Issues on Supporting Distributed Software Processes

Yun Yang

School of Computing and Mathematics
Deakin University, Geelong, Australia, 3217
Email: yun@deakin.edu.au

1 Introduction

Teamwork is a key feature in any workplace organisation. [11] points to a study that developers of large systems spend 70% of the time working with others. In addition, in this computing era, many (non-trivial) tasks are normally carried out by team members, who may physically dispersed, by using various (software) tools. Our ongoing DWebTeam (Deakin Web-based teamwork support) project has the following characteristics: (1) effective teamwork coordination based on reactive systems concepts with strong process evolution and resource management support; (2) seamless integration of team communication and collaboration tools; (3) entirely Web based; and (4) provision of a visual environment. In this paper, we address some issues on designing DWebTeam in particular and software process enactment in general.

With software support, team members can be coordinated automatically by a system, which is normally more effective than managed manually. This could allow for the cooperation among widely dispersed working groups, whose members may be in different organisations and different countries. For example, team members may reside in Asia, Europe and North America. With around 8-hour time differences among locations, 24 hours a day working mode can be facilitated potentially [5]. Even if team members are co-located in the same building, distributed teamwork is still desirable for various reasons such as sharing information and tools over the Internet or Intranet - as part of the Internet.

Nowadays, there is a growing interest to support cooperative work over the Internet and the Web. The emergence and wide-spread adoption of the Web offers a great deal of potential for the development of collaborative technologies as an enabling infrastructure. According to [9], the Web, as enabling technology for software development and distribution, changes the fundamental assumptions ingrained in the discipline. Furthermore, the Java programming language, which has the capabilities of delivering applets over the Web as well as the claim of writing the code once and then running anywhere, has encouraged us to prototype our framework in Java, based on the Web environment. In this case, no particular (up-to-date) software needs to be installed for (client-side) team members since Java applets can be downloaded on the fly and then run directly. Moreover, using combination of Web/Java seems better than using Web/CGI (common gateway interface) [2] in the sense of performance and control/data.
granularity. Therefore, we have treated the Web and Java as an excellent, if not ideal, vehicle to prototype our teamwork support mechanisms in a global distributed environment.

In this paper, we first address issues related to software process technology in practice. We then describe our framework for global teamwork support in section 3. Finally, we conclude our work and point out future work.

2 Issues on supporting teamwork

Generally speaking, a task/process/project is normally composed of sub-tasks which are partially ordered [3]. By partial ordering, it means that a sub-task should and can only start when its previous sub-tasks have been completed. How to manage sub-tasks is the key issue for completion of the entire task. Hence, task-oriented technology is management-centred to facilitate project management focusing on coordination. Teamwork support systems such as process-centred environments have been investigated in various communities such as software engineering, business engineering, information systems and CSCW for more than a decade. For example, process-centred software development environments have been viewed as a recent generation of software development environments and process supported software engineering is by now a well-established research discipline [1]. Similarly, workflow systems have also been investigated intensively and in effect, quite a few commercial products are available. However, there are still many open issues to be solved in a long run [4, 10].

In reality, basic coordination needs to be enhanced significantly to incorporate more flexible and realistic process evolution and resource management. With respect to the first aspect, i.e. process evolution, processes are dynamic entities that need to evolve to take into account changes in the technology, in the goals and requirements of the organisation, in the market place, and customers needs. Processes, particularly software processes, are often highly dynamic and can rarely be planned completely in advance [8]. Therefore, process evolution or change on the fly is essential but not well supported in the existing systems at the moment. Our work [15] has initially discussed how to support process evolution with prevalent databases. With respect to the second aspect, i.e. resource management, given the nature of distributed teamwork, resources including team members, documents, and hardware/software are very dynamic. How to manage these resources on the fly is also highly desirable. Our work in [14] has proposed resource management using trader. For effective coordination, incorporation of these two aspects is important. However, the high-level coordination support, even with process evolution and resource management, is still insufficient to cover the whole spectrum of teamwork. Hence, we need to integrate facilities to support more effective team communication and collaboration [12, 13].

Research into process modelling paradigms has been intensively carried out in most process support systems published in software process related conference proceedings and journals. We view teamwork more or less a reactive system. Examples of reactive systems include most kind of real-time computer embed-