Software Development and Experimentation in an Academic Environment: The Gaudi Experience

Ralph-Johan Back, Luka Milovanov, and Ivan Porres

Turku Centre for Computer Science,
Åbo Akademi University, Department of Computer Science,
Lemminkäisenkatu 14 A, FIN-20520 Turku, Finland
{backrj, lmilovan, iporres}@abo.fi

Abstract. In this article, we describe an approach to empirical software engineering based on a combined software factory and software laboratory. The software factory develops software required by an external customer while the software laboratory monitors and improves the processes and methods used in the factory. We have used this approach during a period of four years to define and evaluate a software process that combines practices from Extreme Programming with architectural design and documentation practices in order to find a balance between agility, maintainability and reliability.

1 Introduction

One of the main problems that hinders the research and improvement of various software construction techniques is the difficulty to perform significant experiments. Many processes and methods in software development have been conceived in the context of large industrial projects. However, in most cases, it is almost impossible to perform controlled experiments in an industrial setting due to resource constraints.

However, university researchers also meet with difficulties when experimenting with new software development ideas in practice. Performing an experiment in collaboration with the industry using newly untested software development methods can be risky for the industrial partner but also for the researcher, since the project can fail due to factors that cannot be controlled by the researcher. The obvious alternative is to perform software engineering experiments inside a research center in a controlled environment. Still, this approach has at least three important shortcomings.

First, it is possible that a synthetic development project arranged by a researcher does not reflect the conditions and constraints found in an actual software development project. This happens specially if there is no actual need for the software to be developed. Also, university experiments are quite often performed by students. Students are not necessarily less capable than employed software developers, but they must be trained and their programming experience and motivation in a project may vary. Finally, although there is no market pressure, a researcher often has very limited resources and therefore it is not always possible to execute large experiments.

F. Bomarius and S. Komi-Sirviö (Eds.): PROFES 2005, LNCS 3547, pp. 414–428, 2005.

These shortcomings disappear if the software built in an experiment is an actual software product that is needed by one or more customers that will define the product requirements and will carry the cost of the development of the product. In our case, we found such customer in our own environment: other researchers that need software to be built to demonstrate and validate their research work. This scientific software does not necessarily need to be related to our work in software engineering.

In this paper we describe our experiences following this approach: how we created Gaudi, our own laboratory for experimental software engineering, and how we studied software development in practice while building software in Gaudi for other researchers. This experience is based on experiments conducted during the last four years. The objective of these experiments was to find and document software best practices in a software process that focus on product quality and project agility.

As we proposed in [1], we chose Extreme Programming [2] (XP) as the framework process for these experiments. Extreme Programming is an agile software methodology that was introduced by Beck in 2000. It is characterized by a short iteration cycle, integration of the design and implementation phases, continuous refactoring supported by extensive unit testing, onsite customer, promoting team communication and pair programming. XP has become quite popular these days, but it has also been criticized for lack of concrete evidences of success [3].

This paper is structured as follows: in Section 2 we describe the Gaudi Software Factory as a university unit for building software in the form of controlled experiments. Section 3 present the typical settings of such experiments and portrays their technical aspects. Section 4 discusses the practices of the software process, while Section 5 summarizes our observations from our experience in Gaudi. Due to space limitations, this article focuses on presenting the qualitative evaluation of these experiments. The reader can find more detailed information about the actual experiments in [5].

2 Gaudi and Its Working Principles

Gaudi is a research project that aims at developing and testing new software development methods in a realistic setting. We are interested in the time, cost, quality, and quantitative aspects of developing software, and study these issues in a series of controlled experiments. We focus on lightweight or agile software processes. Gaudi is divided into a software factory and a software laboratory.

2.1 Software Factory

The goal of the *Gaudi software factory* is to produce software for the needs of various research projects in our university. Software is built in the factory according to the requirements given by the project stakeholders. These stakeholders also provided the required resources to carry out the project. A characteristic of the factory is that the developers are students. However, programming in Gaudi is not a part of their studies, and the students get no credits for participating in Gaudi – they are employed and paid a normal salary according to the university regulations.