Agile SCM for Testers

By Steve Berczuk

While not usually associated with agile software development, good software configuration management practice is essential for teams to apply agile principles and deliver functionality quickly while maintaining high quality. At its core, SCM is about feedback, which is a central principal of any agile process, and one that is also at the heart of testing. The challenge for teams is to develop processes that provide the management and tracking functions of a good SCM process without slowing down development.

SCM is most commonly thought of in terms of version management and branching, and the most common SCM-related question is, "What version management tool should I use?" While version management is an essential part of SCM, it is not the primary goal of an SCM process, but only a means to an end. It is a mechanism for communication among developers, testers and other stakeholders, and for providing feedback about the quality of the product and the state of the code relative to product features.

SCM provides for this feedback through functions such as:

- Identification: Identifying the components of a deliverable
- Control: Ensuring that a deliverable contains the right components and the right features
- Status Accounting: Reviewing statistics about changes, allowing for risk analysis and improvement
- Audit and Review: Validating that changes to delivered software were as expected

Through these functions, an SCM process allows stakeholders on a project, including developers, testers and product owners, to understand the current state of the software. For agile teams, maintaining these functions is especially important, as agile teams depend on rapid feedback to make corrections as needed. For testers on agile projects, it is especially important for an SCM process to support frequent release snapshots along with an identified set of changes so that testers can work effectively.

An agile process needs to implement SCM functions in a low-overhead manner that integrates SCM into every aspect of the development lifecycle. While topics such as testing and build are rarely mentioned in the context of SCM, testing plays a key role in agile SCM processes. Processes such as automated testing, combined with continuous integration, provide the basics of the control, status accounting and audit functions in a way that allows teams to move rapidly. Some of the practices that agile teams can implement to provide the value of SCM with minimal overhead include:

- Fewer, more active codelines: Rather than isolating streams of work on various branches, work toward eliminating the problems that destabilize codelines. Risk is decreased by integrating work sooner, and cost is decreased through merge reduction.
- Frequent commits: By working in small increments, and also by updating workspaces frequently, developers are less likely to accidentally diverge from each other.
- Consistent and continuous build: A consistent build process that can be applied to developer and integration workspaces allows for fewer broken builds and faster completion of work.
- Testing: Automated unit tests that developers can run while coding (and which also run during the build) can identify potential problems sooner, when developers recall the context of a change.
- Deployment: Building a deployment strategy early and making iterations during the development process help enhance the value of integration testing by ensuring that what gets tested is what a customer will see.

Software configuration management is about more than version management; it is a mechanism that allows agile teams to move at a rapid pace while maintaining high quality. By integrating practices such as these in the development and test lifecycle, agile teams can provide stakeholders with a continuous stream of feedback about changes to the application. The availability of frequent builds from an integrated codeline allows testers to review work in progress and provide early, rapid feedback. And the presence of automated tests allows testers to focus on new functionality rather than regression testing. In an agile SCM environment, the focus is on enabling rapid change, ensuring stability through testing, and on erring on the side of identifying problems quickly rather that avoiding change to prevent problems. ☒

***Steve Berczuk** is an engineer at Health Insight Technologies.*