# 3

# Standard Plan for requirements



The concepts and principles of the previous two chapters lead to a division of the requirements into a "*Standard Plan*" made of four parts covering the four PEGS of requirements engineering. The present discussion explains what should appear in each of them.

To use the Standard Plan and its variants, you have access to free resources on this Handbook's web site including templates in major text-processing formats.

The reference for requirements documents has long been the structure outline defined in a 1998 IEEE standard, a response to the simpler needs of systems in another era. The present Standard Plan is a replacement, designed to meet the challenges of today's ambitious projects.

## 3.1 Overall structure

We saw in 1.1.1 that requirements have four major dimensions or PEGS: Project, Environment, Goals, System. The Standard Plan correspondingly includes four *books*. Each book is divided into *chapters* (not to be confused with chapters of this Handbook), themselves made of *sections*. The table below gives the list of books and chapters, with the details coming next.

| The four books of requirements | |
|---|---|
| **Project (P)**<br>P.1 Roles and personnel<br>P.2 Imposed technical choices<br>P.3 Schedule and milestones<br>P.4 Tasks and deliverables<br>P.5 Required technology elements<br>P.6 Risk and mitigation analysis<br>P.7 Requirements process and report | **Goals (G)**<br>G.1 Context and overall objective<br>G.2 Current situation<br>G.3 Expected benefits<br>G.4 Functionality overview<br>G.5 High-level usage scenarios<br>G.6 Limitations and exclusions<br>G.7 Stakeholders and requirements sources |
| **Environment (E)**<br>E.1 Glossary<br>E.2 Components<br>E.3 Constraints<br>E.4 Assumptions<br>E.5 Effects<br>E.6 Invariants | **System (S)**<br>S.1 Components<br>S.2 Functionality<br>S.3 Interfaces<br>S.4 Detailed usage scenarios<br>S.5 Prioritization<br>S.6 Verification and acceptance criteria |

## 3.2 FRONT AND BACK MATTER

The preceding standard structure covers the actual content of the books. Companies typically have specific practices regarding internal and external documents. For that reason, each book may include, in addition to the chapters listed on the previous page:

- Front matter, before the first chapter.
- Back matter, after the last chapter.

Typical *front matter* may include:

- Title of the book (usually occupying the entire first page, with the following elements typically making up the second page).
- A general reminder about the project, a mention that the current book is one of four covering that project, and references to the other three.
- Date of publication of first version and of current version; revision history.
- Table of contents and any other appropriate tables, such as a table of illustrations. (But not the glossary, which is part of the contents: chapter E.1.)
- Copyright notice, intellectual property, distribution information, restrictions on distribution.
- Approval information. (Some organizations require every document to be approved by authorized executives, and sometimes to include an actual signature. This practice has specific relevance to one of the requirements qualities: "Endorsed", 4.14, page 68.)

Some of these elements might instead appear in the *back matter*. Another typical back-matter element is an index. Front and back matter are *metarequirements* as defined in 1.9.4, page 13.The remainder of this discussion ignores front and back matter and focuses on the rest of the four requirements books, consisting of the chapters listed in the overall plan of 3.1.

## 3.3 USING THE PLAN

The following observations help understand the scope of the recommended plan and use it effectively in practice.

### 3.3.1 Forms of requirements conforming to the Standard Plan

The term "book" does not imply a traditional requirements process all focused on producing a linear "requirements document" (here with four parts). In accordance with the Requirements Repository Principle (page 26), requirements can consist of many elements, such as textual documents, PowerPoint slides, use cases, emails and others, as long as they are all recorded in a repository. The division into books means two things:

- Every such element should be marked as belonging to (exactly) **one of the chapters of the Standard Plan** as listed in the present discussion.
- It should be possible to obtain, for each book, a **linear form** — a single document, akin to a part of a traditional requirements document. If the actual requirements elements are scattered in various places accessible from a repository, **tools** can produce the linear form.

This Handbook's site provides templates (HTML, MS Word, LaTex…) for writing the linear forms directly, and tools to help produce linear forms from scattered elements.

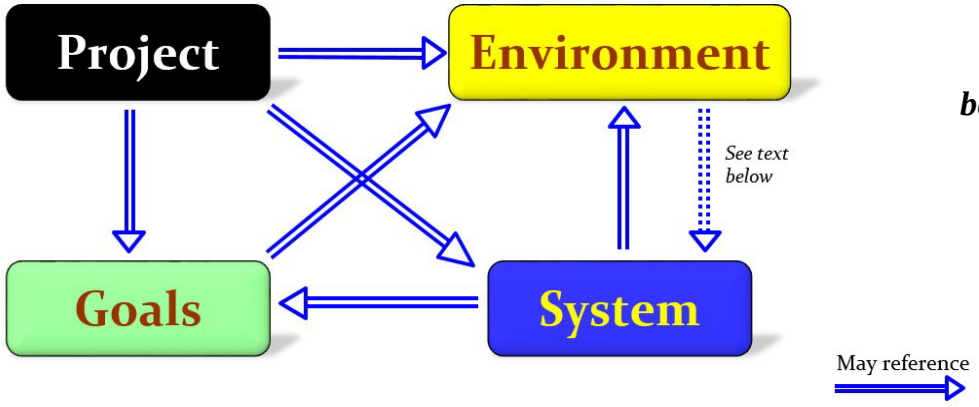
### 3.3.2 Customizing the plan

Many organizations have specific requirements practices and recommendations. The plan is designed to support tailoring the requirements to such policies within a common framework:

- The first two levels, consisting of **books** (P, E, G, S) and **chapters** (P.1, P.2…, E.1 etc.) are universally applicable. That part of the structure should remain as given in 3.1.
- Starting with third-level **sections** (e.g. P.1.1), every organization can refine the structure to fit its specific requirements and software engineering practices.

### 3.3.3 Mutual references

The books may reference each other, but not arbitrarily. The following diagram shows which books may refer to which. (It only governs the chapters of substance, ignoring the front and back matter as introduced in 3.2.)




***Possible references between the books***

The rules and explanations are the following:

- The Project book may refer to all the others.
- The Environment book normally does not refer to the Goals and Project books, since the existing environment predates the project. (But all others may refer to it, as implied by the definition of "environment" at the onset of this Handbook in 1.1.1: "*the set of* [external] *entities* […] *with the potential to affect the goals, project or system or be affected by them*".)
- The dotted downward arrow signals an exception to the preceding rule: the Environment book may refer to the System to describe possible changes to the environment's properties caused by the system. In a payroll system, for example, the environment defines payroll practices; but the system may by itself bring changes to the payroll process, requiring the Environment book to describe these changes. Such cases may appear only in chapters E.5 and E.6 ("effects" and "invariants") of the Environment book.
- The Goals book must be self-contained except for possible references to the Environment. It is indeed essential to explain the goals independently of how the Project plans to achieve them and how the System will meet them.

- The other way around, the descriptions of both Project and System will refer to the Goals.
- The System book must not refer to the Project book, since it is necessary to describe the system independently of when and how it will be developed.
- The other way around, the Project will reference components of the System, in particular as part of the development schedule and tasks (P.3, P.4).

The following sections give more details about the books, in the order Goals, Environment, System, Project.

## 3.4 THE GOALS BOOK

Goals are "*needs of the target organization, which the system will address*" (1.5). While the development team is the principal user of the other books, the Goals book addresses a wider audience: essentially, all stakeholders. As consequences:

- It must contain enough information to provide — if read just by itself — providing a general sketch of the entire project. To this effect, chapter G.3 presents a short overview of the system and G.1 will typically include some key properties of the environment.
- It may contain references to all three other books (satisfying the restrictions of the figure on the previous page) for readers who want to know the details.
- As it addresses a wide readership, it should be clear and minimize the use of specialized technical terms. (Chapter 5 is devoted to style guidelines for requirements writing, and 5.9 specifically to rules governing the Goals book.)

Here is the structure of the Goals book with an explanation of the role of each chapter:

| **Goals book (G)** | |
|---|---|
| **Chapter** | **Contents** |
| ***Front matter*** | Control information (3.2). |
| **G.1 Context and overall objective** | High-level view of the project: organizational context and reason for building a system. |
| **G.2 Current situation** | Current state of processes to be addressed by the project and the resulting system. |
| **G.3 Expected benefits** | New processes, or improvement to existing processes, made possible by the project's results. |
| **G.4 Functionality overview** | Overview of the functions (behavior) of the system. Principal properties only (details are in the System book). |

| | |
|---|---|
| **G.5 High-level usage scenarios** | Fundamental usage paths through the system. |
| **G.6 Limitations and exclusions** | Aspects that the system need not address. |
| **G.7 Stakeholders and requirements sources** | Groups of people who can affect the project or be affected by it, and other places to consider for information about the project and system |
| ***Back matter*** | Control information (3.2). |

Notes on the chapters of the Goals book:

- G.1 is a general introduction. It explains why the project is needed, recalls the business context, and presents the general business objectives. If the project's complexity requires a division into subprojects, that division should be stated here.
- G.2 describes the current situation, upon which the system is expected to improve.
- G.3 presents the business benefits expected from the successful execution of the project. This chapter is the core of the Goals book, describing what the organization expects from the system (hence its highlighting in the table).

  Together, G.1, G.2 and G.3 describe the rationale for the project. It is important to state these justifications explicitly. Typically, they are well understood at the start of the project, but management and priorities can change. Later on, executives may start questioning the project's value. Having a cogent writeup of its intended benefits can help save it from cancellation. Another use of these chapters, G.3 in particular, is to ensure that the project remains focused: if at some stage it gets pushed in different directions, with "creeping featurism" threatening its integrity, a reminder about the original business goals stated in those chapters will help.
- G.4 is a short overview of the functions of the future system, a kind of capsule version of book S, skipping details but enabling readers to get a quick grasp of what the system will do.
- G.5 presents the main scenarios (use cases) that the system should cover. The scenarios chosen for appearing here, in the Goals book, should only be the main usage patterns, without details such as special and erroneous cases; they should be stated in user terms only, independently of the system's structure. Detailed usage scenarios, taking into account system details and special cases, will appear in the System book, chapter S.4 (see below page 42).
- G.6 states what the system will *not* do. This chapter addresses a key quality attribute of good requirements, which we will see in 4.9: the requirements must be **delimited** (or "scoped"). G.6 is not, however, the place for an analysis of *risks* and obstacles, which pertain to the project rather than the goals and correspondingly appears in chapter P.6.
- G.7 lists stakeholders and other requirements sources. It should define stakeholders (1.1.4, 1.10.1) as categories of people, not individuals, even if such individuals are known at the time of writing; for example "company CEO" rather than the person's name. The main

goal of chapter G.7 is to avoid forgetting any category of people whose input is relevant to the project.

To identify stakeholders, you should start from the detailed lists in "Categories of stakeholders", 1.10.1, page 13, which presents general categories applicable to many projects, on both the production and target sides. Starting from this common basis, every project may need its own customized list. See further discussions of stakeholders in chapter 6 of this Handbook.

Chapter G.7 also lists documents and other information that the project, aside from soliciting input from stakeholders, can consult for requirements information. Examples include: people who have relevant information (without being considered stakeholders of the project); documents predating the project, such as emails and minutes of meetings which discussed launching it; information about previous projects; documentation of existing systems in the marketplace pursuing similar goals; information about projects at competing organizations; industry standards; web sites providing background information. For a detailed discussion see "Sources other than stakeholders", 6.3, page 106.

## 3.5 THE ENVIRONMENT BOOK

The Environment book describes the application domain and external context, physical or virtual (or a mix), in which the system will operate.

Here is the structure of the Environment book:

| Environment book (E) | |
|---|---|
| **Chapter** | **Contents** |
| ***Front matter*** | Control information (3.2). |
| **E.1 Glossary** | Clear and precise definitions of all the vocabulary specific to the application domain, including technical terms, words from ordinary language used in a special meaning, and acronyms. |
| **E.2 Components** | List of elements of the environment that may affect or be affected by the system and project. Includes other systems to which the system must be interfaced. |
| **E.3 Constraints** | Obligations and limits imposed on the project and system by the environment. |
| **E.4 Assumptions** | Properties of the environment that may be assumed, with the goal of facilitating the project and simplifying the system. |
| **E.5 Effects** | Elements and properties of the environment that the system will affect. |
| **E.6 Invariants** | Properties of the environment that the system's operation must preserve. |
| ***Back matter*** | Control information (3.2). |

Notes on chapters of the Environment book:

- The glossary (E.1) introduces the terminology of the project; not just of the environment in the strict sense, but of all its parts. As discussed in a later chapter ("The glossary", 6.4, page 107), every organizational project has its own terminology, not always obvious to outsiders and in particular to developers; the glossary collects and defines the relevant terms.

- Chapter E.2 lists the elements of the environment that are relevant for the project and system. These components may include existing systems, particularly *software* systems, with which the system will interact — by using their APIs (program interfaces), or by providing APIs to them, or both. These are interfaces provided to the system *from* the outside world. They are distinct from both: interfaces provided by the system *to* the outside world (covered in S.3 in the System book); and technology elements that the system's development will require (covered in P.5 in the Project book).

- While all chapters are important, the table highlights E.3, Constraints, since this chapter defines non-negotiable restrictions coming from the environment (business rules, physical laws, engineering decisions), which the development will have to take into account. An example, in addition to those of 1.7, is the constraint that a system's user interface must accommodate screen resolutions as low as 1280 x 720 (HD) and as high as 3840 x 2160.

- E.4, Assumptions, defines properties that are not imposed by the environment (like those in E.3) but assumed to hold, as an explicit decision meant to facilitate the system's construction. An example, in addition to those of 1.7, is the assumption that available bandwidth (for a networked system) will be no less than 1 Gigabit per second. Another, to be compared with the preceding constraint example, is the assumption that the system *need not support* resolutions lower than 1280 x 720 or higher than 3840 x 2160. (A possibly risky assumption, especially the second part since screen technology evolves quickly, but one that may be necessary to enable prompt completion of the project.)

  If you ever hesitate between classifying a condition as a constraint or an assumption, go back to section 1.7.4, which gave practical criteria for deciding between these two cases.

- E.5 defines effects of the system's operations on properties of the environment. Where the previous two categories defined influences of the environment on the system, effects are influences in the reverse direction. 1.7 gave an example for a business system: the effect of a payroll system on the payroll process itself, such as changing the dates on which employees are paid. Another example, for an embedded system, is its maximum carbon footprint.

- E.6 defines invariants: properties of the environment that operations of the system may assume to hold when they start, and must maintain. In other words, influences in both directions. An example invariant for a train control system may be "doors must be closed whenever the train is moving". Any operation — for example, starting or stopping the train — may assume this property on inception, and it is also required to maintain it.

## 3.6 THE SYSTEM BOOK

Here is the overall structure of the System book:

| System book (S) | |
|---|---|
| **Chapter** | **Contents** |
| ***Front matter*** | Control information (3.2). |
| **S.1 Components** | Overall structure expressed by the list of major software and, if applicable, hardware parts. |
| **S.2 Functionality** | One section, S.2.*n*, for each of the components identified in S.2, describing the corresponding behaviors (functional and non-functional properties). |
| **S.3 Interfaces** | How the system makes the functionality of S.2 available to the rest of the world, particularly *user* interfaces and *program* interfaces (APIs). |
| **S.4 Detailed usage scenarios** | Examples of interaction between the environment (or human users) and the system: use cases, user stories. |
| **S.5 Prioritization** | Classification of the behaviors, interfaces and scenarios (S.2, S.3 and S.4) by their degree of criticality. |
| **S.6 Verification and acceptance criteria** | Specification of the conditions under which an implementation will be deemed satisfactory. |
| ***Back matter*** | Control information (3.2). |

Notes on chapters of the System book:

- S.1 lists the constituent parts of the system. Another term, which chapter 12 will use as part of the lifecycle discussion (see 12.5, page 217), is **clusters**. A cluster is a major subsystem. The cluster structure may be flat (for a simple system) or multi-level. For example, a web-based system may at the top level be split into "front-end" (Web interface) and "back-end", where the back-end has a data access cluster, a computation cluster and a business logic cluster.
- As the highlighting suggests, the bulk of the System book is typically chapter S.2, describing elements of functionality (behaviors). This chapter corresponds to the traditional view of requirements as defining "what the system does".
- Chapter S.3 specifies how that functionality will be made available to the rest of the world, including people (users) and other systems. These are interfaces provided by the system *to* the outside; the other way around, interfaces *from* other systems, which the system may use, are specified in E.2 in the Environment book (discussed in the comments part of 3.5 above.)

- S.4 describes typical usage scenarios, expressed for example as **use cases** or **user stories** (chapter 7 of this Handbook). Such scenarios are not by themselves a substitute for precise descriptions of functionality (S.3), but provide an important complement by specifying cases that these behavior descriptions must support; they also serve as a basis for developing test cases. The scenarios most relevant for stakeholders are given in chapter G.5 in the Goals book, at a general level; in contrast, S.4 can refer to system components and functionality (from other chapters of the System book) as well as special and erroneous cases, and introduce more specific scenarios.
- S.5 defines priorities, useful in particular if during the course of the project various pressures force the team to drop certain functions. Good requirements prioritize functionality, as discussed in "Prioritized", 4.13, page 67 and "Get stakeholders to prioritize", 6.10.5, page 121.
- S.6 defines the criteria under which the system will be deemed acceptable. Remember (terminology note on page 20) that this Handbook uses "verification" as shorthand for what is more explicitly called "Verification & Validation" (V&V), covering several levels of testing — module testing, integration testing, system testing, user acceptance testing — as well as other techniques such as static analysis and, when applicable, program proving.

## 3.7 THE PROJECT BOOK

Here is the overall structure of the project book:

| Project book (P) | |
|---|---|
| **Chapter** | **Contents** |
| ***Front matter*** | Control information (3.2). |
| **P.1 Roles and personnel** | Main responsibilities in the project; required project staff and their needed qualifications. |
| **P.2 Imposed technical choices** | Any a priori choices binding the project to specific tools, hardware, languages or other technical parameters. |
| **P.3 Schedule and milestones** | List of tasks to be carried out and their scheduling. |
| **P.4 Tasks and deliverables** | Details of individual tasks listed under P.3 and their expected outcomes. |
| **P.5 Required technology elements** | External systems, hardware and software, expected to be necessary for building the system. |
| **P.6 Risk and mitigation analysis** | Potential obstacles to meeting the schedule of P.4, and measures for adapting the plan if they do arise. |
| **P.7 Requirements process and report** | Initially, description of what the requirements process will be; later, report on its steps. |
| ***Back matter*** | Control information (3.2). |

Notes on chapters of the Project book:

- P.1 defines the roles involved in the project. "Role" was defined in 1.5 as a human responsibility. Some of the most frequent roles appeared in "Stakeholders on the production (development) side", page 15. Chapter P.1 of the Standard Plan also describes the actual personnel available, their characteristics, any constraints on their time — for example whether they are full-time on the project or also needed elsewhere — and conditions on the possible induction of further personnel throughout the project.
- P.2 exists because in practice not all technical choices in projects derive from a pure technical analysis; some result from company policies ("*we only use open-source products*" or "*the policy is to use our own cloud storage system, not Microsoft, Google or Amazon solutions*"). While some project members may dislike non-strictly-technical decisions, they are a fact of project life and must be documented, in particular for the benefit of one of the quality factors for requirements: "Justified", 4.2, page 49. P.2 addresses this need.
- The schedule in P.3 defines the project's key dates.
- The tasks and deliverables in P.4 define the project's main activities and the results they must produce, associated with the milestone dates defined in P.3. For a classic Waterfall-like project, the tasks correspond to successive steps such as requirements, design and implementation, but the role of P.4 is more general. For example agile projects do not have a traditional succession of phases such as requirements, design etc., but they have iterations or "sprints", and may have milestones and deliverables defined by a legal contract. As the highlighting suggests, P.4 typically makes up the core part of the Project book.
- Chapter P.5 lists external technology elements, such as program libraries and hardware devices, that the project is expected to require. Although the actual use of such products belongs to design and implementation rather than requirements, it is part of the requirements task to identify elements whose availability is critical to the success of the project — an important element of risk analysis (P.6 as presented next). It will also be advisable to perform early checks of the actual availability and suitability of the technology ("Feasibility prototypes", 6.11.5, page 123).
- P.6 discusses risks associated with the project; it can include a SWOT analysis (Strengths, Weaknesses, Opportunities, Threats) for the project. It is essential to be on the lookout for events that could derail the project, and devise mitigation strategies. Two of the preceding chapters of the Project book yield sources of risk, since they express choices that may have been made without the benefit of a fully rational analysis: imposed technical choices (P.2) and external technology elements as just discussed (P.5). The analysis of risks in P.6 is distinct from chapter G.6 of the Goals book (page 39), which covers limitations and exclusions of the system's goals, although it may need to refer to it. A later section of this handbook discusses the topic of risk further: "Risk assessment and mitigation", 6.11.7, page 126.
- The Process and Report chapter, P.7, starts out as a plan for conducting the requirements elicitation process (covered by chapter 6 of this Handbook), but is meant to be updated as part of that process so that it includes the key lessons of elicitation; see sections 6.8 and 6.9.

## 3.8 MINIMUM REQUIREMENTS

It is reasonable to expect any project to produce requirements; the four-book structure presented in this chapter serves as a checklist for all the elements that may be needed.

Not all requirements efforts will fill in the entire structure at the same level of detail. In line with the general "Requirements Effort Principle", page 31, there is no reason to feel bad if the requirements are not textbook-perfect.

You should also, however, remember another principle, "Minimum Requirements Outcome Principle", page 27, which states the minimum that any project should have to show for its requirements efforts. Regardless of where your lifecycle model (chapter 12) lies in the spectrum from full-agile to full-Waterfall, these are the "upfront" elements which the project should have at its disposal before it can comfortably embark on other software development tasks.

Remember, too, that not all requirements are "upfront". Yet another principle, "Requirements Elaboration Principle", page 25, reminds us to continue working on the requirements as the project proceeds and accumulates new information about both the problem and the solution.

## 3-E EXERCISES

### 3-E.1 Finding the right place

State the chapter from the Standard Plan (such as "P.1") to which each one of the following example requirements would belong:

- "*Some of the general constraints were defined in the preliminary meeting of 15 June 2022, available at*" [URL].
- "*The login record shall be implemented using MongoDB.*"
- "*Here is the basic scheme of interaction for ordering a product:*" [followed by the description of that scheme].
- "*The project shall only use external software products available through an approved open-source license (GPL or Creative Commons)*".
- "*The product shall be available on mobile platforms as well as through an API.*"
- "*Any use of cookies shall conform to the GDPR.*"
- "*As a result of the introduction of the new payroll system, pay periods shall be standardized to monthly for all employees.*"
- "*As the system depends on Windows 11 facilities, meeting the schedule depends on Microsoft fully releasing Windows 11 by end of October, 2021.*"
- "*This function is considered critical to the deployment of the project.*"
- "*Upon exiting a session, the system shall memorize the last explored directory as the restart point for the next session.*"

### 3-E.2 Restructuring a requirements document

(Project exercise.) Consider the sample requirements document given in [Bair 2005]. Rewrite it as three books (excluding the Project book since the document does not address project organization) according to the Standard Plan of this chapter. (The document leaves out some figures, usually using a variable *x*, as in "*reduce the number of support calls from foreign customers by x%*". Make sure to replace such placeholders by actual numbers.)

### 3-E.3 Devising a project plan

Referring again to [Bair 2005] (previous exercise), devise a suitable project plan for the system's development, and write it as a Project book according to the Standard Plan of this chapter.

### 3-E.4 Use cases in different places

(This exercise is about the Standard Plan discussed in the present chapter, but assumes familiarity with use cases, which you can obtain from reading chapter 7.) Consider examples of use cases from a standard reference such as [Jacobson 1992] or [Cockburn 2001]; alternatively, from an actual requirements document to which you have access. Starting from the criteria presented in sections 3.4 and 3.6, state which of those use cases would, in the Standard Plan, go into chapter G.5 of the Goals book ("*high-level usage scenarios*") and which into chapter S.4 of the System book ("*detailed usage scenarios*"). Explain the rationale for your decisions.

## *BIBLIOGRAPHICAL NOTES AND FURTHER READING*

Until now, many industry projects have been applying a recommended structure for requirements document published in 1998 by a committee of the IEEE: [IEEE 1998]. Although it remains good (and short) reading, it does not measure up to the needs of the complex systems we develop today.

Gilb's Planguage work [Gilb 2005-2022] includes a systematic catalog of the elements needed in requirements specifications.