sub-project was organised by the UMIST team and the customer was a warden at one of the UMIST Halls of Residence. Previously, a paper based filing system was used to record student details together with notes taken during interviews with students who have problems.

The system had to be able to store information (such as name, room number, doctor's details, course of study, problems reported and action taken) for up to 360 students in a university Hall of Residence. It had to be possible to access the information based on date, flat, block or room number or student name. Printed reports were to include end of year reports and individual student records.

4. Evaluation

The collaborative software engineering activities (sub-projects) described above were evaluated in terms of administration, software engineering issues and technologies. Student views were obtained using a questionnaire and through the comments and conclusions that they expressed in their final year project reports. In particular, the questionnaire collected opinions on the technologies used (video, audio, chat, whiteboard and shared repository), the software engineering tasks and working as a distributed team. Sixteen of the participants completed the questionnaires.

Other results were obtained through observation and expert opinion (in particular, relating to the quality of the life cycle deliverables).

4.1 Administration

A number of overall administrative problems arose. These included:

- timetable differences sometimes made it difficult for students to attend synchronous working sessions
- there were occasional hardware and software failures leading to ineffective synchronous working
- it became clear during the second collaborative period that the individual goals of some of the students differed significantly from their fellow group members, particularly with respect to completing the implementation component of the sub-project
- maintaining compatible versions, across the three sites, of software being used during system development was a problem. In particular, it had been agreed at the start of the sub-projects that the applications would be developed using Microsoft Access 95. This was provided on the personal computers available to the students at each site. However, this was a problem to some students who had Access 97 on their own computers, as the databases they created using Access 97 were not directly usable with or convertible to Access 95. The problem was compounded when one participating university department upgraded to Access 97 during the sub-projects.

4.2 Software Engineering Issues

The students encountered some groupworking difficulties. In particular, one student failed to contribute to his group's activities, some students regularly failed to turn up for synchronous working sessions and some difficulties arose in reaching agreement (e.g. on specifications, overall life cycle and design methods). The students participating in the projects were self-selected and although they were enthusiastic about their participation they were of mixed ability and had varying degrees of commitment to the work.

These problems are of course typical of group working whether at a single site or across many sites. Unfortunately the distributed nature of the working and of the overall management meant that we were slow to recognise and resolve the problems.

The students were asked to assess how successfully they felt they had completed the task using DTVC (only fifteen participants responded), and then asked to compare their achievements with how well they felt the group would have completed the task under face-to-face conditions. Table 7 summarises their responses.

Supervisors at each institution assessed the products resulting from the period of collaborative work. The requirements deliverable was generally well attempted by all groups, and adequate design was carried out. However, the collaborative results from the implementation phase were of variable quality, some groups producing poor work, whilst others produced good products.

4.3 Technologies

On the whole the students found the CSCW tools used both simple and rewarding. The range of communication options was thought by most of the students to be sufficient to support the group in accomplishing the software engineering tasks required of them. The results outlined below were obtained principally from the questionnaires and student project reports.

Audio: Opinions on the use of the audio facilities provided by the synchronous communications tools were mixed. Occasionally the audio failed to work, but when functional, the quality was found to be acceptable (just). The main reason for audio failure seems to have been incorrect use. Table 2 summarises questionnaire responses.

Whiteboard: The WhitePineBoard v3.0 enabled students to make synchronous annotations to shared documents. The students enjoyed using the tool and found it easy to use although the user interface and range of features available were felt to be limited. For effective use of the tool, the students found it necessary to establish protocols for managing access to the whiteboard. Table 3 summarises questionnaire responses.

Chat: The chat facilities received favourable comments. In particular, chat provided an invaluable backup to the audio provision. In fact, most students used the chat facilities only as a back up to failing audio. For one student who was investigating distributed working without the use of audio it was clearly essential. One of the benefits of chat over audio was that it provided an automatic record (log) of a conversation. Analysis of these logs formed the basis for one student's investigation into percentages of time spent on discussing the sub-project work, problems with the technology and other matters (e.g. socialising). Table 4 summarises the responses about the chat facilities.

Video: Video was generally thought to be useful, however, some performance problems did occur with the result that small or fleeting gestures were lost. Some difficulties arose due to an inability to maintain eye contact over the video and the lack of synchronisation between lip movements and speech reception. Table 5 summarises responses to questions about the use of video.

Shared Repository: All of the groups chose to use BSCW to support asynchronous working. Although Lotus Notes was available as an alternative, the students found it took considerably longer to become proficient in its use.

The features of BSCW that were found to be most useful were:

- the ability to upload and view documents whilst in a conference with the document author;
- document sharing;
- uploading documents in various formats;
- remote file storage;

- the ability to access documents from anywhere.

Features that contributed to group working included:

- versioning;
- scheduling of meetings;
- threaded discussions;
- the ability to check whether co-workers had read files
- the fact that BSCW created a group Intranet for the project.

Students found BSCW easy to use, valued its security features, and commented that it would have been difficult to complete the project without it. It was also possible for staff to monitor student activities by inspecting the BSCW logs that automatically record all user operations in the workspace.

On the negative side, some students commented that it could be slow to use, and thought that for groups of two students, e-mail would be more effective. Questionnaire responses are shown in Table 6.

5. Lessons Learned

The lessons learned are summarised below.

5.1 Administration

The three institutions did not have any previous experience of organising distributed collaborative student projects and took on this aspect of the wider project as the obvious extension of the previous exploratory phases. It is clear that regular monitoring of group progress is necessary and managing collaborators where only one member of the group is local to the sub-project supervisor can present difficulties.

In the planning phase, the value attached to such projects should be standardised across the participating institutions as far as possible. Consideration should be given to the precise deliverables of the project and their method of assessment at each site.

The role of sub-project supervisors, which during this study was filled by the research assistants, is crucial. They should meet regularly with group representatives to discuss overall group progress, difficulties etc. A mechanism for the reporting of faults needs to be established, training in video conferencing skills needs to be provided, and technical support should be readily available to the students.

Problems of group cohesion and working should be anticipated and advice given to the groups. The importance of commitment to the group needs to be emphasised, as well as the need for collaboration throughout the project. When/if problems occur, they should be identified at an early stage. In the case of the failure to contribute by a team member, there should be fall back mechanisms available to support the group efforts, as it is important to ensure that it is possible for good students to get good marks regardless of other group partners skills or efforts. Consideration could also be given to the learning curve for software packages, in our case MS Access, and perhaps training documents should be circulated to the students prior to the projects.

5.2 Software engineering issues

Collaborative working involving three university departments gave students access to a wider field of expertise, which is extremely valuable to small departments. During the period of collaboration, students benefited from the diversity of educational experiences and methods that were brought together.

The collaborative experience proved to be valuable. Students learned about the problems of co-operating without a shared history. Working collaboratively on the sub-projects clearly instilled in the students a professional attitude, which would have been difficult for them to achieve without experience.

Project Management: Students gained project management skills during the period of collaboration. Groups found it necessary to decide on individual roles, and to fit those roles to the length and complexity of the project.

Tasks: One of the clear findings that came out of the sub-projects was the value of having a task that spanned the whole software development life cycle. The time allotted to the task meant that the time-scale was tight, however, it was felt that the correct balance was achieved. Basic, but working systems were produced, though the extent of collaboration diminished in the closing weeks of the project.

Groupworking: Though the teams generally worked well together, pointing out a few potential pitfalls to students embarking on collaborative work could save them from unnecessary difficulties. For example, versions of development tools (e.g. MS Access) need to be agreed at the outset, and strictly adhered to. Also, guidance on use of versioning, branching etc. when using BSCW would help students more effectively organise their workspace.

In order to maintain the quality of co-operation that would more closely resemble joint projects within the same institution, groups should be encouraged to plan for longer and more frequent synchronous working sessions. Advice on a suggested minimum time to meet each week would be useful. Provision for emergency contact procedures should also be made, in case of hardware failure or group member absence (e.g. readily available telephone access, phone numbers pinned up near the DTVC equipment.)

Technologies: It was found to be important to agree file formats for upload to BSCW so all group members were able to use them. In addition, compatible versions of the support applications such as word processors need to be available at all sites.

Versioning is a useful way of managing documents that are being worked on collaboratively. Students used it to varying degrees on BSCW. Though it was not initially used, most of the groups ended up using it extensively whilst working on collaborative implementation. It served the twofold purpose of preventing the proliferation of redundant copies of the database, and enabling other users to quickly locate the latest version of the document in the workspace.

In order to limit technical difficulties during synchronous working sessions, proper use of the equipment needs to be emphasised prior to the period of collaboration, and potential difficulties stressed. Discussion of these could constitute part of an induction session.

6. Conclusions

Overall, the student and staff experiences of collaborating across universities have been very positive. Students enjoyed using the technologies and felt that their employability was enhanced. Staff benefited directly through increased knowledge of Groupware technologies and of organising and managing distributed group projects. Staff also gained through the sharing of information and expertise relating to undergraduate courses (especially on CSCW) and procedures for managing student

projects. In addition, there has been a significant strengthening of research links between the participating departments.

Acknowledgements

We acknowledge the support of the Joint Information Systems Committee's Technologies Application Programme through which the work reported here was funded. We would also like to thank Matt Gumbley for his contributions to the project.

References

- [1] O. P. Brereton, S. Lees, M. Gumbley, C. Boldyreff, S. Drummond, P. Layzell, L. Macaulay and R. Young, 'Distributed group working in software engineering education', *Information and Software Technology*, Vol. 40, No 4, July 1998, pp 221-227
- [2] N. Habra and E. Dubois, 'Putting into Practice Advanced Software Engineering Techniques through Students Project', 7th SEI CSEE Conference, San Antonio, Texas, USA, January 1994, pp 303-316
- [3] D. Gotterbarn and R. Riser, 'Real-World Software Engineering: A Spiral Approach to a Project-Oriented Course', Proc. 7th. Conference on Software Engineering, San Antonio, Texas, USA, January 1994, pp 119-150
- [4] P. N. Robillard, 'Measuring Team Activities in a Process-Oriented Software Engineering Course', 11th Conference on Software Engineering Education and Training, Atlanta, Georgia, IEEE Computer Society, February 1998, pp 90-101
- [5] L. F. Truett, E. Z. Faby, J. W. Grubb, J. P. Lofit and P. C. Shipe, 'Co-ordination of Software Development Among Sites That Are Separated', Proc. COMPSAC-17, Phoenix, IEEE, 1993, pp. 70-75.
- [6] I. Gorton and S. Motwani, 'Issues in Co-operative Software Engineering Using Globally Distributed Teams', *Information And Software Technology*, Vol. 38, 1996, pp. 647-655.
- [7] M. Gumbley, 'Procurement and set-up of low cost video conferencing in a student environment', JTAP project, Department of Computer Science, Keele University, Keele, Staffordshire ST5 5BG, UK 1997
- [8] S. Drummond, 'Use of low cost video conferencing to support distributed student groupwork', JTAP project, Department of Computer Science, University of Durham, South Road, Durham DH1 3LE, UK 1997
- [9] R. Young, 'Setting up video taping of video conferencing applications, screen and audio', JTAP project, Department of Computation, UMIST, PO Box 88, Manchester M60 1QD, UK 1997
- [10] CUSeeMe, <http://www.wpine.com> (15th January 1999)
- [11] WhitePineBoard, <http://www.wpine.com> (15th January 1999)
- [12] BSCW, <http://bscw.gmd.de> (15th January 1999)

Table 1. System configuration

Hardware:	Mini-tower floor-mounted PC (P133, 32MB), 17" or 15" monitors, mouse, keyboard, Connectix Colour QuickCam, headphones and/or speakers, small directional microphones or headsets, Wacom A4 graphics tablet.
Software:	Tools for synchronous working: CUSeeMe 3.0 [10], Microsoft NetMeeting 2.1, WhitePineBoard 3.0 (multi-casting) [11], Reflector Software (WhitePine 2.1), Netscape Conference (4.0) Tools for asynchronous working: BSCW (Basic Support For Co-operative Work) (3.1) [12] Lotus Notes / Domino (4.51)

Table 2. Audio

	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
I did not enjoy using the Audio component.	1	3	5	5	2
I would look forward to using Audio in this way in the future.	3	5	2	4	2
Audio is not necessary for Software Engineering students who need to work in this way.	-	2	2	5	7
Audio greatly helped our group complete the task.	4	2	1	4	5
I found Audio very easy to use.	3	5	2	3	3

Table 3. Whiteboard

	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
I did not enjoy using the Whiteboard.	-	1	3	6	6
I would look forward to using the Whiteboard in this way in the future.	4	7	3	2	-
The Whiteboard is not necessary for Software Engineering students who need to work in this way.	1	2	1	5	7
The Whiteboard greatly helped our group complete the task.	5	7	3	-	1
I found the Whiteboard easy to use.	6	9	-	1	-

Table 4. Chat

	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
I did not enjoy using the Chat component.	1	1	-	7	7
I would look forward to using Chat in this way in the future.	5	6	2	2	1
Chat is not necessary for Software Engineering students who need to work in this way.	-	-	2	8	6
Chat greatly helped our group complete the task.	10	3	1	1	1
I found Chat easy to use.	11	4	-	1	-

Table 5. Video

	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
I did not enjoy using the Video component.	1	1	6	6	2
I would look forward to using Video in this way in the future.	4	7	3	1	1
Video is not necessary for Software Engineering students who need to work in this way.	1	5	4	3	3
Video greatly helped our group complete the task.	2	1	4	7	2
I found Video easy to use.	5	7	1	3	-

Table 6. BSCW

	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
I used BSCW frequently	10	4	1	-	1
I invested time and effort learning the system	4	6	5	-	1
I found the interface intuitive	3	8	5	-	-
I feel confident using BSCW	8	6	2	-	-
BSCW supported the type of SE work we have undertaken	6	8	-	2	-
I found the hierarchical structure of the workspace useful	6	10	-	-	-
I found this structure helped our group organise our work	5	9	1	-	1
BSCW was a useful means of communication between group members	5	6	1	3	1
I would be happy to use BSCW again	8	8	-	-	-

Table 7. DTVC vs. face-to-face

	1: not very successful	2	3	4	5	6	7	8	9	10: very successful
DTVC	-	-	7	1	-	2	1	2	2	-
Face to face	-	-	-	-	1	2	3	3	2	5