



# The AgileCycle Advantage

Author:

Damon Poole, Chief Technology Officer

# AccuRev

## The Agile Vision

Imagine a world where you are predictably delivering high quality products to happy customers faster than ever before and realizing more revenue and higher profits as a result. A world where compliance is automatic, audits are a breeze, and expenses are kept under control.

In this world builds don't break, developers are happy and productively working on the right things. People are working smoothly towards common goals at multiple sites across the globe. Using parallel development, multiple projects are worked on all at the same time, customer updates are made quickly, and there are fewer regressions due to fixes that didn't make it into a new release.

More and more folks are experiencing this world as they upgrade their development tools and development methods. Just as C replaced assembler, C++ replaced C, and Java and C# are replacing C++, new development tools are replacing the old ones and Agile development is replacing traditional development methods.

### Transitioning to Agile Development

Congratulations! You've decided to make the move to Agile development. In order to adopt an Agile process you are sending your people to Agile training and you've engaged the services of an Agile coach. You are reconfiguring your work environment to facilitate the practice of collocated cross-functional teams. But what about an invisible part of your work environment which has a daily impact on your team's effectiveness?

Your software development tools may be just like your carpeting. Installed many years ago, coming up at the seams, patched together with scripts, completely out of date, and nearly invisible. Why not maximize the results of your Agile transition by also reconfiguring your development tool stack and transitioning from your existing tool stack to an Agile tool stack?

As you transition to Agile development, many of the components in your tool stack that were helping you do traditional development are now working against you in two ways. First, they aren't helping you do Agile development well, and second, they are actually working actively against your efforts to be Agile. You may also be missing some support structure that you need and tools that can help you leverage your effort.

### Survey of Agile Tool Stacks

At one extreme of tool stacks for Agile development is a stack that has no support for Agile whatsoever. One example of this is the following combination:

- CVS for source control
- Bugzilla for tracking defects
- Word and or Excel for tracking new features
- Low code coverage, provided by system tests only
- Monolithic build and test scripts
- Automatic full build only available via nightly build

This environment will actually work against your transition to Agile development. It includes two tools that are poorly suited for Agile development – CVS and Bugzilla – and it is entirely missing support for Agile Project Management, unit tests, and Automated Build Management.

At the other end of the spectrum is a tool stack purpose-built for Agile development and linked together into an integrated Agile solution. Along this path there are really just two choices out there today. One choice is IBM's Rational Team Concert. Team Concert is a good system but it is immature, expensive, and there don't seem to be many deployments.

The other option is to build a custom solution using individual tools. For instance, you could use GIT for source control, Hudson for automated build management, and Excel and your existing issue tracking system for Agile project management. There are two problems with this combination. First, the only component purpose-built for Agile development is Hudson. Second, using Excel for backlog management doesn't scale very well. If only there was another way!

## Introducing AgileCycle

To simplify the adoption of Agile and increase the success of existing Agile projects, AccuRev introduces AgileCycle. AgileCycle is a fully-integrated Agile Application Lifecycle Management (Agile ALM) solution offering SCM, Agile Lifecycle Management (including Agile project management), and Build & Release Management together. AgileCycle incorporates true best-of-breed tools from AccuRev, Rally Software, and UrbanCode in order to deliver a comprehensive Agile suite, designed specifically for today's Agile development teams.

## The AgileCycle Advantage

There are many advantages to AgileCycle over competing solutions. One of the biggest advantages is that we've already done a lot of the work that you would have to do on your own. We've leveraged our years of deep software development process experience and industry partner experience to put together a comprehensive Agile solution.

With AgileCycle, organizations get the benefit of a fully-integrated solution of tools and services which support the full spectrum of Agile practices from a single vendor including deployment, integration, Agile coaching, product training, maintenance and support, without the single-vendor lock-in this typically brings.

At AccuRev, our strategy is to provide an open best-of-breed model where we support all industry leading tools. We've sold our own bug tracking and request management software, AccuWork, for years while continuing to invest in a wide variety of out of the box integrations with other issue tracking tools. In addition to these advantages, AgileCycle provides many specific feature advantages.

### Transition to Agile at Your Own Speed

While AgileCycle is uniquely qualified to support Agile development, it also offers support for other methodologies. You can transition to Agile development all at once or a project at a time. Whatever your choice, you can use AgileCycle to manage, track, and measure all of your software development projects.

### Unique Support for User Story Based Engineering

In order to facilitate the use of a backlog, you really need to have all work represented in that backlog. Ideally, all work that is performed should be in the form of user stories and defects and managed via the backlog. Thus, all user stories and defects should be available via your Agile Project Management system. That doesn't mean that they need to be originated there, you may have a separate system for defect management, but all planned defects should be migrated into your Agile Project Management system.

In AgileCycle, thanks to its unique combination of integration and change packages, change packages unify all aspects of your Agile lifecycle. The benefit is that nothing falls through the cracks and all work can be managed, tracked, and measured via the same process.

There are many different methods for linking the work requested by the business to the work done by the development organization to fulfill that request. An all too common method is to have no linkage at all. Other forms of integration include typing a bug number into a comment, tight integration on a file/version level via scripting or built-in capability to do validation and other actions, tight integration via transactions (sometimes called change sets or change lists), and tight integration via change packages. Change packages are the best form of integration in general and also the best suited for Agile development.

A change package represents a complete patch of all changes required to implement a particular change. It is a rollup of all of the changes that have been applied to resolve an issue including a complete audit trail of who made the changes, when they made the changes, and why they made the changes. Each change package is tied to an issue in the Issue Tracking System (ITS).

Unlike other changed-based integrations where all information about the changes associated with an issue are stored in the ITS, AgileCycle uses change packages. The advantage to using change packages is that the progress of the issue and its current state can be seen via the source control system which is a more natural interface for developers.

Change packages simplify the process of backing out changes. Any change can be backed out at any stage of the development process. If the change is being backed out due to a testing failure, it is easy for a developer to pick up work on the change and then reintroduce it when it is finished.

### **Refactoring That is Second to None**

Refactoring is the process of improving the maintainability of code without changing its results. This practice originated with Agile Development but has since become a common coding practice as evidenced by its inclusion as a feature in Eclipse and other IDEs.

Refactoring may occur anywhere and may affect many files. Most SCM tools discourage refactoring by requiring developers to check files out first or losing track of rename and merge operations. If developers find it hard to do, they will avoid it, thus losing out on the benefits of refactoring.

AgileCycle encourages refactoring with an all-writeable and all-real files SCM model, excellent rename support which works even across merges, and full merge and rename tracking. In addition, the Eclipse and Visual Studio plug-ins insure that all refactoring operations performed via the IDE are accurately reflected in the SCM system.

### **Built-in Support For Continuous Integration**

Continuous Integration is the practice of automatically triggering a build and often a build and test whenever a change is made to the code base. This provides instant feedback to developers and helps to keep the code base in a "ready to release" state, free of build problems and test failures. While Continuous Integration originated with Agile Development it has since become a mainstream development practice in its own right.

Setting up Continuous Integration is a simple process using AgileCycle. For example, you can set up a trigger such that on successful completion of a build and test cycle, all of the source file changes are automatically promoted to the next level in the development hierarchy, protecting the integrity of the next stream level.

AgileCycle's Stream Hierarchy makes it simple to set up multiple levels of continuous integration corresponding to whatever process stages you have in your stream hierarchy such as code review, team integration, product-wide integration, etc. For large teams, multi-level CI is highly recommended.

## Advanced Support For Build Management

In Agile, feedback should be as quick as possible. Consider a typical “nightly build.” In many shops, there are three kinds of builds: official builds, nightly builds, and developer builds. The official builds are done infrequently and are the builds that become the official product or officially deployed version of the software. Nightly builds are similar to the official builds, but are used for the sole purpose of providing feedback to development as to the state of the mainline. Developer builds are the builds that developers do for themselves as they do their work.

The official build of a piece of software can be a fairly complicated affair. The build script may consist of 100 or more steps, each of which may fail. Following the build, the test phase may involve the running of thousands of individual automated test, each of which may also fail. Typically, there is a check after the build to see whether or not to run the tests, but often the build and test phases are monolithic processes and are not broken up into pieces. In an Agile environment, there will be multiple teams, each of which need their own build. The best situation is to be able to fail fast. That is, build the pieces of the software which have changed first, run the tests which are most likely to detect systemic failures first, etc. If you think of the process of going from source code to ready-to-install bits as the whole build process, you have a dependency graph of many steps which can be run in parallel across multiple machines. In order to do testing, it is also likely that you will need to use virtual machines in order to automate the fast setup and teardown of test environments.

AgileCycle has direct support for “failing fast.” Via AnthillPro, you can run most of your build and test pieces in parallel and distribute the load of the many builds typical of Agile development across your entire server farm. AgileCycle will take care of all of the details of dependencies, load balancing, and virtual environment management.

## Maintain a Single Agile Workflow Across All Tools

All software projects track the progress of the work to be done using some set of workflow stages. It may be a very complicated workflow that can fill multiple pages, or in the case of an Agile project it may be as simple as “reported,” “backlog,” “todo,” “in progress,” “coded,” “done.” People are used to keeping track of this status on a per-work-item basis in their issue tracking systems and Agile project management systems, but this does not generally translate over to their SCM system. In most cases, the SCM system models all states with a single concept: “the mainline.”

By mirroring your Agile workflow with AgileCycle’s streams, you can get many of the same benefits that tracking state gives you in your change management system in your SCM system. For instance, if a tester wants to do exploratory testing, they can query the change management system to find out which user stories for the current iteration are in the “coded” state. That way, they only test things which developers have indicated are ready for testing. By using streams to model your workflow, that same tester can go to the “coded” stream to check out the sources, build the system, and do exploratory testing on a version of the system containing only “coded” changes instead of including “in progress” changes.

Having streams representing workflow stages also makes it possible to easily find out what is different between stages which aids in tracking down problems. Differences can be determined on both a file and issue basis.

## Keeping Up with Change

When adopting Agile Development, it is likely that your process will change many times along the way. The more flexible and automated your process is the faster you can make the transition and the less it will cost you to implement it. AgileCycle is designed around the philosophy that change is inevitable. In support of this philosophy, everything in AgileCycle is configurable and flexible with good defaults to start from. For example, AgileCycle allows you to reconfigure your process model with fast and easy drag-and-drop operations that remove the need for scripting and guesswork.

## Distributed Agile Development

The short development cycle that results from the short iterations of Agile development requires very close interaction between all participants. Work must be integrated both frequently and rapidly. As a result, when development is taking place at multiple sites, the environment must be set up to give the perception that development is actually taking place at a single site. The obvious solution to this is to use replication.

Some replication solutions require a branch per site. That will not work well in an Agile environment. AgileCycle's optional replication solution provides high performance for remote users and gives the appearance that all users are working at the same site, thus making distributed Agile Development a practical option. You can also use replication to remove the load of build servers from your main server, thus reducing the impact of continuous integration on overall performance.

## Compliance and Agile Development

There are many compliance standards and regulations in effect today, including Basel II, 21 CFR Part 11, COBIT, COSO, HIPAA, ISO 17799, ITIL, Sarbanes-Oxley, SAS 70, and others. It might seem like doing Agile Development is not compatible with compliance requirements. However, AgileCycle plays a vital role in any compliance effort that involves software development.

AgileCycle has many built-in features which automate compliance activities. These features include process visualization, change packages, separation of duties, physical security, standards based authentication and authorization, extensible controls, and support for audits.

## Support For All Leading Issue Tracking Systems

Only AgileCycle provides out-of-the-box integrations with all of the leading Issue Tracking systems on the market today including Atlassian Jira, Rational ClearQuest, MKS Integrity Manager, Bugzilla, Serena TeamTrack, and HP QualityCenter. If we don't integrate with your system, we can work with you to provide it or you can use our integration SDK to do it yourself.

## Conclusion

If you are considering a move to Agile development, or are already doing Agile development, why not consider upgrading to AgileCycle and using the best tools available? You'll be reinforcing your investment in Agile and supercharging your results.



AccuRev, Inc.  
10 Maguire Road  
Lexington, MA 02421

Phone: 1-800-383-8170  
sales@accurev.com  
www.accurev.com