TRAINING GLOBAL SOFTWARE DEVELOPMENT SKILLS THROUGH A SIMULATED ENVIRONMENT

Miguel J. Monasor
University of Castilla-la Mancha, Campus Universitario s/n, 02071, Albacete, Spain

Aurora Vizcaíno, Mario Piattini
Alarcos Research Group, Institute of Information Technologies & Systems, Escuela Superior de Informática
University of Castilla-la Mancha, Paseo de la Universidad 4, 13071, Ciudad Real, Spain

Keywords: Global Software Development, Distributed Software Development, Engineering Education, Educational Environment, Teaching Model, Simulators, Virtual Agents.

Abstract: Training and Education in Global Software Development (GSD) is a challenge that has recently emerged for companies and universities, which entails tackling certain drawbacks caused by the distance, such as communication problems, cultural and language differences or the inappropriate use of groupware tools. We have carried out a Systematic Literature Review in relation to the teaching of GSD which has proved that educators should provide learners with a wide set of realistic and practical experiences, since the skills required are best learned by doing. However, this is difficult as companies are not willing to incorporate students in their projects. In this paper we present an alternative: an environment that will simulate typical GSD problems and will allow students and practitioners to develop skills by interacting with virtual agents from different cultures, thus avoiding the risks of involving non-qualified people in real settings.

1 INTRODUCTION

Global Software Development (GSD) allows development teams to be geographically distributed whilst collaborating on the same projects with the main objective of decreasing costs by finding zones in which a skilled workforce is more readily available (Herbsleb, 2007).

However, this shift entails a number of problems, caused mainly by distance, and time and cultural differences. The language barrier is one of the most significant problems (Toyoda et al., 2009) since communications usually use English as a lingua franca, and the implication of non-native speakers can entail the of use terms, expressions or gestures that may be misinterpreted.

These problems affect the way in which the stages of the software life cycle are carried out, signifying that students and practitioners must be trained in communication protocols, and must develop new skills, such as:

- Thinking about problems from the perspective of the other side (Toyoda et al., 2009).
- Communication protocols and the use of computer-mediated communication.
- Oral and written communication with a multidisciplinary team through a common terminology and language (Toyoda et al., 2009).
- Codes of ethics, leadership and time management.

Companies frequently complain that people who have recently graduated lack the necessary experience since they have rarely been involved in distributed projects. It is therefore necessary to provide learners with real experiences in order to develop both the technical and non-technical skills required in GSD (Swigger et al., 2009).

Instructors do not always have the appropriate tools and methods with which to teach these skills.

In this poster we present a training environment that will minimize the instructors' effort, and will allow learners to be placed in a simulated GSD environment in which they will interact with Virtual Agents (VAs) which can simulate different cultures at any time avoiding the need to coordinate with real members.

2 RELATED WORK

The results of a Systematic Literature Review on the field of GSD education has provided studies, such as that of (Swigger et al., 2009), that present the joint efforts made by universities from different countries to involve students in the development of real projects by using asynchronous tools to communicate with each other. Most of these studies attempt to simulate typical problems found in industry.

We also found teaching environments such as that of iBistro (Braun et al., 2002) which allow informal communication problems to be addressed by exposing students to GSD issues when conducting informal meetings.

There are also blended learning environments such as CURE (Schümmer et al., 2005), based on the use of virtual rooms which may contain pages, communication channels (such as chat, threaded mailbox), and users. Users in the same room can communicate by using synchronous communication channels. (Toyoda et al., 2009) developed an e-Learning training system oriented towards teaching project management activities, along with an application that helps in the teaching of methods with which to solve problems.

We also found collaboration platforms such as Jazz (Meneely and Williams, 2009), that train students by using version control, chat, and work item features.

Problems caused by using English as a lingua franca and cultural differences are also addressed (Favela and Peña-Mora, 2001) and direct contact and interaction are seen as a positive experience to achieve a better understanding and comprehension.

However, these proposals imply a high work load for the instructors. Involving companies or coordinating distant universities is not always possible and requires a great deal of coordination. A typical difficulty is that of managing to reproduce the complexity of real environments, along with cultural and language differences. Even in the best of scenarios, we can always find schedule problems, interaction problems and a lack of student motivation, since they are rarely provided with immediate feedbacks.

3 GSD SIMULATOR

Our proposal is based on the Scenario-Based Learning (SBL) approach which immerses students in a story in which they have to respond certain questions in order to influence the outcome of that story. SBL systems prompt the student to choose a response between several options, and this choice will influence the execution of the scenario. Our simulator uses VAs, which by interacting with learners will prompt them to textually response questions and take certain actions into the system, and these actions will determine the following steps in the scenario according to a predefined scenario workflow.

3.1 Virtual Agents

In our simulator, VAs behave like people from different nationalities with different appearances, cultures and gestures, in order to train learners to confront cultural and language differences through the use of textual dialogues.

The VAs communicate with students using text and text-to-speech capabilities by using a chatbot system, displaying emotions according to their personalities and increasing the students' motivation by appearing to care about their actions.

Each scenario can contain one or more VAs that will play any role in the GSD problem (e.g. customer, requirements analyst, developer, project manager, etc.). The scenario also has a Virtual Colleague (VC) or team mate; a special VA that helps learners to cope with the scenario by guiding them, correcting their actions and explaining their consequences.

3.2 Architecture

The system is based on a client-server architecture and provide separated interfaces for instructors and learners.

The instructors' interface allows them to manage learners, organize teams, assign tasks and examine learners' actions and send notices and emails to individuals or groups. Our aim consists of minimizing the instructors' effort by automating their tasks and providing a wide set of training scenarios. This interface allows existing training scenarios to be edited and new ones to be created, along with defining new VAs with specific cultures and personalities.

The learners' interface provide a set of features that allows communicate with instructors, organize their tasks and simulated meetings, access and execute training scenarios, submit deliverables at prescribed milestones, accessing documents and answering tests.

The server side contains the GSD scenarios, with the information required for their execution, including cultural and language knowledge.

The cultural knowledge base is based on the existing literature of Hall (Hall, 1976) and Hosfstede (Hofstede and Hofstede, 2005) considering the use of titles, presentations and greetings, starting and finishing conversations, motivation and rewards, requests, negotiations, conflicts resolution, etc.

The language knowledge base contains the rules for all the language pairs that the VC will use to correct learners' mistakes, such as those that take place when English is used as a lingua franca: the incorrect use of "false friends", incorrect plural formations, avoidance of passive forms, the absence of the third person, the use of redundant prepositions, the overuse of certain verbs, etc.

3.3 Scenarios

A training scenario consists of a VC and one or more VAs, a set of documents and tests associated to the problem to solve and a *Meeting Workflow* that will guide the interaction with the student.

When the learner starts a virtual meeting, the VC will present him/her the scenario and the VAs involved. Learners will take a proactive role in the meeting and the VC will guide them and will detect cultural problems and the inappropriate use of language, so the learners will learn from their mistakes and learn to solve them. The scenario finalizes when the learner complete all the documents associated to the scenario and fill in a questionnaire that can be automatically or semi-automatically evaluated according to a template.

A scenario is defined by a set of phases that define a small part of the conversation. Each phase has a concrete *conversational knowledge* and also a context specific *language* and *cultural knowledge* which are used in that context for that scenario.

These phases are arranged setting up the *Scenario Workflow*. We also employ decision points in which the student will influence the execution path of the scenario based on their textual responses or actions. The phases can also store information about its priority, and this serves to evaluate the learners' actions and the correctness of their decisions.

Using the scenario definition, the VAs and the VC guide the virtual meeting following a logical sequence according to the learners' actions. This design of the scenarios makes it possible to simulate profound and insightful conversations, avoiding speech repetitions and out of context interventions.

At the same time, this model helps to reduce the loss of time on incorrect phases, prevents the learner from returning to a previous conversation, removes unnecessary complexity in the scenario design and helps to structure the cultural and language problems based on the context of the conversation.

The *Meeting Workflow* can be created and edited from the instructors' interface. For each phase, the instructors can introduce the conversational knowledge, documents associated and the cultural and language knowledge.

Phases of the workflow can also be simple or composed what means that can contain other workflows with the aim of organizing the conversation with a high granularity level.

4 TEACHING REQUIREMENTS ELICITATION

We have focused our first scenario on the Global Requirements Elicitation (GRE) stage, since it is a highly communicative process, particularly affected by poor communication, and cultural and time differences.

In our proposed scenario, Spanish learners, playing the role of analysts, will have to communicate with a virtual customer from United States using English in order to elicit a set of functional and non-functional requirements and develop a specification document of the software. For this purpose, we have prepared our scenario's culture and language knowledge bases to include the typical problems or mistakes for that cultures and languages.

Figure 1 shows a part of a *Meeting Workflow* for our design, in which the VC will guide the learner in the context of the meeting and the Virtual Customer will answer the questions according to the associated conversational knowledge.

During the interview, the learner will determine the next phase in the workflow by chatting to the VAs. Since real GRE interviews usually require more than one meeting, the scenario can also store several meeting's workflows dealing with the same subject in a different phase of the project.

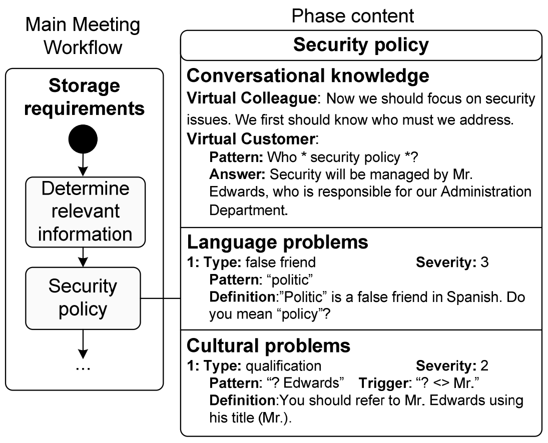
Main Meeting Workflow
Phase content
Storage requirements
Determine relevant information
Security policy
…
Security policy
Conversational knowledge
Virtual Colleague: Now we should focus on security issues. We first should know who must we address.
Virtual Customer:
Pattern: Who * security policy *?
Answer: Security will be managed by Mr. Edwards, who is responsible for our Administration Department.
Language problems
1: Type: false friend Severity: 3
Pattern: "politic"
Definition:"Politic" is a false friend in Spanish. Do you mean "policy"?
Cultural problems
1: Type: qualification Severity: 2
Pattern: "? Edwards" Trigger: "? <> Mr."
Definition:You should refer to Mr. Edwards using his title (Mr.).

Figure 2: Example of GRE meeting workflow.

After the meeting, learners will complete a requirements document in which they will classify all the requirements detected and which, will be automatically evaluated to detect faults such as: ambiguous, incorrect and unspecified requirements.

5 CONCLUSIONS

In this work we present a simulator for training students and practitioners in GSD activities. The usage of VAs implies that the training can be carried out at any moment, thus avoiding the problems that occur in real projects and reducing the training costs.

This simulator helps learners to develop the informal communication skills needed in GSD by putting theory into practice by playing different roles in various scenarios. Although we have focused our initial efforts on requirements elicitation meetings, in our future work we shall consider studying other stages of GSD (e.g. software design, software construction, software testing, etc.) in which other types of meetings might take place.

We plan to validate our environment in a course on Software Engineering with students. Our aim in the future is to gather a wide set of training scenarios in order to obtain a complete and autonomous training environment that will require minimum effort and knowledge on the part of the instructor.

ACKNOWLEDGEMENTS

This work has been funded by the PEGASO/MAGO project (Ministerio de Ciencia e Innovación MICINN and Fondo Europeo de Desarrollo Regional FEDER, TIN2009-13718-C02-01). It is also supported by the ENGLOBAS (PII2I09-0147-8235) and MELISA (PAC08-0142-3315) projects, Junta de Comunidades de Castilla-La Mancha, Consejería de Educación y Ciencia, in Spain, and also by the FABRUM project (PPT-430000-2008-063), Ministerio de Ciencia e Innovación, in Spain.

REFERENCES

Braun, A., Dutoit, A. H., Harrer, A. G. & Brüge, B. 2002. iBistro: A Learning Environment for Knowledge Construction in Distributed Software Engineering Courses. *Proceedings of the Ninth Asia-Pacific Software Engineering Conference.* IEEE Computer Society.

Favela, J. & Peña-Mora, F. 2001. An Experience in Collaborative Software Engineering Education. *IEEE Softw.,* 18**,** 47-53.

Hall, E. T. 1976. *Beyond Culture*, Anchor Press.

Herbsleb, J. D. 2007. Global Software Engineering: The Future of Socio-technical Coordination. *2007 Future of Software Engineering.* IEEE Computer Society.

Hofstede, G. & Hofstede, G. J. 2005. *Cultures and organizations: software of the mind,* New York, NY, USA.

Meneely, A. & Williams, L. 2009. On preparing students for distributed software development with a synchronous, collaborative development platform. *Proceedings of the 40th ACM technical symposium on Computer science education.* Chattanooga, TN, U.S.A.

Schümmer, T., Lukosch, S. & Haake, J. M. 2005. Teaching distributed software development with the project method. *Proceedings of th 2005 conference on Computer support for collaborative learning: learning 2005: the next 10 years!* Taipei, Taiwan: International Society of the Learning Sciences.

Swigger, K., Aplaslan, F. N., Lopez, V., Brazile, R., Dafoulas, G. & Serce, F. C. 2009. Structural factors that affect global software development learning team performance. *Proceedings of the special interest group on management information system's 47th annual conference on Computer personnel research.* Limerick, Ireland: ACM.

Toyoda, S., Miura, M. & Kunifuji, S. 2009. A Case Study on Project-Management Training-Support Tools for Japanese/Chinese/Indian Offshore Development Engineers. *In:* BERLIN, S. (ed.) *Knowledge-Based Intelligent Information and Engineering Systems.* Heidelberg.