*Texto elaborado por António Eloi (Unidade de Gestão do Projeto) de Engenharia de Software 2018/2019 - PL5*

*Nota 1:* Estas informações deverão ser dadas nas aulas teóricas de E.S..

**Requirements Team (page 82-118)**

The requirements for a system are the descriptions of what the system should do - the services that it provides and the constraints on its operation.

The process of **finding out**, **analyzing**, **documenting** and **checking** these services and constraints is called **requirements engineering** (RE).

***User requirements vs System requirements***

* **User requirements** are statements, in a natural language plus diagrams, of what services the system is expected to provide to system users and the constraints under which it must operate.
* **System requirements** are more detailed descriptions of the software system’s functions, services, and operational constraints. The system requirements document (sometimes called a functional specification) should define exactly what is to be implemented. It may be part of the contract between the system buyer and the software developers.

**You need to write requirements at different levels of detail because different readers use them in different ways.**

**Software system requirements are often classified as functional requirements or nonfunctional requirements:**

***Functional requirements vs Non-Functional requirements***

* **Functional requirements** - These are statements of services the system should provide, how the system should react to particular inputs, and how the system should behave in particular situations. In some cases, the functional requirements may also explicitly state what the system should not do.
* **Non-functional requirements** - These are constraints on the services or functions offered by the system. They include timing constraints, constraints on the development process, and constraints imposed by standards. Non-functional requirements often apply to the system as a whole, rather than individual system features or services.

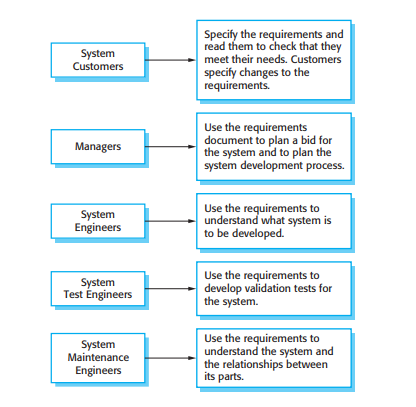
Os requerimentos funcionais descrevem o que um sistema deve fazer. Dependem do software a ser desenvolvido, dos seus utilizadores e o que a empresa normalmente faz quando tipicamente escreve requerimentos. Exemplos:

1. Um investigador deve poder pesquisar por restantes investigadores
2. O sistema deve gerar todos os dias relatórios de número de utilizadores
3. Cada utilizador deve ser identificado pelo nome de utilizador e palavra-passe

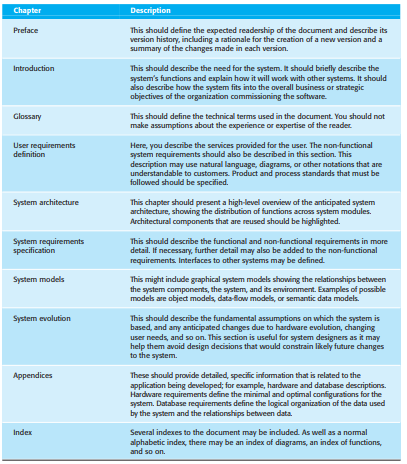
Os requerimentos não-funcionais não estão diretamente ligados com o serviço específico. São requerimentos que fazem parte do projeto todo e não de um simples componente.

***Software requirements document***

The software requirements document (sometimes called the software requirements specification or SRS) is an official statement of what the system developers should implement.



Exemplo de um documento de requerimentos:



O relatório de requerimentos deve também ter as necessidades de todo o programa.

***Interviews***

Formal or informal interviews with system stakeholders are part of most requirements engineering processes. In these interviews, the requirements engineering team puts questions to stakeholders about the system that they currently use and the system to be developed. Requirements are derived from the answers to these questions. Interviews may be of two types:

* **Closed interviews**, where the stakeholder answers a pre-defined set of questions.
* **Open interviews**, in which there is no pre-defined agenda. The requirements engineering team explores a range of issues with system stakeholders and hence develop a better understanding of their needs.

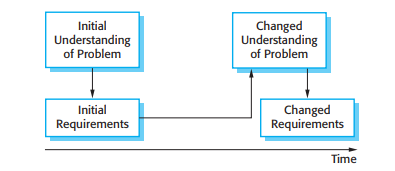
As entrevistas devem por vezes recorrer a cenários e não se deve limitar o entrevistado.

***Requirements validation***

Requirements validation is the process of checking that requirements actually define the system that the customer really wants. It overlaps with analysis as it is concerned with finding problems with the requirements. Requirements validation is important because errors in a requirements document can lead to extensive rework costs when these problems are discovered during development or after the system is in service.

During the requirements validation process, different types of checks should be carried out on the requirements in the requirements document. These checks include:

* **Validity checks -** A user may think that a system is needed to perform certain functions. However, further thought and analysis may identify additional or different functions that are required. Systems have diverse stakeholders with different needs and any set of requirements is inevitably a compromise across the stakeholder community.
* **Consistency checks** - Requirements in the document should not conflict. That is, there should not be contradictory constraints or different descriptions of the same system function.
* **Completeness checks -** The requirements document should include requirements that define all functions and the constraints intended by the system user.
* **Realism checks -** Using knowledge of existing technology, the requirements should be checked to ensure that they can actually be implemented. These checks should also take account of the budget and schedule for the system development.
* **Verifiability -** To reduce the potential for dispute between customer and contractor, system requirements should always be written so that they are verifiable. This means that you should be able to write a set of tests that can demonstrate that the delivered system meets each specified requirement.



There are a number of requirements validation techniques that can be used individually or in conjunction with one another:

* **Requirements reviews** - The requirements are analyzed systematically by a team of reviewers who check for errors and inconsistencies.
* **Prototyping** - In this approach to validation, an executable model of the system in question is demonstrated to end-users and customers. They can experiment with this model to see if it meets their real needs.
* **Test-case generation** - Requirements should be testable. If the tests for the requirements are devised as part of the validation process, this often reveals requirements problems. If a test is difficult or impossible to design, this usually means that the requirements will be difficult to implement and should be reconsidered. Developing tests from the user requirements before any code is written is an integral part of extreme programming.

**Concluding!**

