Abstract. SCM provides many well-known benefits for traditional software development. It enables software teams to develop quality software in a timely and predictable manner. However, some teams who are doing Rapid Application Development, also known as RAD, sometimes feel that standard SCM processes have too much overhead for their quickly moving team members. In fact, some teams choose to forego SCM altogether when in rapid development mode, because they think it slows them down. Is RAD really incompatible with SCM? This paper explores the relationship between SCM and RAD, provides some strategies to keep SCM from hindering RAD, and describes some ways in which SCM can help teams develop applications more rapidly.

1 Introduction

Software is being developed more quickly than ever before. Environments such as Visual Studio provide fully integrated development tools and packaged controls that enable rapid prototyping of complex applications. Tools such as Rational Rose can generate application code based on system models. GUI builders abound, both for traditional software development and web development. Software that once took months or years to develop can be put together in just weeks or even days. Companies publish changes to their web sites daily, sometimes hourly.

These rapid methods of software development are known as Rapid Application Development, or RAD. RAD teams may have many of the following characteristics:

- They are working on a new product that has not yet been shipped.
- Developers often must work closely together on new features.
- Team members are adding new files on a regular basis.
- They have a general sense of the tasks that need to be done, but often discover additional tasks as the features they implement become better defined. The project plan may not be as detailed or well-defined as for traditional software projects.
- Developers may be using tools to quickly prototype or generate their software, such Microsoft’s Visual Studio or FrontPage, or Rational Rose.
- The team is not doing exhaustive product testing, and may not even have a regu-
lar build cycle yet.

The benefits of SCM are well known. However, some RAD teams feel that standard SCM processes have too much overhead for their quickly moving team members. Typical SCM capabilities such as tracking each change individually, insulating developers from each others’ untested changes, and managing build areas may be more than a RAD team needs at a given point in time, especially when developing a new application. RAD teams need to be able to relax tight SCM restrictions when it makes sense; likewise, it is important for them to understand the tradeoffs of doing so, so they can make informed decisions about the risks involved. In addition, some processes that look like overhead can actually help you develop more rapidly.

This paper describes some SCM process alternatives for RAD teams and discusses the tradeoffs of using those strategies. Ultimately, the paper offers some ways that SCM tools can help teams meet their rapid application development goals.

2 SCM Process Alternatives for RAD

One option for RAD teams is no SCM at all. Although most of us would never even consider this an option, many RAD teams do, because they feel that SCM practices slow them down. Of course, the drawbacks of this decision are numerous: the team has no insurance against lost work, bugs may reappear in later versions of the software after they were fixed, developers may overwrite each others’ changes, the product quality may suffer, and the team may not be able to reproduce and maintain the software after it is released. To be successful, it is essential for a RAD team to follow some SCM practices even if they do not use an SCM tool.

At Continuus, we’ve noticed that RAD is becoming much more common at our typical customer site, and we’ve spent some time working with those teams to find ways they can be successful while using SCM. Although we cannot recommend these alternatives as best practices for SCM, they are practices that enable RAD teams to accomplish their goals and yet incorporate some important benefits of SCM. These practices are not specific to any particular SCM tool, although some of them assume the SCM tool or environment has certain features.

The process alternatives for RAD teams discussed here fall into the following categories:

- Alternative build and test cycles
- Less formal change management
- Closer sharing between developers
- Ways to manage rapid configuration changes

Each of these topics is discussed in detail on the following pages.

2.1 Alternative Build and Test Cycles

Mature traditional software development teams often have a process that incorporates two or more stages of testing. The first stage might be categorized as integration testing, where changes from different developers are gathered and tested together.