Tooling Up for **Agile Development**

by Allan McNaughton

Many in the software industry see agile development methodologies as the antidote to the process-heavy development practices that are so commonplace today. Success in agile development is measured by the working code produced, not the number of specifications written, models designed, or meetings attended.

The Agile Manifesto, which is the product of collaborative efforts between seventeen luminaries in the software development community, clearly spells out the principals of agile development. Agile development values:

- Individuals and interactions over processes and tools.
- Working software over comprehensive documentation.
- Customer collaboration over contract negotiation.
- Responding to change over following a plan.

While agile practices are a breath of fresh air to developers bogged down in process, agile development does not mean throwing out processes and tools. On the contrary, the methodology requires delicately balancing the need for process and structure with the basic desire for solid results.

In this paper we will discuss the central theme of agile development practices and learn how Seapine Software's tools can facilitate success in an agile world.

Agile Development to The Rescue

Traditional waterfall-like development methodologies break down a software project into four distinct phases: requirements gathering, software design, coding, and testing. The work then proceeds in a serial or progressive manner until the tasks in each phase are complete--or thought to be complete.

The problem with the waterfall approach is that changing requirements can often render the best laid plans irrelevant. Attempts to minimize change by locking down a requirements list results in software that, while complete, does not meet the customer's actual needs.

Agile methodologies address the needs of the real world because they are based on the already proven strategy of iterative development. Simply put, iterative development works by breaking the project into more manageable chunks (iterations) and focusing the team on creating functional software at the end of an iteration (iterations are 2-8 weeks in duration depending on project size and the agile methodology selected).

This approach stands in contrast to traditional lockstep software development methodologies that produce volumes of specifications, plans, models, and many other non-code artifacts throughout their lengthy planning phases.

Four Steps to Achieve Agile Success

Agile development eschews the traditional "big-bang" approach to building software.

Instead of planning huge releases that take many months (or years) to build, agile methodologies produce smaller releases at a far more frequent rate.

Agile developers select a manageable subset of features to be developed in the timeframe of a single iteration. These features are then elaborated, designed, constructed, and tested during the iteration.

The processes that agile developers follow are remarkably simple. Agile developers must:

- Define the system to build – collect high-level requirements (detailed elaboration is deferred until the "last responsible moment").
- Allocate requirements to the release plan – determine to which release a requirement needs to be assigned.
- Create iteration plans – assign the requirements for a release to specific iterations within that release.
- Execute to meet the iteration plan and deliver – the goal of agile development is to build working software, evaluate the results, adjust the course of direction, and start the process again.

Simplicity Fosters Better Results

Agile development places a premium on simplicity. Getting the job done with a minimum of fuss is paramount. The philosophy extends to the type of tools agile developers use. When there are only 30 days to produce functional (and tested) software, developers cannot afford to spend time fiddling with overly complex tools.

The primary activities of agile software developers are straightforward. The developers must build software, track its progress, and test what they have built. These are the core tasks that generate good results. Seapine Software helps streamline these tasks with development tools that are especially well-tuned to the needs of fast moving agile projects.

Building Software Agilely

Underlying the success of any agile development project—or any software project for that matter--is a solid change management system that tracks each and every modification to the code base. Development projects without such a system can quickly spiral out of control.

Agile developers will find that Seapine Software's Surround SCM offers robust change management capabilities without the crushing complexity of similar tools. Surround SCM combines the most useful features of enterprise-level change management systems in an easy-to-use package that works equally well for both local and remote users.

The iterative nature of agile development is easily supported by Surround's sophisticated branching capabilities. Developers can work either in a private branch to protect their code from changes made by other team members or in a shared branch so changes can be picked up in the nightly build (individual changes and/or change sets can easily be promoted from private branches to the project's shared branch).

When development on the iteration is complete, a snapshot branch can be created that captures the state of the source tree. Surround's flexible branching model allows agile developers to manage assets in a way that closely follows the iterations and releases of the project.

Surround has many additional capabilities that further streamline development, ranging from automatically sending out notification emails when files are changed to calling user-specified scripts when certain operations occur. And, Surround does all of these tasks using a true client/server architecture that provides the security, flexibility, and scalability needed by even the largest agile team.

Tracking Software Agilely

Although agile projects use minimalist development practices, the need to track open issues and work pending is still required. In agile circles, the set of features, defects, and other w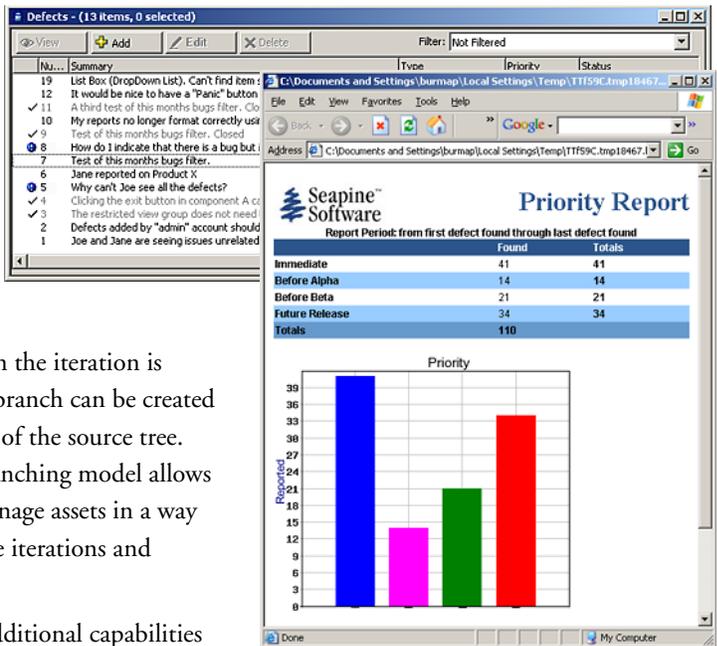ork to complete is referred to as product backlog. Backlog items begin life as part of the product backlog and later get assigned to a release backlog (at which point the items get built).

Although the product and release backlog could conceivably be tracked on a large whiteboard, this information

Figure 2: List View and Priority Report Window in TestTrack Pro

can be managed much more efficiently using Seapine Software's TestTrack Pro--an advanced issue management solution that can track any backlog item (e.g., feature request, change request, defect) through its lifecycle.

The capabilities of TestTrack Pro neatly coincide with the way agile developers work. As product backlog items are uncovered, they are entered into TestTrack Pro. The project team can assign priorities and time estimates as needed. Individual backlog items can then be tracked in detail and assigned to developers for closure within the specified iteration of an upcoming release.

Developers can assign product backlog items to a release in any of a number of ways. Some teams may simply customize the priority field (such as setting that field to "Fix In Iteration 4") to group backlog items. Other developers may choose to link backlog items to a parent release, so the release cannot be closed until each item in the backlog is addressed.

TestTrack Pro does not impose a one-size-fits-all solution on development teams. With its extensive customizations, which include everything from modifying the workflow to adding custom fields and renaming fields, TestTrack Pro can be easily configured to work the way a team actually works.
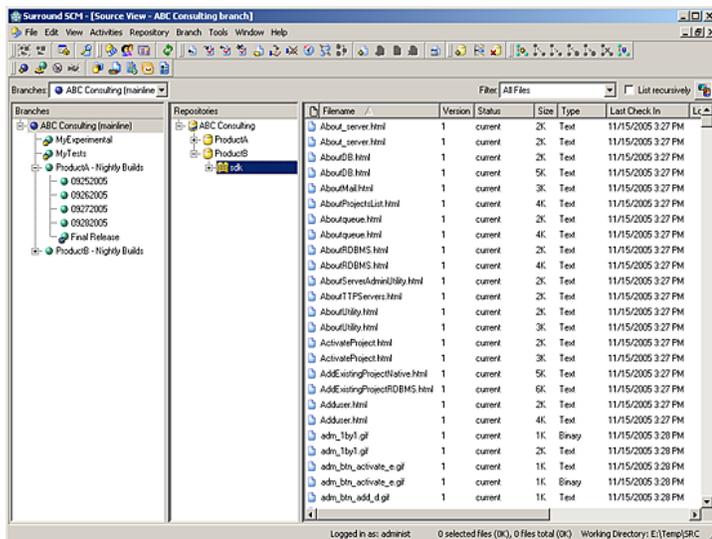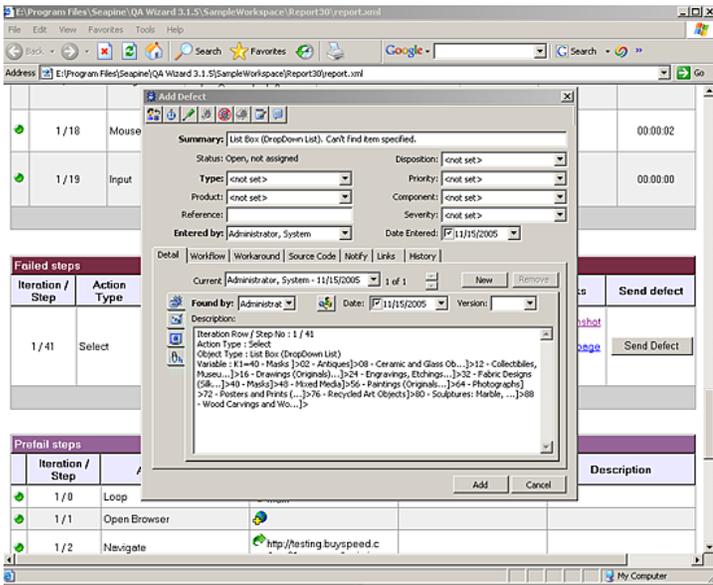

Figure 1: Branch Tree in Surround SCM

Figure 3: QA Wizard and TestTrack Pro Integration

Testing Software Agilely

In agile projects, software testing is not relegated to the end of the development cycle. The focus on creating working software in a timely manner means testers and developers must work closely together throughout the iteration. To achieve the goal of functionally correct high-quality software, agile projects commonly run automated tests on nightly builds.

Seapine Software's QA Wizard adeptly meets the needs of agile teams by making it easy to create automated tests for most applications. QA Wizard's point-and-click interface avoids the steep learning curve common to test automation tools. Testers can even set up checkpoints that compare the value of a specific attribute to an expected result without having to learn a scripting language.

Putting the nightly build through its paces with QA Wizard is a snap. If a test fails, QA Wizard captures the information necessary to help diagnose the problem—including a screenshot of the failure. QA Wizard can even automatically submit the list of test failures to TestTrack Pro. This information can drive the next day's activities and is a perfect topic for the daily standup meeting.

QA Wizard also easily handles application changes, such as modifications to the user interface layout, which break other test automation tools. It does this by binding objects to a parent object within the user interface, not coordinates on a page. This allows QA Wizard to locate the object under test automatically even if it has been moved—a likely occurrence in rapidly changing environments.

Catching the Agile Wave

While agile development practices may not be suitable for every team (highly regulated environments come to mind), they have certainly proven to be a boon to many developers. Armed with winning development tools, such as Seapine Software's Surround SCM, TestTrack Pro, and QA Wizard, teams can now agilely transform themselves into productivity powerhouses that consistently deliver quality software on time and within budget.

About the Author

Allan McNaughton (allan@technical-insight.com), a veteran developer and long-time writer, is the principal analyst at Technical Insight LLC, a firm specializing in the composition of high-technology white papers.

Seapine Software™
Changing the World of Software Development