ACCELERATE DEVELOPMENT BY GETTING REQUIREMENTS RIGHT

Karl E. Wiegers, Principal Consultant at Process Impact
Sandra McKinsey, Senior Product Manager
at Serena Software, Inc.

SERENA™

TABLE OF CONTENTS

Abstract

Requirements management is an often underutilized discipline in software development. Yet the economic benefits—lowered development costs, faster development and better products that delight customers—have plainly been demonstrated in study after study. In this paper, renowned engineering process expert Karl Wiegers makes the case for requirements management as an integral part of any product lifecycle. Metrics are provided to show the positive effect of requirements management, from individual developers to the business to the customer. Serena® Dimensions® RM and Serena Composer are examined as an example of how to bring effective requirements planning and management into product development.

The Proven Way to Accelerate Development

Many software problems arise from shortcomings in the ways that people acquire, document, agree on, and modify the product's requirements. Typical problem areas include informal information gathering, implied functionality, inadequately defined requirements, and a casual change process.

Most people wouldn't ask a construction contractor to build a custom $300,000 house without extensively discussing their needs and desires and refining the details progressively. Homebuyers also understand that making changes carries a price tag. However, people blithely gloss over the corresponding issues when it comes to software development. The widely quoted CHAOS reports from The Standish Group relate the consequences of casual approaches to requirements engineering. Year after year, lack of user input, incomplete requirements, and changing requirements are the major reasons why so many information technology projects fail to deliver all of their planned functionality on schedule and within budget.

Nor do requirements issues only plague the early stages of a software project. One study at a large defense contractor found that 54 percent of all the errors were discovered after unit testing was complete, and that 45 percent of these were requirements or design errors (Davis 1993). Other studies also indicate that 40 to 60 percent of all defects found in a software project can be traced back to errors made during the requirements stage. Nonetheless, many organizations still practice ineffective methods for this essential project activity. The typical outcome is an expectation gap, a difference between what developers think they are supposed to build and what customers really need.

Requirements Stakeholders

Nowhere more than in the requirements process do the interests of all the stakeholders in a software or system project intersect. These stakeholders include end users, customers acquiring the product, requirements analysts, project managers, developers, testers, regulators and auditors, manufacturing staff, sales and marketing, and field support or help desk staff.

Handled well, this intersection can lead to exciting products, delighted customers, and fulfilled developers. Handled poorly, it is the source of misunderstanding, frustration, and friction that undermine the product's quality and business value.

Requirements are the foundation for all the subsequent software development and project management activities. Therefore, all stakeholders must be committed to following an effective requirements process. This process includes activities for:

- Requirements development (eliciting, analyzing, specifying, and validating requirements)
- Requirements management (dealing with the requirements once you have them in hand)

Developing and managing requirements is hard! There are no simple shortcuts or magic solutions. But a toolkit of techniques, coupled with tools to store and manage requirements, provides a powerful defense against the expectation gap.

When Bad Requirements Happen to Nice People

The major consequence of requirements problems is rework—doing something again that you thought was already done. Rework can consume 30 to 50 percent of your total development cost (Boehm and Papaccio 1988) and requirements errors account for 70 to 85 percent of the rework cost (Leffingwell 1997). As illustrated in Figure 1, it costs far more to correct a defect that is found late in the project than to fix it shortly after it was created (Grady 1999). Preventing requirements errors and catching them early, therefore, has a huge leveraging effect on reducing rework. Imagine how different your life would be if you could cut your rework effort in half!

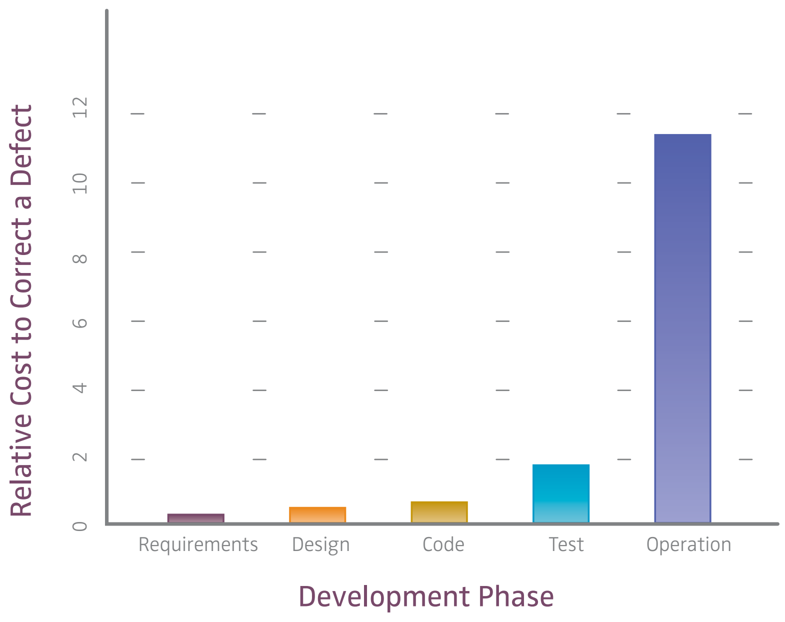

Figure 1: Relative cost to correct a requirement defect depending on when it is discovered

Shortcomings in requirements cause no end of headaches on software projects. For example, creeping requirements contribute to schedule and budget overruns. Now, change is going to happen. Requirements change because initial elicitation activities are imperfect, because business needs evolve, and because customer expectations change once they see the product beginning to take shape. Project plans and commitments therefore need to anticipate change, rather than expecting that an initial stage of gathering requirements is all that’s needed. Requirements creep, though, is the uncontrolled growth of requirements well beyond the original scope boundary.

The way in which requirements changes are incorporated into the product can degrade quality. You need to introduce changes at the highest level of abstraction they affect and follow the links in the traceability chain to see the impact of each proposed change. Traceability data stored in a requirements management tool such as Serena Dimensions RM greatly facilitates this impact analysis.

One of the biggest problems with requirements is ambiguity. Ambiguity results when a requirement can be interpreted in multiple ways. Tools and activities that let stakeholders compare their understanding of the requirements can reveal these ambiguities. Ambiguity leads to wasted effort, as developers or testers interpret the requirement to mean something different from what the customer or analyst intended.

Many projects suffer from gold-plating. Developers might add unnecessary features they think the users will like, or users request excessively complex systems. Tracing each functional requirement back to its origin—such as a user task, a business rule, or a business objective—helps solve this problem.

The result of all of these kinds of requirements problems is to slow the pace of software development. It doesn't do you any good to work very efficiently on the wrong requirements. If you don't get the requirements right, it doesn't matter how well your team executes the rest of the project.

How Good Requirements Accelerate Development

A sign in a high-school chemistry lab reads, "If you don't have time to do it right, when will you have time to do it over?" We rarely take the time to understand a product's requirements clearly enough to reduce the risk of racing into construction. However, we always find the time, money, and resources needed to fix a flawed product. It seems counterintuitive: If you spend more time on requirements engineering in the early stages of a project, is it supposed to save you time later in the project? Absolutely! To convince yourself of the benefits, think about how much effort on your projects is misspent because of miscommunications, misunderstandings about requirements, missing information, or misstated requirements. The case for emphasizing solid requirements practices is an economic argument, not a philosophical or technical argument.

A solid foundation of agreed-upon business requirements and project objectives establishes a shared vision, goals, and expectations. Frequent and intimate user involvement makes software development a collaborative partnership between IT and the customer community. It also reduces the chance of system rejection upon rollout. It's frustrating for everyone to release a product only to find that essential functionality is missing and other capabilities are included that no one will ever use. Good requirements processes ensure that the functionality built enables users to perform essential business tasks. Well-specified requirements also define achievable quality expectations, letting the team implement both the capabilities and the characteristics that will make the users happy.

While typical projects spend perhaps 10 percent of their effort on requirements, investing more has a big payoff. A study of 15 projects in the telecommunications and banking industries revealed that the most successful projects spent 28 percent of their resources on requirements engineering (Hofmann and Lehner 2001). The average project devoted 15.7 percent of its effort and 38.6 percent of its schedule to requirements engineering. NASA projects that invested more than 10 percent of their total resources on requirements development had substantially lower cost and schedule overruns than projects that devoted less effort to requirements (Hooks and Farry 2001). And in a European study, teams that developed products more quickly devoted more of their schedule and effort to requirements than did slower teams, as shown in Table 1 (Blackburn, Scudder, and Van Wassenhove 1996).

	Effort Devoted to Requirements	Schedule Devoted to Requirements
Faster Projects	14%	17%
Slower Projects	7%	9%

Table 1: Investing in Requirements Accelerates Development

Not all of the requirements development effort should be allocated to the beginning of the project. Don't think of it as a "requirements phase," but rather a set of requirements-development activities that are threaded throughout the project's lifecycle. Projects that follow an incremental lifecycle will work on requirements on every iteration throughout the development process. Agile development projects take an incremental approach, rapidly building small portions of the product so users can refine their needs and developers can keep up with changing business demands. Agile projects will have frequent but small requirements development efforts. Regardless of how large an increment your team tackles, they need to understand the requirements for that increment.

Managing Your Requirements

The goal of requirements development is to deduce, capture, and agree upon a set of functional requirements and product characteristics that will achieve the stated business objectives. Requirements management, then, involves everything that happens with the requirements in the product lifecycle once the requirements are developed, as shown in Figure 2. The demarcation between requirements development and requirements management is the defining of a requirements baseline. A baseline is a snapshot in time that represents the agreed-upon set of requirements for a specific product release.

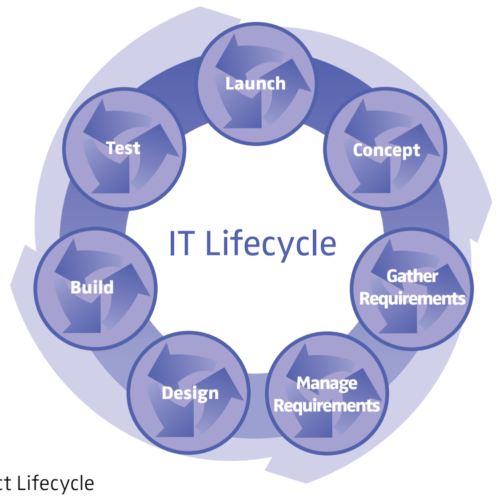

Figure 2: Product Lifecycle

The heart of requirements management is dealing with requirements changes. Each proposed change needs to be reviewed and the likely impact of implementing the change evaluated before it is approved by the appropriate body of decision-makers. This body is typically called the change control board. Change request decisions must be communicated to all who are affected so approved requirements changes can be incorporated into the software in a controlled way. Project plans need to be updated to remain current with the new requirements. The part of requirements management that no one likes—but which is the harsh reality—is the need to negotiate new commitments based on the estimated impact of changed requirements.

How Using a Requirements Management Tool Can Help

The traditional requirements approach is to create documents that contain business, user, functional, and nonfunctional requirements written in natural language, preferably using the organization's standard templates. A document-based approach to storing requirements has numerous limitations:

- It's difficult to keep the documents current and synchronized.
- Communicating changes to all affected team members is a manual process.
- It's not easy to store supplementary information (attributes about each requirement).
- It's hard to define logical links between requirements of various types and other system elements.
- Tracking the status of individual requirements is cumbersome.
- Concurrently managing sets of requirements are planned for different releases or for related products in difficult. When a requirement is deferred from one release to another, an analyst needs to move it from one software requirements specification (SRS).
- There's no convenient place to store those requirements that were proposed but rejected, nor those requirements that were deleted from a baseline. Sometimes you want to remember that these requirements had been proposed, because they might be back in scope one day.
- Reusing a requirement means that the analyst must copy the text from the original SRS into the SRS for each other system or product where the requirement is to be used.
- It's difficult for multiple project participants to modify the requirements, particularly if they are geographically separated.
- Visualization of the actual requirements is not possible.

Requirements planning and management tools such as Serena Composer and Serena Dimensions RM allow users to actually visualize the requirements using rich prototypes and store the prototypes and associated requirements in a multi-user database providing a robust solution to these restrictions.

Serena Dimensions RM becomes even more valuable as time passes and the team's memory of the requirements details fades. Following are some of the areas Serena Dimensions RM can help with:

COLLABORATION

In virtually any enterprise project, several types of requirements will come from multiple sources—customer and end user requirements, object-oriented design models from development, test procedures from QA, defects from customer service, change requests from all of the above, and more. Dimensions RM facilitates and organizes input from all relevant functional areas and stakeholders on a global basis. Once requirements have been captured into Dimensions RM, any stakeholder with appropriate access rights can view the requirements. Users can discuss requirements, submit change requests, and create specific polls querying the user community about the requirements. This information is automatically stored in Dimensions RM so that the rationale behind decisions is preserved.

FAMILIARITY AND USABILITY

Managers don't need to make huge investments in expensive methodology training in order to see the benefits of Dimensions RM. RM actually encourages user participation by leveraging familiar desktop tools like Microsoft Word and favorite web browsers, reducing time to productivity. In addition to lowering costs, this ease of adoption means that managers can see the project benefits of RM sooner. Users can quickly capture legacy requirements from existing Word documents into the RM environment, or use Word or even the web browser word-like view to author new requirements. RM then allows users to simply associate information or attributes with individual requirements. These attributes might include cost, schedule, or other types of data. This data is automatically captured and stored in RM's powerful Oracle™ data repository.

Dimensions RM also allows users to personalize the way data is presented to them through a customizable forms interface and a customized web menus and home page.

LOGICAL CLASSIFICATIONS AND TRACEABILITY

All projects involve different types of information that must be classified and linked appropriately. These include requirements, design, test, schedule, cost, changes, defects, and more. The Dimensions RM repository facilitates information classification, and then links classes together to define logical relationships. Users can graphically organize classes and relationships into a project definition, thus presenting a logical model for requirements management. These classes can also be organized into different categories such as usability or security and reused in many different documents or baselines of requirements.

Once data is captured into the Dimensions RM repository, users can then begin to track that data across development lifecycles, projects, and even organizations. This traceability gives users the power to answer key questions, like:

- What marketing requirements have no product requirements associated with them?
- What functions in the design are not needed and should be eliminated?
- What software applications and modules need modifications?
- What is the impact of changes?

With visual impact analysis for the classification and linkage among requirements, RM users can also take advantage of powerful query and reporting features in the product for higher-level analysis.

EASE OF ADOPTION

With a flexible requirements methodology, Dimensions RM supports both formal and informal requirements management processes, allowing users to evolve as their familiarity with the tool grows. RM can scale from small to very large projects, letting organizations standardize on a single product.

FLEXIBILITY AND INTEGRATION

The inherent flexibility of Dimensions RM is shown in our customers' use of the product for everything from managing software projects requirements, mobile phones, and even IT infrastructure changes. On one hand RM is a tightly integrated requirements capability in Serena Composer, which addresses the needs of requirements planning using prototyping and modeling to visualize the requirements to gain concise agreement. On the other hand it complements Serena® TeamTrack® for end-to-end process management and orchestrating project workflows. Dimensions RM is also tightly integrated with Mercury Quality Center to ensure complete traceability from requirements to test.

Handy data import and data export capabilities make it easy to integrate Dimensions RM with many 3rd party tools and existing environments. This allows users to maximize the value of their current investments, while leveraging the benefits of traceability and requirements management with RM.

About the Authors

Karl E. Wiegers is principal consultant with Process Impact in Portland, Oregon. Prior to starting Process Impact in 1997, Karl 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.

Karl is the author of the books *More About Software Requirements* (Microsoft Press, 2006), *Software Requirements, 2nd Edition* (Microsoft Press, 2003), *Peer Reviews in Software: A Practical Guide* (Addison-Wesley, 2002), and *Creating a Software Engineering Culture*. Karl has also written more than 160 articles on many aspects of software, chemistry, and military history. He has served on the Editorial Board of IEEE Software and as a contributing editor for *Software Development* magazine.

Karl received a B.S. degree in chemistry from Boise State College, and M.S. and Ph.D. degrees in organic chemistry from the University of Illinois. He is a member of the IEEE, the IEEE Computer Society, and the Association for Computing Machinery. He is a frequent speaker at software conferences and professional society meetings.

Sandra McKinsey is a senior product manager for Serena Software. In her role, she is responsible for the requirements and traceability management strategy and product development. With more than 15 years of experience in information technology, Sandra's background includes application development, managing development groups, and managing technology products through their entire lifecycle.

Before joining Serena, Sandra served as a senior product manager at Stellent and Agilent Technologies. Sandra holds a Masters of Business in Marketing from the University of Colorado and a Bachelor of Science in Information Systems from the University of Colorado.

References

Davis 1993: Davis, Alan M. 1993. *Software Requirements: Objects, Functions, and States*. Englewood Cliffs, NJ: PTR Prentice Hall.

Boehm and Papaccio 1988: Boehm, Barry W., and Philip N. Papaccio.1988. ''Understanding and Controlling Software Costs.'' *IEEE Transactions on Software Engineering* 14(10):1462–1476.

Leffingwell 1997: Leffingwell, Dean. 1997. ''Calculating the Return on Investment from More Effective Requirements Management.'' *American Programmer* 10(4):13–16.

Grady 1999: Grady, Robert B. 1999. ''An Economic Release Decision Model: Insights into Software Project Management.'' *In Proceedings of the Applications of Software Measurement Conference,* 227–239. Orange Park, FL: Software Quality Engineering.

Hofmann and Lehner 2001: Hofmann, Hubert F., and Franz Lehner. 2001.''Requirements Engineering as a Success Factor in Software Projects.'' *IEEE Software* 18(4):58-66.

Hooks and Farry 2001: Hooks, Ivy F., and Kristin A. Farry. 2001.*Customer-Centered Products: Creating Successful Products Through Smart Requirements Management.* New York: AMACOM.

Blackburn, Scudder,and Van Wassenhove 1996: Blackburn, Joseph D.,Gary D. Scudder, and Luk N. Van Wassenhove. 1996. ''Improving Speed and Productivity of Software Development: A Global Survey of Software Developers.'' *IEEE Transactions on Software Engineering* 22(12):875-885.

ABOUT SERENA

Serena Software, the Change Governance™ leader, helps more than 15,000 organizations around the world—including 96 of the Fortune 100 and 90 of the Global 100—turn change into a business advantage. Serena is headquartered in San Mateo, California, and has offices throughout the U.S., Europe, and Asia Pacific.

CONTACT

Learn more about the enterprise-wide power of Serena solutions by visiting www.serena.com or contacting one of our sales representatives in your area.

Serena Worldwide Headquarters	Serena European Headquarters	Serena Asia Pacific Headquarters
Serena Software, Inc.	Serena Software Europe Ltd.	360 Orchard Road
Corporate Offices	Abbey View Everard Close	#12-10
2755 Campus Drive	St. Albans	International Building
Third Floor	Hertfordshire AL1 2PS	Singapore 238869
San Mateo, California 94403-2538	United Kingdom	
United States		
800.457.3736 T	+44 (0)800.328.0243 T	+65 6834.9880 T
650.522.6699 F	+44 (0)1727.869.804 F	+65 6836.3119 F
info@serena.com	ukinfo@serena.com	apinfo@serena.com