A Multi-Agent Architecture for Cooperative Software Engineering

Alf Inge Wang, Reidar Conradi* and Chunnian Liu†

Abstract

This paper looks at how Cooperative Software Engineering (CSE) can be supported. We first investigate the process aspects by presenting a traditional process architecture supporting CSE. Then we propose a multi-agent architecture for CSE, which is better in terms of simplicity and flexibility, and particularly useful in modelling and providing support to cooperative activities. We describe an industrial scenario of CSE, and show how to apply the proposed architecture to this scenario. The scenario is based on a software development and maintenance process for a Norwegian software company.

Keywords: Computer-Supported Cooperative Work, Cooperative Software Engineering, Software Process Technology, Multi-Agent Systems

1 Introduction

Most of the work in the software process community has been focusing on how to make people work together in an organised and planned way (partly pre-planned). For high-level processes with little details, it is likely that it is possible to make people work in this manner. However, the development of software involves people that cooperate to solve problems and to do actual work. These kind of processes are very hard to support by traditional software process support tools, because the focus will be more at cooperative aspects than pure coordination of work [22]. In this paper we introduce an architecture to provide support for cooperative software engineering.

Computer-Supported Cooperative Work (CSCW) is a multidisciplinary research area focusing on effective methods of sharing information and coordinating activities. CSCW systems are often categorised according to the time/location matrix [13] (synchronous/asynchronous and non-distributed/distributed). We may add an extra dimension to the CSCW typologies, considering different kinds of cooperative work in the order of increasing complexity of the process support they need [15]:

- Ad-hoc cooperative work such as brainstorming, cooperative learning, informal meetings, design work, etc.. Process modelling support here is implemented through awareness triggers.

- Predefined/strict workflow, in traditional Office Automation style represented by simple document/process flow. Examples of such systems can be Lotus Notes [21], Active Mail [12] and MAFIA [16].

- Coordinated workflow, such as traditional centralised software maintenance work consisting of check-out, data-processing, check-in, and merge steps. There exist several systems supporting coordinated workflow (mostly prototypes), e.g., EPOS [8], MARVEL [2] and APEL [11].

- Cooperative workflow, such as decentralised software development and maintenance work conducted in distributed organisation or across organisations. Here the shared workspace and the cooperation planning are the main extra factors from the process point of view. Example of a system supporting distributed organisations and processes is Oz [3].

By Cooperative Software Engineering (CSE) we mean large-scale software development and maintenance work which falls into the last two categories in the above list. Because of the rapid spread of World Wide Web as the standard underlying platform for CSCW systems, more software companies are moving from the traditional centralised working style to the decentralised one. In the decentralised CSE, communication, negotiation, coordination and collaboration among the various participants are more complicated, because people are not only geographically distributed, but may also work on different platforms, at different times, with different process models. A better understanding about CSE processes is needed as well as a full