

The Big Freakin' Requirements Document Must Die. Here's Why.

The typical requirements document is a long, sprawling piece of literature. Within it, one might find a title page, table of contents, change history, complex headers and footers, legalese, confidentiality notices, and, if you're lucky, maybe even requirements.

Its length is probably, primarily due to the fact that it tries to be everything to everybody. But, the problem is that this big freaking document isn't read entirely by any single person, except perhaps by the person who wrote it in the first place.

Every company refers to these documents as something different: BRDs, PRDs, BPDs, DRDs, SRSs, FRDs... or any other number of acronyms that people have forgotten the meaning of. OMG. To complicate matters, each department, project team... heck, *person*, uses their own template; so, one BRD does not necessarily equal another. But, who can blame the people who write these things? There are deadlines to be met, and all templates do not accommodate the needs of the many. Adjustments are made.

But luckily, from where I sit, I believe that the typical, mega-honking requirements document is nearing its death. And the good news is that this eventuality is closer than most people think. Don't believe me? We have actually witnessed this happening at some of the more progressive companies that we've worked with.

Let's walk through the reasons why I think these documents exist, and the problems that lie within.

To Communicate

In my mind, your primary job – whether you call yourself a "business analyst", "requirements manager", "product manager", or whatever – is to *translate* needs from one level of detail, to another. Thus, to be an effective business analyst, you must be effective translator. This means that you must be able to talk the language of the business, and then effectively analyze and formalize that language for the analytical minds of your implementers.

You then need a vehicle to communicate this translation: So, naturally, you go hunting for that mythical template, and set out to write a document.

Let's then consider what has to be translated. Requirements really come in many different shapes, including:

- flow charts, for describing a series of events, or flow of information;
- lists and tables, for detailing data, and rules;
- images/mockups, for showing what the thing that we're describing is supposed to look like; and
- paragraphs, for describing what's left.

One quickly realizes that a single document can't adequately communicate all of these things. So, what do you do? You go beyond the document, and use a mishmash of the tools you have available: Perhaps you use Microsoft Visio for drawing out diagrams, Excel for lists and tables, PowerPoint for user interface mockups, and Word for everything else. And *then* you have to try and glue it all together (more on that, later).

At some point you send out the document you're working on, because communication is a two-way street (a point that a lot of people seem to forget). Unfortunately, documents are designed for one-way streets. So, instead of your reviewers being kind enough to comment directly in the document, we schedule meetings because they don't know how to make Word work, or they need you there to interpret what you wrote.... perhaps because they didn't read it. The cycle continues: You change the document, you send it out again, and then you cross your fingers and hope it will be reviewed in some intelligent fashion.

Who's reviewing these things? The challenge is that there are many people to please, and thus not one, but *multiple* audiences. Parts of your document are only relevant to certain people: High-level overview stuff is great for managers, but developers could care less about the "Project Vision"; and testers only require certain parts themselves. (Though one might argue that they should read this stuff, in order to provide context for the job at hand).

As a result, there may come a point where you break the document into two documents (at least): One document for business audiences, and another, with a lower level of detail, for a more technical audience.

To Validate Needs are Met

Validation is where I say that what you think I said and what you say I said are the same thing, basically. (Got that?)

To do this, we need to connect the dots, or *trace* between what was originally said, and what it was translated to. This is also the part where it becomes necessary to add identifiers to everything, so we can reference them and find them, no matter where they are. Basically, all this means is that we end up putting unique numbers next to our requirements.

Ideally, we want to see how all these pieces fit together, so we aim to put everything in one document. In particular, what we're looking for is how the high-level needs will get addressed at a lower-level, by the implementers. However, if these two things are in two different documents, this makes things a little more interesting; not to mention we already have other objects spread across flow charts, spreadsheets, and slideshows.

This is where the idea of a "Traceability Matrix" comes in, which is not as cool as it sounds. The matrix is a table, created by hand, that shows how the stuff on the left is connected to the stuff across the top. In practice, however, because our documents become difficult to manage as they grow, these linkages begin to fall apart and we forget to update the table.

To Ensure Everything's within Scope

As our project evolves, we also discover that it would be nice to have *this* thing, and that we might need *that* thing.

By consequence, it quickly becomes apparent that if we don't control the size of the big thing that we're talking about, it will continue to grow, get out of hand, and then dinosaurs will rule the planet again.

Hence, a parking lot is born: This is a table of items that are "Out of Scope" that gets added to the document. This particular table shows us that yes, VP of Something Something, we listened to you ("Look, it's in the document!"), but no, we don't have the resources at the moment.

Of course, when the next project comes along, all of these items are buried in that document we worked on last year, and now everybody's forgotten about them, like fossils.

To Manage Changes to Requirements

Change happens. It's inevitable. And as you know, keeping track of changes in documents isn't easy: We have to record these changes, why they occurred, and who made them.

From the reviewer's perspective, they would like to see a big-picture view of what changed between the last time they saw the document, and what they're looking at now. Fine. This is where the "Revision History" table enters the picture.

But from *your* perspective (as the one who wrote the document), it might be helpful to see what you had done before you went out to Starbucks after lunch, and promptly forgot everything. In an effort to try to keep a better record of things, we turn "Track Changes" on, and make sure our beloved document is in a so-called "content management system", such as SharePoint.

Great. But what if lots of people are working on this thing? And what if your changes are spread across multiple documents?

To Meet Regulatory and Corporate Obligations

Documents are called into existence for regulatory, and other process-related reasons. In some industries, a process has to be in place, and a document marks a tangible point in this process.

Managers still want to put their monogrammed gold pen's ink on something tangible, sign on the dotted line, and have everything recorded so that somebody (namely the BAs that report to them) can be held accountable in case something goes wrong... which will happen, because nobody understood, or questioned the contents of the document in the first place.

Hence the list of "Approvers", with room for each stakeholder to sign and date the document, appears. The signatures on the printed document don't really mean anything, of course. Instead, this is something we like to call "the illusion of acceptance".

To Define Requirements Collaboratively

Whether you like your co-workers or not, you have to work with them to gather, document, and translate stuff. (There goes all hopes you had of keeping things under control.)

As you know, building a document with other people is currently not a fun process: Documents are emailed back and forth, opened in Word, edited, comments are added, changes are tracked, you save the document, and then upload it to SharePoint, or email it back.

Recently, newer versions of Word are starting to make it easier to collaborate on documents with other people. Of course, reality tells us that even though these features are becoming available, your company's culture, and IT department, isn't moving as fast. (If you're like most large companies, you're probably stuck on Office 2003. If you're not, consider yourself lucky!)

The Death of the Requirements Document

So, what can we do to get rid of the big freaking requirements document? What's needed is a solution that addresses the reasons as to why they exist, namely:

To Communicate. We need to be able to express requirements in various forms, in one place, not across several tools. We should also be able to present/split/deliver what we have, based on the audience we're talking to.

To Validate Needs are Met. We must be able to tie needs and their detailed requirements together, through traceability. However, this should be a by-product of our work, not something we have to manually create and maintain. We should also be able to illustrate and animate to help communicate, and not just depend on legal-style paragraphs of text.

To Ensure Everything's within Scope. We should capture various levels of scope in one place, and only show what is needed. (Don't keep them separate.) Being able to easily see the relationships between all the detailed stuff, and what higher-level stuff it came from helps us to ensure that everything is within the agreed upon scope, as well.

To Manage Changes to Requirements. We should be able to manage change, see those changes over time, at checkpoints, and drill down to specific changes, as necessary.

To Meet Regulatory and Corporate Obligations. If possible, we should be able to move through an approvals process, without the need for a document. However, we should be able to generate a document if, and when it's needed.

To Define Requirements Collaboratively. We must be able to allow multiple people to work together, without tripping over each other.

In the end it's become clear that writing down and drawing out requirements in documents just isn't working.

The funeral for the big freaking requirements document must happen, if we are to move forward.