11 Tips to make the requirements definition process more effective and results more usable

This article discusses what I believe are the key techniques for making requirements definition process repeatable from project to project.

1. Do timely 'scoping' to organize work

From the requirements point of view, the ultimate reason for "scoping" is to ensure the integrity of requirements that you produce. Without correctly "scoping" your work, you may produce the requirements that are not needed or redundant with the requirements produced by others. You may also omit some important requirements (Figure 1). Either reason may translate into poor designs or even solutions that do not work.

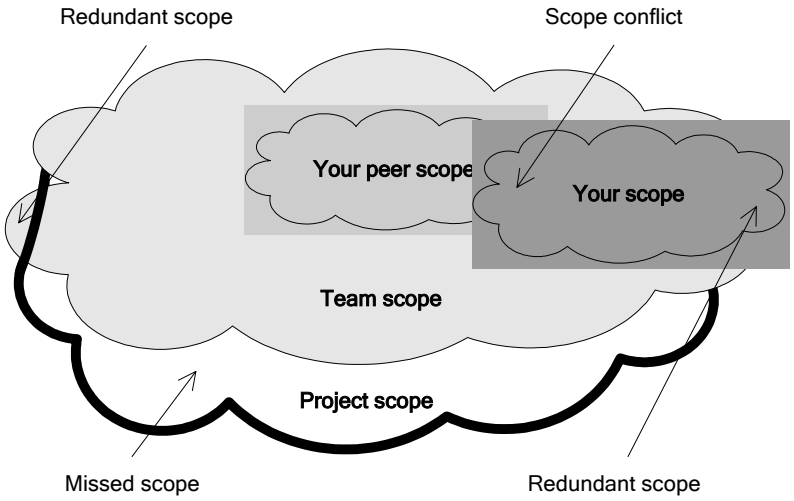


Figure 1 Requirements 'scoping' anomalies

2. Assess what categories of requirement are right for a project

Understanding implementation differences and standardizing on only certain categories of requirements is essential to a project as it affects the selection of methodology and tools.

The most commonly used flavors of requirements are functional and non-functional, however other requirement types may need to be occasionally captured as well, such as constraints, and operational, contract, and process requirements. (See Figure 2 below.)

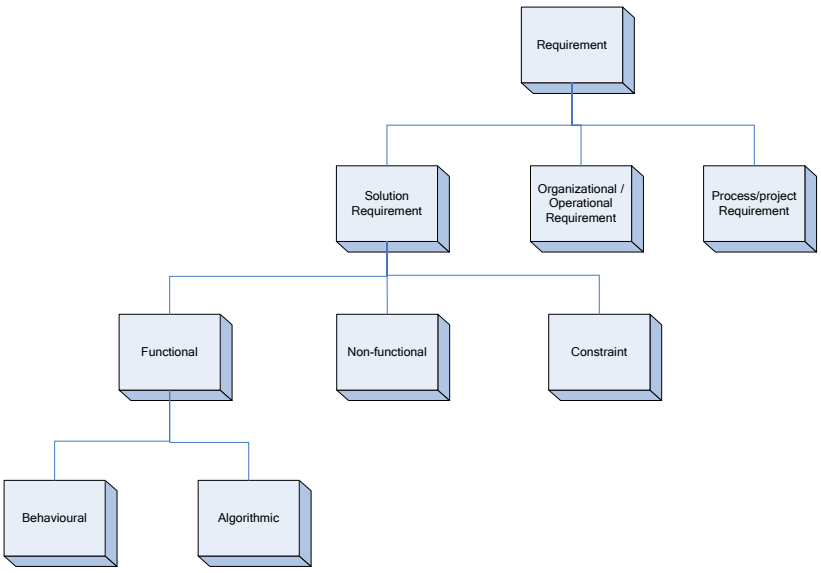


Figure 2 Requirement categories

"Project and process requirements" have a project perspective and include things like tasks, objectives, and methodology. "Organizational and operational requirements" span the entire enterprise and include, among other things, training, monitoring, and configuration issues.

Within the solution implementation space functional requirements describe flows and activities, while non-functional requirements describe their execution characteristics, and constraints describe their limitations.

Additionally, a project may need to capture events, needs, dependencies and other solution or enterprise requirements which would otherwise be captured during the pre-project work.

3. Adapt a standard methodology that deals with requirements

Normally, a company would either reuse a standard methodologies (Figure 3) or develop its own methodology that, usually, inherits from a standard methodology. An advantage of an of-the-shelf standard methodology over a custom one is, however, in a much larger pool of information and guidance available to its users.

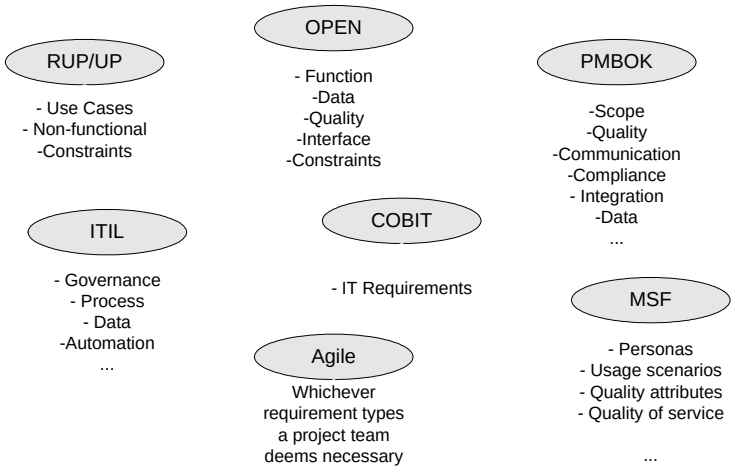


Figure 3 Popular methodologies and frameworks and their requirements scope

Understanding that types of requirements that are expected is a good departing point in a process of choosing the right methodology. Here are some other questions to ask:

- does the methodology support all identified categories of requirements ?
- how comfortable are you and your team are with the methodology ?
- how much guidance on the methodology is available to the team ?
- how well the methodology is supported by tools ?

4. Choose a requirements tool that fits the methodology

Tools are as critical because they provide the process guidance and ensure that the methodology is correctly followed.

The effort required to support any given discipline may spread over multiple project phases which, in turn, affects the tool usage (Figure 4).

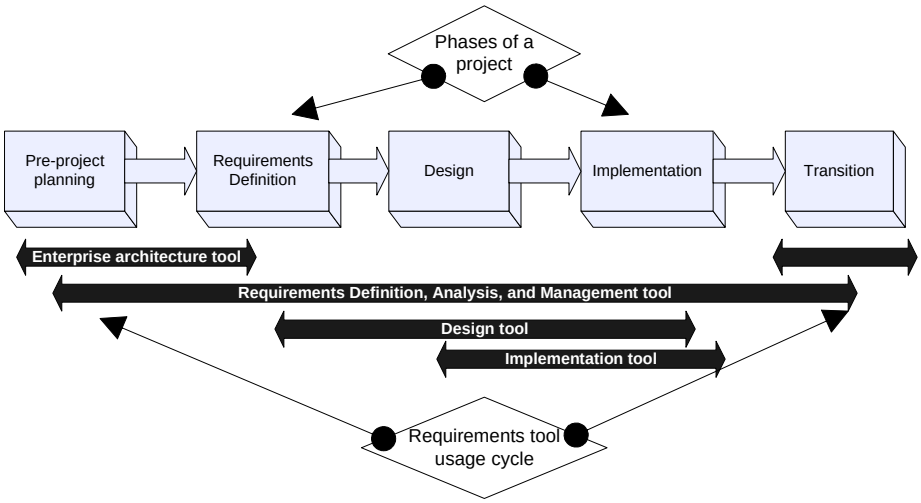


Figure 4 Illustration of domain tool usage throughout project phases

The tool selection for the Requirements discipline has to revolve around the requirement categories identified in the beginning of the project and the chosen project methodology. Since functional requirements are formulated differently then non-functional requirements and constraints they also demand a different tool.

5. Implement a requirements grouping strategy to support growth

A need for the requirements categorization is usually recognized when their number reaches a threshold at which managing them from the same pool becomes uncomfortable. A typical solution then is to group requirements into logical classes, according to a pre-defined criteria (Figure 5). A criteria used for grouping may vary from project to project and typically depends on such factors as a project type, identified requirement categories, and a project methodology. Some of the most common criteria are:

- by importance
- by requirement delivery factors (iteration and priority)
- by resource assignment (John Smith, and Vitalie Temnenco)
- by expected physical components (screens, web pages, and reports)
- by common solution layers (integration, user interface, and business logic)
- by business context (function, area, and process)

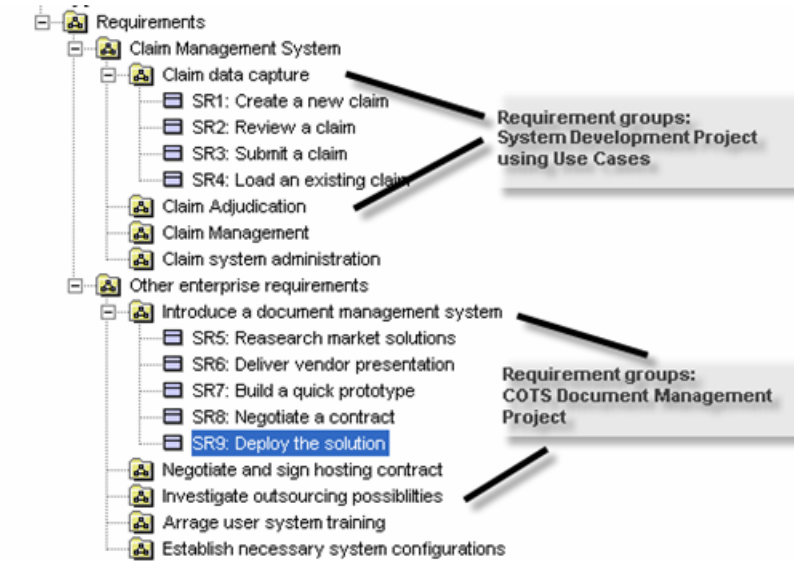


Figure 5 Examples of using grouping to classify requirements

6. Maintain a balance between different types of requirements

The problem often is that when a functional methodology is selected as a principal in project, the non-functional requirements are largely ignored (or vice-versa), which results in ill-suited implementations that cannot fulfill, presumably, the poorly documented needs.

All requirement categories are important to a successful implementation and must be paid adequate attention during a project (Figure 6). Ideally, all requirements should be confined in a single document where they can be cross-linked for quick traceability. In practice, some methodologies separate requirements into different documents, according to their type.

Notwithstanding the actual constraints and project configurations, a balance between distinct requirement categories must be maintained. Maintaining a balance is the key to delivering solutions that meet users' needs.

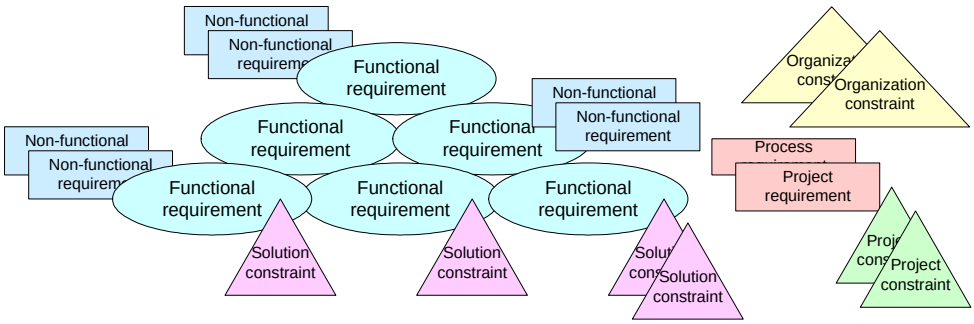


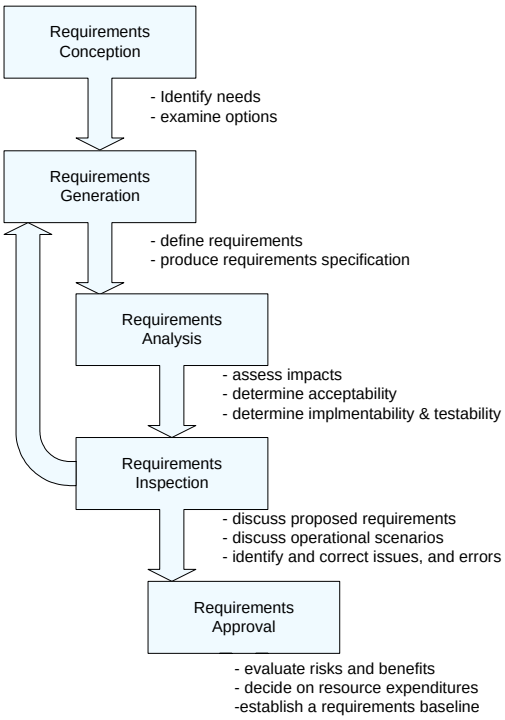
Figure 6 Conceptual requirements composition

7. Introduce a requirements definition process

Requirements management is an evolutionary process that kicks in when the requirement is discovered and completes after it is implemented and tested.

Few realizations of a requirements definition process were put together by different companies which resulted in some terminology differences. Generally, a process is an iterative activity that looks something like shown in the Figure 7. While larger projects may require a process that consists of multiple phases and activities, smaller projects may get away with less rigor.

From a point of view of a solution implementation methodology, such as RUP or ITIL[1], requirements definition starts long before an implementation project commences. The Requirements Conception, therefore, does not belong to a solution implementation methodologies' lifecycle. It belongs to a domain of Enterprise Architecture which has its own implementation methodologies, such as the TOGAF[2] [5]. The enterprise architecture is a domain of its own with the main objectives of identifying solution needs and creating an implementation blueprint, which leaves the solution definition requirements gathering to the implementation projects.



[1] ITIL stands for the Information Technology Infrastructure Library. For more information about ITIL go to www.itil.org and www.itilcommunity.com

[2] TOGAF stands for The Open Group Architecture Framework. For more information about TOGAF go to www.opengroup.org

Figure 7 Sample Requirements Definition Process

8. Start high-level, low-detail and iterate

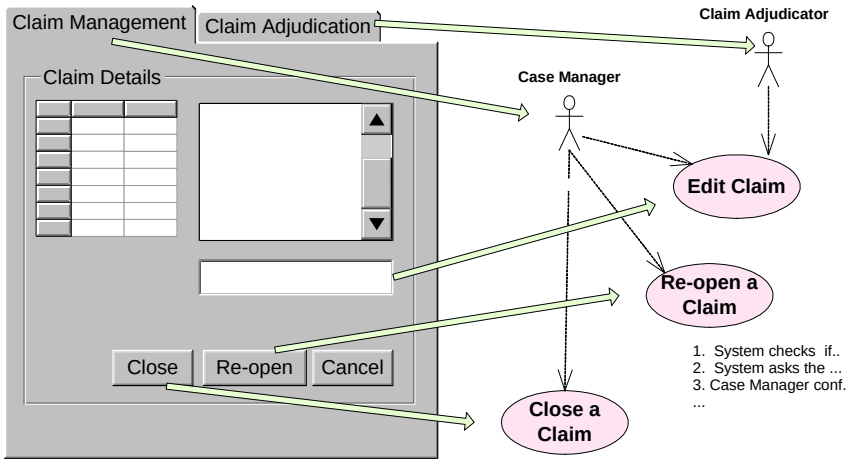
During the requirements definition process capture the high-level knowledge first, as it will further drive the definition of more granular requirements. Although this is generally a "common sense" principle, it is also an often violated one. Do not push for more detail until you, as well as the other stakeholders (!), are comfortable with the present level of detail, as this is key to maintaining requirements in-check, and users satisfied. As far as the requirements definition process is concerned, you should never proceed to the next iteration [3] of the Requirements Generation until the Requirements Inspection phase (Figure 7) is successfully completed and all issues and errors resolved with the stakeholders.

9. Use interface prototyping to elicit and analyze requirements

Interface prototyping has very different objectives from proof-of-concept prototyping. Few main objectives of user interface prototyping are:

- To elicit requirements through visual presentation and dialog with users and more specifically to discover new actors and use cases
- To gather users feedback on visual look and navigation
- To educate users and reduce implementation risks associated with look and feel and behavior exposed through the GUI

The first objective listed is of the most help for requirements analysis as it allows discovering new and refining existing requirements, actors, and use cases (Figure 8). An alternative is to model requirements visually using UML or another graphical presentation scheme.



[3] Do not mix up an iteration of the Requirements Definition Process (RDP) with a concept of an iteration that exists in RUP and other project methodologies as they are not the same thing. Aligning the iterations of the RDP with iterations of RUP is a project-level activity.

Figure 8 Example of using interface prototyping to elicit requirements

10. Model requirements visually

Draw a picture or two to promote the requirements you created. Not only the functional requirements, such as use cases, may be depicted, but the non-functional requirements and constraints can be visualized as well (Figure 9). If visual modeling is not a part of the methodology you selected, then you may still visualize requirements using a standard notation or a free format.

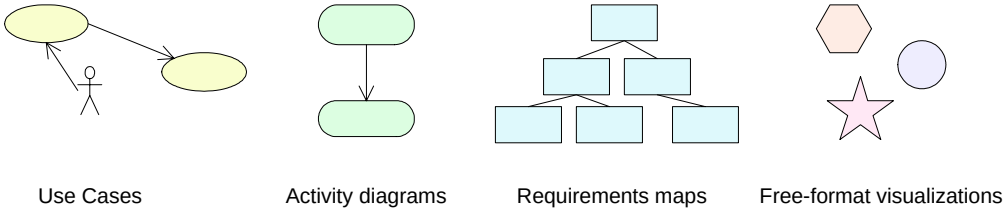


Figure 9 Sample types of requirement visualizations

Note that some requirement tools support automatic requirements visualization [4] a feature that may help to reduce duration and improve precision of your work. Interface requirements prototyping is yet another visualization technique that may be useful from time to time.

11. Consider using requirements for planning purposes

A well organized requirements specification is a highly valuable source of information about the project. The simplest thing to do would be to take the requirements structure you created and export it as a hierarchy that can be loaded in a project management tool. Majority of requirement tools let you do just that, while some, like Optimal Trace, ship with the pre-defined format map for the Microsoft Project tool (Figure 10).

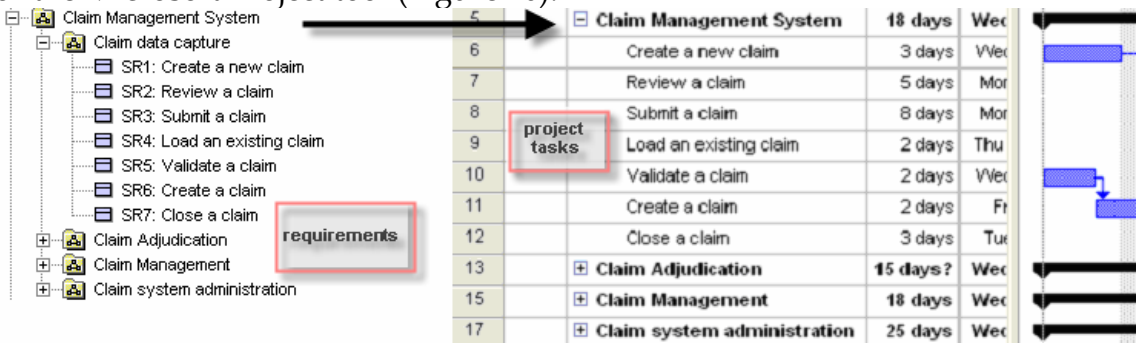


Figure 10 Example of using requirements for project planning

Conclusion

I have to run, but here are some more tips:

- Start requirements work early
- Obtain as much theoretical knowledge of the business domain as possible
- Study the existing solutions
- Analyze usage scenarios and using environment for the solution
- Use collaborative tools, such as forums to discuss requirements
- Start and complete requirements elicitation with the dialog with the users
- Do not create multiple editable instances of the requirements document