jama build great products™

THE BUSINESS VALUE OF BETTER REQUIREMENTS

REQUIREMENTS EXPERT KARL WIEGERS EXPLAINS THE BENEFITS OF REQUIREMENTS

Not every manager is convinced that his team needs to do a better job on requirements development and management, or that such an investment will pay off. Numerous industry studies, however, indicate that requirements issues are a pervasive cause of project distress. The often-quoted CHAOS Reports from The Standish Group indicate that three of the biggest contributors to projects that fail or are "challenged" are lack of user input, incomplete requirements and specifications, and changing requirements and specifications. This white paper describes the business value an organization can achieve from investing in developing better requirements for its software projects, as well as how to think about the return on investment from that effort.

[1]Adapted from Karl E. Wiegers, **Testing the Requirements,** Microsoft Press, 2006.

AN ECONOMIC ARGUMENT

The case for implementing better requirements practices is an economic or business argument, not a philosophical or technical position. Think about how your company's bottom line is affected by requirements-related problems. Then use that understanding to justify investing in better requirements practices that can pay off for the long term.

A recent case study of requirements failure was the Federal Bureau of Investigation's new case management software system, called VCF. This project was abandoned after $170 million had been spent because the delivered software was full of defects and off-target functionality. As one investigator wrote:

> I suspect what happened with the VCF is that in the rush to put in place a system, you think you got your requirements nailed, but you really don't. It was a classic case of not getting the requirements sufficiently defined in terms of completeness and correctness from the beginning. And so it required a continuous redefinition of requirements that had a cascading effect on what had already been designed and produced. (Goldstein, Harry. 2005. "Who Killed the Virtual Case File?" IEEE Spectrum 42(9): 24–35).

Numerous studies have examined the effects of errors in requirements on software projects. They consistently find that nearly half of the discovered defects originated as requirement errors. The typical outcome of errors in the requirements is an expectation gap, a difference between what developers build and what customers really need. Clearly, any domain that is the root cause of approximately half of the problems on software projects deserves our attention.

The main reason errors in requirements are so damaging is that they force the development team to perform extensive rework to correct the errors. It's well-established that the cost of correcting a software error increases dramatically the later it is discovered, as shown in Table 1. An error, omission, or misunderstanding in the requirements forces developers to redo all the work they've already done based on the incorrect requirement. Therefore, any technique that can reduce requirement defects and prevent some of this wasted effort is a high-leverage investment indeed. One analysis of the potential return on investment from better requirements suggested that requirement errors can consume between 70 and 85 percent of all project rework costs.

Table 1: Relative Cost to Correct a Requirement Error

STAGE WHEN ERROR IS DISCOVERED	RELATIVE COST TO CORRECT
Requirements development	**1x**
Design	**2-3x**
Construction	**5-10x**
System or Acceptance Test	**8-20x**
Operation	**68-110x**

WHAT BETTER REQUIREMENTS CAN DO FOR YOU

In addition to avoiding some of the negative consequences described above, better software requirements provide numerous benefits. These include selecting the right projects to fund, facilitating estimation, enabling rational prioritization, developing higher quality designs, and testing more effectively.

1. Selecting Projects to Fund

Good preliminary requirements enable senior managers to make effective business decisions as organizations decide which among a set of potential projects to fund. Better requirements allow more accurate projection of business returns. Once a project is funded, better requirements allow project managers to more sensibly partition tasks among their teams and even among individual team members.

2. Facilitating Estimation

Well-understood requirements can help your team estimate the effort and resources needed to execute a project. Reliable estimation requires some historical correlation between requirements size and effort.

3. Enabling Prioritization

Documented requirements allow the team to prioritize its remaining work. Most projects need to make compromises to ensure that they implement the most critical and most timely functionality. A prioritized requirements baseline helps the team incorporate those changes that will deliver the maximum customer value. One study revealed that just 54 percent of the originally defined features were delivered on an average project. If you can't implement all of the requested functionality, make sure the team implements the right portion.

4. Developing Designs

Requirements are the foundation for design. Therefore, well understood and well-communicated requirements help developers devise the most appropriate solution to the problem. High-quality requirements also ensure that the development team works on the right problem. Many developers have experienced the frustration of implementing functionality that someone swore they needed, only to find that no one ever used it. One survey indicated that fully 45 percent of the delivered software product features were never used. Wasting less time implementing the wrong functionality accelerates the project and maximizes its business return.

5. Testing Effectively

Well-defined and testable requirements allow testers to develop accurate test procedures to verify the functionality. Prioritizing requirements tells testers which ones to concentrate on first. Assessing requirement difficulty and risk helps testers know which functionality should receive the closest scrutiny.

6. Tracking Project Status

A comprehensive, traced set of requirements helps the stakeholders know when the project is done. A body of work is complete when all of the requirements allocated to it are either verified as being correctly implemented in the product or deleted from the baseline. Defined business requirements also allow the stakeholders to determine whether the project has met its goals.

7. Accelerating Development

Believe it or not, investing more effort in developing the requirements can actually accelerate software development. This seems counterintuitive, but it's true. Defining business requirements—the expected business outcomes the product will provide—aligns the stakeholders with shared vision, goals, and expectations. Effective user involvement in establishing the requirements reduces the chance that users will reject the new system upon delivery. Following are some published illustrations.

- In a study of 15 banking and telecommunications projects, the most successful projects spent 28 percent of their resources on requirements engineering, whereas the average project in the study devoted just 15.7 percent of its effort to requirements.
- Increasing the fraction of the total budget devoted to requirements on a group of NASA projects led to substantially lower overrun of both cost and schedules; see Table 2 (Hooks, Ivy F., and Kristin A. Farry. 2001. Customer-Centered Products: Creating Successful Products Through Smart Requirements Management. New York: AMACOM).
- In a European survey, the fastest project teams spent about twice as much of their schedule (17 percent versus nine percent) and effort (14 percent versus seven percent) on requirements activities as the slower teams.

Table 2: Cost and Schedule Overruns on Some NASA Projects

% OF BUDGET SPEND ON REQUIREMENTS	NUMBER OF PROJECTS	AVERAGE PROJECT COST OVERTURN
< 5%	**5**	**125%**
5 to 10%	**7**	**83%**
> 10%	**6**	**30%**

Accurate requirements ensure that the functionality built will let users perform their essential business tasks. The requirements also establish achievable quality expectations. This lets the team implement both the capabilities and the product characteristics—the nonfunctional requirements—that will make users happy. Additionally, emphasizing requirements development is cheaper than relying on beta testing to find requirements problems. Fixing problems that late in the game is far costlier than correcting them earlier.

THE RETURN ON INVESTMENT FROM BETTER REQUIREMENTS

Managers often ask me what return on investment (ROI) they can expect from the money they spend on training, process improvement, and tools for requirements engineering. I can't give them a nice, tidy answer. As with so many questions in software, the correct answer is, "It depends." However, we can explore some of the factors that contribute to determining what ROI an organization can expect from better requirements.

The Investment

If you want to determine the ROI from any activity, you need to track both what you invested in the activity and the benefits—reduced costs, accelerated schedules, increased sales, whatever—you enjoyed as a result. Unfortunately, few software organizations collect this sort of data. It's not hard to track the money and time your organization spends developing improved requirements. Measuring the payback is trickier.

Following are some of the actions you might take to improve your requirements processes and hence the product requirements themselves. Track what you spend on these activities to determine your investment.

Assessing Current Practices

All process improvement should begin with some kind of appraisal. You need to learn how your teams are dealing with their requirements issues today and how those current approaches do and do not yield the desired results. You might begin with the "Current Requirements Practice Self-Assessment," available at: www.processimpact.com/goodies.shtml.

Developing New Processes & Templates

Once you've identified specific requirements practices that are ripe for improvement, your teams need to devise processes that will work better for them. This might involve writing entirely new processes, modifying current processes, and selecting templates for your key requirements deliverables. You can start with the sample templates for a vision and scope document, use case document, and software requirements specification available from www.processimpact.com/goodies.shtml. Adapt these to meet your project needs as appropriate.

Training the team

It's not reasonable to expect your team members to work in new ways if they haven't been taught how to perform the new practices. All business analysts and others who must deal with requirements should receive some basic training in requirements engineering concepts and practices. The team members also need guidance in the effective use of your own processes and templates.

Acquiring Books & Other Resources

Requirements engineering is a complex domain with many concepts and practices. Your BAs will benefit from having reference books and articles at hand so they can refresh their knowledge and get ideas for handling new challenges they encounter. Books are a high-leverage investment, assuming that team members actually read them and apply what they learn.

Employing external consultants

Some software organizations pursue improved requirements approaches on their own. Others prefer to acquire assistance from experienced consultants who have worked with a variety of companies. Consultants aren't cheap, but they can help your team members solve problems much faster than they might solve those problems on their own.

Buying requirements management tools

Written requirements documents have numerous limitations. As your requirements engineering activities become more sophisticated, you might elect to store requirements in a database rather than in traditional word processing documents. Nearly three dozen commercial requirements management tools are available. These tools make it far easier for you to store attributes that provide a rich understanding of each requirement, to track requirements status, to deal with changes, and to record requirements traceability information. Chapter 23 of my book More about Software Requirements (Microsoft Press, 2006) offers many tips on getting the most from your tool investment.

Of course, the greatest investment you make in developing superior requirements is the time your team members spend eliciting, analyzing, documenting, validating, and managing the requirements for their products. As we saw in the previous section, this is likely to accelerate your project, amply rewarding the additional investment in quality requirements.

The Return

I can't predict what ROI you're going to get from your investment in developing better requirements. There are too many variables, most of which depend on the performance of your own teams. However, I

can help you think through what the payback might be for your organization. The return you can expect to receive depends on the price your projects are currently paying for shortcomings in your requirements. If your teams are forced to perform extensive rework to deal with overlooked or inaccurate requirements, you'll obtain a better return than if requirements issues are just a minor nuisance. Before you dive into requirements process improvement, consider the following questions.

- What fraction of your development effort is expended on rework? (Few software organizations can answer this question. Those that have measured it find that rework can consume 30 to 50 percent of all the effort expended on a software project. Some rework is unavoidable and adds value but much rework constitutes wasted effort.)

- How much does a typical customer-reported defect cost your organization? A system-test defect? (Every organization should know this. To my knowledge, only one of my clients has measured it, though. They spent an average of $4,200 to deal with each customer-reported defect. Contrast this with their average cost of just $200 to discover a defect by inspection.)

- What fraction of user-reported defects, and what fraction of defects discovered through system testing, originate in requirements errors? (Defect root cause analysis is an excellent technique for determining where your leverage lies for quality improvement.)

- How much maintenance cost—defect correction, unplanned enhancements—can be attributed to missed requirements or other types of requirement defects?

- How much do you think you could you shorten your delivery schedules if your project teams could reduce requirement defects by, say, fifty percent?

The goal of software process improvement is to improve the bottom line of your business by lowering the costs of building and maintaining software. Adopting practices that result in fewer requirement defects will reduce the amount of development rework your teams must perform. Performing less rework has a direct payback through reduced product development costs and quicker time to market. Techniques that get your BAs and customers working closer together lead to products that better meet customer needs. Superior requirements development also lowers your maintenance and support costs. Many products have to be modified immediately after release when the customers realize some critical functionality is missing or wrong.

Besides these obvious practical benefits, improving your requirements approaches leads to less tangible outcomes that are valuable but hard to measure. Experiencing fewer miscommunications on a software project reduces the overall level of chaos. Less chaos lowers unpaid overtime, increases team morale, boosts employee retention, and improves the team's chances of delivering on time. And all of the benefits I've described here have the potential of leading to higher customer satisfaction. What is that worth to you? It might be hard to measure, but it's real.

ABOUT KARL WIEGERS

Karl has provided training and consulting services worldwide on many aspects of software development, management and process improvement. He has authored 5 technical books, including Software Requirements, and written more than 175 articles. Prior to starting Process Impact in 1997, he spent 18 years at Eastman Kodak Company. His responsibilities there included experience as a photographic research scientist, software applications developer, software manager, and software process and quality improvement leader. Karl has led process improvement activities in small application development groups, Kodak's Internet development group, and a division of 500 software engineers developing embedded and host-based digital imaging software products. http://www.processimpact.com.

ABOUT JAMA SOFTWARE

From concept to launch, the Jama product delivery platform helps companies bring complex products to market. By involving every person invested in the organization's success, the Jama platform provides a structured collaboration environment, empowering everyone with instant and comprehensive insight into what they are building and why. Visionary organizations worldwide, including SpaceX, The Department of Defense, VW, Time Warner, GE, United Healthcare and Amazon.com use Jama to accelerate their R&D returns, out-innovate their competition and deliver business value. Jama is one of the fastest-growing enterprise software companies in the United States, having exceeded 100% growth in each of the past four years, during which time both Inc. and Forbes have repeatedly recognized the company as a model of responsible growth and innovation. For more information please visit http://www.jamasoftware.com.