Agile software development
White paper
January 2009

Rational. software

IBM

The top three secrets to successful agile software development.

Galina Mishiev, go-to-market manager, Rational software, IBM Software Group

Adeel Omer, go-to-market engineer, Rational software, IBM Software Group

Contents

Introduction

Agile software development methodologies have taken the development community by storm. A recent agile adoption survey of Dr. Dobb's subscribers found that 69 percent of respondents were already using one or more agile techniques.[1]

And it's easy to see why agile has become popular. It focuses the efforts of development teams on results and deliverables. This underscores the ultimate goal of commercial software development—providing software solutions that fulfill unmet marketplace needs, drive competitiveness and help ensure profitability for the company. The iterative and incremental nature of agile—with daily builds, or even multiple builds throughout the day—enables the enterprise to get applications to the marketplace faster, rather than having to dislodge competitors later.

Agile is highly collaborative and self-organizing, meaning the people doing the work are the people doing the organizing. It provides just the right amount of process to match the needs and culture of the development team.

Successful agile development teams have found that they can produce quality applications. The methodologies are cost-effective and time efficient, with less process burden. Teams can meet customer needs and build close working relationships with stakeholders to rapidly adapt to changing requirements and gain a competitive advantage against the competition.

Scaling agile for larger development opportunities

A key facet of agile development is that it is highly adaptable to different organizations. Agile methodologies are already providing a number of benefits to early adopters among small, colocated development teams. And the obvious success that smaller teams are having with agile approaches is being noticed by larger teams and organizations that wish to implement its principles and techniques in their own environments.

Highlights

Although agile practices are often considered only useful for small development teams, they can be scaled to address the needs of large teams.

But the broader adoption of agile methodologies in the global development community places greater demands on its use to meet some ongoing, real-world challenges. Agile methodologies need to adapt to the many business, organization and technical complexities that development teams are now facing. These challenges include:

- *Keeping information flowing properly among larger, distributed development teams.*
- *Making effective handoffs—especially when the development team is offshore or outsourced to a third party.*
- *Accommodating platform-specific requirements or multiple hardware resources.*
- *Conforming to regulatory requirements—including the Sarbanes-Oxley Act—and internal governance mandates.*

So how do we achieve the full potential of agile development? More importantly, how do larger development teams—often distributed around the world—get the most from it?

Fortunately, the principles underlying agile development can be modified for larger teams. And tools to improve product quality, team efficiency and on-time delivery have now made this possible.

When you apply agile practices to larger teams, you need to have an effective implementation plan.

Incorporating traditional development processes to get the most out of agile development

Development teams still need to have a plan for the overall effort. They need to have consistent, streamlined processes. They need to provide documentation for audits. And they need to train incoming staff.

Highlights

The IBM vision for scaling agile practices includes three fundamental elements: process, technology and expertise.

Further, as agile development practices scale to larger projects and organizations, development teams become more spread out around the globe. They work in more offshore locations. And they become more outsourced. This increases the need for more sophisticated coordination than is required for smaller, colocated teams.

This white paper discusses what IBM has found to be the most useful processes to incorporate into agile development and the latest solutions that make that possible. The IBM vision encompasses three fundamental building blocks for getting the most from agile development:

- *Process*
- *Technology*
- *Expertise*

Secret 1—Process helps smooth out the complexities of software development

First, it's important to apply the right amount of process to the development effort by implementing practices. Practices provide useful units of knowledge that help solve one or more commonly occurring problems. A practice can be adopted independently and improved from other practices so that an organization can adopt one or a few practices at a time.

Practices can help you map business objectives to areas that need improvement.

Further, a practice can map to business objectives and pain points so that you can identify needed improvements—speeding time to market, enhancing quality or increasing productivity. The effectiveness of adopting the practice and its results can be measured to take the necessary corrective actions.

Implementing practices makes it easier to adopt improved development techniques while avoiding processes that are too heavy, large or complex. Independent practices can be learned and implemented over time. And the adoption and results of the practice can be measured more quickly.

Highlights

Incremental adoption of practices is the key for successfully employing them. You can start small and add on as needed. It's best to begin by assessing your organization, its projects, and its needs and pains. Then choose which practices you need to adopt first.

IBM Rational Method Composer software helps integrate industry-proven best practices

Development teams need to facilitate collaboration between business and IT teams to help organizations more effectively manage software projects and maximize returns on technology investments. Better team communication improves project predictability and helps better manage and mitigate risk. Increased project responsiveness and resilience help keep development on track and the company more competitive.

IBM Rational® Method Composer software represents a major evolution of IBM's process solutions. It includes and extends the IBM Rational Unified Process®, a software process framework that has guided some 500,000 developers around the world in a broad range of software and systems development projects.

Rational Method Composer includes a customizable process library and tooling to help you manage, author, configure and deploy effective processes tailored to your project needs. The process library offers proven and reliable guidance for software and systems development, management and governance with more than 100 selectable and customizable process best practices that can be applied to a variety of processes and domains.

Rational Method Composer software includes a customizable process library to help you develop processes that support business needs.

Highlights

Secret 2—Software development technologies can help deliver ship-ready code

Focusing on the quality of deliverables provides the best measure of the success of the software development process. Begin with visual modeling and development tools to help define the application at a much higher level, and then maintain it at that level throughout the process.

Visual modeling and development tools help teams envision what the overall software picture looks like.

Modeling helps the development team understand what the overall software picture looks like and how the pieces fit together to make the whole. Just like a completed puzzle consists of many pieces that must fit together perfectly in order for the whole picture to emerge, visual modeling helps ensure that every piece is in the right place.

But in an agile environment it's also critical to transform the model to the code and to enable reverse transformations. Teams may need to refactor code as requirements change or because they uncover some unanticipated roadblock. Visual modeling by itself may lack semantically relevant information. It might require a lot of manual work that could derail timelines and result in outdated requirements. The end pieces may not reflect the intended design.

Your visual modeling tools should help you rapidly develop applications so you can seize competitive advantage.

New solutions are available to help teams more effectively use visual modeling. When it comes to rapidly developing applications for your company to seize competitive advantage, it becomes increasingly important to leverage the features that make things simpler for development teams.

Highlights

New features include annotations editors, wizards and quick fixes, enabling fast development with minimal human error.

A domain-specific modeling language enables you to use a semantically rich visual language that is specific to your business and easily understood by business stakeholders. These specifications are not just imprecise blobs of information. They are documentation that can be used to generate code, automatically update models as changes are made to code, and provide automatically generated documentation that supports compliance and other mandates.

A domain-specific modeling language and an integrated test environment can help your development teams avoid the human interaction that slows down development and increases errors.

It's also important to test for quality at every point so that you can consistently deliver ship-ready code. Integrated test environments enable and encourage independent testing. Static code analysis and application profiling help identify roadblocks at a much earlier stage. Coding guidelines can now be built into the tooling. And teams can test code when it's running to monitor performance of the overall application.

Incremental development—a hallmark of the agile approach—is the key to satisfying the marketplace and beating the competition. Tools enable you to integrate across virtually all stages of the product development lifecycle. Teams can track progress, assignments, and code and configuration changes. An integrated view of the workforce can be taken to efficiently allocate tasks.

Integration solutions help development teams manage complex relationships

Teams often have to express the design of their IT systems in order to provide communications across departmental boundaries, including between development and operations. Conventional methods—e-mail, spreadsheets, slides, diagrams—lack the methodology needed to communicate this design, as well as a system to validate the communication. This leads to a variety of configuration issues that are seen throughout the application lifecycle, with unchecked deployments made in the target environments. This gap worsens when a solution moves between different stages of the application lifecycle.

Highlights

Rational Software Architect features a deployment architecture platform that helps developers visually understand IT systems and their relationships.

The IBM Rational Software Architect solution provides a powerful deployment architecture platform to design and understand IT systems and their complex relationships visually. This platform is built on an extensible, strongly typed model that provides a framework for describing practically any IT system.

IDEs cut time to marketplace for high-quality applications

Software testers have a particular problem with context switching. A tester may have to use a static document with a representation of the model on it. After using the model to author the code by hand, the tester must then build and export it. Then the code is deployed to a test machine, and the server is restarted. If a bug is discovered, the tester needs to go about the whole cycle again. This process can occur several times, and—when coupled with code reviews by other people—uses up huge amounts of time and results in suggestions of massive changes, which often means schedules are extended or poor-quality applications are shipped.

But a do-it-all integrated development environment (IDE) helps generate the code foundation for the developer while providing realtime analysis and suggestions on improving code. The tooling also automatically builds the code, deploys it on a self-contained test server and provides statistics on how well the application is performing. By eliminating the need to switch between development and testing contexts, an IDE allows the developer to improve quality and constantly test the application without having to use any other resources.

Highlights

Secret 3—It's possible to leverage far-ranging expertise and still maintain a fast development pace

Collaboration among development teams spread around the world is increasingly important. Traditionally, if developers needed help from their teammates they simply grabbed them in the office and started working on the same problem together. This kind of collaboration is no longer possible when teammates are sitting in offices halfway around the world.

Collaboration is essential for effective software delivery, so a traditional agile environment needs to be augmented to support collaboration within large teams.

Teams need to share everything from requirements to the code that's being written at all stages of development. They also need to maintain traceability from requirements to code so that they can accurately understand the impact of change, especially in this dynamic process.

An agile development environment makes this even more challenging since its iterations have much shorter cycles than traditional methodologies. Agile teams need better reporting, traceability and collaborative debugging.

Debugging code in a distributed environment is particularly challenging. Complex applications are broken into components. Team members are often focused on just one component of a larger application. When required to debug unfamiliar code, they often have to begin at the beginning. This means they have to set up the environment, reproduce the problem and start the whole process over again for a new problem. Global development—in different time zones—makes this process even more inefficient. Common communications media—like e-mail—are disjointed.

Modern development tools support widespread collaboration by integrating all aspects of the development process.

Fortunately, modern development tools, based on open platforms, help integrate all aspects of the development process using a common server. An IDE can support everything from requirements management and requirements composition to test management and integrated development. Team members

Highlights

can easily communicate with one another and share their progress. They can determine what tasks are involved, see where the code exists, and have an integrated view of defect and requirements management.

In fact, field experience has shown that development teams using high-level implementation features for agile processes can cut development and test cycles by 60 percent just by integrating the test environment with the development environment.[2]

Team Debug allows developers to establish a debug session on the IBM Jazz server, which in turn allows another developer anywhere else in the world to use Rational Team Concert to pick up the session.

Team Debug enables collaborative debugging without having to reproduce the problem

Many developers have found themselves in situations where they are trying to debug the same problem for days. This is especially problematic when developers have little expertise in an area they're called upon to debug, or they're trying to learn the code and perform debugging at the same time.

A new capability from IBM, Team Debug helps the developer establish a debug session in the IBM Rational Application Developer solution, adding the debug session to the IBM Jazz™ server, which manages it from that point on. As long as this debug connection between the Jazz server and the developer is not lost, Team Debug can help the developer transfer the session or suspend it temporarily.

Once the developer reaches the point where he or she needs help, IBM Rational Team Concert™ software provides a way to look up a team member and transfer the session. Alternatively, the developer can drag and drop the debug session to the chat window using an instant messaging tool. Once suspended in its current state, the debugging session will not advance until someone restarts the session.

IBM Rational solution	Function	URL
IBM Rational Application Developer for WebSphere® Software	Helps Java™ developers rapidly design, develop, assemble, test, profile and deploy high-quality Java/Java Platform, Enterprise Edition; portal; Web/Web 2.0; Web services; and service-oriented architecture (SOA) applications	**ibm.com**/software/awdtools/developer/application
IBM Rational Software Architect for WebSphere Software	Provides an integrated platform that facilitates communication and collaboration, helping development teams innovate and accelerate delivery of software projects	**ibm.com**/software/awdtools/swarchitect/websphere
IBM Rational Team Concert	Provides a collaborative software delivery environment that empowers project teams to simplify, automate and govern software delivery	**ibm.com**/software/awdtools/rtc
IBM Rational Method Composer	Provides a flexible process management platform with one of the industry's most comprehensive tooling sets and a rich process library to help companies implement effective processes for successful software and IT projects	**ibm.com**/software/awdtools/rmc
IBM Rational Asset Manager	Helps create, modify, govern, find and reuse virtually any type of development assets, including SOA and systems development assets	**ibm.com**/software/awdtools/ram

IBM has developed the right solution to help you adopt agile development successfully.

Why IBM?

IBM provides technology, best practices and industry expertise to help companies succeed in an agile development environment—regardless of their size and complexities.

IBM Rational software can help you:

- *Bring together critical capabilities for better quality and faster, iterative development with end-to-end automation.*
- *Empower distributed agile teams with realtime information and automatic handoffs that keep teams in sync.*
- *Manage tests, defects and project progress through a realtime, single project view.*
- *Increase developer productivity with faster software iterations and the ability to validate quality from individual IDEs to increase the frequency of code check-ins and accelerate troubleshooting.*
- *Make compliance management painless by embedding documentation into everyday work.*

For more information

To learn more about how IBM can help you get the most from agile software development, contact your IBM representative or IBM Business Partner, or visit:

ibm.com/rational/agile

© Copyright IBM Corporation 2009

IBM Corporation
Software Group
Route 100
Somers, NY 10589
U.S.A.

Produced in the United States of America
January 2009
All Rights Reserved

IBM, the IBM logo, ibm.com, and Rational are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at **ibm.com**/legal/copytrade.shtml

Java and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

References in this publication to IBM products or services do not imply that IBM intends to make them available in all countries in which IBM operates.

The information contained in this document is provided for informational purposes only and provided "as is" without warranty of any kind, express or implied. In addition, this information is based on IBM's current product plans and strategy, which are subject to change by IBM without notice. Without limiting the foregoing, all statements regarding IBM future direction or intent are subject to change or withdrawal without notice and represent goals and objectives only. Nothing contained in this documentation is intended to, nor shall have the effect of, creating any warranties or representations from IBM (or its suppliers or licensors), or altering the terms and conditions of the applicable license agreement governing the use of IBM software.

IBM customers are responsible for ensuring their own compliance with legal requirements. It is the customer's sole responsibility to obtain advice of competent legal counsel as to the identification and interpretation of any relevant laws and regulatory requirements that may affect the customer's business and any actions the customer may need to take to comply with such laws.

1 Dr. Dobb's Portal, *Has Agile Peaked?*, Scott W. Ambler, May 7, 2008.

2 IBM Rational Community of Practice.

RAW14101-USEN-00