Standards are documents created by a team of experts in a specific area so that best practices and guidelines are formalized and disseminated among practitioners. As you may imagine, these documents are fairly specialized, but at the same time, they provide generic guidelines that sometimes need to be instantiated to a specific situation. There is a large number of standards and various international bodies that supervise the process of creating these documents. The typical way to refer to these documents is through its reference number.

Standards can be seen as *convenient meeting points*, which is proposed descriptions on certain area that a body of experts initially agreed on, so that other people may agree as well. For example, if two companies are developing a software tool together, instead of devoting time to decide the structure of a Software Quality Assurance Plan, they may directly use one of these standards. When a company or institution adopts a standard for certain procedures, they usually make it explicit to other stakeholders.

Although standard documents are terse and challenging to read in depth, they usually provide a comprehensive description of all the aspects that need to be taken into account within a specific area and, most importantly, help to organize numerous tasks. In our case, we will focus only on two of these documents: IEEE 730-2002 and ISO/IEC 25010:2011. It follows a very brief description of each of the standards.

**4.5.1. The IEEE 730-2002 Standard**

This standard describes what is a Software Quality Assurance Plan and the type of document that is required to describe such plan. The 17 page document gives details on how to prepare a Software Quality Assurance Plan (also known as SQAP) as well as the sections in the document and the type of information that should contain. These are the 16 sections with the subsection titles:

1. Purpose
2. Reference Documents
3. Management: Organization, Tasks, Roles and responsibilities, Quality assurance estimated resources
4. Documentation: Purpose, Minimum documentation requirements, Other documentation.
5. Standards, practices, conventions, and metrics: Purpose, Content
6. Software reviews: Purpose, Minimum requirements, Other reviews and audits
7. Test
8. Problem reporting and corrective action
9. Tools, techniques, and methodologies
10. Media control
11. Supplier control
12. Records collection, maintenance, and retention
13. Training
14. Risk management
15. Glossary
16. SQAP change procedure and history

The content of some of the sections is clear (Purpose, Glossary), but there are others in which a greater level of detail is needed.

**4.5.1.1. Management**

In this section of the document you must elaborate on what is the management structure in place to deploy the quality process. More precisely, the information must describe:

* Organization: Describe the structure that will be responsible for deploying the quality assurance and control mechanisms.
* Tasks: Describe which portion of the production cycle is covered by the SQAP, which tasks are included in the plan, the entry and exit criteria for each task and the relation with the major stages of the project.
* Roles: Identify the responsibilities and how are they divided into these roles.
* Estimated Resources: Describe the resources needed to deploy the plan including assurance and control aspects.

**4.5.1.2. Documentation**

Documentation is one of the most important resources to guarantee quality in a software project. When writing documentation, all interested parties clarify the details of different parts of the system, helping gather a common view of the strategy. The minimum documentation requirements include:

* Software Requirement Specification: A description of what should the system do.
* Software Design Description: A description in blocks of the intended implementation to comply with the software requirements.
* Verification and validation plans: Describe the processes what will be used to verify and validate the application. In here, tasks such as code analysis, inspections, and testing are included.
* Verification and validation results report.
* User documentation.
* Software configuration management plan.

**4.5.1.3. Reviews**

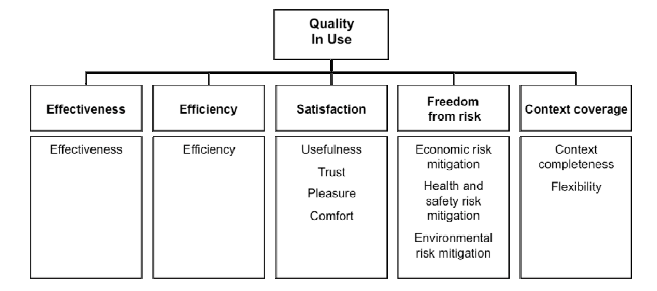
This is one of the most important sections of the SQAP. It must define all the possible review processes, how are they conducted, its scheduling of tasks and the actions required to implement these reviews. The following reviews are considered a minimum:

* Software specifications review (SSR)
* Architecture design review (ADR)
* Detailed design review (DDR)
* Verification and validation plan review
* Functional audit
* Physical audit
* In-process audits
* Managerial reviews
* Software configuration management plan review (SCMPR)
* Post-implementation review

**4.5.2. The ISO/IEC 25010:2011 Standard**

The ISO/IEC 25010:2011 Standard is an updated version of the prior ISO/IEC 9126. They are both international standards for the evaluation of software quality. The document defines the necessary and desired quality characteristics that should be included in a software product. When compared with IEEE 730-2002, this standard provides two models of software quality instead of the structure of a Software Quality Plan. The first model contains the aspects related to quality when referring to the use of an application. The categories and sub-categories of this model are:

* Effectiveness
* Efficiency
* Satisfaction: Usefulness, Trust, Pleasure, Comfort
* Freedom from Risk: Economical risk, Health and safety risk, Environmental risk.
* Context Coverage: Completeness, Flexibility

The following figure illustrates these quality aspects.

Additionally, a quality model is also provided that contains the categories in which quality should focus. If a product contains the procedures to plan, assess and improve quality on each of these categories, then the quality assurance can be considered exhaustive. These categories are (Source ISO/IEC 25010:2011 Standard):

* Functional suitability: The system provides the functions in the requirements.
  + Functional completeness: Application functions cover the objectives.
  + Functional correctness: Application correctly implements the expected functionality.
  + Functional appropriateness: The functions are adequate to achieve the requirements and objectives.
* Performance efficiency: The application executes efficiently with the given resources.
  + Time behavior: The response time and processing time are acceptable.
  + Resource utilization: The product uses the expected resources while executing.
  + Capacity: the limits of the product are within acceptable ranges.
* Compatibility: System must be compatible with other systems, hardware or software environments.
  + Co-existence: The application shares resources with other products.
  + Interoperability: The system exchanges information with other systems.
* Usability: How effective is a system used in terms of efficiency and satisfaction.
  + Appropriateness recognizability: The system is appropriate for the user needs.
  + Learnability: The system contains information on how to be used.
  + Operability: The system is easy to operate.
  + User error protection: The system includes mechanisms to protect users when they make a mistake.
  + User interface aesthetics: Interface is pleasing to the user.
  + Accessibility: The system can be used by people with different capabilities.
* Reliability: How the system performs its function over a period of time.
  + Maturity: The system meets the reliability requirements.
  + Availability: The system is available when required.
  + Fault tolerance: The system reacts as expected in the presence of hardware or software failures.
  + Recoverability: The system may recover from a failure.
* Security: The system provides the right degree of access to data to users.
  + Confidentiality: The data is only accessed by the authorized user.
  + Integrity: The product prevents unauthorized access to data.
  + Non-repudiation: The actions in the system can be proven to have taken place.
  + Accountability: Actions can be traced to a unique entity.
  + Authenticity: the identity of a subject or resource can be proved to be the one claimed.
* Maintainability: How feasible is to modify the system by the maintainers.
  + Modularity: The system is divided into components and change in one of them has minimal impact on the rest.
  + Reusability: The system assets can be reused by another system.
  + Analysability: It is easy to assess the impact of a change in the system.
  + Modifiability: The system can be modified without degrading its quality.
  + Testability: Test criteria can be established.
* Portability: How feasible is for the product to be deployed in a different platform.
  + Adaptability: The system can easily be adapted to new versions of the environment.
  + Installability: The system can be easily installed and uninstalled.
  + Replaceability: The product can be replaced easily by another product with the same purpose.

The following figure illustrates these quality model.