Automatically Identifying that Distributed Programmers are Stuck

Jason Carter and Prasun Dewan
Department of Computer Science, University of North Carolina, Chapel Hill
{carter_jl,dewan} @ cs.unc.edu

Abstract

We hypothesize that it is useful and possible to automatically identify that distributed programmers are stuck by extending existing software development environments using a general architecture.

.

1. Motivation

Often programmers get "stuck" while coding, unable to make much progress despite all efforts to address some issue. It would be useful if an interested remote party could become aware of this situation. For example, instructors could use this information to (a) offer help to student programmers who are too shy to ask for it, (b) determine how much progress they are making, and (c) identify difficult problems.

An educational setting provides particularly compelling applications of this idea because an important goal is to help students and monitor their progress. In fact, the true benefits of this idea could actually occur in industry. A manager of a team could use this information to identify problematic software components and better estimate completion times. Even more interesting, based on recent research, it is possible to argue that this information could significantly improve programmer productivity.

In a study comparing co-located and distributed software development, Herbsleb found [2] that the productivity of co-located teams was significantly higher than that of distributed teams primarily because co-located developers were more apt to help each other finish their tasks. A related study by Teasley et al [4] found that the productivity of a team located in a single "war-room" was much higher than that of one spread out in different cubicles. A major reason, was that if someone was having difficulty with some aspect of code, another developer in the war-room "walking by seeing the activity over their shoulders, would stop to provide help" [4]. Thus, the two studies above show that the greater the distance between developers, the more difficult it is to determine if they need help.

One approach to help distributed teams is described in [1]. Developers use a programming environment that allows them to be aware of the methods on which their team members are working. They could use this information together with project and user-specific information to determine if some team member is stuck.

It would be much more attractive if this deduction could be made automatically by logging developers' interaction with the system. An important step in this direction is made in [3], which describes a logging-based tool for monitoring student progress. Student teams use a wiki to interact with several tools including CVS, newsgroups, and a metrics module that analyzes students' data. The wiki allows students to track their development tasks and analyzes tasks such as file modifications to measure the workload of teams. We are extending this idea to automatically determine when distributed programmers are stuck.

2. Acknowledgements

This research was funded in part by NSF grants IIS 0312328, IIS 0712794, IIS-0810861, and HRD-0450099 UNC-Chapel Hill AGEP Program.

3. References

[1] Hedge R. and Dewan P. *Connecting Programming Environments to Support Ad-Hoc Collaboration* in 23rd IEEE/ACM International Conference on ASE. 2008.

[2] Herbsleb, J.D., et al. *Distance, dependencies, and delay in a global collaboration*. *in Proc. CSCW* 2000**.**

[3] Liu, Y. and Stroulia, E., *A Lightweight Project-Management Environment for Small Novice Teams*, *in Proc.* of 3rd International Workshop on Adoption-Centric Software Engineering, 2003, pp. 42-48.

[4] Teasley, S., et al. *How does radical collocation help a team succeed?* in *Proc. CSCW* 2000.