Evolving an Infrastructure for Engineering, Communication, Project Management and Socialization to Facilitate Student Global Software Development Projects

Olly Gotel
Pace University
New York, USA
ogotel@pace.edu

Vidya Kulkarni
University of Delhi
New Delhi, India
vkulkarni@cs.du.ac.in

Des Phal
Royal University of Phom Penh
Phnom Penh, Cambodia
phaldes@gmail.com

Moniphal Say
Institute of Technology of Cambodia
Phnom Penh, Cambodia
say.moniphal@gmail.com

Christelle Scharff
Pace University
New York, USA
cscharff@pace.edu

Thanwadee Sunetnanta
Mahidol University
Bangkok, Thailand
cctth@mahidol.ac.th

ABSTRACT

With the rapid increase in offshore outsourcing of software development, Global Software Development (GSD) has become the need of the hour. This, in turn, necessitates preparing computer science students to face the challenges associated with GSD. Today's information technology, in the form of communication networks and tooling opportunities, provides us with a supposedly ready infrastructure to support GSD. However, selecting an appropriate combination of open source tools that cross cultural boundaries and account for unique in-country connectivity situations is not a trivial task. In this paper, we describe our experience of evolving an infrastructure for student GSD projects over a period four years, culminating in an environment to accommodate the needs of five different teams from four globally dispersed universities in countries straddling many technological divides.

Keywords

Global software development, Software development tooling, Software engineering education.

1. INTRODUCTION

In 2005, faculty from Pace University in New York City and the Institute of Technology of Cambodia (ITC) initiated a collaboration to enable undergraduate computer science students from the two academic institutions to work together on the development of software systems for the Cambodian market [1]. The Cambodian students acted as clients and the US students acted as developers. The motivation was to have the students gain a real-life experience of offshore outsourcing, and so to begin to understand its potential opportunities and challenges. The model of collaboration was extended in 2006 to include the participation of graduate students from a database course at the University of Delhi in India [2]. The focus was now on incorporating global supply chains and right sourcing into the model, thus exposing the students to the management of sub-contracts and dealing with the communication and coordination issues that may arise from this type of setting. The Indian students therefore acted as sub-contractors for the database design component of each student project. In 2007, the scenario was further refined to emphasize scale and deployment, thereby exposing students to integration planning and resource sharing [3, 5]. The model was such that the US developers worked on separate components of a single software system to be eventually integrated for the Cambodian clients. Teams of Indian sub-contractors responded to a competitive Request For Proposal for the database design, one design of which was to be selected and integrated by the US developers. US graduate students also participated as mentors and auditors in an attempt to highlight quality matters [4]. Lessons learned from these three previous years of collaboration influenced the setting of 2008.

The Cambodian client did not accept the final software product of 2007. This was due to the unsatisfactory fulfillment of requirements. In 2008, teams of developers in Cambodia, India, Thailand and the US were therefore put in friendly competition to re-develop the software, capitalizing upon the work to date and increasing the chances of one system being successfully selected for deployment. To address the prior quality issues, each global team was assisted by US graduate students acting as software quality assurance (SQA) coaches and auditors, to control the quality of the process and final product. An additional dimension was the incorporation of socialization activities to permit the students to bond during the project, improving engagement to help better achieve a common goal. The wiki of the project can be found at: http://atlantis.seidenberg.pace.edu/wiki/gsd2008.

This paper outlines the infrastructure that evolved over the four years to bring students with different roles together to work on a GSD project of this nature, accounting for different backgrounds and prior exposures to technologies,. Section 2 provides the context for the 2008 setting of the project. Section 3 justifies the choice of tools and describes those that were used to support the different engineering, communication, project management and socialization activities. Section 4 presents the findings on practical use and impact, and Section 5 presents a list of do's and don'ts when configuring an infrastructure for student GSD projects.

2. CONTEXT

The students who partnered on this project were undergraduate and graduate students in computer science and software design and development from: the Institute of Technology of Cambodia (ITC), the University of Delhi in India, Mahidol University in Thailand, Pace University in the US (NYC and Pleasantville PLV campuses) and the Royal University of Phnom Penh (RUPP) in Cambodia. Table 1 illustrates the roles and responsibilities of the students. Each global development team (five overall) was composed of a Cambodian client and a client coach, the development team and a developer coach and auditors, and was requested to build a version of MultiLIB, a library system for the computer science department of ITC. The requirements comprised 71 functional and 4 non-functional requirements. The software of highest quality, measured in terms of requirements satisfaction, was selected by the client and client coaches and deployed into operations at ITC. The socialization team was composed of the US students of Pleasantville (US PLV) and the RUPP students.

Each team followed a loose waterfall-based software engineering process with iteration and feedback cycles to account for the unfamiliarity of the developers with software engineering practices, to coordinate the work of students distributed in five different locations and to facilitate the SQA activities of the coaches and auditors. The project spanned eighteen weeks – including four weeks for requirements, four weeks for design, six weeks for coding/testing and five weeks for deployment. The software was delivered and tested incrementally by one-week iterations; developers held planning meetings at the beginning of each iteration where they committed to implement a certain number of requirements.

3. TOOLING ENVIRONMENT

The use of a standardized tooling environment has been emphasized in the last two years of our collaboration to improve the coordination of the work of the students across locations. The introduction of the tooling has been supported by the design of tutorials and dedicated sessions with hands-on laboratories. The tooling used on the project can be categorized into engineering, communication, project management and socialization. This section describes the tooling environment and its evolution, and provides justifications for our choices.

Engineering tooling. The use of an Integrated Development Environment (IDE) has been encouraged from the beginning of this initiative in 2005. Therefore, the engineering tooling environment converged to the use of Eclipse/Netbeans/Visual Studio with JUnit/NUnit for unit testing and subversion for code version control, given these IDE were integrated in the curriculum at the various institutions. Students therefore built on their programming skills; the Thai students used Visual Studio as the programming language they had mastered to date and would use on the project was to be C# and ASP .NET, whereas all the other students developed in Java and JSP. The use of subversion was emphasized this year to facilitate integration and incremental code delivery to the coaches and auditors for review and to the client for deployment. Java.net issue tracker was introduced in 2007 and used this year by the developers, coaches and auditors to manage issues and validate the software respectively.

Communication tooling. Communication between the students was carried out using emails and chats. Each student created a Yahoo email and pseudo to communicate during the project. In 2008, nine mailing lists were created – one for each development team, one for the client and client coaches, one for the auditors and coaches, one for the socialization team and one for the instructors. The instructors monitored use of the mailing lists to detect any early impediments on the project. The local development teams often used phone to communicate, but the global teams did not even though they were encouraged to do so; the belief was that videos and voice conversations would be too difficult to set up.

Table 1. Global teams, roles and responsibilities on the project
(UG and G denote undergraduate and graduate students)

Countries, roles and numbers	**Responsibilities**
Cambodian Client (ITC) (5 UG)	Own and manage requirements; Solicit software systems from five development teams; Review and give feedback on all aspects of developers' work; Interact with client coaches and development teams; Test candidate software and select the one of highest quality; Deploy the selected software in Cambodia.
US client coaches (5 G)	One coach for each client. Mentor the client team; Help with the production and maintenance of the requirements document; Manage requirements changes, versioning, change awareness and protocols associated with them.
Cambodian developers (ITC) (4 UG)	Five development teams in friendly competition for the development of the highest quality software. Assess the risks associated with the delivery of the system that satisfies the requirements; Describe the software engineering process to be followed; Propose design options; Iteratively implement, integrate and test software; Reflect on process and communication protocol; Interact with client, mentors and auditors and integrate feedback.
Indian developers (5 G)	
Thai developers (4 UG)	
US NYC developers (5 UG)	
US PLV developers (4 UG)	
US developer coaches (5 G)	One coach for each development team. Mentor the developers; Provide coaching with techniques and practices introduced in classroom; Raise early concerns to developers and instructors; Formulate a test plan and test the system of the sponsored team.
US auditors (16 G) Thai SQA trainees (4 UG)	Three US auditors for each development team; One US SQA manager to oversee and coordinate the audit efforts. Propose an audit plan; Audit and report on deliverables and process to assure and improve quality; Test the system; Interact with developer coaches and developers to raise issues and provide timely feedback. Four Thai SQA trainees observed the quality assurance tasks of the US auditors. The objective of such observation was to establish continuous improvement on testing processes locally.
Cambodian socialization team (RUPP) (2 UG)	Introduce the US PLV development team to many aspects of Cambodian life and culture.

Project management tooling. Shared Google calendars and timezone software were used for sharing milestones and time awareness across locations. The use of wikis as the coordination backbone of the project was introduced in 2007. Wikis were created in place of web pages to provide a standardized environment to permit the contributions of students at each location, to share documents (written and videos) and software artifacts, to increase overall team awareness, assist with SQA activities and facilitate deployment. Eight wikis were created using the same template - one for each development team, one for the client and client coaches, one for the auditors and coaches, and one for the socialization team. Blogs that were used by the students from 2005 to 2007 for weekly reflections were abandoned in 2008 and replaced by posting progress and reflections directly on the wiki.

Socialization tooling. Photos of all the students were posted on the wikis so that students could put a face to a name. During the two first weeks of the project, students were required to meet in chat sessions to get to know each other and establish the relationships that nurture distributed global teams to work efficiently and gain trust. Participation in these chats was not required of the coaches and auditors, a decision that was detrimental to the team bonding in some of the global teams. After the first few weeks of scheduled chats, students were then free to socialize and have informal chats to sustain the relationships. In an attempt to study the impact of socialization in GSD projects, in a controlled manner, the US PLV team was the only team intensively exposed to the Cambodian culture of the client. In previous years, all the students exchanged videos about themselves and gifts, so this aspect could not be studied in detail. Note that this study is not elaborated on in this paper. These students received country- and city- specific gifts and were required to meet every week with the RUPP students in Second Life to discuss predefined topics related to Cambodia. At the end of the project, all students were invited to the Pace Second Life Island for a party with virtual food, drinks and country-specific music and activities to celebrate their achievements. The party was well attended by the US developers, the Thai and RUPP students, and all the instructors.

Choice of tools. The project selected open source and free tooling to facilitate access by the students across the locations. The tools were also chosen for their widespread use in industry, their relative simplicity of use and the functionality they provide to support necessary distributed GSD practices. They were installed and configured by the instructors. Lessons from the three previous years suggested the need to give higher importance to communication, project management and socialization tools,. Past experience also indicated the importance of redundancy in communication channels and so the need for different types of communication tools, including emails, chats and wikis. Difficulties were faced in selecting an appropriate UML design tool due to the complexity of the existing tools in this domain. The only design tool that was recommended to the students was DB designer 4 to propose ERD design options. Java.net was chosen for its ready-to-use subversion and issue tracking facilities. These tools were all installed on each institution's server.

4. FINDINGS

Five versions of MultiLIB were implemented. The Cambodian, US PLV, US NYC, Indian and Thai teams implemented 24, 33, 64, 66 and 68 of the functional requirements respectively. Effective use of the tooling was largely responsible for the achievements of the US NYC, Indian and Thai teams.

Tooling use. Students were new to many of the technologies introduced. Based on the exit survey, the Indian and Thai developers perceived the IDE as the most difficult technology to learn; the US developers had difficulties with the design tools and the Cambodian students found java.net too complex. The developers unanimously designated the IDE as the 'killer' engineering tool though. Yahoo IM was designated the 'killer communication tool for Cambodian and US NYC developers whereas emails were crucial for the other developers. All developers asked for the two crucial tools on the project cited an engineering and communication tool, except the Indian developers who named their IDE and java.net. Subversion and unit testing were used only in a limited manner. Students found it difficult to become familiar with continuous code integration and regression testing practices. The number of revisions in subversion was low, ranging from 13 (Thailand) to 29 (US-NYC) revisions, and the number of developers who committed code ranged from 1 (India, Thailand) to 2 (US) – which is in contradiction with the philosophy of the use of such tools. There is a need for more context and training. Java.net issue tracker was used effectively to submit bugs by the auditors and developer coaches, and to manage bugs by the developers in India, Thailand and US NYC.

Communication problems between the client and the developers arose, not only from the difficulties of access to the Internet of the Cambodian students, but also from the differences in times, semester and vacation misalignment, and class loads. Communications between the US-based developers and auditors was not as effective as it could have been possibly because the team bonding was not created at the beginning of the semester (overlooked since co-located) and because problems with the localization of deliverables on the wiki frustrated some of the auditors. Milestone deliveries were typically not announced on the mailing lists by some of the development teams because the developers assumed shared awareness of calendars. One setup of the wiki that would have been helpful was the notification of any post on the wiki. Additionally, auditors had to maintain the auditors' wiki and the wiki of the developers they audited with some redundancy. The coaches had more communications with the developers and clients due to their mentorship roles.

In the past years, we found that socialization provides a foundation for our GSD projects, but attention to such efforts declines with the pressure of approaching deadlines. The relationship created by establishing a regular meeting of the socialization team in Second Life permitted the US PLV students to feel more comfortable with the client and establish a friendly relationship. However, this did not have the positive impact on local team bonding and quality that was anticipated. An interesting finding for socialization though was the enthusiasm of the participants in the Second Life party at the end of the semester, which positions Second Life as a promising tool for socialization on GSD projects if used at the right junctures.

Table 2 summarizes the extent of the use of the various tools, categorized as High, Medium or Low (H, M, L). Empty cells indicate that the tool was not used because not required for the role or due to the difficulties in getting up to speed with it. Tools used during audit reviews, deployment and the Second Life party, are classified by R, D and P respectively.

Tool perceptions. Quotes from the students' exit survey are provided below to illustrate some of their main concerns associated with the tooling and their respective roles on the project. The quotes are in accordance with the findings above.

- Several Cambodian client students stated that they would have preferred: “more activities and less email replies”.
- A Cambodian developer reported that: “What I liked most on the GSD project is that it improved my experience such as: knowing the development process, database design, class design, how to implement a database in a DBMS, OOP concepts, new technologies (JSP, Servlets, Java beans, Struts, MVC but only the basics) and working in group.”
- The Indian developers mentioned that what they liked the most on the GSD project were: “the new tools that [they] came to know”. They did not have issues in learning the new tools and said that it: “hardly took a day to get the hand of them.”
- The Thai students liked the “project’s wiki” most on the project.
- The US developers affirmed that they “never learn[ed] to use any diagrams such as class diagrams, ERD, activity and sequence diagrams. Diagrams make programming much easier and allow you for a clear view of what you need to implement.”
- The coaches and auditors had converging opinions and mentioned that there were: “Too many places to check for communication. It became hard to get oriented with the whole environment” and “to find needed information. Sometimes things slipped by because of this.” They also added that “Emails and the wiki provided a massive confusing place to communicate and it was hard to have traceability in [their] communication as a result”.

Table 2. Tooling, roles and use on the project

(H. M and L = high, medium or low tool use; R = tools used during audit reviews, D = tools used during deployment and P = Second Life Party)

Tool	Client	Client coach	Cambodia	India	Thailand	US NYC	US PLV	Developer coach	Auditors	Socialization team
MS Word	H	H R	H	H	H	H	H	H	H R	
UML/DB Design			H	H	H	H	M	R	R	
IDE	D		M	H	H	H	H	R	R	
MySQL	D		H	H	H	H	H	R	R	
Java/JSP/C#	D		M	H	H	H	H	R	R	
Subversion	D			M	M	M	M	R	R	
JUnit/Nunit				L	L	L		R	R	
Java.net				H	H	H	L	M	M	
Emails	M	H	M	M	H	H	H	M	M	L
Chats	M	H	M	M	H	H	M	M	L	M
Wikis	H	H	M	H	H	H	H	H	H	L
Demo video			H	H	H	H	H			
Second Life				L (P)	H (P)	M	H			H

5. DO’S AND DON’TS OF TOOLING

Do	Don’t
Assess the experience and level of exposure of the students to the Internet, as well as to engineering, communication, project management and socialization tooling, and account for the learning curve in the schedule.	Underestimate the importance of communication, project management and socialization tooling by only giving attention to engineering tools; socialization has to have a central place and be sustained throughout the project.
Explain the rationale behind the choice of each tool and setup to students.	Discard Internet access difficulties at different locations and the different perceptions on the usefulness of various tools.
Create a shared consensual tooling environment across locations to reflect what is used in industry and accommodate the mix of roles, cultures, and levels of prior exposure to the Internet and technologies.	Overestimate the extent of the use of the tooling by the students; they use less than 50% of their functionality and have to be guided along the way to use them appropriately.
Provide customized training prior to using the tools, either online or/and on-site – complemented by tutorials and access to resources.	Rely on too many tools and think that tools will solve all the encountered problems; people and processes are essential behind the tools.
Get multiple and redundant communication channels privileging one-to-many rather than one-to-one communications. Wikis work best in conjunction with short status meeting via chats or conference calls.	Rely on having students install the tooling; the tooling environment must be ready to be used by the students from day one, and set up by the instructors or preferably the IT department.

6. ACKNOWLEDGEMENTS

This work is supported by a National Collegiate Inventors and Innovators Alliance grant (#3465-06), “Incubating the Next Generation of GSD Entrepreneurs” and a Campus Second Life scholarship. We thank all 159 students who have been involved in this global initiative to date.

7. REFERENCES

[1] Gotel, O., Scharff, C. and Seng, S. “Preparing Computer Science Students for Global Software Development”. In *Proc. IEEE Annual Frontiers in Education Conf. Borders: Intl, Social & Cultural (FIE 2006),* San Diego, California, USA, 2006.

[2] Gotel, O., Kulkarni, V., Neak, L., Scharff, C. and Seng, S. “Introducing Global Supply Chains into Software Engineering Education”. In *Proc. Intl. Conf. on Software Engineering Approaches For Offshore and Outsourced Development (SEAFOOD 2007)*, Zurich, Switzerland, 2007.

[3] Gotel, O., Kulkarni, V., Neak, L. and Scharff, C. “Working Across Borders: Overcoming Culturally-Based Technology Challenges in Student Global Software Development”. In *Proc. Conf. on Software Engineering Education and Training (CSEE&T 2008)*, Charleston, South Carolina, USA, 2008.

[4] Gotel, O., Kulkarni, V., Neak, L. and Scharff, C. "Students as Partners and Students as Mentors: An Educational Model for Quality Assurance in Global Software Development". In *Proc. Conf. on Software Engineering Approaches For Offshore and Outsourced Development (SEAFOOD 2008)*, Zurich, Switzerland, 2008.

[5] Gotel, O., Kulkarni, V., Neak, L. and Scharff, C. "Integration Starts on Day One in Global Software Development Projects". In *Proc.of Int. Conf. on Global Software Engineering (ICGSE 2008)*, Bangalore, India, 2008.