

A Model of Planning and Enactment Support in Complex Software Development Projects

Sigrid Goldmann

University of Kaiserslautern
Knowledge-Based Systems Group
Kaiserslautern, Germany
sigig@informatik.uni-kl.de
<http://wwwagr.informatik.uni-kl.de/~sigig/>

Abstract

In recent years, software development has become increasingly complex as requirements increased, and geographical dispersion of software development enterprises made distributed software development necessary and/or desirable. This fact complicates not only the enactment of software development projects, but also makes project planning and management much more difficult. Especially keeping the plan up to date, and distributing information about changes during project planning as well as plan enactment to everybody concerned, become more difficult with increasing project complexity while at the same time growing in importance. Decisions made by individuals, project planners as well as software developers, are often made and implemented informally, their rationales being known to the person making the decision and maybe a few others, but never documented in a way that makes them available for later reference. This means that reasons for a decision might be lost, complicating later changes.

We propose an approach to facilitating not only project enactment but also project planning and monitoring, by tracking all decisions made during project planning and enactment and managing dependencies between these decisions. In order to identify those dependencies relevant for a decision, we established an extendable *Model of Planning and Plan Enactment*, which explicitly describes the activities likely to occur while planning and enacting a software development project, as well as standard dependencies between these activities. We formalized this model by adapting an existing dependency management system, the *Redux Model of Design* [1], to record planning and plan enactment decisions and their dependencies. Furthermore, we are currently identifying heuristics to automatically capture typical dependencies, and defining rules to provide automatic planning support where possible.

This approach allows to interleave planning and plan enactment, and to feed enactment data back into the plan, either by automatically reacting to enactment events and plan changes, or by notifying the appropriate person(s). Thus, we provide extensive support for the process of planning and enacting software development projects, in the form of dependency management, user notifications, and, where possible, automation of selective process steps.

[1] Charles J. Petrie: *The Redux' Server*. In Proceedings of the International Conference on Intelligent and Cooperative Information Systems (ICICIS), Rotterdam, May 1993.