



BITS Pilani presentation

BITS Pilani
Pilani Campus

Dr. N.Jayakanthan



BITS Pilani
Pilani Campus



SE ZG501

Software Quality Assurance and Testing

Lecture No. 3

Characteristics to measure quality of a requirement



- **Necessary:** They must be based on necessary elements
- **Unambiguous:** clear enough to be interpreted in only one way.
- **Concise:** They must be stated in a language that is precise, brief, and easy to read.
- **Coherent:** They must not contradict the requirements.
- **Complete:** They must all be stated fully
- **Accessible:** They must be realistic regarding their implementation (time ,budget ,resource)
- **Verifiable:** inspection, analysis, demonstration, or tests.

Specifying Quality Requirements: The Process

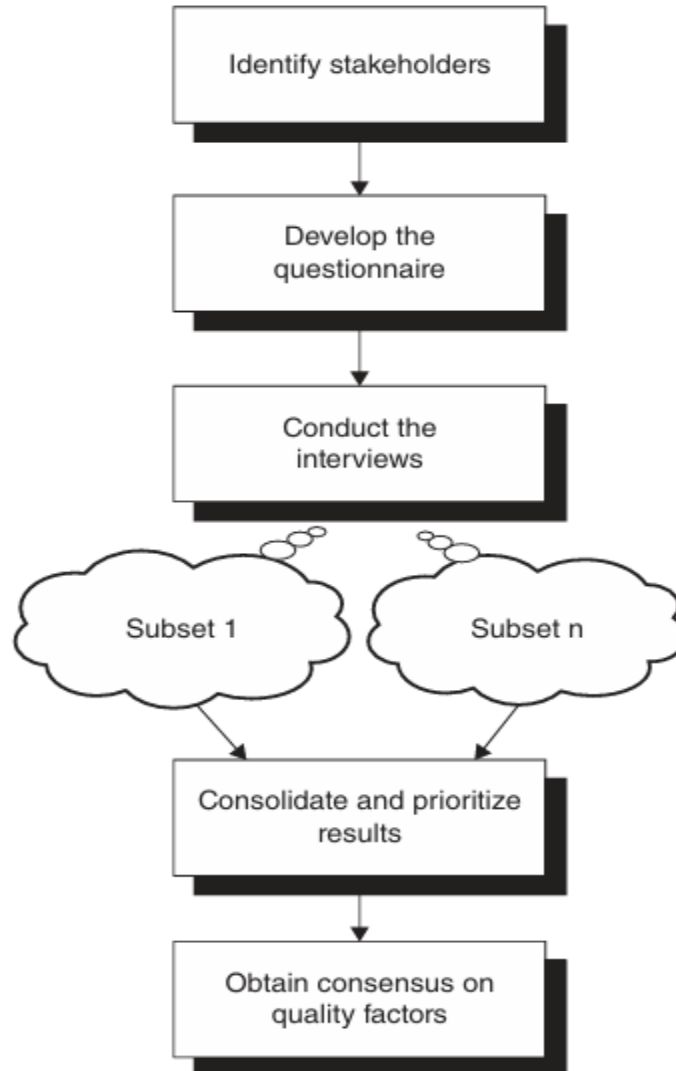


Figure 3.6 Steps suggested for defining non-functional requirements.

- Stakeholders are **any person or organization that has a legitimate interest in the quality of the software.**

These needs and hopes may **change** during the system life cycle and must be checked when there is any change.

- Developing the **questionnaire** that presents the external quality characteristics in terms that are easy to understand for managers.

Table 3.3 Example of Quality Criteria Documentation

Quality characteristics	Importance
Reliability	Indispensable
User-friendliness	Desirable
Operational safety	Non-applicable

Next, for each characteristic, the quality measure must be described in detail and include the following information

- quality characteristic;
- quality sub-characteristic;
- measure (i.e., formula);
- objectives (i.e., target);
- example.

The last step in defining quality requirements involves having these requirements authorized through consensus.



Evaluation of the Functional Capacity of Software

Users often choose this characteristic. It is defined in the specifications as the ability of a software product to carry out all specified requirements. The *ability* sub-characteristic is chosen by the team and described as the percentage of requirements described in the specification document that must be delivered (%E). The measure established is

$$\%E = (\text{Number of functionalities requested} / (\text{Number of functionalities delivered})) \times 100.$$

It is necessary to identify target values as an objective for each measure. It is also recommended to provide an example of the calculations (i.e., measurement) to clearly illustrate the measure. For example, during the writing of the specification, the project team indicates that the %E should be 100% of the requirements described in the specifications document and that they are functional and delivered, without defects, before the final acceptance of the software for production.

Alternatively, with a lack of measurable objectives, the general policy is to accept the most stable version of the code having the necessary functionality.

ISO 25010 [ISO 11i]

REQUIREMENT TRACEABILITY DURING THE SOFTWARE LIFE CYCLE



Throughout the life cycle, client needs are documented and developed in different documents, such as **specifications, architecture, code, and user manuals.**

Throughout the life cycle of a system, many changes regarding client needs should be expected.

Every time a need changes, it must be ensured that all documents are updated.

Traceability is a technique that helps us follow the development of needs as well as their changes.

Software Engineering Standards



Other engineering domains such as mechanical, chemical, electrical, or physics engineering are based on the laws of nature as discovered by scientists.

Hooke's Law

$$\sigma = E \cdot \varepsilon$$

Newton's Law

$$x(t) = \frac{1}{2}a \cdot t^2 + v_0 \cdot t + x_0$$

Boyle-Mariotte's Law

$$p_1 \times V_1 = p_2 \times V_2$$

Curie's Law

$$E = -\vec{\mu} \cdot \vec{B}$$

Refraction Law

$$\eta_1 \cdot \sin(\theta_1) = \eta_2 \cdot \sin(\theta_2)$$

Gravitational Law

$$\vec{F}_{A \rightarrow B} = -G \frac{M_A M_B}{AB^2} \vec{u}_{AB}$$

Ohm's Law

$$V = RI$$

Coulomb's law

$$F_{12} = \frac{q_1 q_2}{4\pi\epsilon_0} \frac{r_2 - r_1}{|r_2 - r_1|^3}$$

Figure 4.1 A few laws of nature used by some engineering disciplines.

Unfortunately, software engineering, unlike other engineering disciplines, is not based on the laws of nature. Software engineering, like other disciplines, is based on the use of **well-defined practices for ensuring the quality** of its products.

In software engineering, there are several standards, which are actually guides for management practices.

A rigorous process serves as the framework for developing and approving standards, including international ISO standards and those from professional organizations like IEEE.

Standard

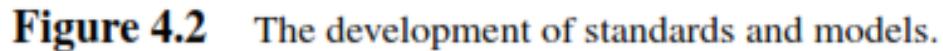
A set of mandatory requirements established by consensus and maintained by a recognized body to prescribe a disciplined and uniform approach, or to specify a product, with respect to mandatory conventions and practices.

The four principles for the development of ISO standards are:

- **ISO standards meet a market need.**
- **ISO standards are based on worldwide expertise.**
- **ISO standards are the result of a multi-stakeholder process.**
- **ISO standards are based on consensus.**

The ISO standards are developed by consensus

- That all parties were able to express their views.
- The best effort has been made to take into account all opinions and solve all problems (i.e., all the submissions in a vote of the draft of a standard).



- American Department of Defence (DoD) created the “DoD-STD1679A” military standard
- IEEE, the International Organization for Standardization (ISO) European Space Agency (ESA) developed standards
- “*Capability Maturity Model*” (CMM®): developed at the request of the American DoD, by the Software Engineering Institute (SEI) in order to provide a road map of engineering practices to improve the performance of the development, maintenance and service provisioning processes.

Figure 4.3 illustrates the evolution of standards that are maintained and published under the responsibility of the appointed subcommittee for standardized processes, tools, and supporting technologies for software engineering and systems:

The Continuous Evolution of Standards

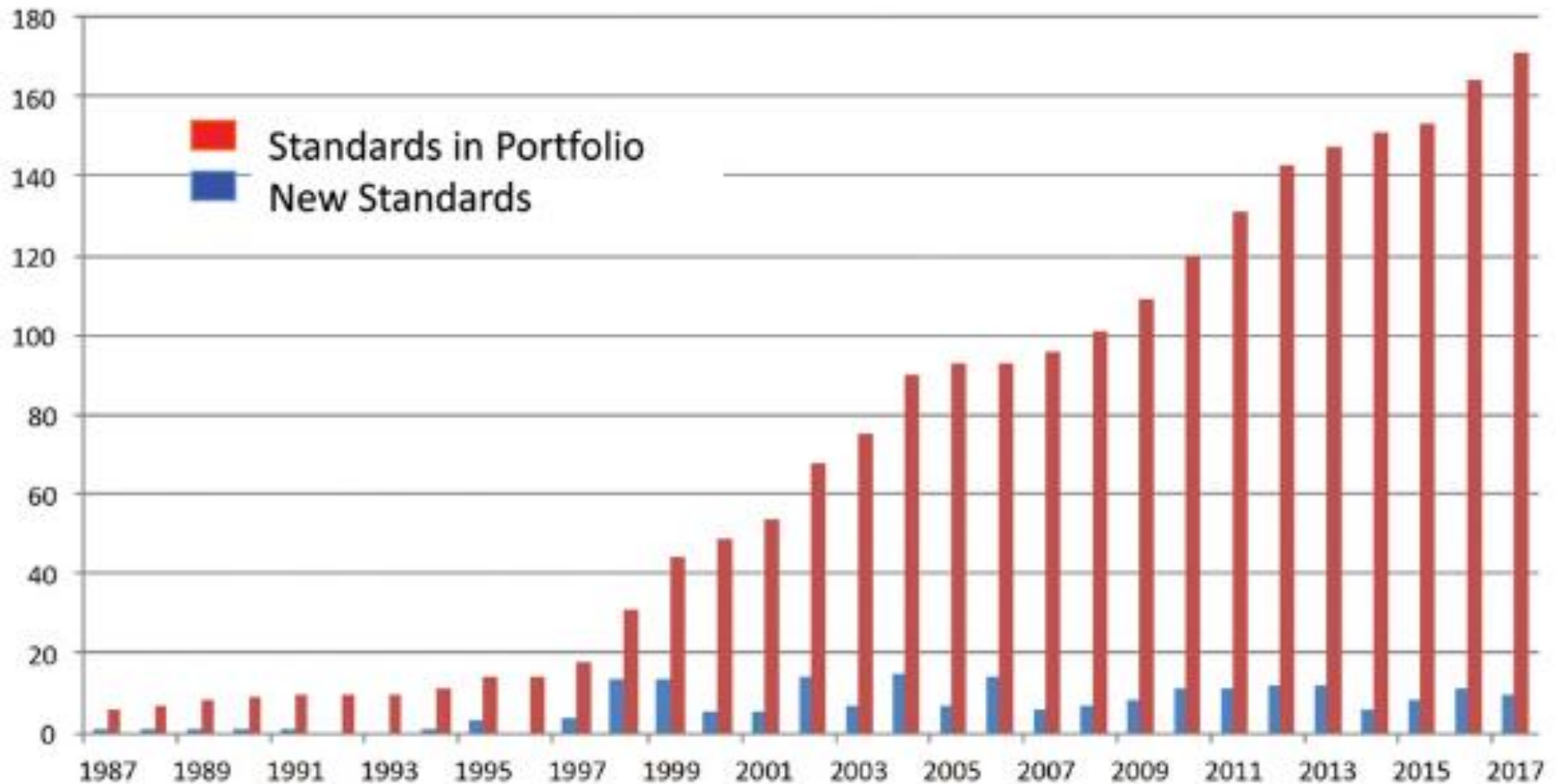


Figure 4.3 The evolution of standards SC7 [SUR 17].

MAIN STANDARDS FOR QUALITY MANAGEMENT



Standards related to the management of software quality:

ISO 9000 [ISO 15b] and ISO 9001 [ISO 15].

The application guide for software, The ISO/IEC 90003 standard.

Standards in the ISO 9000 family include:

- ISO 9001:2015 - sets out the requirements of a quality management system
- ISO 9000:2015 - covers the basic concepts and language
- ISO 9004:2009 - focuses on how to make a quality management system more efficient and effective
- ISO 19011:2011 - sets out guidance on internal and external audits of quality management systems.

- The ISO 9001 standard provides the **basic concepts, principles and vocabulary** of quality management systems (QMS) and is the basis for other standards for QMSs [ISO 15].
- The “**Quality Management Principles**” (QMP) are a set of values, rules, standards, and fundamental convictions regarded as fair and that could be the basis for quality management.

The seven QMP of the ISO 9001



- Principle 1: Customer focus
- Principle 2: Leadership
- Principle 3: Involvement of people
- Principle 4: Process approach
- Principle 5: System approach to management
- Principle 6: Factual approach to decision making
- Principle 7: Mutually beneficial supplier relationships



ISO 9001 uses the process approach, the Plan-Do-Check-Act (PDCA) approach, and a risk-based thinking approach [ISO 15].

- The **process approach** allows an organization to plan its processes and their interactions.
- The PDCA cycle allows an organization to ensure that its processes are adequately resourced and appropriately managed and that opportunities for improvement are identified and implemented.
- The **risk-based thinking** approach allows an organization to determine the factors that may cause deviation from its processes and its QMS in relation to expected results, to implement preventive measures in order to limit negative effects and exploit opportunities when they arise.

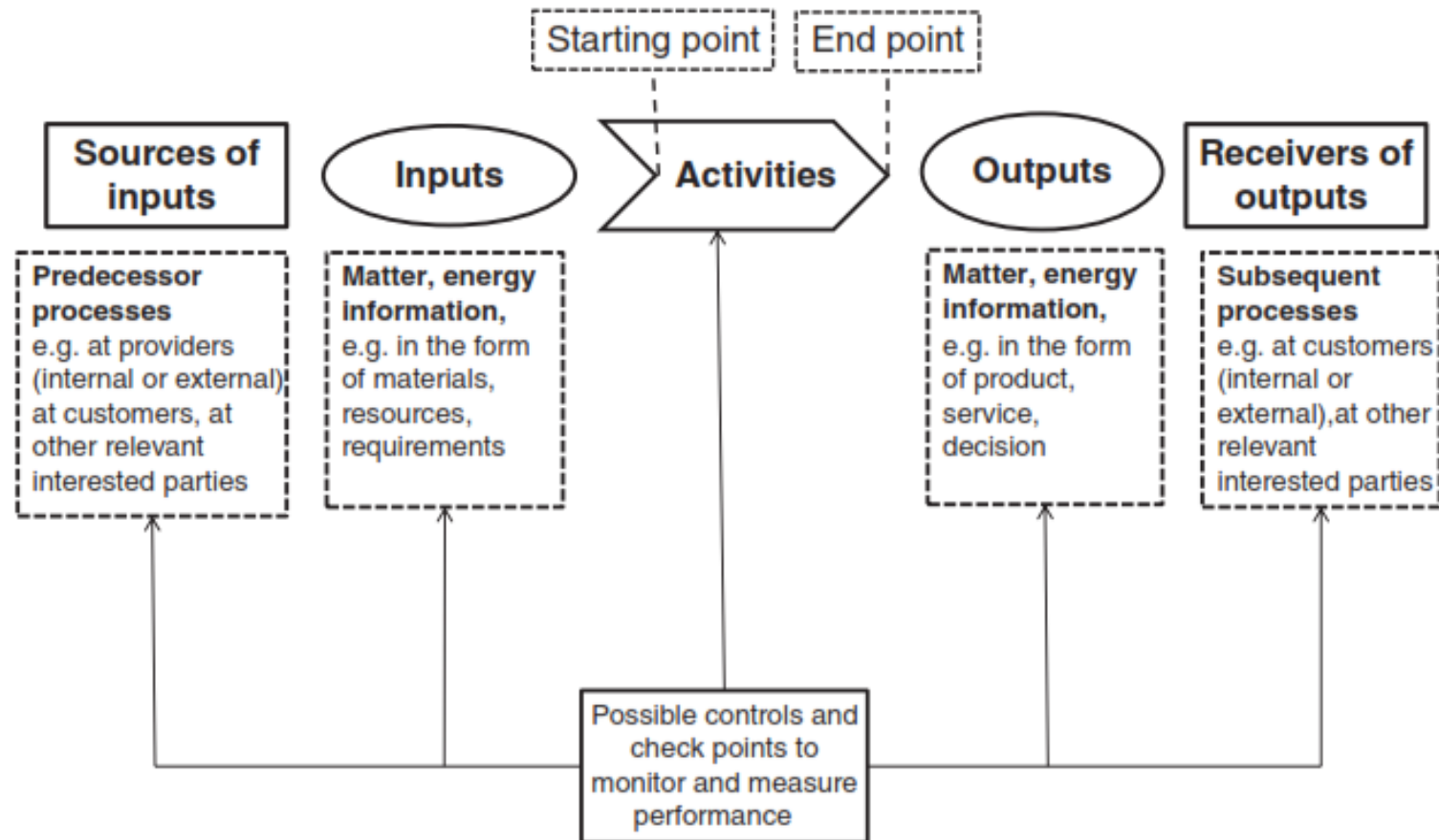


Figure 4.4 Elements of a process [ISO 15].

ISO 9001 describes the elements of the PDCA cycle as follows :

- **Plan:** establish the objectives of the system, processes and resources to deliver results in accordance with customer requirements and policies of the organization, identify and address risks and opportunities;
- **Do:** implement what has been planned;
- **Check:** monitor and measure (if applicable) processes and the products and services obtained against policies, objectives, requirements and planned activities, and report the results;
- **Act:** take actions to improve performance, as needed.

ISO/IEC 90003 Standard



International Electrotechnical Commission.

- Provides guidelines for the application of the ISO 9001 standard to computer software.
- It provides organizations with instructions for **acquiring, supplying, developing, using and maintaining software.**
- Explains what a software audit is for the organization wishing to set up a QMS as well as for the QMS auditor.

ISO/IEC/IEEE 12207 STANDARD



Establishes a common framework for software life cycle processes.

It applies to the acquisition of systems and software products and services, development, supply, operation, maintenance, and disposal of software products and the development of the software part of a system whether performed internally or externally to an organization

ISO 12207 [ISO 17] defines four sets of processes as shown in Figure 4.5:

- **Two agreement processes between a customer and a supplier;**
- **Six organizational project-enabling processes;**
- **Eight processes for technology management;**
- **Fourteen technical processes.**

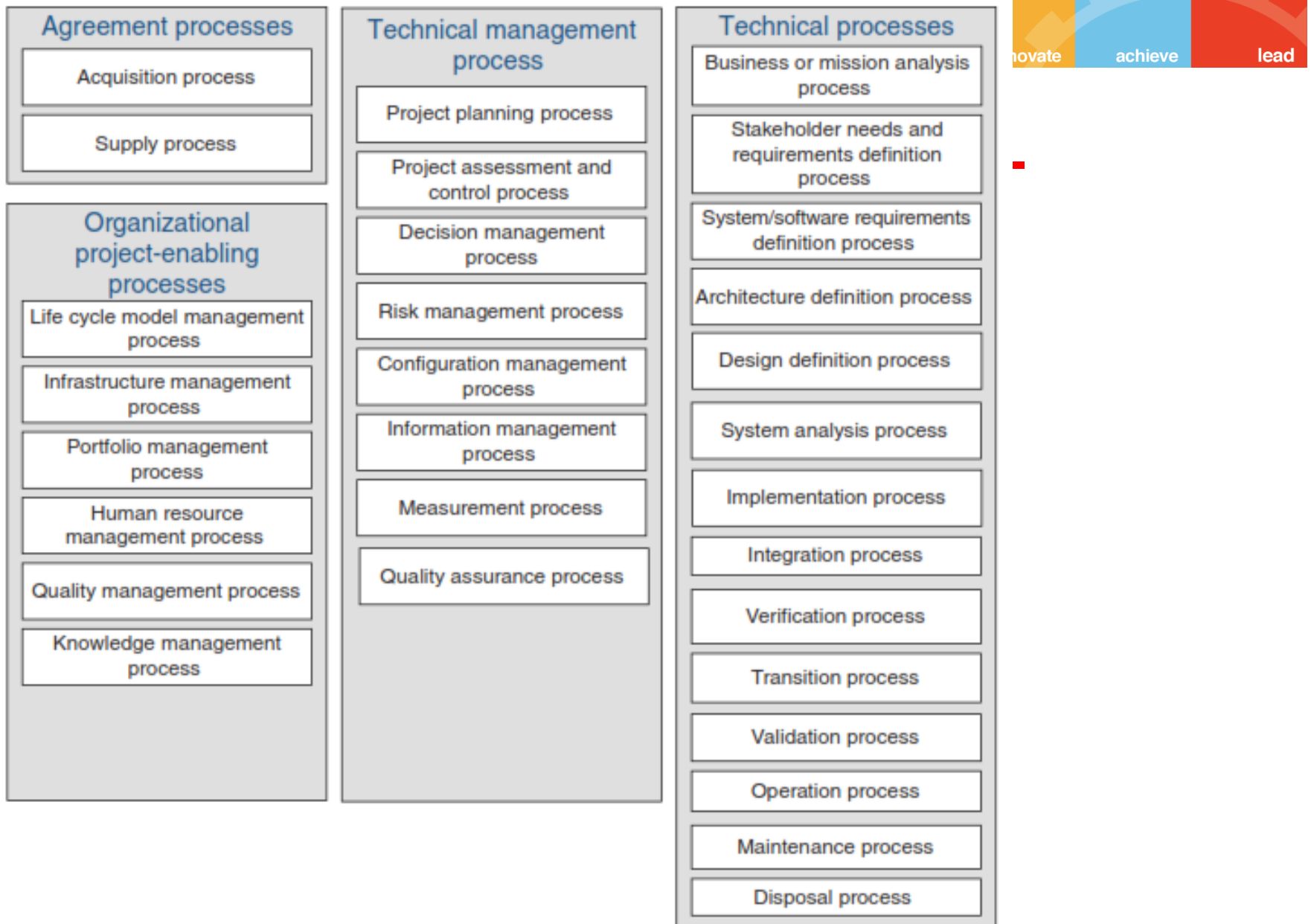


Figure 4.5 The four life cycle process groups of ISO 12207 [ISO 17].

The ISO 12207 standard can be used in one or more of the following modes



For an organization: It aids in establishing a desired process environment, supported by appropriate methods, procedures, techniques, tools, and trained personnel.

For a project: It assists in selecting, structuring, and employing elements from established life cycle processes to deliver products and services effectively.

For an acquirer and a supplier: It facilitates the development of agreements concerning processes and activities, ensuring clarity and mutual understanding.

For organizations and assessors It serves as a process reference model for conducting process assessments, which can support organizational process improvement initiatives.

IEEE 730 STANDARD FOR SQA PROCESSES



QA according to IEEE is a set of proactive measures to ensure the quality of the software product.

The IEEE 730 provides guidance for the SQA activities of products or of services.

The SQA process of the IEEE 730 is grouped into three activities: the implementation of the SQA process, product assurance, and process assurance.

Activities consist of a set of tasks.

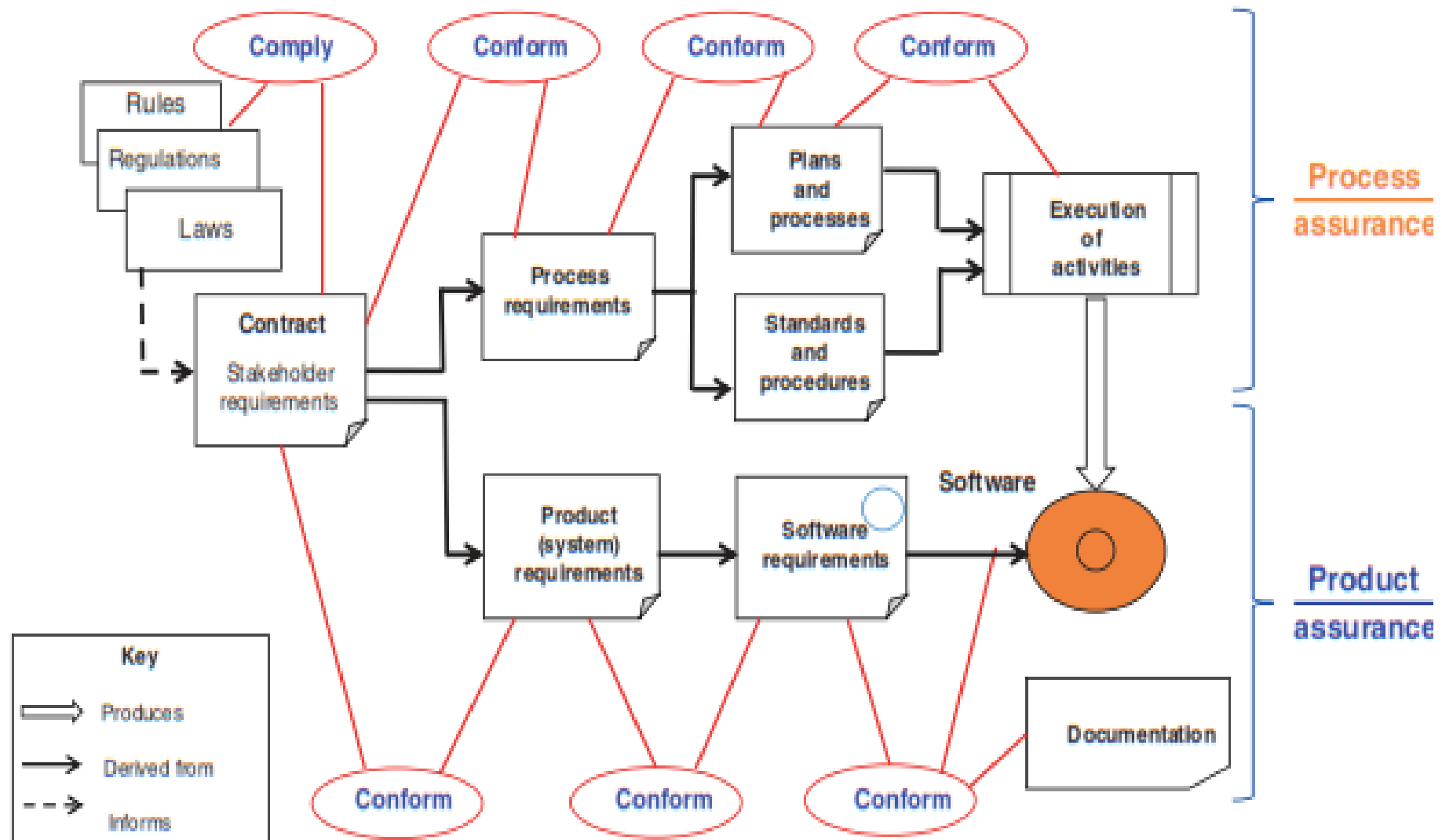


Figure 4.6 The links between requirements and the artifacts of a project [IEE 14].

IEEE 730 [IEE 14] describes what must be done by a project;

- it assumes that the organization has already implemented SQA processes before the start of a project.
- The standard includes a clause that describes what is meant by compliance.

Product Assurance Activities

- Evaluate plans for compliance to contracts, standards, and regulations;
- Evaluate product for compliance to established requirements;
- Evaluate product for acceptability;
- Evaluate the compliance of product support;
- Measure products.

Process Assurance Activities

- Evaluate compliance of the processes and plans;**
- Evaluate environments for compliance;**
- Evaluate subcontractor processes for compliance;**
- Measure processes;**
- Assess the skill and knowledge of personnel.**

Capability Maturity Models (CMM®).



- Tool used to improve and refine software development processes.
- Framework that is used to analyze the approach and techniques followed by any organization to develop software products.
- It also provides guidelines to enhance further the maturity of the process used to develop those software products.

The Capability Maturity Model Integration (CMMI)



- An advanced framework designed to **improve and integrate processes across various disciplines** such as **software engineering, systems engineering, and people management**.
- Helps organizations **fulfill customer needs, create value for investors, and improve product quality** and market growth.

The **CMMI** model was developed as **two versions**:

Initial staged version and **continuous version**, which is the first CMM model for systems engineering

CMMI-DEV The objective of this model is to encourage organizations to check and **continuously improve their development project process and evaluate their level of maturity** on a five-level scale as proposed by the staged CMMI model.

The **CMMI for Development (CMMI-DEV)** covers a broader area than its predecessor by adding other practices, such as systems engineering, and the development of integrated processes and products.

The objective of this model is to encourage organizations to check and continuously improve their development project process and evaluate their level of maturity on a **five-level scale**

Two other CMMI models were developed based on architecture, **CMMI for Services** (CMMI-SVC) and the **CMMI for Acquisition** (CMMIACQ)

The **CMMI-SVC** model provides guidelines for organizations that provide services either internally or externally.

The **CMMI-ACQ** model provides guidelines for organizations that purchase products or services.

All three CMMI models use 16 common process areas.

- For each level of maturity, a set of **process** areas are defined.
- Each area encompasses a set of **requirements** that must be met.
- These requirements define which elements must be produced rather than *how they are produced*

Thereby allowing the organization implementing the process to choose its own life cycle model, its design methodologies, its development tools, its programming languages, and its documentation standard.

This approach enables a wide range of companies to implement this model while having processes that are compatible with other standards.

Maturity levels and process areas for each maturity level in the CMMI-DEV model.



Maturity Level 1: Initial

Processes are usually ad hoc and chaotic.

Maturity level 1 organizations are characterized by a tendency to overcommit, abandon their processes in a time of crisis, and be unable to repeat their successes.

Maturity Level 2: Managed

When these practices are in place, projects are performed and managed according to their documented plans.

Process areas:

- Requirements management
- Project planning
- Project monitoring and control
- Supplier agreement management
- Measurement and analysis
- Process and product quality assurance
- Configuration management



Maturity Level 3: Defined

Processes are well characterized and understood, and are described in standards, procedures, tools, and methods.

Process areas:

- Requirements development
- Technical solution
- Product integration
- Verification
- Validation
- Organizational process focus
- Organizational process definition
- Organizational training integrated project management
- Risk management
- Decision analysis and resolution

Maturity Level 4: Quantitatively managed

The organization and projects establish quantitative objectives for quality and process performance and use them as criteria in managing projects.

Process areas:

- Organizational process performance
- Quantitative project management

Maturity Level 5: Optimizing

An organization continually improves its processes based on a **quantitative understanding of its business objectives and performance needs.**

Process areas

- Organizational performance management
- Causal analysis and resolution

CMMI model structure.



Each process area has generic and specific goals, practices, and sub-practices.

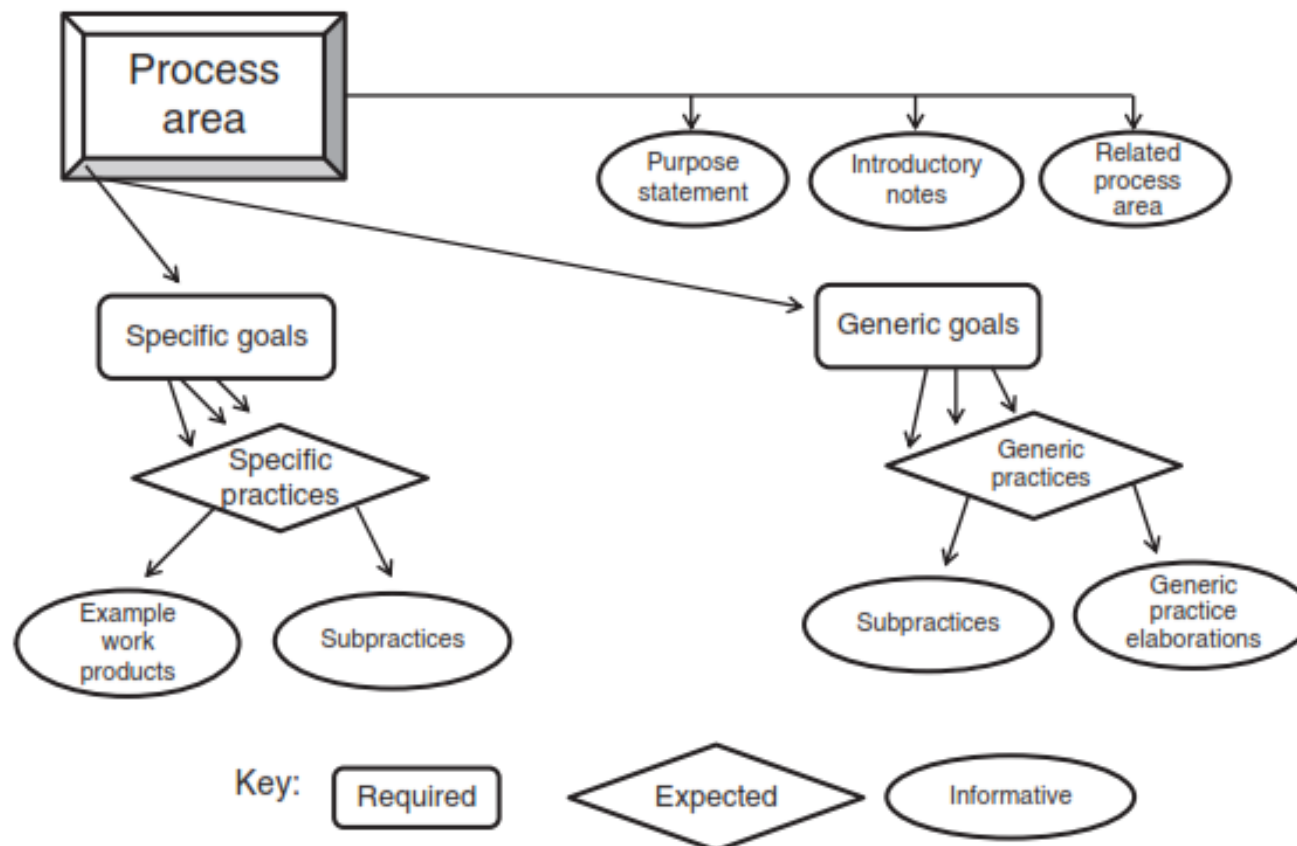


Figure 4.8 Structure of the staged representation of the CMMI [SEI 10a].


Level	Focus	Key process area	<div>Quality productivity</div> <div>  <div>Risk rework</div> </div>
5 Optimizing	<i>Continuous process improvement</i>	Organizational performance management causal analysis and resolution	
4 Quantitatively managed	<i>Quantitative management</i>	Organizational process performance quantitative project management	
3 Defined	<i>Process standardization</i>	Requirements development Technical solution Product integration Verification Validation Organizational process focus Organizational process definition Organizational training Integrated project management Risk management Decision analysis and resolution	
2 Managed	<i>Basic project management</i>	Requirement management Project planning Project monitoring and control Supplier agreement management Measurement and analysis Process and product quality assurance Configuration management	
1 Initial			

Figure 4.9 The staged representation of the CMMI® for Development model.

ITIL Framework



The ITIL framework was created in Great Britain based on good management practices for computer services.

It consists of a set of five books providing advice and recommendations in order to offer quality service to IT service users.

IT services are typically responsible for ensuring that the infrastructures are effective and running (backup copies, recovery, computer administration, telecommunications, and production data)

- strategy;
- design;
- transition;
- operation;
- continuous improvement.

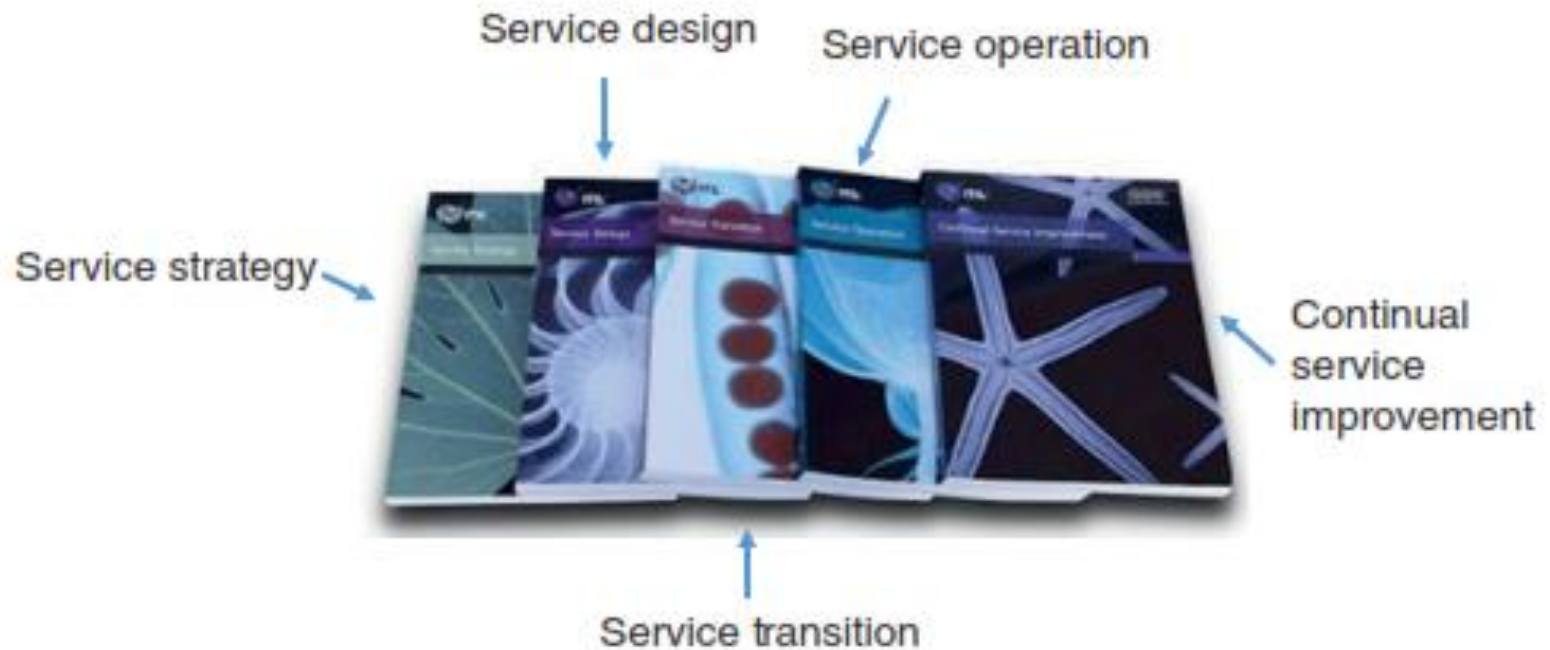


Figure 4.11 The main ITIL guides.

The ITIL framework offers guidance and best practices for managing the five stages of the IT service lifecycle: **service strategy, service design, service transition, service operation and continual service improvement.**

Support processes described in the ITIL are focused on daily operations. Their main goals are to resolve the problems when they arise or to prevent them from happening when there is a change in the computer environment or in the way the organization does things.

Support center function



- Incident management
- Problem management
- Configuration management
- Change management
- Commissioning management

Five processes for service operation:



- Service level management
- Financial management of IT services
- Capacity management
- IT service continuity management
- Availability management

Given the major recognition of ITIL worldwide, an international standard based on ITIL came into being: ISO/IEC 20000-1.

The principles of ITIL were successfully conveyed to many companies of all sizes and from all sectors of activity

The main subjects handled under ITIL



- **User support**, which includes the management of incidents and is an extension of the concept of a Helpdesk;
- **Provision of services** which involves managing processes that are dedicated to the daily operations of IT (cost control, management of service levels);
- **Management of the production environment infrastructure** which involves implementing the means for network management and production tools (scheduling, backup, and monitoring);
- **Application management** which consists of managing the support of an operational program;
- **Security management** (confidentiality, data integrity, data availability, etc.) of the SI (security process)

THANK YOU