Chapter 1.22
Software Configuration Management in Agile Development

Lars Bendix
Lund Institute of Technology, Sweden

Torbjörn Ekman
Lund Institute of Technology, Sweden

ABSTRACT

Software configuration management (SCM) is an essential part of any software project and its importance is even greater on agile projects because of the frequency of changes. In this chapter, we argue that SCM needs to be done differently and cover more aspects on agile projects. We also explain how SCM processes and tools contribute both directly and indirectly to quality assurance. We give a brief introduction to central SCM principles and define a number of typical agile activities related to SCM. Subsequently, we show that there are general SCM guidelines for how to support and strengthen these typical agile activities. Furthermore, we establish a set of requirements that an agile method must satisfy to benefit the most from SCM. Following our general guidelines, an agile project can adapt the SCM processes and tools to its specific agile method and its particular context.

INTRODUCTION

In traditional software development organisations, software configuration management (SCM) is often pushed onto the projects by the quality assurance (QA) organisation. This is done because SCM in part can implement some QA measures and in part can support the developers in their work and therefore helps them to produce better quality. The same holds true for agile methods—SCM can directly and in-directly contribute to better QA on agile projects.

Software configuration management (SCM) is a set of processes for managing changes and modifications to software systems during their

entire life cycle. Agile methods embrace change and focus on how to respond rapidly to changes in the requirements and the environment (Beck, 1999a). So it seems obvious that SCM should be an even more important part of agile methods than it is of traditional development methods. However, SCM is often associated with heavily process-oriented software development and the way it is commonly carried out might not transfer directly to an agile setting. We believe there is a need for SCM in agile development but that ist should be carried out in a different way. There is a need for the general values and principles of SCM, which we consider universal for all development methods, and there is a need for the general techniques and processes, which we are certain will be of even greater help to agile developers than they are to traditional developers.

There are some major differences in agile projects compared to traditional projects that heavily influence the way SCM can—and should—be carried out. Agile methods shift the focus from the relation between a project's management and the customer to the relation between developers and the customer. While traditional SCM focuses on the projects and company layers in Figure 1, there is a need to support developers as well when using SCM in agile projects. Shorter iterations, more frequent releases, and closer collaboration within a development team contribute to a much greater stress on SCM processes and tools.

Agile methods are people-oriented rather than process-oriented and put the developer and the customer in focus. As a consequence, SCM has to shift its primary focus from control activities to that of service and support activities. The main focus on audits and control needs to be replaced by a main focus on supporting the highly iterative way of working of both the team and the developers, as seen in Figure 2. From a QA point of view, the control measures are moved down to the developers themselves with the purpose of shortening the feedback loop in agile methods. So SCM does not directly contribute to the QA on an agile project, this is the task of the processes that the agile method in question prescribes. However, by supporting said processes and making them easier and safer to practice SCM indirectly is a big factor in QA on agile projects.

The traditional process-oriented view of SCM has also lead to several misconceptions of agile methods from an SCM point of view. The lack of explicit use of SCM and its terminology has lead quite a few people to conclude that agile methods are not safe due to an apparent lack of rigorous change management. However, a lot of

Figure 1. The different layers of SCM

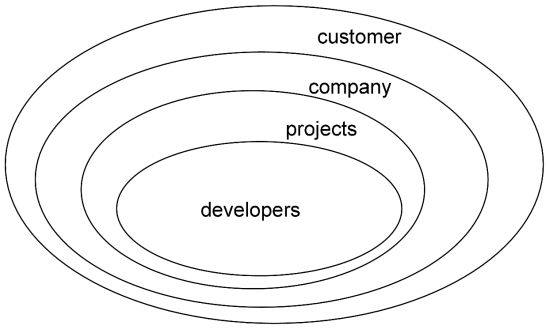

Figure 2. The main development loops in agile