Abstract. Work processes in team based software development need to be structured to minimise and resolve conflicting or divergent work. Current software development methodologies propose ways for dividing the whole task of software development between team members. This paper suggests a different way of working by introducing modes of collaboration (MoCs), which support concurrent and collaborative work. A MoC defines how tight two people can work together and how much the rest of the group can demand to know about a programmer. Different MoCs are ordered in a spectrum from single user’s offline usage up to concurrent editing of the same source code. Special emphasis is put on balancing gains and efforts that are related to a specific MoC. The second part of the paper presents how MoCs are implemented in the distributed co-operative software development environment TUKAN. TUKAN includes synchronous co-operative tools and awareness widgets, which operate on a spatial representation of the software under construction. TUKAN provides tools for each MoC and allows programmers to switch between MoCs.

Introduction

Nowadays, software development is usually carried out in teams. Many modern software development methodologies emphasise this fact by introducing special forms of collaboration. For instance, the eXtreme Programming methodology (XP) (Beck, 1999) introduces pair programming sessions, where two program-
mers share one computer and solve the programming task together. The adaptive software development process (ASD) (Highsmith, 1999) is another methodology that focuses on collaboration within the team.

Anyhow, besides the collaborative aspects, programming is implicitly an activity performed by (many) individual users, as writing a book or composing a piece of music (Weinberg, 1971). The discrepancy between isolated work and group work is therefore inherent to software development. Even within XP’s pair programming sessions the participants frequently select one of two possible roles: One is coding (driving) and has the keyboard while the other person observes, comments and corrects the programming activity of the first. Environments that want to support programmers in their job of programming should therefore provide different modes of collaboration matching the roles and phases within the software development process and should ease the transitions between them.

When programming in medium sized to large teams, these teams are frequently distributed across many locations. Even small teams are often composed of experts who work at different locations. This introduces new challenges to the organisation of the programming work. Version management tools and conference systems can be used in these settings, but these tools fail to provide awareness on each other’s work. Conflicting changes and the fact of solving the same problem over and over again are consequences of this lack of awareness.

To assist programmers in programming as a team we propose a new tool called TUKAN. TUKAN provides different modes of collaboration and awareness, which meet the different needs of the programming team at different phases of the collaboration.

In a previous publication on TUKAN, we first identified different points of collaboration during the process of software development (Schümmer and Schümmer, 2001). These points of collaboration (PoCs) will serve as a basis to define different modes of collaboration (MoCs) for software development in this paper. After this, the latter part of the paper describes transitions between the different MoCs and how these transitions and the MoCs were implemented in TUKAN. We will then present some experiences that we gained from first experiments with the usage of TUKAN in programming groups. A section on related and future work concludes this paper.

**Different Modes of Collaboration**

Based on the PoCs that we found in Schümmer and Schümmer (2001), we identified different collaboration modes that are used during distributed team programming. Following the abbreviation PoC for “point of collaboration”, we call the mode of collaboration MoC. A MoC is a lightweight mode, which defines possible collaborative activities. Changing MoCs can therefore be considered as a lightweight activity (comparable to the effort of changing different modes of op-