Managing Project Risk

Best Practices and Tool Integration in Software Development

Managing Project Risk

BEST PRACTICES AND TOOL INTEGRATION IN SOFTWARE DEVELOPMENT

Rational, the Rational logo, Rational Unified Process, RUP, Rational Developer Network, SoDA, TestFactory, ClearCase, ClearQuest, Rational Rose, RequisitePro, Rational Suite, AnalystStudio, TestStudio, and Rational Suite ContentStudio, are trademarks, registered trademarks or services marks of Rational Software Corporation in the United States and/or other countries. All other names are used or identification purposes only, and are trademarks or registered trademarks of their respective companies.

All rights reserved. Made in the U.S.A.
© Copyright 2002 by Rational Software Corporation.
TO800; rev. 8/02. Subject to change without notice.

Managing Project Risk

BEST PRACTICES AND TOOL INTEGRATION IN SOFTWARE DEVELOPMENT

Developing software can be a risky business. For experienced project managers, this is nothing new; for some time now, effectively managing risk in software development has been recognized as a key factor in delivering software successfully. But even dramatic advancements in software development techniques, architectures, and platforms proven to be effective in reducing risk for teams of all sizes and tools have not stopped many projects from being cancelled, or delivered late, or from running over budget. The complexity of the applications being developed has outpaced the resources and abilities of many software development teams. As a result, only a small fraction of software development projects are completed on time and within budget. Project managers and team leaders — from seasoned veterans to those spearheading a project for the first time — are focusing closely on reducing risk in software development.

Through years of close work with thousands of development organizations, Rational Software has developed a comprehensive solution that has proven to be effective in reducing risk for development teams of all sizes. This solution is based on best practices of software engineering — including an iterative development process — and a tightly integrated set of development tools that helps teams improve communication and leverage assets throughout the entire project lifecycle.

We've prepared this guide for project managers and team leaders who understand the importance of minimizing risk and want to help their teams realize their full potential. The content demonstrates the value of integrated tools in risk management and guides you step-by-step through a sample project lifecycle. This guide will help you understand how Rational's process, award-winning tools, and key tool integrations can work in concert to help you and your team decrease risk, increase predictability, and ensure that your next project is delivered on schedule.

If you have questions or comments, or if you would like to arrange a demonstration or evaluation of Rational's solutions, please visit us on the Web at www.rational.com.

Contents

Managing Project Risk

BEST PRACTICES AND TOOL INTEGRATION IN SOFTWARE DEVELOPMENT

Iterative Software Development: Managing Project Risk and Uncertainty

Today, software projects are initiated with significant levels of uncertainty:

1. When will code and content be stable?
2. What runtime failures will be uncovered?
3. How do project managers make accurate project progress forecasts to stakeholders and the team?

Because of this uncertainty, it is nearly impossible for project managers to adhere to a predictable project schedule. Yet, successful managers have been able to consistently guide their teams to complete projects on time by adopting proven techniques for effectively dealing with project uncertainty — by following best practices and using integrated tools.

The classic waterfall approach to software development — in which the project proceeds sequentially from requirements analysis through design, coding and testing — postpones addressing risks until late in the project, when it is much more costly to make changes and fix problems introduced in earlier phases. In the search for more predictable projects and risk reduction, many organizations have found the waterfall approach unworkable.

From an overall business perspective, the success of many organizations is becoming increasingly dependent on the success or failure of the software they build — regardless of whether it is intended to be packaged and sold, to be used internally, or to drive business transactions. In this environment, managing risk is not only a sound development practice, but also a vital business practice.

What Is Risk in Software Development?

Software development teams encounter many different kinds of risks. By definition, risk involves an exposure to potential hazards. It is uncertainty that underlies risk. Factoring in this uncertainty and examining the unknown and ill-defined is accomplished by creating and prioritizing a set of risks based on what is known about the project. For example, after an early risk assessment you might conclude:

Risk 1: Requirements evolve over a project. Project scope creep is inevitable

Risk 2: We have no straightforward way of integrating our developments assets. There's lots of manual rework

Risk 3: We know that we will be using XML extensively, but we have only one developer with XML experience

With little effort, you can probably think of a few important risks that your current software development project is facing. Of course, the order of the items on the risk list, as well as the items themselves, will change over time. The important thing is to take steps to address risk and uncertainty as soon as possible, before you spend a lot of time, effort, and money on a flawed approach.

Reducing Risk Through Best Practices

In contrast to the waterfall approach to software development, an iterative approach seeks to identify project risks — both technical and business-related — early in the lifecycle, when it is possible to address them most efficiently. The iterative

development process is driven by continuous discovery, invention, and implementation, during which the project team completes development of project artifacts in a predictable and repeatable way, iteration by iteration. An iterative process enables your team to mitigate risks earlier because it unearths and tests for problems earlier. As you progress through the initial iterations, you exercise many aspects of the project, including tools, off-the-shelf software, your process, and the skill sets of individuals on your team.

The main advantage of iterative software development is that it increases the predictability of the schedule, the final product and the entire process. As an additional dividend, it can also be expected to produce higher quality products — that satisfy the real needs of end-users — because the requirements can evolve along with the design and the implementation. In fact, developing software iteratively is one of six best practices of software engineering that are commonly used throughout the industry by successful organizations. This time-tested approach to software development is embodied in the Rational Unified Process®, Rational's comprehensive framework for software development. It is a cornerstone of Rational's complete solution, which combines best practices, unified tools, and services that accelerate implementation.

Rational Software Best Practices

- *Develop Iteratively* – to identify and eliminate risks before they threaten your project.
- *Manage Requirements* – to ensure resilience in the face of inevitable change.
- *Use Component Architectures* – to make your architecture tangible to all practitioners.
- *Model Visually* – to attain and preserve a high-quality architecture.
- *Continuously Verify Quality* – to ensure quality throughout the development life cycle.
- *Manage Change* – to enable efficient parallel development within teams and across the enterprise.

Integrated Tools Pave the Way

Adopting an iterative development approach does not necessarily mean the project manager will have an easier job. In some situations, iterative development requires more careful planning and will likely place an additional burden on project managers. But the benefits of iterative development far outweigh the costs, especially when the development team is equipped with a set of tightly integrated, team-based tools that support the entire lifecycle. With Rational's integrated tools, software development artifacts, such as requirements specifications, architectural and design models, and test cases, can be leveraged and reused repeatedly, and all team members have clearer, more reliable mechanisms for conveying and understanding project details. Integrated tools help teams of analysts, developers, and testers work closely together, by providing an infrastructure that allows each team member access to common project requirements, defects, test cases, and more. In addition, tool integration increases the return on investment by enabling round-trip engineering and synchronization of requirements, design elements, and test artifacts. In short, Rational's broad array of integrated tools work together to minimize project risk and ensure successful project delivery in several vitally important areas:

- Increased productivity and efficiency
- Faster development cycles and improved ability to meet deadlines
- Improved team communication
- Increased quality
- Simplified project management