SCORE Project: Awareness Tool for Distributed Software Team

INTRODUCTION

Geographically-distributed software development teams are increasingly common, in both industrial and open-source contexts. Some of the difficulties faced by distributed team members is maintaining awareness of the evolution of the software design and implementation, work items and bugs, and teammates' activities. In collocated work environments this information is often supplied by intentional conversations, peripheral awareness of others' conversations, and opportunistic discussions. The absence of these mechanisms leads to impeded knowledge flow and thus difficulties in coordination.

The purpose of this project is to attempt to use technology to partially fill this knowledge flow gap for distributed team. This might be addressed by a "developer dashboard", an information display that a developer could use to quickly get an idea of the recent changes to the software, work items, coworker activities, etc. The display might take the form of a web page, a dedicated screen, a sidebar gadget, etc. Ideally the most important information would be available at a glance, whereas the developer could investigate deeper into the details by interacting with the system.

APPLICATION DOMAIN AND SCENARIOS

Consider a team of 4-6 developers working closely together on a project, spread around the globe. The developers might be the entire software development staff of a small company, a team working on a subsystem of in a larger project in a medium- or large-sized company, or volunteer contributors to an open-source project. However it should be considered that each developer considers this project to be his or her primary work activity, making daily contributions to the evolution of the software.

A developer on the team arrives at work and logs into his main work machine. He quickly checks the dashboard to see what has been happening with the project. He sees that another dev has checked out some code that he was planning to refactor. He checks the detail of the change and finds that his ideas for refactoring have been partially implemented by the other dev. He sends an email to the other dev proposing his refactoring ideas. He also notices that new bugs have been filed against the subsystem he is currently working on, and examines the bugs to see how they relate to his current work and plans. Additionally he takes brief note of the areas and work items that his other teammates are working on.

PROJECT GOALS (REQUIREMENTS)

1. Once configured, the system should work with little intervention.
2. The system must be reliable.
3. The system may optionally use dedicated hardware (server, displays, etc.) as needed.

THE INTENDED OUTPUT OF THE PROCESS

1. The team may use any development process that they like, though must provide an initial plan identifying how they will approach the project.
2. An agile process is preferred, but a principled waterfall approach is also acceptable.
3. High-reliability is a requirement, so testing must be an integral part of the development
4. It is important that the UI be relevant to the customer, so the team should be principled about frequent informal UI testing and check-point meetings with the customer.

TOOLS AND STANDARDS

1. The team uses Microsoft Visual Studio Team System 2008 to manage their source code repository, their work items, and their bugs.

INTERACTION BETWEEN STAKEHOLDER AND DEVELOPING TEAMS

1. The team will provide monthly updates to the stakeholders.
2. The team and the stakeholders will maintain an open channel of email communication on an as-needed basis. The contact for the project is ginav@microsoft.com.
3. Microsoft will provide each selected team with a copy of Visual Studio Team System 2008 for their use on this project.
4. At most, two teams total will be selected to work on this project and/or its companion, "Design Rationale Investigation Tool."