

7 ESSENTIAL TIPS TO ENSURE SUCCESS WITH RM:

The path to building great software is through requirements management.

It's easy to forget, but the world relies on great software. Software operates the cars we drive, the planes we fly in, the cell phones we can't live without and the tools we use every day to get our jobs done. Software is everywhere.

As a software professional, you know all too well that software development isn't easy. A software product is never completed. There's always an opportunity to improve functionality and there's no shortage of challenges to overcome along the way:

- Lots of people involved in the process
- Customers have difficulty articulating their real needs
- Requirements constantly change
- Teams are spread across multiple geographies
- There's growing pressure to release products faster
- The complexity of software doubles every 2-3 years
- More projects fail than succeed

Whether you're building a revenue-generating product or an internal system, your company's overall success largely relies on your software team's success. And, the path to building great software goes through requirements management. Organizations that embrace this concept enjoy greater results. They experience fewer errors and frustration, faster planning and development cycles and they're able to deliver higher quality products to their customers.

The goal of this guide is to provide you seven essential tips to help you be more successful with requirements management. For some, these tips might be new. For others, these tips will serve as a good reminder of the fundamentals that are easy to lose sight of during the heat of a project. In addition, this guide includes links to other free resources that you can leverage right away in your requirements management process.

ONE: STAY CONNECTED, and you'll eliminate most issues.

So much attention is placed on the high failure rates of software projects and for good reason. Any time there's billions of dollars at stake and failure rates ranging between 60%-80%, people are going to pay attention. But, what you don't hear as much about is the root cause. Last year, in The State of Requirements Management Report we polled over 200 professionals about the top challenges they faced in eliminating project failure, and the resounding theme boiled down to one thing – communication. If you can get connected and stay connected throughout the entire development process, you can eliminate the vast majority of issues.

collaboration	Keeping your entire team connected throughout the development process.
traceability	Keeping the requirements, artifacts & other related information connected.

There's two parts to staying connected. First, there's the connectedness of your team, which has been popularized recently as "collaboration" – new buzzword, same meaning. Analysts, project managers, developers, testers, product managers, executives, stakeholders and customers - is everyone on the same page about what you're building and why?

Keeping everyone connected is often easier said than done, but it's absolutely critical to the success of your project. Depending on the size and location of your team, you can do this manually through meetings, phone calls and documents or you can use a tool to help keep your team connected. It depends on your situation and the complexity of what you're building. See number seven for the tipping points for when a tool might be valuable.

Second, there's traceability – the act of connecting up the requirements and other artifacts such as use cases, test cases, tasks, defects and even user documentation – all the details that are related to each other within a project. For complex development projects, there can easily be hundreds or thousands of items involved and it's critical to establish the traceability relationships between these items – both upstream and downstream.

For example, when a high-level business requirement changes 30 days into a project, through trace relationships you can then immediately assess the impact it has on any downstream functional requirements, tasks and defects that a developer or tester might be working on. This helps minimize errors and costly rework because the team members affected are aware of the specific change and its impact.

Implementing traceability and a change control process that's appropriate for your situation is one of the most important steps to ensuring success. As a simple first step to establishing change control, you can use a change request form manually to document changes right now.

[[click here for a change request template & other resources](#)]

TWO: TAKE ACTION NOW, don't wait for your process to be "perfect."

It's easy to fall victim to "process perfectitis" – a condition reached by teams that get paralyzed by process and analysis versus delivering working software. How many times have you heard someone say, "Well, we'll get to that project as soon as we really lock down our process?" Is any process perfect? More importantly, should that really be the highest goal of your team?

Whether your team is practicing some flavor of Agile or not, there's one thing we can all take away from the principles of Agile – it's that working software is the primary measure of progress. Don't get us wrong, optimizing your process is important, very important. We're constantly tweaking our process. However, if you have a better process and no product, you still have nothing to show your customers.

Doing something is better than nothing. Start small, identify a few critical requirements and take the approach of continuous improvement where you build, reflect, refine and repeat. Then, with each release cycle you'll learn more about the needs of your customers and continuously improve and expand upon the software solution you deliver to them. If you think your team suffers from process perfectitis, look for these symptoms:

- Requirements definition phase seems to drag on and on and on
- In the last month more time when spent talking about process, while your product stayed the same
- Lack of a decision-maker to make the call when to move forward with development

THREE: DON'T REINVENT THE WHEEL, leverage existing resources.

Even though every company, project & team are unique, the resources needed to help you be successful in most cases already exist. In minutes you can do a search on Google and find a wealth of best practices information. To short-cut your search, here are a few websites with resources to check out.

process impact	Karl Wieger's website with many templates, videos and other RM resources.
Tyner Blain	Scott Sehlhorst's insightful and detailed blog on software development.
Requirements Defined	Active requirements management message board hosted by Seilevel.
RQNG	Online network of professionals who manage requirements and share best practices.
Volere	The requirements template by James & Suzanne Roberston provides a checklist for requirements at several levels of detail.
Alistair Cockburn	One of the authors of the Agile manifesto and a resource for agile techniques.

FOUR: ELIMINATE AMBIGUITY by writing good requirements.

One of the things you can do immediately is make a “do not use” word list and post it up on the walls in your office. Visual reminders will help you to avoid using ambiguous terms when writing requirements. Karl Wieggers, a well-respected requirements management consultant, in his book *Software Requirements* provides a good list of ambiguous terms to avoid in requirements specifications. Here’s a snapshot of a few of them:

fast	Specify the minimum acceptable speed which the system performs some action.
flexible	Describe the ways in which the system must change in response to changing conditions or business needs.
acceptable, adequate	Define what constitutes acceptability and how the system can judge this.
simple, easy	Describe system characteristics that will achieve the customer’s needs and usability expectations.
shouldn’t	Try to state requirements as positives, describing what the system will do, instead of what it won’t do
robust	Define how the system is to handle exceptions and respond to unexpected operating conditions

Source: *Software Requirements* by Karl Wieggers, 2nd Edition, Microsoft Press, 2003.

[[click here to get the full list of ambiguous terms from Karl Wieggers](#)]

FIVE: RECONNECT with your customers.

You don’t have to be an expert to capture the voice of your customer – just committed. This may sound obvious, but it’s easy to lose sight of customer needs as a project gets underway and the team gets to work building the solution. Keep in mind, we use the word “customers” to define the end-users of the product you’re building – these customers could be external consumers for commercial products or internal users in the case of internal IT systems where other departments and employees are your customers.

Capturing the voice of the customer isn’t a one-time effort. Most project teams do a thorough requirements gathering session at the beginning of a project, but rarely does the customer interaction carry through to the end. Successful requirements management practices include constant communication with customers. Otherwise, you risk falling into the trap of delivering a product that end-users reject because it doesn’t resonate with how they expect to use it. There’s definitely an art to eliciting feedback and requirements from customers and clearly some people are better at it than others. There’s a plethora of books and courses out there to provide training for this specific skill. However, you don’t need to be a requirements management expert to capture the voice of your customers (continued on next page).

The fundamental skill required is commitment. Commit to picking up the phone every week and talking to customers. Commit to getting out of your office and sitting down with customers in their real environments. These are things everyone on the team can do, and should do. Even in Agile it's not always possible to have an on-site customer present, so you have to commit to getting that feedback other ways. Here's a quick list of dos and don'ts about how to stay connected to your customers.

Do	Don't
Be a journalist and ask open-ended questions.	Think you know best about what customers want.
Talk to your customers each week.	Assume past experience represents current needs.
Be open, and be flexible to change.	Assume customer needs are static.
Just pick up the phone & call customers.	Elicit requirements & feedback at a project's start
Listen with an open mind during elicitation	Sell customers on your idea for how it should work.
Sit with a customer and watch how they work	Assume customers know how to articulate exact needs.
Follow-up with customers when their feedback has been implemented in the product.	Forget to capture & share the evidence you gather with your team.

SIX: PRIORITIZE OBJECTIVELY. Avoid building what customers don't need.

Development time is valuable. There's nothing more frustrating for everyone than wasting time building features that customers don't actually use and don't provide value back to your company.

This is where requirements prioritization is essential. You need to avoid the common pitfalls of building features that seem cool or that someone thought a customer might need. Too often, requirements prioritization happens subjectively. The team holds a meeting and debates over the requirements and the loudest voice wins; or a request comes in from a salesperson who just spoke to a customer and the most top of mind request now becomes the hottest priority du jour. With each new feature request or high-level requirement, ask these questions to determine if this is a must-have or a nice-to-have feature:

- What percentage of our customers will benefit from it?
- Does it fit our brand values and enhance a competitive differentiator?
- What is the trade-off if we prioritize this ahead of other requirements?

It's best to establish an objective prioritization model that quantifies the variables that matter most and that each high-level requirement gets evaluated against. That way, by getting agreement on the scoring model, it's easier to get consensus on the highest priority requirements your team should focus on, objectively.

[[click here to get a feature prioritization matrix template](#)]

SEVEN: MINIMIZE OVERHEAD by selecting the right tools.

When you're a small team in the same office developing a fairly straight-forward product, you can use a whiteboard, task cards and daily face-to-face meetings to manage requirements. A specialized tool in this case could create unnecessary overhead. Likewise, if your team is building a product where the requirements are all agreed upon up-front and won't change much throughout the course of development, then documents and periodic status meetings may work just fine.

As projects grow in complexity and teams grow in size and geography, so do the communication challenges and overhead of trying to keep everyone and everything in sync. It's in these scenarios, where a requirements management tool can add value because the overhead of using the tool is far less than the manual overhead it takes to keep track of changes, manage trace relationships, update documents and communicate constantly with everyone on the team.

Here's a checklist of a few common tipping points where a specialized tool makes sense and can help reduce overhead by automating the process of keeping people and all the related information connected:

Variable	Tipping Point	Benchmarks
Complexity	The more complex the project, the greater the need. If you have over 100 requirements.	72% of teams have projects that on average have 100+ requirements.
Team size	The bigger the team, the greater the need. If you have over 25 people involved.	Over 40% of teams have at least 25 members and stakeholders.
Location	The more geographically distributed the team, the greater the need. If 10%+ are virtual.	Over 60% of teams have at least 10% of their team working in different locations.
Change	The more changes, the greater the need. If you spend 10%+ of your time managing change.	Over 2/3 of teams spend 10% of more of their time managing change.
Traceability	If traceability is a priority for meeting standards or government regulation, a tool is valuable for automating the management of trace relationships, change control and version history.	

Benchmark data: *The State of Requirements Management Report* by Jama Software and Ravenflow, 2008.

Thanks for reading the guide, and we hope you enjoy the tips. Let us know if you have any feedback because we want to hear from you. As you may have noticed, this guide highlights other companion resources and templates that are free and available for you to download. Simply click the link below to access them.

[[click here for free templates & other resources](#)]

About the Author



John Simpson, Director of Customer Outreach & Marketing

John represents the voice of the customer in Jama's product strategy and communications. He has over 14 years experience working at software and Web technology companies including Microsoft, WebTrends, Omniture and ZAAZ. He has contributed to several books, whitepapers and presentations on marketing and technology.

About Jama Software

Thousands of users worldwide. Billions in R&D projects managed within Contour.

Jama Software is the leader in collaborative requirements management solutions for improving enterprise collaboration and managing complex software development projects. Its Web application, Jama Contour, helps organizations manage the entire requirements management lifecycle through an intuitive, easy-to-use interface that brings people, process and data together to ensure software quality is delivered as specified.

Customers, from agile start-ups to the largest and most sophisticated technology and IT organizations in the world, turn to Jama to help drive innovation, improve the decision-making process and harness the collective genius of all stakeholders involved in building great software. For more information please visit: <http://www.jamasoftware.com>.