

The Seinfeld Approach to Requirements

By Barbara Davis

Software industry stats clearly show there is an urgent need for dramatic and immediate improvement to the way we develop our products.

Only 16.1% of software and technical projects are successful, and that of the successful 16.1%, only 20% of the implemented features are used all the time and 40% are *never* used at all! That means that only 1.24% of all proposed features are actually implemented and used! Further, 2.48% of all proposed features are implemented and never used!

The single-most common reason for these alarming statistics is requirements. A lack of practice formalization and holistic view of the causes and effects of failure, have led to the shotgun approach to requirements. Periodic successes with this approach have, in turn led Business Analysts to believe that the fault lies elsewhere in the process.

In a nutshell, we approach requirements the same way we approach any other communication or conversation. When you think about it, most communications and conversations are one-sided and ego-centric. Seinfeld was an excellent example of multiple people carrying on conversations in the same room without either listening or being heard. Translate this into requirements gathering and you get ambiguity, miscommunication and missed requirements.

The solution is that we need to streamline the requirements process and follow the old SMART ideology. The process itself must be Specific, Measureable, Achievable, Repeatable and Timed. Think of requirements like a recipe. If the recipe called for some flour and a little bit of sugar, and said to cook until done. Could you possibly make the recipe? Not unless you took the time to go through countless trial and error cycles and created a prototype in each cycle. Wait that does sound like the average development project... hmmm....

We'd have better luck creating recipes and requirements in the same way we conduct a science experiment. Define the expected outcome, determine the specific measures we will use and verify the results. Now, I realize you're going to argue that we verify in testing. The fact is, testing verifies the product against the defined requirements. When they differ or the requirements are wrong, we have testers pondering if they should be testing the product as it was developed or testing it against the defined requirements. The truth is that if you develop the product defined in the requirements, there is nothing left to wonder.

Unfortunately, what happens is that requirements are poorly defined, validated and communicated, development occurs based on the misunderstood requirements and testing exposes the gaps between them and struggles with reconciling the differences.

It's one thing to gather and document requirements, but that's really only half of the task. We also have to verify the requirements before we start the design & development process. When you consider that 70-80% of the cost of re-work in testing is used to fix requirements, and that the cost of fixing requirements can be as much as \$1000 to 1 requirement when you get to this stage, you could

potentially save approximately 1.43% of your project budget for each requirement by performing your due diligence up front.

So now that you know *what* you need to do, you need to know *how* to do it and what it looks like. The CRRSP requirements methodology – or Comprehensive & Robust Requirements Specification Process, details the exact tools, techniques and timing to get the best results possible.

Using a combination of lean management and aviation software certification techniques, CRRSP follows a clear process summarized into four categories: Research & Elicitation, Analysis, Elaboration & Specification and Validation.

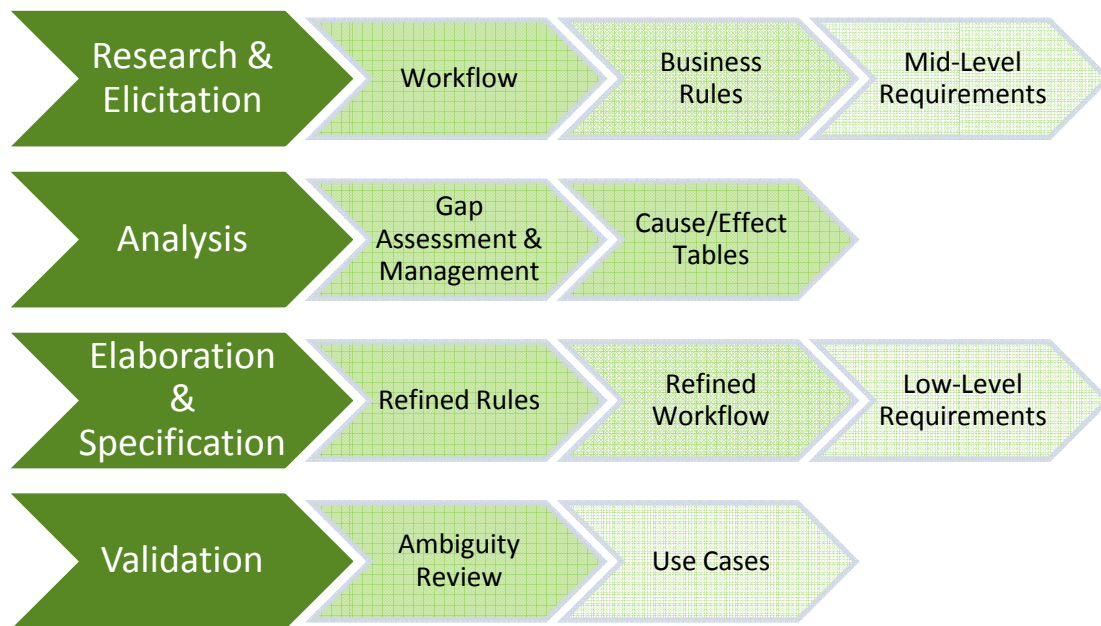


Figure 1: Comprehensive & Robust Requirements Specification Process

Each category utilizes a set of tools and techniques that are familiar to many business analysts; however, they have been refined and enhanced to deliver the highest impact without adding additional bulk to the project. For the most part, BA's take the shotgun approach to their job, simply because there are no hard and fast rules about how to do it well and there is a relatively short history to draw substantive data from.

The Research & Elicitation stage is comprised of workflow, business rules and mid-level requirements collection. That basically means you conduct your current state "inventory" and begin to drill down into the details of the high level requirements defined during discovery. The unique piece here is the idea of mid-level requirements.

We often get hung up on how far we drill down and how granular requirements need to be in order to be useful. Adding the mid-level requirements stage, allows the analyst to drill down to another level of detail without going in too deeply before they can assess the level needed. Mid-level requirements provide the analyst with the ability to stop and re-assess the detail of the requirements.

In the Analysis stage, the analyst spends a lot of effort mapping data, scenarios, decision tables, mind mapping and conducting a Gap Assessment between the current & future states. Here the unique steps

are the application of mind-mapping and cause/effect table techniques, as well as Gap Assessment & Management. The mind mapping and cause/effect tables are applied to ensure completeness of the elements and scenarios, while gap assessment is used to outline the gaps between current & future states, associated risks & impacts, and manage the recommended outcomes to closure.

Typical cause/effect tables looks like the following samples:

Cause	Effect 1	Effect 2...	...Effect 20	Priority/Scope (Optional)
User Input	What is the first thing that happens when the user inputs data?	What is the next thing that happens when the user inputs data?	What is the next thing that happens when the user inputs data?	What is the priority rating of this item? How will it impact the project?
Requirement	What are the states or outcomes of my requirement?	What are the states or outcomes of my requirement?	What are the states or outcomes of my requirement?	Is this item listed here to demonstrate consideration, but not in scope?

Figure 2: Cause Effect Table with definitions

State	Filed (in doc mgmt syst)	Not Filed (not in doc mgmt syst)	Status	Priority
Existing	X		Active (used & edited)	Medium
Existing		X	Active	Medium
Existing	X		Active Historical (used but not edited)	Medium
Existing		X	Active Historical	Medium
Existing	X		Obsolete Historical (not used or edited)	Low
Existing		X	Obsolete Historical	Low
Non-Existing		X	Active	High
Non-Existing		X	Active Historical	High
Non-Existing		X	Obsolete Historical	OUT OF SCOPE
Future	X		Active	OUT OF SCOPE
Future	X		Active Historical	OUT OF SCOPE
Future	X		Obsolete Historical	OUT OF SCOPE

Figure 3: Cause Effect Table live sample for a project to assess the existence of documentation and create it where needed.

During the Elaboration & Specification stage, the analyst refines and drafts the low-level requirements. It is important to note, that for mid-level requirements that have enough detail to define the functionality, are now included as low level requirements. They will all be submitted to strict validation techniques in the next stage.

Validation is the final stage of the CRRSP methodology. It defines specific techniques for final validation of the requirements draft. The truth is that validation does occur in the analysis stage as well because application of the cause/effect table allows the analyst to produce a quick visual of scenarios and can

illustrate the completeness of all possible combinations inputs and outputs in a scenario. But here in the Validation stage, the primary step is the Ambiguity Review.

The Ambiguity Review consists of a peer review by other analysts and team members. They are given specific criteria by which to assess the requirements: are they clear, consistent, accurate and complete. The reviewer's findings are logged into the Ambiguity Log. While simple ambiguities can be cleaned up in short order once they have been logged, complex ambiguities will need to be tackled in the Ambiguity Walk Through sessions. It should be noted that ambiguities remain open in the log until the person that originally entered them is satisfied that the response fully clarifies the requirement.

Walk through sessions can take up to two weeks and time blocks of 4 hours per day need to be allotted to them. You will need to facilitate the sessions according to specific rules in order to make the best use of everyone's time. Set rules about what constitutes a break-out session. Typically these include items requiring further analysis and items requiring input from another person that is not in the room (such as a decision maker).

The results of utilizing these detailed activities are dramatic. The biggest thing you will note outside of the decreased number of bugs is the level of collaboration your team and business will have after you have completed them. These tasks will specifically address communication at every level of your project in a way that brings people to the same understanding at the same time and they will be well-prepared to move into design, development and testing with confidence. The Seinfeld style of communication ends as abruptly as Kramer himself bursts in and makes a bold statement that forces the others to stop and respond to it directly.