Interactive Web-based Applications

Enforcing Communication and Cooperation in Distributed Teams

Alessandro Soro asoro@crs4.it
Manuela Angioni angioni@crs4.it
Davide Carboni dcarboni@crs4.it

CRS4 Center for Advanced Studies, Research and Development in Sardinia
Parco Scientifico e Tecnologico, POLARIS, Ed. 1, 09010 PULA (CA - Italy)

1. From Document Sharing to Real Time Collaboration

Software engineering requires tools and applications to share data and information. Configuration management systems, integrated development environments, frameworks for automatic building and testing, mailing lists, wikis, are examples of tools that support group-work and collaboration[8]. Agile methodologies put a strong emphasis on the adoption of such tools as fast feedback and fast reaction to change are key factors in their philosophy. This is particularly true in a dispersed development scenario, in which teams (or team members) are not co-located, and must communicate indirectly, usually by means of the Internet; (offshore) outsourcing or open source development projects are typical examples of such scenarios. But the need for collaboration is not limited to document sharing. Instant messaging[1] and chat applications[2] allow synchronous communication and are widely used for online meetings, other tools and addons for popular applications allow cooperation in various forms, from group surfing on the Web, to remote desktop.

In this paper we show the work ongoing at CRS4, on the topic of collaboration tools. In par 2 we describe DJ-Lab, a plugin for the popular integrated development environment IntelliJ Idea that supports the practice of remote pair programming; par. 3 describes XP4IDE, that automates the activity of tracking of XP managed development projects and integrates in the IDE a view of the project tasks; in par 4 is presented WebRogue, an application for virtual presence in Web sites, that allows web users to see the other people connected to a web server and communicate and cooperate in various ways. Finally in par 5 the philosophies of these applications are analyzed to spot analogies and difference, potential evolutions, technical and human limitations, and track a path for future development.

2. DJ-Lab: Distributed Classroom

DJ-Lab[3] is an addon for the IntelliJ Idea integrated development environment that suports the practice of distributed pair programming[6]. With DJ-Lab, two instances of the IDE, running on different desktops, can be synchronized so that every action performed in one of them (writing, selection, cut&paste...) are replicated identical and in real time on the other. Synchronous is supported through a chat bar integrated in the main window of the IDE. Users can append comments to specific sections of code, that are signaled by an icon on the left side and displayed as a tooltip when the mouse is passed over them. With DJ-lab two developers sitting at different desktops can

cooperate in writing Java code as if they where actually working at the same PC. Just like in co-located pair programming, developers have different roles: the one that actually writes the code is the *driver* while the other reviews the code written and is called the *navigator.* The actions performed by the driver are replicated identical and immediately to the navigator, so that they always have a coherent view of the task.

3. XP4IDE: Managing & Tracking XP Projects in the IDE

XP4IDE[4] is an Internet based tool which connects to an XP Project Management Application and presents a view of the XP process data embedded in the GUI of the Integrated Development Environment; IntelliJ Idea and Eclipse are supported. The role of XP4IDE is twofold: from one hand it provides the developer with a view, directly integrated into the IDE, of process data, User Stories, Tasks, acceptance tests and artifacts (classes, documentation files, Make files or Ant files, test cases, etc.), all managed and saved by the project management application. On the other hand, it measures the specific time spent on user stories, tasks, artifacts and sends this information to the server, automating the tracking activity.

The whole system architecture is quite simple: the XP project management application XPSwiki[7] performs as back-end in a client-server architecture where the application protocol is built on top of SOAP/TCP/IP. The integration between XPSwiki and XP4IDE provides the following advantages:

- each actor in the XP process access to process data with the right user interface for the right phase. Developers can use the IDE during development, while customers and managers can use XPSwiki by means of a simple web browser. Both of them can use the web interface during the *planning game*;
- it is possible to associate the artifacts to the tasks in which they are created and developed. This way, the tool collects data about the time spent on every artifact, every task and every user story;
- measuring effectively the effort spent and presenting this information to developers, is a way to build a knowledge base useful to refine and improve the ability of developers to estimate stories in future planning activities;
- beginners have immediately a view of the XP process. Concepts, roles and process data are presented in a structured way, minimizing the learning curve.

4. WebRogue: Virtual Presence in Web Sites

WebRogue[5] is an application for virtual presence over the Web. It provides the Web Browser with a chat sub-window that allows users connected to the same Web site to meet, share opinions and cooperate in a totally free, non moderated and uncensored environment. Each time the user loads a Web page in the Web Browser, WebRogue opens a discussion channel in a centralized server application, that is completely decoupled from the Web server, using the URL of the Web site as a key. Thus whenever a new page is loaded the user can see who is connected, as if entering a physical site. Interactivity is supported by means of two type of commands.

Communication commands allow synchronous interaction as with chat or instant messaging software.

say sends a message to all the users that are watching the same page. We consider a Web page as the analogous of a room, where the Web site represents the establishment of the company. Messages sent using the *say* command are heard by anyone in the same page, as if speaking loud in the same room.

whisper sends a message to a specific user, the message is encrypted for privacy and can be signed for reliability.

scream sends a message to all the users connected to the Web site.

Social commands allow cooperation: group surfing, exchange of visit-cards and wait in line.

follow Users can decide to surf the Web in groups, whenever a member of the group loads a new page, a message is sent by WebRogue to the other browsers that will change location accordingly.

handshake Just like in common instant messaging applications, WebRogue can keep a list of contacts. After handshaking with another user it is possible to see if he or she is online and send a message even if the users are connected to different pages.

wait Users can wait in line to get attention by someone. For example to talk to the clerk in an e-commerce Web site. The *wait* command supports this necessity.

WebRogue lets Web surfers meet in Web sites, just like people meet in real life, without any control or censorship. It is designed with the spirit of free and spontaneous association in mind, to encourage users to share information and opinions. Unlike chat or IM software users are not supposed to subscribe any service or to authenticate, and there is no need to contact a dedicated service to open a channel for discussion. As users with similar interests are likely to consult the same web-sites, the web-site itself becomes a meeting point for them.

5. What next?

Common paradigms. The tools described simulate co-location using the Internet to overcome spacial separation. The points they have in common are:

- *synchronous messaging* is supported to simulate direct conversation;
- *asynchronous messaging* allow leaving notes that other users can read, just like bulletin boards are used in real life;
- *screen replication* allows users to concentrate and work on the same topic (a Web page, a Java class) just like if they where sitting side by side at the same desk;

These paradigms could be applied to any desktop application, but the implementation of this functionalities can be very tricky without a support from the working environment (i.e. a suitable API and plugin architecture).

Technical factors. Although the Web and its protocols is not the best choice as a support for interactive applications, it turns out to be the only practical solution. In a typical environment any channel other than the Web is blocked by firewalls and in many situations a Web browser is actually the only application available. The strength of Web based application is that they are accessible through virtually any computer that has an Internet connection, and, as such, are the closest approximation to an ubiquitous application. Any collaboration tool should, to be usable, rely on well established, Web oriented protocols, like SOAP.

Human factors. Communication is a must to cooperate, but the job at hand has the focus of the users. Communication tools should integrate seamlessly in existing environments and be as transparent as possible. For this reason plugins and addons for existing popular applications (like integrated development environments and web browsers) should be preferred over separated tools.

Synchronous vs Asynchronous. Tools like those described here help communication and cooperation where people are not co-located, integrating synchronous and/or asynchronous messaging into existing applications, but none of the

two paradigms addresses completely the problem, just like e-mail and instant messaging are both indispensable companions of day by day work, but they are complementary rather than interchangeable. Communication & cooperation tools *must* provide both synchronous and asynchronous messaging in order to be complete.

The tools described here put together well known functionalities that have proved their usefulness over the years: messaging and chat exist since the creation of the Internet and have hundreds of million of users worldwide, screen replication and desktop sharing have many commercial implementations. We believe that this technologies are mature enough to be promoted for systematical integration in desktop applications, in order for them to be available anywhere, just like cut&paste or drag&drop.

6. References

[1]. M. Day, J. Rosenberg, H. Sugano,
A Model for Presence in Instant Messaging (RFC2778), February 2000
available online http://www.ietf.org/rfc/rfc2778.txt

[2] C. Dewes, A. Wichmann, A. Feldmann,
An Analysis of Internet Chat Systems
In Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement Miami Beach, FL, USA, 2003. pp. 51-64, ACM Press, 2003.

[3]. A. Soro, R. Sanna, M. Angioni, D. Carboni, G. Paddeu,
DJ-lab: Laboratorio di Informatica Distribuito.
In Proceedings of DIDAMATICA 2004.
Ferrara, Italy, May 2004, Pages 389-395, Pub. Consorzio Omiacom, 2004.

[4]. A. Soro, M. Angioni, D. Carboni, M. Melis, S. Pinna, R. Sanna,
XPSuite: Tracking and Managing XP Projects in the IDE.
In ACM SIGSOFT 2004, Workshop on Quantitative Techniques for Software Agile Processes (QUTE-SWAP), Newport Beach, CA, US. Nov. 2004, ACM Press, 2004

[5]. A. Soro, I. Marcialis, D. Carboni,
WebRogue: Meet Web People.
In Proceedings of the 2nd IADIS International Conference on Web Based Communities (WBC 2005), Carvoeiro, Portugal, February 2005, IADIS Press, 2005.

[6]. D. Stotts,L. Williams, N. Nagappan,P. Baheti, D. Jen, A. Jackson,
Virtual Teaming: Experiments and Experiences with Distributed Pair Programming,
In Proceedings of Extreme Programming and Agile Methods - XP/Agile Universe 2003, 3rd XP and 2nd Agile Universe Conference, New Orleans, LA, USA, August 10-13, 2003. pp 129-141.

[7]. S. Pinna, S. Mauri, P. Lorrai, M. Marchesi, N. Serra,
XPSwiki: an Agile Tool Supporting the Planning Game,
In Proceedings of the 4th international conference XP2003,
Genova, Italy, 2003, Sprigler, pp 104-113

[8]. M. Angioni, D. Carboni, R. Sanna, A. Soro
Strumenti per lo Sviluppo Distribuito di Software
available online http://www.crs4.it/nda/data/papers/