Requirements definition and management
White paper
October 2009

Rational® software

Getting requirements right: avoiding the top 10 traps.

Contents

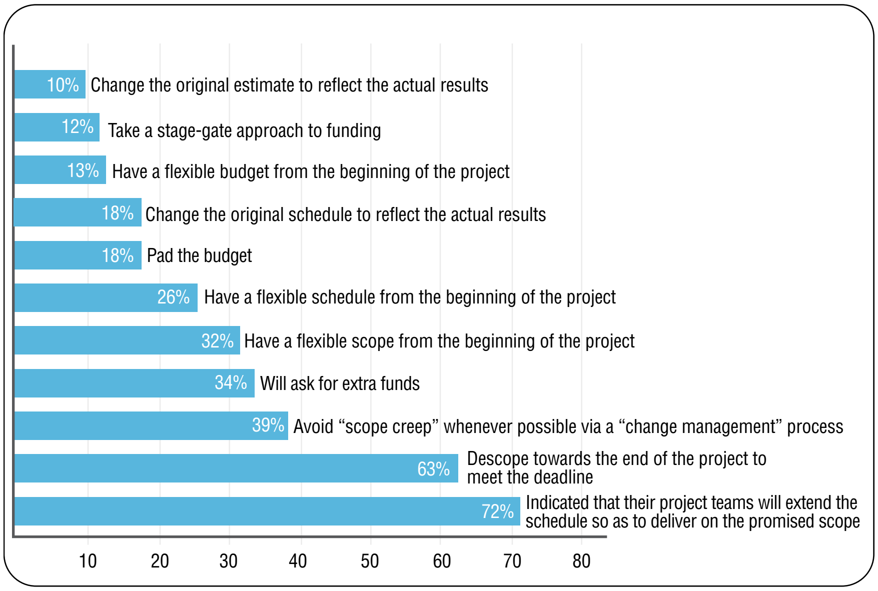
Don’t get caught

A trap is a position or situation from which it is difficult or impossible to escape. Getting caught in software development requirement traps can have dire consequences for the survival of your business. The traps of bad requirements definition and management result in cost overruns, missed deadlines, poorly designed products and, ultimately, a failure to deliver what the customer needs.

Unfortunately, requirement traps are all too common, and their influence is considerable. The State of the IT Union Survey[1] explored the development management practices that teams applied to either stay out of trouble or address problems after they are discovered. The online survey of respondents reveals how often development teams anticipate that they’ll fall into traps. They pad budgets and schedules. They change initial estimates to match actual results. And they request additional resources. Figure 1 reveals the lengths teams and team leaders will go to deal with these traps.

Highlights

State of the IT Union Survey[2]



Development teams anticipate the traps of requirements management and intentionally subvert the creation of accurate budgets and timelines.

Poor requirements management results in inconsistent project outcomes.

Products today increasingly depend on software to deliver their value. A brake sensor on a car, a home appliance, a medical device—these products all contain software that plays an increasingly important role. The products are becoming smarter, and they involve more people and teams outside the typical value chain. Smarter management of the requirements process—the foundation of effective software delivery—is increasingly important for delivering these smart products.

This white paper presents 10 common and devastating mistakes that project teams make in defining and managing requirements. More important, it discusses how to avoid these traps so you can get your requirements right and develop the right product on time and within budget.

Highlights

Trap 1: scope creep

Scope creep usually involves features being added to previously approved product designs without corresponding increases in resources, schedule or budget. The potential causes vary depending on the organization, the project and the individuals, but creep tends to occur when project requirements are not defined and managed properly.

Capturing business requirements can control scope creep.

Uncontrolled growth can also occur when development teams create solutions before determining what the business really needs. The business requirements of a project are typically seen by development teams as being too high level and vague and not applicable to them. They want to focus on detailed product requirements. However, business requirements are real requirements, and they need to be sufficiently detailed to avoid scope creep. By meeting the product requirements and not the business ones, teams fail to develop solutions that provide value and will overcompensate by adding features.

Avoid the trap

- *Map detailed, real business requirements to product features that satisfy those requirements and provide value.*
- *While accommodating change, be vigilant about focusing on business requirements and reprioritization.*
- *Identify and work with stakeholders early and often to understand the business requirements, stakeholder priorities and the effect on stakeholders when changes occur.*

The payoff

Avoiding scope creep makes planning easier, helps keep budgets intact and helps keep projects on schedule. Most important of all, it helps generate desired return on investment (ROI).

Highlights

Guide discussions away from focusing on features.

Asking customers about outcomes helps determine requirements.

Trap 2: asking customers what they want

This seems counterintuitive. Teams are supposed to talk to customers and give them what they want in a product. But customers tend to talk about features, not what they truly need. The truth is that people often don't know what they want. And when customers don't know what they want and developers don't understand the problem, poorly conceived solutions—and products—can result.

Avoid the trap

- *Ask customers why they need a particular solution. It may lead to a better understanding of their expectations and better discussions about specific requirements.*
- *Guide the discussion away from focusing on features—ask customers what they want the software to do. Create a separate discussion for how resulting products will be used.*
- *Identify the right stakeholders. The target audience or end user may not be the person who is responsible for the project or invested in its outcome. Find—and listen to—the right mix of users, customers, executives who fund the project, government representatives who impose regulations, project teams, support teams and others.*

The payoff

Asking customers what outcome they are seeking helps you determine the true and realistic product requirements that will deliver value to them.

Highlights

Uncertainties and outside factors may influence requirements.

Plan for change in order to reduce risks.

Trap 3: inability to adapt to change

Setting requirements in stone early in development can be a recipe for disaster that results in an enormous waste in resources and an inability to stay on schedule. As recent changes in the global economy illustrate, outside influences can quickly change project requirements. Organizations have quickly changed their focus to doing more with less. Speculative or experimental pet projects are off the table, replaced by those with strong potential ROI. Yet in any economy, there are always new requirements to contend with as business priorities change, new government regulations are enacted and new stakeholders are identified. Project teams need to accommodate those changes.

Avoid the trap

- *Expect and plan for requirements that change throughout your development process.*
- *Reprioritize requirements based on shifting circumstances such as business need, customer importance, estimated effort and cost.*
- *Have a fine-grain plan that you adjust at regular intervals.*
- *Keep your stakeholders informed as changes occur—get their input for prioritization and the rationale behind it.*

The payoff

Accommodating and planning for change in project requirements helps mitigate risk and decreases costly rework.

Highlights

Communicate with stakeholders in ways they can easily understand.

Effective communication helps optimize time and effort.

Trap 4: failure to communicate effectively

Ineffective communication is often a root cause of project failure. The perspective of development teams, customers, end users and executives is different for each group, as are their needs for communication. If you don't express requirements using methods your stakeholders can easily understand, you can't possibly gain consensus on requirements.

Avoid the trap

- *Know your audience and communicate in ways that help them understand the information. Make use of diagrams, user stories, sketching and storyboards.*
- *Create glossaries, document templates and feedback forms that are clear, concise and easy to use.*
- *Use prototyping to help stakeholders visualize the solution. This can either augment text or completely replace it depending on the level of detail required.*
- *Elicit feedback from all of your stakeholder representatives, and remember that one or two people tend to be the most vocal. Don't make the mistake of overlooking others' feedback.*
- *Always respond to feedback, preferably with some clear statement of status such as, "Will incorporate," "Placed this on a wish list" or "Unable to accommodate this now." If you ask for input, acknowledge it.*

The payoff

Effective communication makes the most efficient use of everyone's valuable time and helps avoid misunderstandings that derail projects.

Highlights

Stakeholders often have differing priorities.

Regular communication helps deliver product value.

Trap 5: failure to communicate frequently

Failure to communicate with stakeholders early and often leads to one primary problem: rework. Developing a product for customers without consulting them while that product is being developed is just asking for trouble. The biggest reason it happens is that we often think we know what our customers want well enough that we don't need to consult them. And stakeholders usually have different priorities.

Sometimes teams don't communicate with stakeholders because they prefer to avoid confrontations. But if you want a positive end result and minimum risk and rework, it is important to collaborate with stakeholders not just at the beginning of a project but throughout the entire process, from start to finish.

Avoid the trap

- *Identify key stakeholders, including customers. Choose a representative from each group to communicate with regularly.*
- *For large, in-person stakeholder workshops, consider using a skilled facilitator. Good facilitators are worth their weight in gold in keeping everyone on track for attaining the meeting objectives.*
- *Establish regular checkpoints with your stakeholders. Determine at the beginning of the project how often you need to check in, and also schedule time for keeping stakeholders up-to-date as unexpected changes occur.*
- *Make it easy for your stakeholders to provide feedback. When they do, let others see their feedback to generate better discussions.*

The payoff

Regular communication reduces risks, increases team productivity and avoids rework. Ultimately it helps deliver the product the customer really wants.

Highlights

Trap 6: unwieldy documents and too much information

Do you have time to review and give feedback on a 200-page document? Probably not—and most likely, neither do your stakeholders. Doing more work than necessary and adding unnecessary detail to documentation costs both time and money. It is unproductive for the person creating it and a hindrance to the people looking for information they need to get their work done. Two common missteps include adding too much detail to requirements too early in the process and requiring more traceability than is necessary to facilitate effective lifecycle management.

When it comes to documentation, think quality, not quantity.

Think quality, not quantity. Better to add just enough detail to your requirements and identify just enough traceability to get the job done—not the entire job, but the next piece or iteration that needs to be completed.

Avoid the trap

- *Large, dense documents are not very consumable by stakeholders. Invest in communication tools that efficiently gather and disseminate requirements information.*
- *Use visual techniques to model business and product requirements. Business process diagrams and use case diagrams, storyboards and sketches can help cut through text-heavy clutter.*
- *Provide a glossary of industry terms, acronyms and domain-specific terms to facilitate communication.*

Give users glossaries to help avoid confusion.

- *Create transparency of feedback. When your stakeholders are able to review each other's feedback, the discussion is richer, problems come to light, missing requirements are identified and necessary details get filled in.*
- *Add just enough information to your documents so the rest of your team members can complete their work.*
- *Remember that requirements management is an ongoing process. There will be other opportunities to add more detail to requirements and to capture more traceability later when it may be more appropriate.*

Highlights

Focused documentation and feedback helps increase efficiency and accuracy.

The payoff

Focused documentation and feedback loops increase the efficiency of all stakeholders. Reduction of extraneous details in both requirements and traceability increases quality by focusing on the most important information—while postponing further detail until it's needed.

Highlights

Trap 7: hidden project artifacts

Developers and testers often aren't aware of the project vision or can't locate the documentation for the architecture or the business requirements—that is, if they're created at all. Without easy access to these foundational documents, how can we possibly expect them to deliver models, code and tests that solve the right customer problems? We can't. Transparency to all project artifacts is critical to the success of any software project.

Avoid the trap

Keep a central repository for project artifacts.

- *Keep all project artifacts in a central repository that is accessible by project team members. Having to search multiple sources for relevant documents is frustrating and time-consuming.*
- *Make sure documents are categorized and managed in such a way that they are easy to find.*
- *Ensure that when changes occur, the team is informed. Automatic notifications can help deliver this information.*

The payoff

Centralized management of information can facilitate collaboration.

Accessibility and management of information and transparency of project artifacts reduces rework, diminishes waste and promotes reuse because it makes collaboration and communication easier.

Highlights

Trap 8: ambiguous requirements

Ambiguous requirements are the result of unclear or missing critical information. This leads to confusion and rework. Project teams spend too much time trying to get clarification so they can design, code and test. It's very difficult for architects to develop relevant models, developers to write defect-free code and testers to develop the right test cases without clear requirements. Unfortunately, reworking requirements is so common that it can become an accepted practice—rework is just built into the schedule and budget.

Ambiguity pushes risks onto the next phase of projects.

Ambiguity also tends to push risks into the next phase of the project. Requirements then have to be reworked, and that poses problems to project schedule and cost. This trap is especially damaging to fixed-cost projects.

What causes ambiguity? Poor writing, inaccurate information and the assumption that the audience understands just as much as you do.

Avoid the trap

- *Use a writing reference, such as Elements of Style by William Strunk Jr. and E.B. White—which has long been considered the authoritative reference on writing crisp, clear text—before and during requirements creation.*
- *Create a glossary, and make sure every acronym and technical term is included.*
- *Pretend you are writing for a student just out of college or someone who recently joined the company. Don't assume the audience will know every term and understand every concept.*
- *Augment text with visuals. It's a great way to express both simple and complex concepts.*
- *Step back. After writing any draft document, put it down for a while and read it later. Fresh perspective can reveal ambiguities.*

Set aside a first draft and come back to it later with a fresh perspective.

The payoff

Clear, understandable requirements are the foundation of your software project.

Highlights

Measuring outcomes allows for continual process improvements.

Trap 9: failure to measure and assess requirements processes

Defining and managing requirements can be a complex task. Missing and ambiguous requirements can easily result in missed schedules, cost overruns and decreased productivity and quality as downstream project deliverables fail to provide value to stakeholders. Don't wait for disaster to strike—assess your project status on a regular basis.

Organizations must have the ability to review, assess and improve their requirements process. Having accurate insight into data, processes and practices is a key component of success. Measuring project and process outcomes allows for continual process improvements across the software delivery lifecycle, which reduces project failures and lowers business costs.

Avoid the trap

- *As part of your process, conduct a "lessons learned" feedback session at the end of each development iteration or release.*
- *Also do a "lessons remembered" session before starting a new project. To encourage continual improvement, you need to not only capture lessons learned at the end of an iteration or release but also reinforce those lessons as you move forward.*
- *Define and collect metrics that ensure your success. For example, measure the impact of changes to your requirements, test case coverage, priority, cost and effort of business, and product requirements. As you become more experienced with measurement, you'll find just the right combination of metrics that allows you to continually improve your requirements process.*

The payoff

Ongoing measurement of project performance reveals small problems before they become big issues.

Highlights

Trap 10: isolating your requirements

Viewing requirements as isolated entities, failing to capture relationships between requirements and other artifacts, and failing to recognize dependencies between requirements leads to increased project risk and rework. Reprioritizing one requirement without considering its effect on other requirements results in increased project risk and costs.

Changes in one requirement may have significant downstream impact.

For example, a risky trap that organizations often succumb to is not capturing the relationship between project requirements, business requirements and other downstream deliverables such as models, test cases and defects. When you fall into this trap, you deliver a product that doesn't satisfy stakeholder needs.

Avoid the trap

- *Identify relationships between requirements and then manage them together.*
- *Create the right level of traceability between requirements and downstream deliverables that balances the traceability needed for effective lifecycle management with support for productivity.*
- *As you make changes to requirements and reprioritize them, consider the effect of these changes to related requirements and your downstream deliverables.*
- *Use tools that allow you to easily visualize the relationships you've identified.*

Identifying and managing relationships between requirements helps lower risk.

The payoff

Identifying and managing relationships between requirements and other artifacts mitigates project risk; helps ensure alignment between your business requirements, product requirements and downstream deliverables; and results in lower development costs.

Highlights

Conclusion

There are many traps in software. Some of the most expensive ones occur in the requirements space because that is where the foundation for your entire project is laid. Lack of planning, lack of communication and collaboration with stakeholders, ineffective requirements elicitation, and requirement management techniques all lead to problems that can paralyze your software project. When we don't measure how we're doing and continually make improvements, the risk escalates quickly and the project gets out of control—something you may never recover from.

IBM Rational software requirements solutions can help you avoid common software development traps.

IBM Rational® software requirements solutions incorporate many best practices that help you avoid these common traps and enjoy the payoffs.

IBM Rational Requirements Composer software, for example, facilitates team collaboration. It is a requirements definition and management solution, perfect for teams that want to focus on business and product requirements definition and only require basic requirements management. The software supports textual and visual techniques designed to elicit, elaborate and validate requirements in a collaborative manner with stakeholders. It also provides capabilities for managing complex requirements information and as well as some process guidance.

IBM Rational RequisitePro® software in the IT space and IBM Rational DOORS® software in the system space provide traceability and impact analysis that keeps analysts, architects, developers and testers aligned to business needs and requirements throughout the software delivery lifecycle.

Becoming familiar with the traps identified here, recognizing them within your organization and avoiding them—preferably at the beginning of your software projects—will help you not only survive in business but also go on to increase marketshare and ROI as you serve the needs of your customers and other stakeholders; help your project delivery teams increase their productivity, quality and reuse; and enable innovation.

For more information

To learn more about IBM Rational requirement solutions for software development, contact your IBM representative or IBM Business Partner, or visit:

ibm.com/software/rational/offerings/irm

Rational Requirements Composer software:

ibm.com/software/awdtools/rrc/

Rational Requirements Composer free trial download:

https://jazz.net/projects/rational-requirements-composer/

Software delivery best practices e-kit: Do more with less:

ibm.com/services/forms/preLogin.do?lang=en_US&source=swg-rtl_sdekit

Top 10 software delivery secrets to doing more with less:

ftp://ftp.software.ibm.com/common/ssi/sa/wh/n/raw14139usen/RAW14139USEN.PDF

ROI calculators:

ibm.com/software/rational/offerings/testing/roi/

Read our RDM blog:

rationalrdm.wordpress.com

Follow us on Twitter: @RationalRDM

Become a Rational Requirements Composer Facebook fan:

http://www.facebook.com/home.php?#/pages/Rational-Requirements-Composer/47378244431?ref=ts

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

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

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 documentation is provided for informational purposes only. While efforts were made to verify the completeness and accuracy of the information contained in this documentation, it is 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. IBM shall not be responsible for any damages arising out of the use of, or otherwise related to, this documentation or any other documentation. 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, 2] Scott W. Ambler, “State of the IT Union Survey,” *Ambysoft*, July 2009, http://www.ambysoft.com/surveys/stateOfITUnion200907.html