Quality Management

Show Me One Good Requirement

Northway Solutions Group
9005 Overlook Blvd • Brentwood, TN. 37027 • Ph: 866.611.8762 • Fax: 866.611.8762
www.northwaysolutions.com

The Journey to Good Requirements

Many, many years ago, I was a budding test manager and this particularly obnoxious business analyst had been writing requirements for years. Not a good recipe for collaborative discussion on the relative quality the requirements. The BA sent me the business requirements specification for me to approve by the end of the business day. I recall how surprised I was at the short timeline to approve the requirements that I was seeing for the first time. I spent the balance of the day reviewing the requirements and documenting an ever-growing list of questions and issues I had with the requirements. With every issue that I documented a pain grew in the pit of my stomach knowing that as some point I would have to share this list with this more experienced, politically connected, and passionate business analyst.

As the end of the day neared along with my deadline, I bit the bullet and approached the BA with my list of issues. I explained that I was unable to provide my approval of the specification, and handed over the list of issues and questions I had prepared. After the BA was self-extracted from the ceiling, a string of words reserved for sailors could be heard across cube-land. The BA commented on the origins of my birth; questioned the existence of any form of intelligence; and explained how much longer they had been doing their job than I. The conversation ended with the BA throwing my comments in the trash and muttering "If you're so good, just show me one good requirement". As I collected the pieces of my fragile ego off the floor, I thought to myself "OK, I will. I'll show you how easy it is!" Over the next several days, my indignation turned to humility as reality sunk in. I had mistakenly assumed I could easily write good requirements since I knew what was wrong with the requirements I had reviewed. This experience (along with many others) provided me a career defining passion for business analytics, a discipline I still am trying to master.

Identifying what makes a good requirement is difficult and (at times) obscure. It is much easier to judge the results of good requirements than to write good requirements in the first place. Let me give you an example. I would venture to say that every one of us could bite into a nice warm piece of homemade apple pie and tell if it was good or not. We could most certainly tell if it was absolutely bad! You don't need much training to tell what tastes good or bad. However, if you were asked to bake that same homemade apple pie from scratch, many of us would struggle. Some would not even attempt that challenge. We would soon realize baking is a very specialized skill that takes a lot of practice and training. While the recipe provides the high-level guideline and the potential for a great pie, it is the skill of the baker that determines the outcome. Years of practice and training have honed the ability of the baker to make the recipe fit the creation. The baker knows how to identify what parts of the recipe or critical and what parts can be adjusted to make the best pie. These skills were honed through the taste buds of their product's consumers.

The same concept applies to requirements. It is much easier for us to sit in judgment of a requirement or set of requirements than it is to be the creative force behind the elicitation, elaboration and documentation of those requirements. We have no problem sitting on our high throne and casting stones on the hard work of business analysts, without having a sound appreciation for the pressures and challenges they face gathering good requirements. Many of our business analysts are expected to do multiple jobs, of which, requirements are a very small part. We often account for the time to write the requirements, but rarely is the effort to elicit, elaborate, and analyze the requirements accounted for in project schedules. In the remainder of this white paper I will try to highlight some characteristics of a good requirement in hopes that it will help both business and project stakeholders understand the amount of effort that business analysts are faced with on every project. These characteristics of good requirements will also help those reviewing requirements to understand the difficulty there is in developing a sound and complete requirement specification.

The Recipe for Requirements

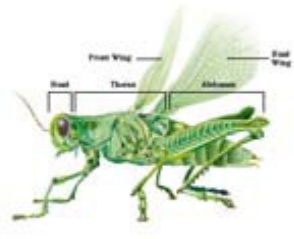
In this fast paced "time-to-market" industry we often are under extreme schedule pressure while collecting user requirements from our business stakeholders. Our business partners provide us a set of very high-level objectives. "The system must have the capability to capture customer information". At first glance, this may appear to be a good requirement; however, as you start to analyze the requirement it becomes apparent the requirement is not complete.

Having complete requirements helps reduce the chance that surprises will happen late in the development cycle

Which system is the requirement referencing? Specifically, what customer information should be captured? What will be done with the information? How will it be captured? How will it be maintained and for how long? What happens when customer information changes? This leads to the first characteristic of good requirements. Requirements must be ***complete***. Getting to complete requirements is probably the most challenging part of the requirements gathering process. It takes a concerted and committed effort to keep focused on digging until all aspects of a given requirement are identified. The job of the business analyst is to keep asking "specifically…" and add one or more of the following questions: "how?", "what?", "when?", or "under what conditions?" These questions are repeated until there are no more answers to discover. Completeness is the great mystery the detective that lives deep inside every business analyst must try to solve. Having complete requirements helps reduce the chance that surprises will happen late in the development cycle

Requirements often only identify what a feature should do. For example, a requirement might state, "upon approval, the balance shall be updated to the new balance amount". When a requirement identifies only what a feature or function must do it is only half complete. This is the second characteristic of requirements. Requirements must be ***balanced***. A requirement needs to identify not only what a solution must do, but equally important what it must not do. In the example above, the requirement will only be complete when the actions are identified for balances that are not

approved. In requirement definition, this is an area that gets little focus. We, as business analysts, are so focused on defining what a system must do; we often skim over identifying the error conditions that the system must address. Spending the time to map out the business process in workflow diagrams or SIPOC diagrams will help you to identify all of the conditions and feedback mechanisms necessary to balance the requirement.



A requirement that identifies the need to be "user friendly" is an example of why the third characteristic of good requirements is important. Requirements need to be ***precise***. Vague words, such as "fast", "intuitive", or "accurate" are difficult to understand or interpret, and are subject to the interpretation of the reader. I often use the example of the bug. If I were to tell everyone to draw a picture of a bug we would have several different renderings (mostly impressionistic) of that subject matter. The problem is that it is very unlikely that anyone will know exactly which bug the requirement author intended. Is it an insect? How about a car? Maybe it is a device that you put in a phone to listen in secret. In any case, it will be difficult for even two people to come to the exact same conclusion. Even if two of us assume an insect, are we envisioning the exact same insect? Would our customers understand if they get any solution as long as it was a software solution? Spend the time to ensure the requirements are not open to interpretation. This is not an area where you want your reader to activate their imagination.

Requirements often define information about either calculations or data that is collected as a part of the business process. A requirement may state, "Capture user name", but fails to define any additional information about the conditions that place limits on user name. This leads to another characteristic of good requirements. Requirements should be ***bounded***. All requirements should have defined boundaries (i.e. must be at least one character and no more than 20 characters long). Boundary definition continues to be an area of weakness in

requirement specifications. Requirements may specify data that should be collected, but fails to set limits on that data. The business analyst should always ask questions about fields, lists, and other data that is collected ensuring that key attributes are collected about the data - such as, minimum value, maximum value, data type, default value, and if the value is required. Consideration should be given to actions that should happen when values go out of the defined range.

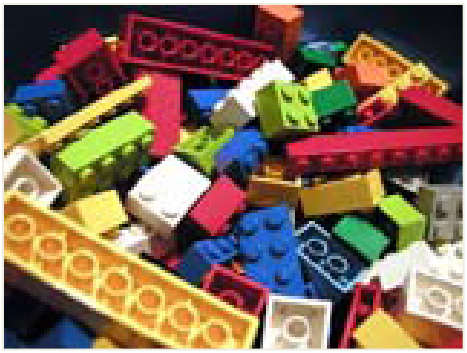
The creative ability of both business analysts and customers never ceases to amaze me. The human mind has the unique ability to assess the environment that it is in, address the current challenges and chart a plan to a unique solution. With this creative mind is the potential to define requirements that are not feasible to implement within current technology or environment constraints. This leads to the fifth characteristic of good requirements. Requirements should be ***feasible***. If a requirement is functionally sound, but cannot be implemented, the customer will receive less than they expect. The business analyst should review each requirement statement with relevant staff responsible for implementation of the final solution to verify that the request is within the realm of possibility. This check will lead to a very solid working relationship between the business analyst and the technology staff as well as ensure that the expectations of the end user are effectively managed.

One of the challenges with documenting requirements is ensuring the requirement is not stated multiple times in the specifications. If a requirement is stated in more than one requirement statement, the requirement may be confusing to readers. This leads to another characteristic of good requirements. Requirements must be ***unique***. The business analyst should ask himself or herself, "Is this the only requirement that defines this particular feature or function?" Is there is a valid reason to restate a particular requirement? If so, give special care to ensure that individual statements are not in conflict.

One of the key indicators of good requirements is that they should be ***testable***. You should be able to build one or more test cases that will completely verify all aspects of this requirement. If you are unable to write a set of test cases for a requirement that fully tests all aspects of the requirement, it is likely missing key aspects that define a good requirement. If a requirement is

not testable, then it is not buildable. If you can write a test case that fully tests any specific requirement, then the development staff should be able to write code that implements the requirement. Writing test cases from your specification as a part of reviewing your test case is a good habit for any business analyst. This is not wasted effort, as these test cases become a part of either the user acceptance suite of tests or the system testing suite.

The final characteristic of good requirements has to do with understanding the relative importance each requirement represents to the requestor. This characteristic of good requirements ensures all requirements are ***prioritized***. Knowing the priority of each requirement helps the implementation team ensure they are focused on delivering the most critical features, and ensures that testing focus is placed on the areas of greatest value. The business analyst should work with the business sponsor to prioritize each requirement or group of requirements as "high – must have", "medium - should have" or "low – nice to have". This prioritization will help both the business and the delivery organization ensure critical features and functionality is delivered to the end users.



Let's revisit the characteristic of ***precision***. As we look at documenting requirements for off-the-shelf or packaged applications, precision in requirements becomes very clear. In many cases, a packaged application can be highly configured. It is very similar to having a box of Lego® building blocks. Each block in the set has been well defined and verified by the company before sale to the customer. The number of solutions that can be built with the blocks is only limited by the imagination of the person constructing the solution. There may be a million solutions, but only one in the mind of the builder. If we are to apply this to software development, we have a packaged application that can be configured in an unlimited number of final configurations to meet the solution needs of the customer. However, there is only one "right" solution that the user has in mind. Any solution that deviates from that one "right" solution results in a level of dissatisfaction on the part of the customer.

They have an expectation that the organization that is building their solution has the expertise to pull from them all of the information they need to accurately implement their vision in the packaged application they purchased. Precision matters. "As long as I bypass an artery, it shouldn't really matter which one" doesn't work so well when doing open heart surgery.

Precision matters. "As long as I bypass an artery, it shouldn't really matter which one" doesn't work so well when doing open heart surgery

To pull all of these characteristics together, let's look at a sample non-functional requirement. As I look at this requirement statement I begin to formulate a quick set of questions. These are not the only questions that I may formulate, but should lead me on a journey of discovery. Let's look at the components of a good requirement. **Complete**: What does the author mean by the term "system"? Is this a software system? Is it a piece of industrial equipment? Or, is the system an institution? **Balanced**: If response time must be reasonable (whatever that may mean) for all critical transactions, what about the non-critical transactions? Can they take forever? What happens if the response time isn't reasonable? What does the system do then? **Precise**: What does reasonable mean? I am sure the early settlers thought the response time of the Pony Express was reasonable. **Bounded**: What is the specific response time that must be achieved? What are the specific critical transactions? **Feasible**: If search is one of the critical transactions, and the response time objective is set too low, is it even feasible to meet that objective? **Testable**: How do I write a test case to verify reasonable? What is the limit? If I don't have a list of critical transactions, how do I choose the right ones? Do I assume all transactions are critical? **Prioritized**: What is the priority of this requirement? What happens if it isn't delivered in the solution? What is the impact of it not being delivered? What other requirements are impacted if this requirement isn't met?

Let's make an attempt to write an improved version of this same requirement. You will notice that it is actually a set of requirements, not just a single requirement.

BR1.0 The response time for all WilsonsOutlet.com website transactions identified as critical shall be no greater than 8 seconds.

BR1.1 Response time is defined as the time from when an end user submits an action to the time the response is visible to the end user.

BR1.1.1A transaction is one complete end-to-end action initiated by an end user

BR1.2 The following transactions are identified as critical:

BR1.2.1 Navigation page views
BR1.2.2 Login
BR1.2.3 Contact Us Info Submittal

BR1.3 The response time for all other transactions (non-critical), including search, shall be no greater than 20 seconds

BR1.4 Transactions with response times that do not meet the defined criteria shall be logged for remediation.

These are just a few of the characteristics that go into the development of good requirements. These characteristics can be used to help guide the business analyst to a well-defined set of requirements. However, just like a recipe for baking a pie, it is up to the baker to apply the correct skill to making a pleasing final product. It takes practice, commitment to improve, and constant feedback on the results to hone the business analyst into a great requirements chef. And just like the baker, the business analyst has to make a lot of bad requirements before they learn the true nature of truly good requirements. For those of you, who are reviewing requirements, think about how difficult it would be for you write just one good requirement. The goal of our organizations should not be to get documented and approved requirements, but to get documented, complete, precise, balanced, bounded, feasible, unique, testable, prioritized, and approved requirements.

Getting to good requirements is the responsibility of every single person in the requirements value chain. Business analysts must strive to discover complete and accurate requirements through the elicitation process. Business representatives must stay engaged in the process to

ensure that there is "no stone left unturned" in the requirements. Development must participate in the review process, and raise red flags when requirements are in any way vague or open to interpretation. Testers must review the requirements and ensure that each requirement is testable. Organizations that place more value on the slinging of code than the elicitation of a solid basis of requirements, will always struggle to deliver quality solutions to the marketplace. Good requirements don't just happen by accident; they are gutted out through a tireless pursuit for perfection. As my Dad would always say, "strive for perfection, settle for excellence".

Good requirements don't just happen by accident; they are gutted out through a tireless pursuit for perfection.

Enjoy the journey.

About the Author

Brian Copeland is currently the QA Practice Director for Northway Solutions Group. With nearly 30 years of senior level experience in the software development industry specializing in organizational transformation and development, Brian has been instrumental in the testing of critical business systems, from mission critical applications to commercial software.

References

James C. Hunter. 1998. The Servant – A Simple Story About the True Essence of Leadership. New York: Crown Business Publishing

Karl Wiegers, PhD. 1996. Creating a Software Engineering Culture. New York: Dorset House Publishing

About Northway Solutions Group

Northway Solutions Group is a HP Elite Solutions Partner and Reseller for HP Software products Northway employs certified consultants with real-world experience who provide long-term solutions to the toughest business challenges. This includes providing training and implementation service of HP Software products

Contact

9005 Overlook Blvd
Brentwood, TN 37027 USA
Phone: 866.611.8762
Web: www.northwaysolutions.com
Email: Info@northwaysolutions.com