

Requirements for Developers (or There and Back Again)

In software development, the destination is delivering software that meets customer needs. But how do you reach that goal? It's much easier when that goal is defined in terms of "requirements". This article looks at requirements from a developer's perspective – why they should care, how effective requirements management can be achieved, and what the benefits are for the developer and all project stakeholders.

Reaching the pot of gold – why should you care about requirements?

Industry analysts report that 70 to 80 percent of software development project failures can be attributed to poor requirements management. Is this just another statistic that tools vendors use to justify their products? However, if you think about it, doesn't it make perfect sense? Requirements define what the software product or application has to do in order to meet the needs of the customer or the market, and if you don't meet customer needs then surely that's a failed project? Requirements also define the scope of the project and if that runs out of control you'll see budget and schedule overruns, which also can be seen as failures. If you set out on your software development quest without a clear idea of your destination – the chances are you'll end up chasing rainbows and never reach the pot of gold.

Requirements set the end goals for the project, but what value do they add to a developer's day-to-day work? Clearly, customer or market requirements set the context for the work of everyone on the project team. When designing or writing software, you are much more likely to create a solution that the customer will appreciate if you know the full context of what they are trying to achieve rather than just being told, "we need a blue widget in the top right hand corner". Not only do requirements ensure that developers stay on track, but by having the whole team with a good understanding of what they are building and why, individual developers can get more satisfaction out of knowing the value their work will bring.

Your mission, should you choose to accept it- why you need to manage requirements

Obviously requirements are important, but the industry analysts talk about "requirements management" – what's the fuss about "managing" requirements? Well, it's not good if only one person knows the requirements – that person could forget them or leave the team and take vital information with them. It would be like your software development project was a "mission impossible" with the tape of requirements self-destructing after one listen. Requirements need to be recorded in a form that is persistent throughout the project lifecycle and is accessible by all project stakeholders.

This isn't to say that they are cast in stone from the start of the project – of course requirements may change due to changes in business needs or evolve as the precise needs become clearer. However, requirements change has to be managed – remember that requirements define the scope of the project and uncontrolled "scope creep" must be avoided. And if requirements change, you need to be sure that everyone impacted by the change is aware of it – development, QA, documentation, operations, etc.

Each requirement needs to be clearly and uniquely identifiable so that it can accurately be referenced. It's also valuable to be able to record additional information against a requirement. For example, project management might define the priority of each requirement

and development may want to communicate effort estimates to project management. See Figure 1 for an example of a simple requirement form.

You may have an environment where edits to project artifacts must be audited in order to comply with industry or government regulations, but even if you are not mandated to do so, it's useful to know who changed what, when and why.

But I'm not shooting for the stars – is requirements management relevant to the IT developer?

Requirements management as a discipline has its roots in the aerospace industry and is well established on complex system engineering projects (where it's also known as "requirements engineering") in industries such as defense, telecommunications, medical devices and automotive. But is it relevant to IT and the software developer?

IT is increasingly coming under the microscope of business executives to ensure that it's delivering value to the business and is not just a cost center. IT applications may not be as complex or safety-critical as space rockets, but they are today absolutely mission-critical to the operations of many businesses. IT needs to meet the needs of its business customers and to do this it needs to have a better understanding of their requirements and an improved ability to respond to business changes.

Developments in sourcing of IT skills bring new challenges in collaboration between development teams and their business stakeholders. Often teams are distributed around the globe – you can no longer walk down the hall to talk to the end users or the business analyst to discuss requirements. If development is outsourced, good requirements specifications and the ability to ensure that requirements changes are communicated accurately become even more critical.

New approaches to development, such as Service Oriented Architecture (SOA), demand tighter collaboration between business and IT to ensure that services developed deliver value to the business. Agile development methodologies do not mean you can throw away requirements – in fact, it's even more critical to ensure that you stay in control of what's being delivered when you are moving rapidly.

Another driver in IT today is that of regulatory compliance. Industry and government regulations like Sarbanes-Oxley, UK Companies Act, HIPAA, Basel II, etc. mean that IT projects must adopt more stringent software development processes. Demonstrating that an IT system is conformant to its business requirements and complies with any applicable legislation is now often critical to avoid harsh penalties.

Make sure you're properly equipped for the journey – the right tools for the job

The basic requirement (no pun intended) for a requirements management process is that it ensures what is delivered is conformant to customer needs. But this doesn't mean that the same requirements processes and tools are applicable to all types of development projects. Software development projects following an iterative approach with monthly or weekly

releases need a different requirements process than systems development projects such as a new aircraft with a lifecycle of years and the complexity of thousands of parts.

Recent surveys of IT professionals have indicated that over 60 percent are not using a dedicated requirements management tool. Indeed the most popular tools for documenting requirements are Microsoft Word and Excel. Some of the reason for this is that requirements management tools have been focused on the needs of customers with more mature requirements practices and have been seen as too complex for many IT shops.

This is now changing with some vendors realizing that one size doesn't fit all in requirements management, and providing a portfolio of requirements solutions that span the needs of rapid IT software development through to complex systems engineering projects. There are now new Web-based solutions (see Figure 2) providing out-of-the-box processes for requirements management on fast-paced software development projects. These have the added attraction that they can be implemented as Software-as-a-Service (SaaS) solutions where the vendor hosts the software so the customer no longer has the overhead of installation and administration.

Retracing our steps – Traceability and Impact Analysis

We've looked at the need for requirements management in IT and highlighted some of the basic capabilities for managing requirements. But the value of requirements management isn't just in storing information about customer needs or stakeholder requests. Stakeholder requests expressed in business terms usually need translation and elaboration into technical requirements for which a solution can be designed. In a simple software application, these two levels of requirements may be sufficient, whereas in a complex system there may be many layers of requirements – customer, system, sub-system, software, hardware etc. In either case you should establish traceability – a "breadcrumb trail" of which technical requirements were derived from which stakeholder requests.

This is important because you can use the traceability links to know why the technical requirements exist, to check that you have full coverage of the stakeholder requests (look for stakeholder requests with no links to technical requirements) and to check for "gold-plating" (look for technical requirements with no links to stakeholder requests). Traceability is also a critical capability in establishing a complete development project audit trail for regulatory compliance.

Traceability isn't just useful for audit trails and coverage analysis, though. It's inevitable that requirements will change – to adapt to a new business environment, to take advantage of new technologies, or simply because the requirements weren't correctly defined in the first place. Establishing traceability means that when a change is proposed you can do a thorough check to see all of the related and more detailed requirements (and if you connect requirements to them, also development tasks, design models and tests) that could be impacted by the change. Then it's possible to scope out the effort involved to decide whether the change can be made within the project plan or if the plan needs to change. You can also be sure that if the change is approved, you won't miss any part of the application that needs to be updated, ensuring the change is correctly and completely implemented.

Requirements management tools make the creation and maintenance of traceability less of an overhead compared to using simple documents and spreadsheets. They also automate the

production of traceability and impact analysis reports to help you ensure that requirements and the impact of changes to them are never overlooked or misunderstood.

And they all lived happily ever after – customer, project manager and developer satisfaction

Effective requirements management practices and tools aid collaboration in software development teams – the requirements are the common goals for the entire team. They enhance alignment between IT and the business because the requirements represent the needs of the business. If the software delivered meets the needs of the business, then the customer is happy. The project manager knows that he/she has been successful. And the whole development team can gain satisfaction in a job well done and know that they have been working on software that has delivered real value to its users and the business. This is not just a fairy tale ending, but the reality of what can be achieved by the employing the right requirements management practices and tools on your projects.

Figure 1. A simple template for recording requirements

Figure 2. A Web-based requirements management tool for fast-paced software development.

Author: Andy Gurd, Telelogic.