The Project Documentation Management

Methods and best pratice for a structured redaction and reuse of the project documentation

Luc Dumont *

June 29, 2021

Naept Editions

* A LaTeX beginer

Disclaimer
You can edit this page to suit your needs. For instance, here we have a no copyright statement, a colophon and some other information. This page is based on the corresponding page of Ken Arroyo Ohori's thesis, with minimal changes.

Colophon
This document was typeset with the help of KOMA-Script and LaTeX using the kaobook class.
The source code of this book is available at:
https://github.com/fmarotta/kaobook
(You are welcome to contribute!)

Publisher
First printed in May 2019 by Naept Editions

The harmony of the world is made manifest in Form and Number, and the heart and soul and all the poetry of Natural Philosophy are embodied in the concept of mathematical beauty.

– D'Arcy Wentworth Thompson

Contents

List of Figures

List of Tables

Stakes and issues

Why writing project documentation is boring? 1

"We'll have to do documentation again ? Really !"

We hear this a lot when we tell a teammate he will have to write some documentation for the project. We are engineers and we have had to deliver many many documents. So let's be honest, documentation is boring. But when it is well done, it is a very valuable asset. It allows to make once mind, settle the project, detail, save, index and share ideas and innovative solutions (God bless Ctrl+F). Documentation would be so much less painful if it wasn't for some productivity traps that are so time consuming and generate pressure. Fortunately they can be avoided . . .

Here is a non exhaustive list of what is unpleasant when writing a document, and how to overcome it.

1.1 Writing about a topic we do not master

There is often only one person dedicated to write all the documentation of a project. It avoids to manage a planing for several editors. Besides the final product will have one consistent tone. But this one person may not master all the different fields of technology taking part in the project to give a complete description of them. This situation has been encountered by a person we met.

A former teammate engineer destimonial

I had been asked to write the tests to do on a product to ensure it meets our customer needs. It was a complex task because I was completely unfamiliar with one of the involved technologies. I had to ask to the expert and I worked hard to understand how it was working to finally write some relevant tests. It took me several iterations of them but it ended up working. I up skilled a lot on this technology but we may had lost some time and I got some cold sweats.

Giving someone a task he isn't skilled enough to perform without some learning can lead to an unsecured mindset. But sometimes, by lack of available experts, we cannot do otherwise.

Today there are more and more collaborative softwares easing the teamwork (you can share a document and comment it in real time with Microsoft Word or Google Docs for example). We have met lots of companies making their own tutorial videos to share some specific knowledge between collaborators. Besides, the "reuse", which means starting from previous documents to write the new ones, is becoming the rule. It is always easier to update a previous revision and adapt it to the new project, than starting the document from scratch.

1.2 The WYSIWYG editors

WYSIWYG stands for: “What You See Is What You Get”. It means what you see on screen is its real rendering. This is how works the main text editors like Microsoft Word. This makes those softwares very intuitive and easy to use. Everybody knows how to write something on Word!

But when you write project documentation, you probably need to add comments, track modifications, draw tables, put some automatic fields (page numbers, etc)...

This leads the software to use lots of RAM to be able to render the final view of your document in real time with all those features. It does it seamlessly for a short file. But your computer might start to sweat when your document reaches the hundred of pages. Then the text editor reactivity is getting slower and can finally crash (true story)!

LaTeX

To avoid hitting the save button every minutes, there are text editors with different philosophy. LaTeX is one of those and it is free. But it requires some learning first. Its way to do is to differentiate the content from the document layout in different files. On one side you write your text with tags in it. Those tags will indicate where you want the graphical features such as a figure, a table, a title, etc. On the other side. At this moment your text doesn’t look like the final rendering at all ! On a second file you configure the display of those graphical features. Finally you run LaTeX on those files and it builds your document. You may have to run several times LaTeX with different settings to get the exact document you had in mind. But then you will never have to correct the layout. It will never change because of a text modification. Content and layout do not interfere.

Ok, I confess it seams a little more touchy than Word. But you should give it a try because it is so much more powerful and rather easy to master ! Fortunately the WYSIWYG text editors get auto-save feature that are more and more effective.

1.3 Validation workflows

Usually every document shall be delivered to the customer. To ensure the document reaches the quality standard of the company, it must be checked by several reviewers through a validation workflow.

But the reviewers expectations may vary if they don’t know the project with the same level of details. So a simple document review can turn into an argumentation about the project itself. Here we go again, those reviews must be planed:

1. Select reviewers for the project and stick with them all along. You can assign them to specific parts of the document corresponding to there expertise.
2. Schedule the reviews in a finite period of time.
3. Collect their remarks in one place, like a spreadsheet for example. This will ease their management en then the document update.

1.4 The level of detail

Whatever the product, its specifications can be two pages long or one hundred. The difference lays in the level of details.

There is no miracle spec. They can be very light, it is cheap but you risk to put the project on hold with non-anticipated issues. They can be very detailed and plan everything but not only they are very expensive but every review will be very long.

So you have to meet halfway. A good practice is to make a first iteration allowing someone who doesn't know the project to understand it and start working on it. Then, very regularly, with small steps, update the document throughout the project. Be careful not to overkill the quality, it is not a piece of art, it is a tool. Documentation must not drain you effort over more critical activities.

1.5 The lack of interest

Some documents have no other purpose than being archived. They are just new entries to some dead database and will never be referred back to ever again. They are artifacts from an obsolete but persistent workflow. The french philosopher Julia de Funes says that loss of meaning happens when the purpose disappears in favor of its mean.

It is time to question this workflow because those documents have a cost (an not only a financial cost). What impact on the collaborator mindset will have this lack of interest for his work?

On the other hand, such a documentation can be highly valuable! It can be a precious asset to kickstart similar new projects. If they are just saved and correctly indexed in a database (like Sharepoint), a simple request on its search engine will rapidly identify the most relevant documentation to reuse. And by constantly updating this database, the company not only saves but grows its knowledge, its expertise. It turns this previously frozen load into a living strategic asset. The author becomes aware that he is building knowledge other will trust and use. It is a far more gratifying activity!

1.6 Working blind

The blank page syndrome is not a novelist privilege. Being in charge of a complex specification document can be petrifying if we don't know where to start. Should I start by this element or this one ? Haven't I forgot anything ? Is it precise enough ? It is really clear ? Should I explain this or is it obvious?

It is always a good move to start from something known and verified. We take a previous document, we drop what is out of context, we keep the frame and what is reusable and we start from here. The old parts will help us, guide us to be more exhaustive about the new project.

The next step is to build templates with pre-filled sections. It avoids us to rewrite over and over the same elements in all documents. Those templates can have entitled sections for every possible aspect of the projects. All of them will rarely be filled but they help ensure we forgot nothing. Project after project, those templates are updated and the document quality rises up.

The website FYI presents a very large amount of templates and dashboards for every productivity tools to Getting Work Done! (their moto)

1.7 Conclusion

After all, we see that documentation might not be a problem at all. Organization, effective management and methodology can easily turn what was a painful activity into a real engine for productivity and motivation!

Why writing documentation is boring

2

Writing a specification document can feel ea"We'll have to do documentation again ? Really !"

If only that could be so easy. Sometimes it is. But usually not. We will go through 4 major difficulties in this process from a simple McDonald's order to a wedding planning!

12:15 AM in a McDonald's fast-food:
– Hello, I'd like a Big Mac, a Coke and a large portion of fries please.
2 minutes later :
– Here is your order, have a nice day.
– Thank you, goodbye.

8:30 PM in a fancy restaurant:
– Honey, will you marry me ?– Yes !!!
One year later, 2 weeks before wedding:
– Great Scott! Nothing is going well! The whole table disposition from the wedding planer has to be remade : Anthony cannot bear his brother and my cousins have canceled. All the tablecloths are taupe but they should be beige. We totally forgot to organize the brunch. And you remember your Mexican music band? Wee cannot afford it!!!

Disclaimer

This article isn't in any way a celebration of the well-know fast food license. But it is a very good example of how a customer experience can be fully optimized

Either for a project as quick as a fast food order, or for one as ambitious asa wedding planning, the first step is always the same. We list all that we want and give it to some professionals who will take care of everything.In both cases, we identified 4 characteristics that are very important to master if we want the most seamless communication possible. Let's dive into them.

2.1 The context

In the McDonald's order placement scenario, the context issue is solved by your presence into the restaurant. If, by any odd circumstance, you would order a burger in a shoe shop for example you would get a good laugh from the seller or maybe a suspicious look.

In the wedding case, we observe that we haven't explain enough of the family situation to the wedding planer: the enemies brothers and the not so reliable cousins.

It is very important to give the most complete context for a project to our supplier, to detail our activity. Like that, even before entering the core of your need, he will be able to tell if he is qualified enough for the job or not. To do so, you can present:

- Your company, its activities, some technical concepts you need your supplier to understand
- Your market and your customers
- The project your need is about

2.2 The vocabulary

At McDonald's, the vocabulary to order a menu is clearly defined thanks to the panels above the cashiers. They show all the burgers and side dishes available. Their combinations are rather finite and the customers have all the time to build their order while they wait in line.

For wedding, we can have very precise expectations, like beige tablecloths for example. But if we haven't correctly defined with the planer what we have in mind when we talk about the beige color, we can end up with taupe!

So clearly stating the vocabulary is essential. There is a probability that you will employ the same wording but with a slightly different meaning. That difference can generate big consequences on the budget or the planning by the end of the project before you understand why.

A simple glossary of the most important elements, placed in the opening of the document, can ensure all the stakeholders that they speak the same language. Do not hesitate to update it during the project in any doubt appears.

2.3 The maturity

This is the most common characteristic in the fast food orders: how many customer arrive at the cashier and are still building up their order. But it's in the name: fast food! If the order isn't complete, it won't take the customer too much time to come back with a new order.

Maybe the Mexican band wasn't necessary for an already beautiful wedding. Or at least, it wasn't totally affordable. It would probably be wiser to abandon this "requirement" to secure the event.

More seriously, it is difficult to precisely estimate the maturity of a project at the beginning. Is this item relevant? Useful? Is it a "must have" or a "nice to have"? Do we really need a hammer to kill a fly? Or a rolled newspaper is enough? We can employ the 5 why technique: questioning a need 5 times in a row to find out its root, its prime cause. We can also openly talk about it with our supplier and give him the opportunity to make suggestions, and for us to trust its expertise.

2.4 The exhaustiveness

We are always asked if we want anything more in a fast food. It is to drive us to buy more? It is to be sure we, the customers, are always pleased? I will leave you to this mystery. But this question has the benefit to ensure an higher maturity, completeness of our order, avoiding us to come back for something we would have forgotten.

Oh, oh, oh! We totally forgot to think about the brunch for the day after! We need a whole new budget for it and talk about it with the caterer! This wedding planner is very lame. He should have warned us.

This issue goes hand to hand with the maturity of our needs. Exhaustiveness asks us "Did you think of it all? Have you not forgot something?" Very hard to tell in advance. To secure those risks and avoid any "holes in the racket" (we say "trou dans la raquette" when something is missing), we can check what we did in previous projects. We can consult their documentation and use it like a starting checklist. Some parts could be completely reusable and some other may need minor updates. In the same mindset, we can build templates for the projects: pre-filled documents with commonly used chapters and checklists. Some will be abandoned, some will be kept, augmented, optimized

Once again, we can brainstorm with our supplier and summon his expertise. His fresh look upon the project can enlighten some dark areas.

Tools

Needs assessment : some good practices

3

When we write a needs assessment or a specification document for a supplier, we try to be as complete as possible. The document turns into a list of expected features. Here are some good practices to build the most efficient request.

3.1 Sorting the requirements by themes

The more requirement we have, the more difficult it will be to get the big picture. So we have to sort them by their subject, by the technology they are about, or by the expertise they require.

For example, to specify a bike, we gather as follow:

- All the requirements about the wheels
- All the requirements about the brake system
- All the requirements about the gear management
- ...

Therefore the supplier will easily be able to choose his best qualified experts for the job. The needs assessment document can now be shared with all of them, and each of them can take on the part about his field of expertise.

3.2 The shorter, the better

When writing a needs assessment, we may intend to write long descriptions for several reasons:

- we are in a train of thoughts
- we describe several connected elements
- the subject is complex and requires lots of details

But the problem happens when the customer brings an evolution to its initial assessment. We have to find out what are the impacts on our features specifications. Without an accurate indexing of the requirement and a clear view of their content, we will have to read the whole document again. That can take a lot of time.

But if, from the beginning, we have tried to write the shortest possible requirements, each of them deals only with a specific point, it will increase their quantity but it will also strongly optimizes the following:

Their **indexing**: one requirement describes one limited subject.

The easiness of their **update**: we know at one glance if the requirement must be updated because of an evolution.

Their **traceability**: it is way easier to link the need to the feature.

Their **testing**: the shorter the requirement, the easier to prove that the final product matches it.

Their **reusability**: in a future project, if the client defines a similar need, it will be very easy to reuse the requirements from a previous project. You may have to perform some adaptations but they will be limited. Just like a piece of Lego you can reuse in a new construction.

3.3 A needs assessment requirement must describe a need, not a feature

When we write a needs assessment, the purpose is to subcontract the product development to a supplier. We do so because we do not have the resources or the knowledge to do it ourselves.

But if we have a very detailed idea of what we want, we may be tempted to write a full feature specification. Which is a totally different document and we advise you to agree with your supplier about it. Unless, be careful not to give too much specific details and force your supplier. By giving him to much technical details, you won't summon the expertise you were asking in the first place by hiring him. It is risky for the project to mix your need and the solution specification in the same document :

- Are you sure the features you have in mind will match your needs? (Have you ever heard about the XY problem?)
- Your feature solutions, are they all compliant?
- What are you expecting from your supplier? An expertise or just some manpower?
- What about its added value?
- What about its responsibility?

That's why it is important to distinguish your needs from the final product features.

3.4 Using "shall"

English language is very generous on how to express what we want. Some of the manners can be ambiguous and may compromise the message we want to deliver.

- "I would like", "I wish"... Is this a strong order or the expression of a nice to have?
- "I want"... Is it a personal request? Is it part of the contract?
- "It has not to"... Should we understand it has a strong prohibition or the freedom not to do it ?

First of all, you can choose any verb but the most important is to choose one and stick to it during all the project. A commonly used one we recommend is to employ "shall". It is simple, kind of binary and rather not questionable:

- "XXX shall be..." or "XXX shall do...": it describes what is expected

- "XXX shall not be. . . " or "XXX shall not do. . . ": it describes what must not be expected

3.5 Pay attention to all the possible interpretations

A same text can be understood differently depending on the reader's expertise, experience, position and even his current state of mind. So there is a chance that he has a slightly different interpretation than yours. This risk must be considered seriously as it can lead to severe project shift. Do your words express your thoughts? Is there any room for a misunderstanding? Do you share with your supplier the same meaning for every concepts of the project? The road to hell is paved with good intentions.

So, before delivering a document, a simple peer review by one of your teammates will very efficiently reduce the risks of misunderstanding. And adding a glossary at the beginning of the document to define the sharpest concepts will reduce the risks even more.

3.6 Is your need testable?

Is it even possible to check that the product matches every specific requirement?

Imagine that one of your requests asks for "the fastest" car. Alright. How can your supplier ensure you that the car he delivered is "the fastest"? This is a tricky demand hardly verifiable. First of all, how fast can a car go today? What if this speed is different than the one you had in mind? Who is right?

It is much safer to build the requests with measurable elements. These won't let any doubt about what you are expecting and will be undeniably verifiable.

"The car shall be able to reach the speed of 250 kph". The delivered car goes up to 300 kph. Ok! It matches the need, the requirement is validated!

For more details about tests, read our article about validation plans. It is also in french, we will publish it in english soon too.

Reading sheet, an easy tool for collaborative review

4

Finally ! You did it ! The document is finished. But before definitely freeze it, it has to be validated. So you send it to several people who will read it and comment it. But one week later, you have dozens of e-mail from them and 4 different corrected copies of your document. Those exchanges have brought more questions than answers. The reading sheet allows you to structure the validation process and to take every remark into account.

Depending on the size and complexity of the project, it is required to validate its documentation by one or several people:

- The project manager who is responsible for the customer
- A peer reviewer who checks the technical content
- A quality expert who ensures the document matches the company's quality criteria
- ...

The customer who tells if the document meets his needs. Without a clear workflow, every reviewer can comment the document in a different way:

- by mail
- by telling you his remarks at the coffee machine
- within his own document copy
- in the MS Word comment section
- by hand-writing his remarks on a print
- ...

Managing so many different channels can jeopardize the final quality of the document. How to be sure none of the remarks is lost? Is there some duplicated ones? Are they all consistent? Etc.

4.1 One file to collect them all

The reading sheet is a tool to collect all the remarks about a document (or anything else) in the same place and organize them to address them all.

This is a simple spreadsheet, an MS Excel file, in which every line is a reviewer remark. It contains several columns to fill. Those columns are indicators allowing to sort the remarks and ease their management. In the following paragraph we will detail the most commonly used columns.

Table 4.1: Headers for a reading sheet

ID	Location	Category	Severity	Description	Reviewer	Responsible	Status	Comment	Action

4.2 The columns

Here is an example of a reading sheet based on the feedback of our own experiences and the ones of people we met.

This list isn't exhaustive. Some of them might be useless and you might need some other depending of your process. Please share with us in the comment section those data you use in your own reading sheets.

ID

It is a simple but unique number, incremented for each remark. The purpose is to identify it. If a remark must be erased, its ID has to be erased too and it cannot be used for another one or it could lead to misunderstanding.

Location

Any remark has to be precisely located! The author must not be forced to read the whole document to find the object of the remark. It wouldn't be very efficient. A first way of doing this is to locate it with the document page. But it isn't the best practice because if a section is added to the document, for example, the updated can turn to be totally irrelevent. The ideal way is to locate the remark with the smallest paragraph title its object belongs too. Like this, it doesn't matter if the paging changes. The author will quickly find it with the table of content.

The location shall be given by the reviewer.

Category

All the remarks are not from the same kind. They can be about:

- a comprehension question
- a spelling mistake
- an author misunderstanding of the need
- etc.

Therefore remarks can be sorted by categories. It will be easier to manage them. Fro example, the author may want to fix all misspelling ones before going to the semantics.

It is important to have a limited number of categories for the sorting to be useful. But on the other hand, it is required that list to be exhaustive enough to cover any kind of remarks. Here are some possible categories:

- Question
- Misspelling

- Nonsense
- Layout
- Error

Category shall be given by the reviewer.

Severity

Severity is a second sorting criteria. Two remarks from the same category can have different weight. This weight can be defined by the severity.

For example, the severity scale can go from level 1 telling the remark is painless for the project, to level 5 telling the pursuit of the project is stopped as long as this remark isn't solved.

It is really important that this severity scale is clearly defined and shared with everyone before starting the reviews.

Severity shall be given by the reviewer.

Description

This is the description of the remark itself. It is important this description is as detailed as possible. You can insert a short document extract to ease its understanding, highlight important words. It shall be precise enough for the author to not have any doubt about the issue.

Description shall be given by the reviewer (obviously).

Reviewer

If the author knows who gave each remark (several reviewer can add remarks to the reading sheet), he will know who to ask if he needs further information to solve the issue.

Reviewer shall be given by. . . the reviewer.

Responsible

Here we talk about the person in charge of solving the remark. As this person can be different for some remarks (many people may have written the document together), the person responsible must be defined for each remark.

But it is not up to the reviewer to decide who is in charge of his remark, beside he may not have a clue about who is competent for it. If there is only one author for the document, so it's a no-brainer: responsible = author. If there are several authors, there must be a manager driving the team. So it is up to him to tell who is in charge of each remark.

Status

It simply is the status of the remark. Depending on your review workflow, this status can take several values. Here are some examples of possible values:

- **Open** (set by the reviewer)
  When a new remark is added, it is "open" for resolution, meaning it shall be processed by the responsible.
- **Duplicated** (set by the person responsible)
  If by any chance a remark tells the same thing than another one, the responsible gives this status to it. It happens when several reviewers use the shame reading sheet to gather their remarks.
- **Accepted** (set by the person responsible)
  It is a temporary status. On a first hand, the responsible may have a look on all remarks at once. He may want to check for the duplicated ones or to see if all of them are relevant. When he wants to tell if a remark is relevant and he is willing the solve it, he sets this status.
- **Rejected** (set by the person responsible)
  But when he believes a remark is not relevant, he can set it to"Rejected". He shall explain why it is irrelevant in the comment section. Then the reviewer will decide if he agrees.
- **Corrected** (set by the person responsible)
  The responsible has accepted and fixed the document according to the remark. He tells it is "Corrected" then the reviewer can check if he is OK the correction.
- **Closed** (set by the reviewer)
  If the correction is satisfying, the reviewer can set the remark to "Closed". Case closed!

Comment

This section can be used by the responsible to bring a comment on the remark resolution. He can explain here how he corrected it. This is where he shall explain why he eventually rejected the remark. If the remark consisted in a simple question, this is where the answer shall be brought.

Be careful to not turn this comment section into a chatroom about the remark.

If required, the comment can be added by any stakeholder.

Action

Some remarks and resolutions can have a higher scope than just updating the document. They may require to update the whole project, change the contract, the planning, the budget, the product. . .

So one remark can have some ripple effect implying the use of a project dashboard action, a bugtracking issue (see our french article, but soon translated). . . Those item have their own identification references. So you can use the Action column to quote those references and ensure the traceability of the cross-application resolution.

If required, the action shall be given by the responsible.

4.3 Best practices

Index the reading sheet

As soon as you create a new reading sheet, give it a unique reference. It will allow everyone to quote it in the document it talks about and ease their management if you have several of them.

Use one for each document delivery

Every time you send a document to someone, if this someone can comment it, join a blank reading sheet. So he can add its remarks directly into it, saving you the task to copy and past them from his e-mail.

Traceability

When you start to update a document with all your reviewers remarks, indicate in the document that it has evolved thanks to the reading sheet <insert its reference>. So anyone can track back the origin of those evolutions.

Precision

Never hesitate to give the most exhaustive description for a remark. The responsible shall have enough information to not need to call you back to remove a doubt.

Once more, the location in the document of the object of the remark shall be as precise as possible for the responsible shall not lose time finding it.

4.4 Pros and cons

Pros

You can build a reading sheet out of any spreadsheet editor (MS Excel, Google Docs, OpenOffice. . .). So that is a really easy tool available to anyone.

It is not only applicable to a document review but, actually, to any kind of reviews: source code, graphic elements, product. . . As soon as you need to collect remarks, the reading sheet is relevant.

Even if there are several reviewers, you can send them each a reading sheet and then gather all of them into one (you can even write a script to do it automatically). But with the today online editors, it is even easier to work on a shared document.

Cons

Be careful not to use a reading sheet and its comment section as a forum with an endless message chain. Comments shall just be comments. Otherwise just pick up the phone.

Obviously it is another file to manage in the project and you may already have lots of them.

You better have a big screen or even two. It is nice to have the document you are reviewing on a side and its reading sheet on the other side. Switching from one to the other on a single screen might drive you crazy.

4.5 Conclusion

The reading sheet is a very easy and versatile collaborative tool. It allows you to structure and easily manage any reviewing process. Even if more and more software come with comment features, a reading sheet is a quick and simple solution to implement with anyone regardless of having the last fancy webapp. You can find a reading sheet template for free on our Github repository.

The traceability matrices 5

In a customer request for proposal, the need is usually given through a list of requirements. The supplier sends back a proposal showing his solution through a list of specifications. The coverage between requirements and specifications can take different shapes:

- 1 to 1: one specification covers one requirement
- 1 to many: several specifications are used to cover only one requirement
- Many to 1: several requirements are covered by one specification.

But in practice, it is rather a mix of the three leading to a maze of coverage. Fortunately the traceability matrix shows up to bring some order.

5.1 Traceability, this entangled mess

A document size can vary a lot depending on the different fields of activity it deals with. To explain this, I will take an example I know well. Consider a request for proposal (RFQ) about an aeronautic calculator. Please don't leave yet! I'll make it clear.

In a plane, there are dozens of calculators. Those are computers without a screen or a keyboard. They manage everything, from the air conditioning to the brakes, the doors, the electrical power distribution, etc. Whether their tasks are complex or very complex, their request for proposal can easily reach a hundred pages and hundreds of requirements. Several expertises are involved to describe all the requirements. For example:

- electronics for the circuit boards
- mechanics for the case housing the circuit boards
- developers for the embedded firmware of the circuit boards
- etc.

Then this RFQ is sent to a supplier. He will study it with his teams to build the specifications. Sometimes several requirements can be covered by the same specification.

Many to one

Customer requirement 1: the circuit board shall be hermetically protected
Customer requirement 2: the case shall have the following measurements : X, Y, Z

Supplier specification 1: the case shall be from the brand B and the model M

Therefore the supplier specification hits two birds with one stone because the chosen case matches both the requirements.

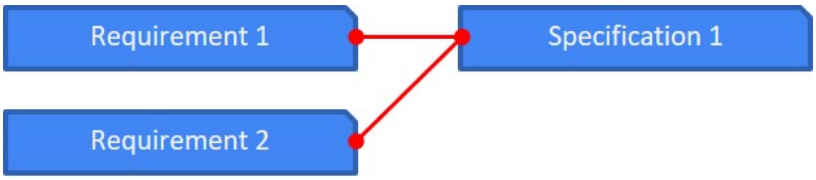

Figure 5.1: 2 requirements covered by 1 specification

One to many

But sometimes, it is the other way around! The requirement is so complex, it requires several specification to be fully covered:

Customer requirement 3: the circuit board shall be fully operational while the plane is flying (it'd better be).

Supplier specification 2: the electronic components of the circuit board shall be from the S series. The S series components stay operational even at temperatures below -50°C (this is the outdoor temperature at cruising altitude).

Supplier specification 3: the case shall have an S shield to protect the circuit board from electromagnetic fields.

Supplier specification 4: the case shall be mounted on suspending plot to protect the circuit board from mechanical vibrations.

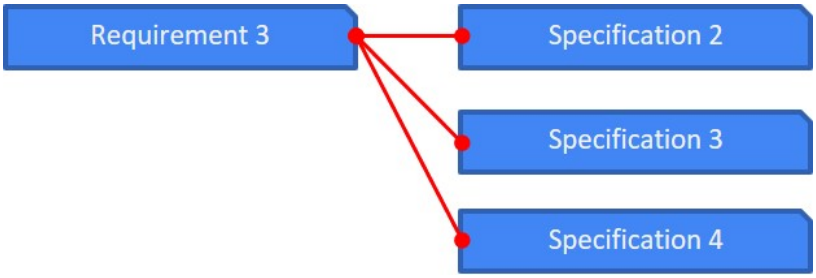

Figure 5.2: 1 requirement covered by 3 specifications

So we see here that the complexity of the coverage between requirements and specifications can quickly increase. Just like a spaghetti dish!

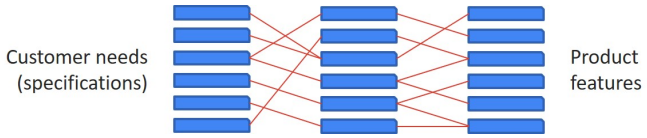

Figure 5.3: Many requirements entangled to many specifications

5.2 Ok but how exactly?

The structure

A traceability matrix is an two-columns table where are exhaustively listed every coverage link between every requirements and every specifications. Our previous example would give the following matrix:

On the left, there is the list of the customer's requirements in the same order they appear in the RFQ. There are given by their title only (or their ID) to avoid an overloaded table. On the right, for each requirement, the list of the covering customer specifications.

Table 5.1: Downstream Traceability Matrix

Customer Requirements	Supplier Specifications
Customer requirement 1	Supplier specification 1
Customer requirement 2	Supplier specification 1
Customer requirement 3	Supplier specification 2 Supplier specification3 Supplier specification 4

It is very important that on the left column, each requirement appears once and only once. The purpose here is to ensure every requirement is covered by at least one specification. On the contrary, on the right column, we observe the supplier's specification 1 is listed twice. This is absolutely normal as this specification matches both customer requirement 1 and customer requirement 2.

The point of view and coverage quality

We can say this matrix adopts the customer's point of view. It goes from the RFQ to the specifications, that is why it is called the **Downstream Traceability Matrix**. Once completed, we know exactly the coverage of the customer need. And if one requirement as no specification match, we know the specification document isn't complete.

But be careful as the devil always hides in the details! The traceability matrix can only tell if a requirement is covered by a specification. I cannot tell if this specification IS ENOUGH to cover the requirement. Let's consider again the customer requirement 3. We see it is covered by the following specifications:

- Supplier specification 2
- Supplier specification 3
- Supplier specification 4

For example, if the supplier specification 4 is omitted, from the matrix point of view, the customer requirement is still covered. But in reality, without the specification 4, the product wouldn't match the need.

5.3 The other traceability matrices

We can change the point of view of the matrix to observe the project's traceability from another angle to ease its management.

Upstream Traceability Matrix

We saw the customer's point of view with the downstream traceability matrix. Let's check the supplier's with the Upstream Traceability Matrix.

On the left, we list the supplier's specifications in the order they appear in the specification document delivered to the customer, avoiding making

Table 5.2: Upstream Traceability Matrix

Supplier Specifications	Customer Requirements
Supplier specification 1	Customer requirement 1 Customer requirement 2
Supplier specification 2	Customer requirement 3
Supplier specification 3	Customer requirement 3
Supplier specification 4	Customer requirement 3

Table 5.3: Non-covering Specification Matrix

Non-covering specification matrix
Supplier specification 5
Supplier specification 6
Supplier specification 7

duplicates. On the right, facing each specification, we list all the requirements it covers. This time, it is OK if there are duplicated requirements. It tells that one requirement is covered by several specifications, proving the supplier's solution is strong and relevant.

This point of view shows if all the specifications cover the requirements and how many requirement are covered by each specification (we will see that sometimes a specification covers nothing)

So in the project's roadmap, specifications covering the highest number of requirements can be prioritized. In our example, it can be strategic to develop supplier specification 1 to rapidly cover several requirements. Consequently, we can faster reach the minimum valuable product to show the customer. Then we iterate upon it by adding more and more features (aka specifications).

Non-covering specification matrix

It happens that the supplier designs additional specifications covering no requirement. Strange isn't it?!

Not necessarily. Those orphan specifications may describe constraints for the supplier independent from the customer's need. But the supplier want to mention them on the specification document to share them with its customer.

For example, the supplier is used to a software, he can tell his customer he will use it through a specification. So that specification will be mandatory without covering any need. Or the supplier has a special discount for specific components, so he tells thought a specification that he will use this brand over another. Last example: reading the RFQ, the supplier identifies a hidden need that doesn't appear in the customer's requirements. He can add it in his specifications.

He could put those orphan specifications in the upstream matrix, but they would face no requirement. That could bring doubt upon the document: is it a mistake? Is it normal?

To avoid those questioning, the best is to create a new table listing only those non-covering specifications.

Table 5.4: Evolution Matrix

Evolutions	Comments
Customer requirement 2	Deleted
Customer requirement 3.2	Updated
Customer requirement 4	New

Table 5.5: A deletion

Evolutions	Comments
Customer requirement 2	Deleted

Evolution

It is inevitable! Whether the RFQ or the specification document are intended to evolve. Because the need is more mature, because of the discussions and new ideas come out or bring corrections. . .

In the document revision 2, some requirements have been updated. But not all of them. So what? Should we read the whole document again? The 200 pages? To only find 3 or 4 modifications?! No! Because there is an evolution matrix!

This matrix doesn't connect two different documents, it links two revisions of the same document! For example, let's imagine the RFQ has had 3 modifications as follow:

A deletion

The customer has simply deleted a requirement. It was no longer required or what it described was non longer needed. A deletion in a 200 pages document might be missed. But its impact on the project might be huge because all the specifications that used to cover it are no longer needed either. Therefore they have to be deleted too to avoid useless costs. Besides the planning may be updated too!

I highly advise to keep this deleted requirement in the downstream matrix. But instead of keeping the covering specification, replace them with the comment "deleted" in the specification column. It might be redundant but at least it won't be forgotten.

An update

A previously existing requirement has been updated. It shall be highlighted because this evolution might have an impact on all the specifications that used to cover the previous revision of this requirement.

Table 5.6: Downstream Traceability Matrix with a deleted requirement

Customer Requirements	Supplier Specifications
Customer requirement 1	Supplier specification 1
Customer requirement 2	Deleted
Customer requirement 3	Supplier specification 2 Supplier specification3 Supplier specification 4

Table 5.7: An update

Evolutions	Comments
Customer requirement 3.1	Updated

Table 5.8: Downstream traceability matrix with an updated requirement (we observe here that the covering specifications have been updated too for the specification document to be relevant)

Customer Requirements	Supplier Specifications
Customer requirement 1.1	Supplier specification 1.1
Customer requirement 2.1	Deleted
Customer requirement 3.2	Supplier specification 2.2 Supplier specification 3.2 Supplier specification 4.2

We can add a revision number to the requirement's ID to show its level of update : "Customer requirement 3**.2**".

Besides, this would be a good practice to add this revision number to all the requirements. By convention, the original revision of a requirement would be the number 1 : "Customer requirement X**.1**" (You can choose .0 as the number of the first iteration. It doesn't really matter, as long as you stick to your convention through the project).

A new one

The customer has added a new requirement to detail his need. Just like the other evolutions, without further indication a new requirement can be missed in a 200 hundred page document. Because a new requirement imply the definition of new specifications, its creation shall appear in the evolution matrix, otherwise you may fall through the net. Or as we say in french you get a "trou dans la raquette" (a hole in the racket).

Long story short, any evolution described in the evolution matrix can, and shall, be also described in the other traceability matrices. It will produce duplicated information but allows you to double-check the consistency of your documents. If you find a loss going from one to another, then there might be a bug in the project. If there isn't, then you have a very clear view on the project's management.

Ok but what if there is a third revision of the document?

We still talk about the evolution matrix between two revisions of the same document. First of all, it is totally fine to have a third, a fourth, etc revision of a document. Documentation is a living entity, the blueprint of your project. On some projects, I have already had up to seven revisions for the same document, it's usual business.

So if you have a third revision (v3) then you should make an evolution matrix showing the updates between v2 and v3.

Table 5.9: A new requirement

Evolutions	Comments
Customer requirement 3.1	Updated

Table 5.10: Downstream traceability matrix with a new requirement (4.1) which has been immediately covered by a new specification (5.1)

Customer Requirements	**Supplier Specifications**
Customer requirement 1.1	Supplier specification 1.1
Customer requirement 2.1	Deleted
Customer requirement 3.2	Supplier specification 2.2 Supplier specification 3.2 Supplier specification 4.2
Customer requirement 4.1	Supplier specification 5.1

Table 5.11: Matrices to add to your document depending on its revision

Matrices	**v1** **Original document**	**v2**	**v3**	**...**	**vN**
Downstream	D_1 compliant with U_1 and NC_1	D_2	D_3	...	D_N
Upstream	U_1	U_2	U_3	...	U_N
Non **C**overing	NC_1	NC_2	NC_3	...	NC_N
Evolution	At this point there is no evolution matrix	$E_{1->2}$			

It is not mandatory, but it can be helpful to keep all the evolution matrices in every new revision. Like this, you'll have a complete historic of your document without having to open all the previous revisions to have the big picture. But beware, in the end, it might be a lot of matrices!

Appendix

A Heading on Level 0 (chapter)

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

A.1 Heading on Level 1 (section)

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

Heading on Level 2 (subsection)

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

Heading on Level 3 (subsubsection)

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift –

not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

Heading on Level 4 (paragraph) Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like "Huardest gefburn"? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

A.2 Lists

Example for list (itemize)

- First item in a list
- Second item in a list
- Third item in a list
- Fourth item in a list
- Fifth item in a list

Example for list (4*itemize)

- First item in a list
  - First item in a list
    - First item in a list
      - First item in a list
      - Second item in a list
    - Second item in a list
  - Second item in a list
- Second item in a list

Example for list (enumerate)

1. First item in a list
2. Second item in a list
3. Third item in a list
4. Fourth item in a list
5. Fifth item in a list

Example for list (4*enumerate)

1. First item in a list
    a) First item in a list
        i. First item in a list
            A. First item in a list
            B. Second item in a list
        ii. Second item in a list
    b) Second item in a list
2. Second item in a list

Example for list (description)

First item in a list
Second item in a list
Third item in a list
Fourth item in a list
Fifth item in a list

Example for list (4*description)

First item in a list

First item in a list

First item in a list

First item in a list

Second item in a list

Second item in a list

Second item in a list

Greek Letters with Pronounciation

Character	Name	Character	Name
α	alpha *AL-fuh*	ν	nu *NEW*
β	beta *BAY-tuh*	ξ, Ξ	xi *KSIGH*
γ, Γ	gamma *GAM-muh*	o	omicron *OM-uh-CRON*
δ, Δ	delta *DEL-tuh*	π, Π	pi *PIE*
ϵ	epsilon *EP-suh-lon*	ρ	rho *ROW*
ζ	zeta *ZAY-tuh*	σ, Σ	sigma *SIG-muh*
η	eta *AY-tuh*	τ	tau *TOW (as in cow)*
θ, Θ	theta *THAY-tuh*	υ, Υ	upsilon *OOP-suh-LON*
ι	iota *eye-OH-tuh*	ϕ, Φ	phi *FEE, or FI (as in hi)*
κ	kappa *KAP-uh*	χ	chi *KI (as in hi)*
λ, Λ	lambda *LAM-duh*	ψ, Ψ	psi *SIGH, or PSIGH*
μ	mu *MEW*	ω, Ω	omega *oh-MAY-guh*

Capitals shown are the ones that differ from Roman capitals.