Working Across Borders: Overcoming Culturally-Based Technology Challenges in Student Global Software Development

Olly Gotel	Vidya Kulkarni	Christelle Scharff	Longchrea Neak
Pace University	*University of Delhi*	*Pace University*	*Institute of Technology*
New York,	*New Delhi, India*	*New York,*	*of Cambodia*
NY, USA		*NY, USA*	*Phnom Penh, Cambodia*
ogotel@pace.edu	*vkulkarni@cs.du.ac.in*	*cscharff@pace.edu*	*longchrea@itc.edu.kh*

Abstract

Facilitated by the Internet, global software development has emerged as a reality. The use of shared processes and appropriate tools is considered crucial to alleviate some of its issues (e.g., space and time differences), homogenizing the environment of development and interaction, and increasing the likelihood of success. Since 2005, Pace University in the United States has been collaborating with the Institute of Technology of Cambodia (ITC) and the University of Delhi in India to bring students together to work on global software development projects. This paper reports on our experiences and lessons from spring 2007 when the focus was on these students working together on the development of a single software system. One key objective was to investigate how to create a shared and open source tooling environment to support a distributed development process that has evolved over two years. The setting is unique in that it seeks to accommodate students from a mix of established, developing and emerging countries who, as a consequence, have had varying levels of exposure to the Internet and use it in non-similar ways. The findings, lessons and recommendations from our study are reported in this paper. Not surprisingly, when the perceived professional value of assumed 'everyday technologies' is dissimilar across cultures, preparation for the communications tooling needs more attention than the engineering tooling. This has important implications for the emphasis placed on 'process' and 'soft skills' in the respective classrooms, and highlights some challenges facing emerging countries as they strive to become players in the global workforce.

1. Introduction

In the present competitive world, students need both technical knowledge and social skills to succeed in the software development workplace [2,3,7,9,10]. This need is driven not only by workforce demands for high-performance employees who can collaborate and communicate to plan and instrument software development projects, but also by the need to help students learn about civic responsibility and master their new roles as global citizens. To achieve such goals, academia has long recognized the importance of teaching through the use of projects that exercise critical skills and require the use of essential tools. Amongst the many benefits of project-based learning are the relationships that are fostered through student teamwork [8,11]. Cultivating such a range of competencies is essential for software engineering education and more so for global software development comprising teams from multiple geographic locations [1,4].

Necessitated by economic and market forces, the last decade has witnessed a growth in the global dispersion of software development work, signifying the emergence of Global Software Development (GSD) as a business necessity. Several factors have

accelerated this trend, such as the lowering costs of telecommunications and the increasing availability of high-bandwidth Internet connections. As a result, software development is increasingly a multi-site, multicultural, globally distributed undertaking, and engineers, managers and executives face numerous, formidable challenges on many levels [3,9,10]. It is imperative that software engineering education and training is contextualized to prepare students to face and address these challenges.

This paper focuses on the efforts of three academic institutions across three countries that have been working together to prepare students for GSD over a period of three years. In 2005, the group project for the software engineering course for undergraduate computer science students at Pace University in the US was adapted to include global teamwork, by collaborating with the Institute of Technology of Cambodia (ITC). We set up five projects so that ITC students acted as customers and Pace students acted as developers, and students experienced a real-life offshore outsourcing experience, albeit with the traditional roles reversed. Students gained a balanced and first-hand view of the advantages, disadvantages and potential of offshore outsourcing in the process [5].

In 2006, Pace University and ITC partnered with the University of Delhi in India to enable the extended global team to subcontract a component of their project to an offshore third-party, thereby capitalizing upon institution and country-specific skills and learn about global supply chain management. Student teams developed three systems targeted to the Cambodian community and students learned how to divide up a project into components for parties to work on across time zones and cultures [6].

More recently in 2007, students from the three countries worked together on a single project with the intention that the system they developed would be deployed into operations within Cambodia. The project ran from January to May, and the website of the project is available at *http://atlantis.seidenberg.pace.edu/wiki/gsd2007*. This project required that the students worked in teams on separate components of a larger project that required integration; the focus was on scaling up and the true need for both local and global integration. This project also incorporated a competitive bidding process for an outsourced component of the work and made use of graduate students as mentors and auditors to help assure the quality of the distributed global project. While the previous two years focused on evolving a suitable process for distributed development and communication across the three locations, this third year focused on increased socialization activities and on creating a shared tool-based environment to support student working. The research question was: "What are the prerequisites and training needs for successful use of tool support, and does this differ across locations?"

Background details about the project are provided in Section 2. Section 3 outlines the tools used to support the various engineering and communication activities. Section 4 presents the findings regarding tool adoption and use. Section 5 presents our lessons and recommendations for others planning to work across diverse cultural borders.

2. Project details

2.1. Collaborators and courses

The students participating in this project were from: ***The Institute of Technology of Cambodia*** (ITC) (*http://www.itc.edu.kh*) -- the software engineering course for fourth year computer science undergraduates; ***Pace University*** (*http://www.pace.edu*) -- the capstone software engineering course for computer science undergraduates and the software reliability and quality assurance course for software development graduates; ***The University of Delhi*** (*http://www.du.ac.in*) -- the database applications course for computer science graduates.

2.2. Project

The internal library of the Department of Computer Science at ITC provides many resources to its students, such as books in English and French, DVDs and e-books. To date, the library management tasks and related transactions have not been computerized; the available resources are managed in an Excel spreadsheet by the departmental secretary (the 'Librarian'). The objective of this project, called MultiLIB, was to develop a multi-purpose web-based library management system and account for specific policies surrounding borrowing. It was to be partitioned in the following way: ***Librarian / Administrator side*** – to focus on that part of MultiLIB to be used for managing the library policies, all the resources, the accounts in the system and all the issue / return transactions; ***Guest / Student / Professor side*** – to focus on that part of MultiLIB to be used to view the information on the available and new resources in the library, to reserve, rate and recommend the resources and to consult accounts; ***Innovation side*** – to focus on that part of MultiLIB to be used by students to view the electronic resources, such as e-books, audio and video. The role of this team was to be forward looking and consider future innovative features of the system.

2.3. Student teams

The extended (or global) team for each of the core sides of the project was composed of US undergraduates and graduates, Cambodian and Indian students. The team composition and their respective roles are illustrated in Table 1.

Table 1. Global and local team structure for the project.

Students	Numbers	Teams and composition	Roles and responsibilities
Cambodian	13	Librarian / administrator side (5); Guest / student / professor side (4); Innovation side (4)	Determine requirements for each side of the system; Review and give feedback on all aspects of US students' work; Test software and submit bug reports; Report on US team; Present work and experience; Demonstrate US software.
US Under-graduate	8	Librarian / administrator side (4); Guest / student / professor side (4)	"Capture" requirements and handle changes; Manage RFP process; Propose design options; Describe and reflect on process and communication protocol; Implement, integrate and test software; Interact with mentors and auditors and integrate feedback; Integrate feedback from Cambodian students; Report on Cambodian and Indian teams; Present work and experience; Demonstrate developed software.
US Graduate	6	US undergraduate mentors (1 for each US team); Integration mentor (1); US undergraduate auditors (2 for each US team)	Mentors: Provide coaching with techniques and practices introduced in classroom; Internal pair of eyes for quality; Raise early concerns to instructor. Auditors: Review and report on deliverables and process; External quality gate keepers; Keep mentors informed of any emergent problems.
Indian	6	3 bidding teams of 2 students	Answer RFP and bid for the database work; Provide US students with database design and SQL code to be integrated into overall system design; Report on US team; Develop software (see Section 2.5).

2.4. Software development process

The project milestones were organized around one week for initialization of the project and team bonding, five weeks for requirements, three weeks for design, three weeks for coding / construction and two weeks for testing. The software engineering process model was a loose waterfall model with iteration and feedback cycles, for instructor control and visibility, and to support teaching. In the requirements phase, requirements were captured using questionnaires and chat discussions. Templates were designed for the requirements, design and testing documents to help the students standardize their work. The students organized themselves with a project leader, a documentation leader, a communications leader and a quality assurance manager.

2.5. Communications

Communications between the Cambodian and US students were initialized during the first week of the class by exchanging Yahoo emails and instant messenger pseudos, and

setting up mailing lists and the dates and times of the first chats. Communications between students in India and the US were initialized at the end of the US requirements phase. This was because the Indian students were acting as third-party suppliers for the database design and required to work from the requirements document. This was to avoid the US students being overwhelmed and confused by two different sets of students with different roles right at the beginning of the project, and to permit them to concentrate on establishing a quality relationship with the Cambodian 'client'. There were no initial communications between Cambodia and India. The intention was just for the Indian students to design and develop the database component of the system in an outsourced manner. However, the Indian students decided they wanted to build the entire software system in parallel to the US, so contacts with Cambodia were established just before the testing phase of the Indian software.

2.6. Socialization activities

Lessons from our previous studies (consistent with the ones from the literature on distributed projects [3,9,10]) highlighted the need to focus attention on activities to help students bond during the course of the project, to improve collaboration and trust amongst team members and engage them in achieving a common goal. At the beginning of the semester, the US and Cambodian students purchased country-, city- or culture-specific gifts for each other. They sent handwritten postcards with country-specific heritage landmarks on them for the other students to research and locate on a country map. They also produced videos about their city, school and themselves for the other students to learn about their environment and goals. Gifts were also exchanged between the US graduates and the Cambodian students, US undergraduates and Indian students, and Cambodian and Indian students. Despite this, there was not enough sustained focus on these relationships, and the emphasis soon became a localized concentration that students considered necessary to get the software implemented.

3. Processes, tools and training

Over the past two years, use of technology was not the focus of our studies. Technologies have been used to support engineering activities, were introduced in the classroom and students were free to use them. During 2007, the instructors focused on encouraging consistency in the use of tooling and in providing more targeted training in the use of communications tools. Given the fact that the environment across the three institutions is varied, not only in terms of technology and Internet access, but also in term of electricity availability, a simple set of open source tools needed to be used.

The engineering tooling converged on Eclipse (with JUnit *http://www.junit.org* and Subversion *http://subversion.tigris.org*) as the development platform, and java.net (*http://java.net*) for bug tracking. Since the students needed to develop a shared system architecture and user interface for the overall project, communication and collaboration tools were to be critical to their efforts; the teams had to share their work products and synchronize their tasks. Communication tooling comprised mailing lists, chats, blogs, wikis and videos, in addition to face-to-face communication. The drivers and requirements behind such a mashup of technologies are explained in Table 2. Use across the three countries is also indicated and categorized as high, medium or low (i.e., H, M, L), as determined by the instructors.

2007 was the first year in which tool training was actively provided for the students and where specific technology use was required. Prior to the project, the US instructors visited Cambodia to meet the students and explain their role. They also introduced the students to the technologies that would be used to support the development and

communication processes, including Eclipse (*http://www.eclipse.org*), blogs and wikis. Additionally, classroom and online tutorials were provided as needed.

Table 2. Project tasks, tooling and use on the project.

Activity	Tool	Rationale and tasks supported	US	Cambodia	India
Requirements	MS Word	Requirements gathering, elicitation and validation.	H	H	N/A
	Chats	Synchronous communication.	H	H	L
	Email	Asynchronous communication.	H	H	H
Design	SmartDraw UML or similar	To model design options and achieve a better understanding of how the system should behave and correspond to client needs; To facilitate communication with database designers in India; To produce an ERD.	H	H	H
Implementation	Eclipse	To take advantage of the IDE features, JUnit and Subversion plugins, and the externalization mechanism.	H	L	L
	Java/JSP	To encourage students to build on their Java / JSP skills.	H - Java	NA	H - JSP
	Apache Tomcat	To use an open source servlet container that would be easily deployable in Cambodia.	H	NA	H
	MySQL	To use an open source DBMS that would be easily deployable in Cambodia.	M	N/A	H
Testing	JUnit	To automatically run unit tests and validate units of the software.	M	N/A	L
	Java.net	To validate software; clients and developers used the Issue Tracker facility of java.net to report, then fix and manage bugs respectively.	H	H	H
Configuration Management	Subversion	To facilitate code sharing, change and version management.	M	N/A	L
Project Management	Wiki	To contain all documents and software artifacts; To increase milestone visibility and awareness; To agree on shared architecture, approve database design, gain feedback on user interface mockups, clarify deployment environment, communicate responsibilities, and manage the integration and system-level testing documentation.	H	L	H
	Blog	To allow problems to be addressed by the teams and instructors.	H	L	M
RFP Process	MS Word	To write RFP and acceptance / rejection letters with justifications.	H	N/A	N/A
	Email	To solicit bids, manage RFP process and select winning design.	M	N/A	M
	Chat	To clarify RFP requirements.	L	N/A	L
Socialization	Video	To get to know each other and put a face on a name.	H	H	L
	Chat	To get to know each other and facilitate spontaneous conversation.	H	H	L
Mentoring	Face-to-face	To provide technical and team management assistance.	H	N/A	N/A
Auditing	Wiki	To review artifacts produced and check compliance.	H	N/A	N/A
	Face-to-face	To conduct interviews.	H	N/A	N/A
Instructor Oversight	Wiki	To monitor progress and deliverables.	H	L	H
	Blog	To address problems and have students elaborate on particular questions asked by instructors for assessment purposes.	H	L	H
	Survey	To gather students' perception on project, about RFP process, mentor / auditor experience and overall experience.	H	L	H

4. Findings

The final version of the requirements specification included thirty-four functional requirements and eleven non-functional requirements. The US students implemented eighteen functional and three non-functional requirements in the delivered software system. In contrast, the Indian students implemented twenty-eight functional requirements and four non-functional requirements. Based on java.net Issue Tracker, thirty-nine issues were submitted by the Cambodian students for the US software system and forty-seven issues were submitted for the Indian software system, half of these issues being defects and half of them being requests for enhancement. Although they preferred the Indian software system because of its user interface and implementation of security features, the Cambodian students rejected both of the software systems that were developed for them -- they mandated that all the requirements be implemented for deployment.

4.1. Use of engineering tools

Technical tools such as UML software, Tomcat, MySQL and java.net had a high level of adoption and were more widely used than the communication tools (see Sections 4.2 and 5). Eclipse, with the Junit and Subversion plugins, was not used extensively by the Indian students to develop their software variant. Eclipse is not yet used in the University of Delhi curriculum, but it is integrated in the Pace University curriculum; it is used in CS1 and CS2 for Java programming with a focus on test-first and in the programming paradigm class to

facilitate experimentation with several languages in the same IDE (e.g., Python, Prolog and SML). Driven by the need to deliver deployable software, students realized the importance of java.net, even though it was a new tool for students to utilize. It eased the communication between the development and testing teams, and allowed issues to be described in detail.

4.2. Use of communication tools

Mailing lists, emails and chats: Overall, six mailing lists were created during the project. Three mailing lists, including undergraduate and graduate US and Cambodian students were created for the three sides of MultiLIB. During the project, a mailing list (including all the US and Indian students) was created to facilitate communication during the RFP process. Another mailing list, including all the US and Cambodian students was also created to support the integration activity. One more mailing list was established including all Indian and Cambodian students, just before the testing phase of the Indian software. Use of these channels of communication peaked during the requirements activities and then slowly declined (see Figure 1). Cambodian and Indian students, going through classes as cohorts, privileged face-to-face meetings to emails. Not all chats were logged – students recorded scheduled chat activities rather than spontaneous ones. Often, when communication was needed the most on the project (e.g., during the RFP process), students in the various countries had exams or national, school or cultural holidays. Planning for these time-outs in classroom settings should not be overlooked.

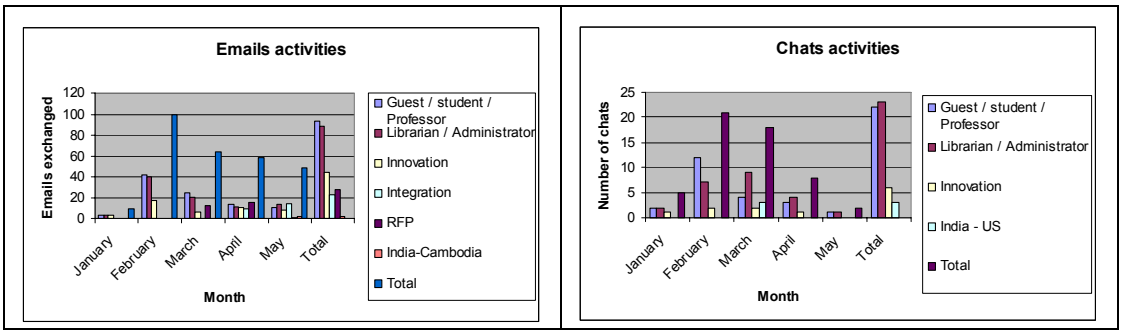

Figure 1. Email and chat activities during the project.

Wikis: A wiki was established and maintained for each side of the project. Each wiki contained all the documents and artifacts that were produced by the sub-team, including a description of the software engineering process followed, the contact information of the global sub-team and its members, the requirements, design and testing material, and PowerPoint and video presentations of the team's work. This material was all editable by the students at the three locations. By the end of the semester, the mantra was *"Update your Wiki morning, noon and evening"* to ensure timely information exchange and accurate project status information. In addition, all the material that was associated with the integration aspects of the project was posted on a separate wiki shared by the three sub-teams. Students at the three locations took advantage of the wikis in dissimilar ways (see Section 5). Wikis allowed easy to access to information as and when needed, and it was a student's responsibility to check the wikis regularly. Despite this, there were problems like when the Indian students worked on an out of date version of the requirements document although a new version had been made available.

This illustrates the importance of redundancy in channels of communication -- the update of the wiki should have been augmented by sending a notification email.

Blogs: The US and Indian students maintained a weekly reflective blog about their progress on the project. However, the Cambodian students did not post their progress to a blog, even though trained on the technology. Indian students, also new to blogging, considered it as a powerful medium. One Indian student mentioned in the survey: "*after this project was assigned to our team I never had the idea that we will be provided with such interaction and freedom to express through blogs and wikis…*"

5. Lessons and recommendations

5.1. Consider wikis as the coordination backbone for student projects

Compared to the previous years, this switch in emphasis to use a wiki and provide wiki training improved the following aspects of the overall project experience:

- It permitted all the students and instructors to get up to speed on the project quickly;
- It empowered students at each location to contribute ideas;
- It increased productivity, since students found wikis easier to maintain than web pages (used in previous years), in turn making updates simpler and the turn-around time to get feedback shorter, whilst avoiding claims of lost documents in email exchanges;
- It provided a way to share the expectations of students at the different locations as to each other's roles and responsibilities, thereby monitoring each other's progress;
- It facilitated quality assurance activities as the US graduates had an up-to-date picture of the global project at any one moment in time, and were able to track its progress; and
- It promoted a better understanding of the software engineering process, practices and tools that were being used in the project.

5.2. Awareness of differences in curriculum emphasis and its consequences

There are different emphases in the computer science curricula of the three countries involved in this study. In the US, process, soft skills and teamwork are focused upon across the curriculum. In India and Cambodia, technical skills are given more importance. Consequently, adoption and use of technical and communication tools is not viewed in the same way. There is a lag in the recognition and acceptance of the value of social and softer skills to computer science – it begins to be recognized in India through offshoring experiences, but Cambodia is a long way behind.

5.3. Address culturally-based technology perceptions

Despite prior training and early use, the Cambodian students were reluctant to use wikis during the development stages of the project -- they had to be reminded to check the wikis and to post their work. By contrast, while new to wikis and in receipt of less prior training, the Indian and US students had no such reluctance, although the Indian students regarded such tools to be more peripheral to those actively used to support technical development. Similarly, blogs were used by the US and Indian students, but not by the Cambodian students. Students from different countries have had varying levels of exposure to the latest Internet-based collaborative working tools and have had different experiences of (and scholastic emphasis placed upon) the kinds of communal working that such tools facilitate. This results in divergent perceptions as to the potential value of learning about and using such tools for professional software development purposes. Training is not sufficient – instructors need to provide evidence of use of these tools in the workplace for students to see their value.

7. Conclusions

Beyond giving students exposure to global working, studies such as this one provide a laboratory to transfer important lessons. Many of the challenges faced on student projects of this nature, such as perceptions as to the value and use of new technologies, are barriers to entry that emerging nations need to overcome. Divergent perceptions as to the potential value of using the latest collaborative communication tools for professional purposes comes from emphasis in the classroom and this study draws attention to the prerequisites required to encourage their use. Our study showed that training on tools is necessary but not sufficient to trigger adoption of the tools by students. It must be coupled with evidence of the use of these tools in the industry for students to appreciate their importance. Additionally, faculty need to align their teaching with the reality of the global software development market, and understand the challenges and skills shortage it faces to address them in the curriculum. In spring 2008, we plan to look to the wider educational context of the three countries and the respective emphases placed on different skill sets in an attempt to expedite this process.

8. Acknowledgements

This work is supported by a National Collegiate Inventors and Innovators Alliance grant (#3465-06), "Incubating the Next Generation of Global Software Development Entrepreneurs" (2006-2008). We thank the US, Cambodian and Indian students involved in this project. We are grateful to Doug Tidwell and Chris Nelson of IBM, and Gary Thompson of Sun Microsystems, for helping to introduce technology into the classrooms.

9. References

[1] Aspray, W., Mayadas, F. and Vardi, M.Y. *"Globalization and Offshoring of Software"*. A Report of the ACM Job Migration Task Force, 2006.

[2] Chen, L., de Souza, C.R.B., Hupfer, S., Patterson, J. and Ross, S. "Building Collaboration into IDEs". *ACM Queue*, 1(9), December/January, 2003-2004.

[3] Coar, K. "The Sun Never Sets on Distributed Development". *ACM Queue*, 1(9), December/January, 2003-2004.

[4] Damian, D., Hadwin, A. and Al-Ani, B. "Instructional Design and Assessment Strategies for Teaching Global Software Development: A Framework". *Proceedings of the 28th International Conference on Software Engineering (ICSE 2006)*, Shanghai, China, May 20-28, 2006.

[5] Gotel, O. Scharff, C. and Seng, S. "Preparing Computer Science Students for Global Software Development". *Proceedings of the 36th IEEE Annual Frontiers in Education Conference. Borders: International, Social and Cultural (FIE 2006),* San Diego, California, USA, October 28-31, 2006.

[6] Gotel, O., Kulkarni, V., Neak, L., Scharff, C. and Seng, S. "Introducing Global Supply Chains into Software Engineering Education". *Proceedings of the 1st International Conference on Software Engineering Approaches For Offshore and Outsourced Development (SEAFOOD 2007)*, Zurich, Switzerland, February 5-6, 2007.

[7] Mawdsley. J. "Tools for Distributed Development". *Dr. Dobbs Journal*, September 26, 2007.

[8] Michaelsen, L.K. "Getting started with team-based learning". In L.K. Michaelsen, A.B. Knight, and L.D. Fink (Eds.), "*Team-based learning: A transformative use of small groups"*. Westport, CT: Praeger, 2004.

[9] Olson, J. S. and Olson, G. M. "Culture Surprises in Remote Software Development Teams". *ACM Queue*, 1(9), December/January, 2003-2004.

[10] Sarker, S., and Sahay, S. "Implications of Space and Time for Distributed Work: An Interpretive Study of US-Norwegian Systems Development Teams". *European Journal of Information Systems*, Vol. 13, No. 1, pp.3-20, 2004.

[11] Williams, J.C., Bair, B., Borstler, J., Lethbridge, T.C., and Surendran, K. "Client sponsored projects in software engineering courses". *Proceedings of the 34th SIGCSE Technical Symposium on Computer Science Education (SIGCSE 2003),* Reno, Nevada, USA, February 19-23, 2003, ACM Press, New York, NY, pp.401-402.