Poster presented at ICCE2007 (15th International Conference on Computers in Education)

A Survey on Software Development Support Based on Collaborative Learning Theories

Atsuo HAZEYAMA[a], Hiroyuki AOKI[a,b], Yachiyo ISHIKAWA[a], Hiroyuki ITO[a], Katsunori OGANE[a], Jun OKAZAKI[a], Yusuke KOBAYASHI[a], Masato MIURA[a], Ichiro YAMASHITA[a]

[a] Graduate School of Education, Tokyo Gakugei University, Japan

[b] Department of Computer Science Education, Korea University, Korea

hazeyama@u-gakugei.ac.jp

ABSTRACT

Lots of theories underlying collaborative learning have been proposed. Software development is in nature collaborative activities. Based on the background, studies on software development support and/or analysis of collaborative software development based on collaborative learning theories have emerged. This paper briefly introduces some studies which adopt the theories. We analyze the state-of-the-art of the field.

Keywords

Survey, software development support, collaborative learning theories

INTRODUCTION

With the advance of information and communication technologies, studies on computer supported collaborative learning have been paid attention. Lots of theories underlying collaborative learning have been proposed. Vygotsky considered that a person learns by interacting with others. Situated learning also focuses on aspects of collaboration among learners [1]. Inaba and Toyoda described overview of collaborative learning theories and introduced some advanced learning environments and research issues [2]. We focus on software development. Software development is knowledge intensive and collaborative in nature [3]. This paper gives survey on studies that associated collaborative aspects in software development with learning theories. We will construct a bridge between both fields.

1. OVERVIEW OF STUDIES

We searched for papers from major international conferences and workshops on software engineering (such as ICSE: International Conference on Software Engineering), computer-supported collaborative learning (CSCL), and computer-supported cooperative work (such as ACM CSCW: Computer-Supported Cooperative Work)/GROUP) in this study.

As the result, we collected five papers at the time of July 2007. Two papers are based on Cognitive Apprenticeship [4], two on Legitimate Peripheral Participation (LPP) [1] and one on Distributed Cognition [5]. We describe overview of each study in the following.

1.1 Studies Based on Cognitive Apprenticeship

1.1.1 Instructional model for project-based software engineering course

Huang et al. proposed an instructional model based on cognitive apprenticeship for a project-based software engineering course [6]. They classified six teaching methods in cognitive apprenticeship into activities conducted by student teams and those by instructors, and showed their interactions. When students encounter some problems in conducting a project, the instructor presents a model to the students as scaffolding. The students learn by imitating the presented model and through practice. The instructor also gives them hints, advice, and/or feedback as coaching. The students make reflection, exploration and/or articulation by responding to the actions from the instructor. The instructor gradually fade out by observing students' progress.

1.1.2 An object-oriented modeling learning environment

Tholander and Karlgren developed an object-oriented modeling learning environment based on two learning theories, cognitive apprenticeship and situated learning [7]. Based on the analysis of behavior of problem resolution by expert modelers, they extracted cognitive elements for expert and designed the following tools:

Problem solving track by experts

Tholander and Karlgren video-recorded experienced modelers when they solved a complex conceptual modeling task. The recordings were used in two ways: to create tracks illustrating how experts solved the problem and to extract critical points in the problem solving process. The purpose of the tool is to present the authentic practices by experts and their languages to solve the problem. This tool corresponds to modeling and scaffolding in cognitive apprenticeship.

Pattern library

The tool presents the analysis pattern in a visualized form so that learners can create abstract models by languages experts use. It corresponds to modeling that novices imitate behavior of experts.

Assistance for reflection support

To facilitate reflection or meta-cognitive thinking, an assistant agent gives critical comments, advices,

and/or questions with respect to a task in progress.

1.2 Studies Based on Legitimate Peripheral Participation

1.2.1 Seeking the Source: Software Source Code as a Social and Technical Artifact

de Souza et al. developed a visualization tool, Augur that mines a software repository and provides views of artifacts and activities in a simultaneous way in order to investigate the relationships between artifacts and activities in open source software (OSS) development [8]. As the result, they found a developer moved from a peripheral role to full participation.

Augur provided views with respect to relationships among developers characterized by dependency relationships among code modules. de Souza et al. told that movement from peripheral to central was observed by inspecting the same OSS project at two different time points, and movement was found by identifying a developer who called the code other developers implemented and then whose codes were called by other developers.

1.2.2 Mining Version Histories to Verify the Learning Process of Legitimate Peripheral Participants

Huang et al. adopted LPP as a representation model for describing interactions of OSS development process from the revision history [9]. They provided a quantitative indicator to evaluate role transition of developers in OSS development. From revision histories, they constructed social network graphs that represented the relations between developers of a project. Then, they computed the distance centrality of each node (node represents developer). The more degree of the measure, the more important the role in the project. They could split project developers into two groups: core and peripheral in some OSS projects.

1.3 Studies Based on Distributed Cognition

Software developers are required to utilize various external information in their work. Ye regarded software development as a kind of distributed cognitive processes and constructed support tools that supported two distributed cognitive processes proposed by Hollan et al [3].

Ye clarified cognitive problems when two distributed cognitive processes are used in knowledge acquisition in software engineering: (1) knowledge on the existence of useful external information, on retrieval of the right external information, on application of the new information and on different information needs, and (2) the social problem. To solve the problems, Ye developed two tools, CodeBroker and STeP_IN. CodeBroker gives awareness to a Java developer with respect to reusable components that are relevant to his/her task but (s)he do not know. STeP_IN provides the social support for understanding the reusable component of interest. For that purpose, STeP_IN provided the following three functions: finding example programs, browsing the archives of past discussions, and sending questions to selected experts.

2. ANALYSIS

We identified three types of approaches, i.e., educational support based on collaborative learning theories, construction of a software engineering environment based on the theories, and identification of phenomena which was suggested by a collaborative learning theory from the analytical result of the OSS development process. We also found application of three theories, i.e., Cognitive Apprenticeship, Legitimate Peripheral Participation, and Distributed Cognition were adopted out of five case studies.

3. SUMMARY

We gave a survey of studies on software development support and analysis of collaborative software development based on collaborative learning theories. As we collected only five studies in this paper, we continue to collect papers and would like to construct a framework of software development support based on collaborative learning theories.

REFERENCES

[1] Lave, J., and Wenger, E. (1991) Situated Learning - Legitimate Peripheral Participation, Cambridge University Press.

[2] Inaba, A., and Toyoda, J. (1999) Underlying Learning Theories and Recent Research on CSCL, Journal of Japanese Society for Information and Systems in Education, 16, 3, 166–175 (In Japanese).

[3] Ye, Y. (2005) Socio-Technical Support for Knowledge Collaboration in Software Development Tools, Proceedings of Workshop on Software Engineering and Usability Engineering.

[4] Collins, A., Brown, J. S., and Holum, A. (1991) Cognitive apprenticeship: Making thinking visible. American Educator, 6-46.

[5] Hollan, J., Hutchins, E., and Kirsh, D. (2000) Distributed cognition: toward a new foundation for human-computer interaction research, ACM Transactions on Computer-Human Interaction, 7, 2, ACM Press, 174-196.

[6] Huang, S. T., Cho, Y. P., and Lin, Y. J. (2005) ADDIE Instruction Design and Cognitive Apprenticeship for Project-based Software Education in MIS, Proceedings of APSEC2005, IEEE CS Press, 652-662.

[7] Tholander, J., and Karlgren, K. (2002) Support for Cognitive Apprenticeship in Object-Oriented Model Construction, Proceedings of CSCL2002.

[8] De Souza, C., Froehlich, J., and Dourish, P. (2005) Seeking the source: software source code as a social and technical artifact, Proceedings of ACM GROUP2005, ACM Press, 197-206.

[9] Huang, S.-K. and Liu, K.-M. (2005) Mining version histories to verify the learning process of Legitimate Peripheral Participants, Proceedings of MSR2005, ACM Press, 84-88.