TECHNICAL REPORT

ISO/IEC TR 15271

First edition
1998-12-01

Information technology — Guide for ISO/IEC 12207 (Software Life Cycle Processes)

Technologies de l'information — Guide pour l'ISO/CEI 12207 (Processus du cycle de vie du logiciel)

Reference number
ISO/IEC TR 15271:1998(E)

Contents

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

The main task of technical committees is to prepare International Standards. In exceptional circumstances a technical committee may propose the publication of a Technical Report of one of the following types:

— type 1, when the required support cannot be obtained for the publication of an International Standard, despite repeated efforts;

— type 2, when the subject is still under technical development or where for any other reason there is the future but not immediate possibility of an agreement on an International Standard;

— type 3, when a technical committee has collected data of a different kind from that which is normally published as an International Standard ("state of the art", for example).

Technical Reports of types 1 and 2 are subject to review within three years of publication, to decide whether they can be transformed into International Standards. Technical Reports of type 3 do not necessarily have to be reviewed until data they provide are considered to be no longer valid or useful.

ISO/IEC TR 15271, which is a Technical Report of type 3, was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 7, *Software engineering*.

Information technology — Guide for ISO/IEC 12207 (Software Life Cycle Processes)

1 Scope

1.1 Purpose

The purpose of this Technical Report is to provide guidance on the application of ISO/IEC 12207.

This Technical Report elaborates on factors which should be considered when applying ISO/IEC 12207 and does this in the context of the various ways in which ISO/IEC 12207 can be applied. The guidance is not intended to provide the rationale for the requirements of ISO/IEC 12207.

The three fundamental life cycle models are discussed and examples of tailoring are provided.

1.2 Audience

This Technical Report is written for those who will use or apply ISO/IEC 12207 in contractual situations, on a project irrespective of size or complexity, in an organization as a self-evaluation or for software process improvement initiatives.

This Technical Report discusses how ISO/IEC 12207 may be used in relation to various types of software and indicates which processes may be relevant in each case.

This Technical Report supports ISO/IEC 12207 when it is used as a requirements document and also for use as a template for guidance. (An example of the latter is when ISO/IEC 12207 is self-imposed as part of a process improvement exercise.) The whole Technical Report should be understood but it may be used in relation to particular situations by referring to specific clauses.

1.3 Prerequisites

The prerequisites to using this Technical Report are:

a) Availability of ISO/IEC 12207;

b) Familiarity with ISO/IEC 12207;

c) Familiarity with the relevant organizational policies;

d) General knowledge of software management, software engineering and software life cycle models.

2 References

This Technical Report makes reference to the following standards:

ISO/IEC 12207:1995, *Information technology — Software life cycle processes.*

ISO/IEC 9126:1991, *Information technology — Software product evaluation — Quality characteristics and guidelines for their use.*

ISO/IEC TR 15504 (all parts), *Information technology — Software process assessment.*

3 Notation

Diagrams depicting the processes and activities of ISO/IEC 12207 follow the style used in ISO/IEC 12207 as shown in Figure 1.

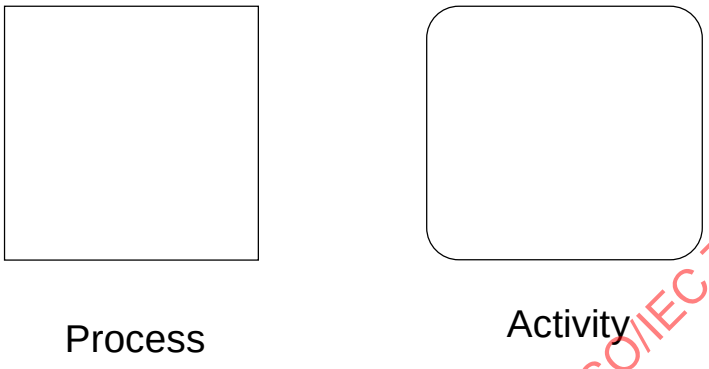

Figure 1 — Drawing notation

4 Basic concepts behind ISO/IEC 12207

4.1 Engineering discipline

The application and practice of software engineering is a relatively young discipline when compared to the traditional branches of engineering. As a result, the control which usually accompanies traditional engineering projects has not always been achieved for software.

The underlying philosophy of ISO/IEC 12207 is that aspects such as software development and maintenance should be conducted in a manner which exhibits engineering discipline. Following this approach allows the establishment of a framework which has clear linkages to the system engineering environment i.e. one which includes software, hardware, people and business practices.

4.2 Software life cycle architecture

ISO/IEC 12207 establishes a top-level architecture of the life cycle of software from conception through retirement. The architecture is constructed with a set of processes and interrelationships among these processes. The processes are based upon two primary principles: modularity and responsibility.

4.2.1 Modularity

The processes in ISO/IEC 12207 are modular, in that they are:

a) Strongly cohesive. All the parts of a process are strongly related;

b) Loosely coupled. The number of interfaces among the processes is kept to a minimum.

In principle, each process is dedicated to a unique function within the life cycle and may employ another process for a specialised function. The following presents the rules for identifying, scoping, and structuring the processes:

a) A process must be modular i.e. one process should perform one and only one function within the life cycle and the interfaces between any two processes should be minimal;

b) Each process is invoked in the architecture;

c) If a process A is invoked by a process B and only process B, then A belongs to B;

d) If a function is invoked by more than one process, then the function becomes a process in itself;

e) It must be possible to verify any function within the life cycle model;

f) Each process should have an internal structure defined sufficiently so as to be executable.

4.2.2 Responsibility

In ISO/IEC 12207, the terms “organization” and “party” are nearly synonymous. An organization is a body of persons organized for some specific purpose, and may be as diverse as a corporation, agency, society, union or club. The size of an organization may vary from one person to many persons. When an organization, as a whole or a part, enters into a contract, it is a party. Organizations are separate bodies, but the parties may be from the same organization or from separate organizations.

Each process in ISO/IEC 12207 is considered to be the responsibility of a party. An organization may perform one or more processes. A process may be performed by one organization or more than one organization, with one of the organizations being identified as the responsible party. A party executing a process has the responsibility for that entire process even though the execution of individual tasks may be by different people.

The responsibility feature of the life cycle architecture facilitates tailoring and application of ISO/IEC 12207 on a project, in which many persons may be legitimately involved.

4.3 The nature of the processes

The processes are grouped into three broad classes:

— Primary;
— Supporting;
— Organizational.

4.3.1 Primary processes

The primary processes are:

— Acquisition;
— Supply;
— Development;
— Operation;
— Maintenance.

In practice, the Acquisition process causes the initiation of the software life cycle. The Supply process responds by performing the Development, Operation, and/or Maintenance processes.

4.3.2 Supporting processes

The supporting processes are:

- Documentation;
- Configuration management;
- Quality assurance;
- Verification;
- Validation;
- Joint review;
- Audit;
- Problem resolution.

A supporting process may be employed by another process which thus supports the former with a specific purpose.

4.3.3 Organizational processes

The organizational processes are:

- Management;
- Infrastructure;
- Improvement;
- Training.

An organization may employ these processes organization-wide to establish, implement, and improve a life cycle process.

4.3.4 Process refinement

Each process is further defined in terms of its own constituent activities, each of which is further defined in terms of its constituent tasks. An activity within a process is a set of cohesive tasks. Within ISO/IEC 12207 there are:

Table 1 — Process breakdown

Class	Processes	Activities	Tasks
Primary	5	35	135
Supporting	8	25	70
Organizational	4	14	27
Total	**17**	**74**	**232**

A task is expressed in the form of a requirement, self-declaration, recommendation, or permissible action. For this purpose, ISO/IEC 12207 carefully employs certain auxiliary verbs to differentiate between the forms of tasks:

— “Shall” is used to express a binding provision between two or more parties;

— “Will” to express a declaration of purpose or intent by one party;

— “Should” to express a recommendation among other possibilities;

— “May” to indicate a course of action permissible within the limits of ISO/IEC 12207.

4.4 Processes and projects

ISO/IEC 12207 describes the set of processes used on large and/or complex software projects. However, ISO/IEC 12207 is designed to be tailorable for a software project of any type and of lesser sizes and complexities. It is also designed to be used whether the software is a stand-alone entity, or a part of the total system.

The processes, activities, and tasks in ISO/IEC 12207 are arranged in their most general, natural positional sequence. This positional sequence does not dictate the life cycle model sequence. It is intended that the software project select, order, tailor and iterate the processes, activities, and tasks as applicable or appropriate.

On the same project, ISO/IEC 12207 may be separately applied more than once. For example, in a given software development project, an acquirer may request a supplier to perform software development with the acquirer and the supplier executing one application of ISO/IEC 12207. The supplier may then request its sub-contractor to perform all or part of the software development. The supplier (now in an acquisition mode) and its sub-contractor (in supplier mode) execute a separate application of ISO/IEC 12207. In both situations, it is necessary to tailor ISO/IEC 12207 to reflect the arrangements.

See clause 6 Application on projects for further details.

4.5 Processes and organizations

An organization (or a party) derives its name from the process it is currently performing, for example it is called an acquirer when it performs the Acquisition process.

The processes in ISO/IEC 12207 form a comprehensive set to cater for a wide variety of organizations. An organization, small or large, depending on its business purpose, can select an appropriate subset of the processes (and associated activities and tasks) to fulfil that purpose. ISO/IEC 12207 is intended to be applied internally by an organization or contractually by two or more organizations. In order to facilitate application of ISO/IEC 12207 either internally or contractually, the tasks are expressed in contractual language. When applied internally, the contractual language is interpreted as self-imposed tasks as described in clause 7 Application in organizations.

ISO/IEC 12207 is to be harmonized with an organization’s policies and standards that are already in place. It is usually the case that an organisation has been utilising its own existing standards and specific techniques for software development. When applying ISO/IEC 12207 within an organisation, it is therefore important to clarify the relationship between ISO/IEC 12207, the organisation's own standards, and the various techniques that have been employed.

Figure 2 shows one possible example of such relationships which may be useful when applying ISO/IEC 12207 within an organisation. ISO/IEC 12207 is located at the first level, standards in the organisation are located at the second level and the third level is for detailed development activities, techniques, and tools that are specific to a project. The terms defined and used in the second and the third levels are required to conform to ISO/IEC 12207.

Resolution of any conflicts is left to the organization applying ISO/IEC 12207 and may involve developing a mapping and if necessary, filling any gaps.

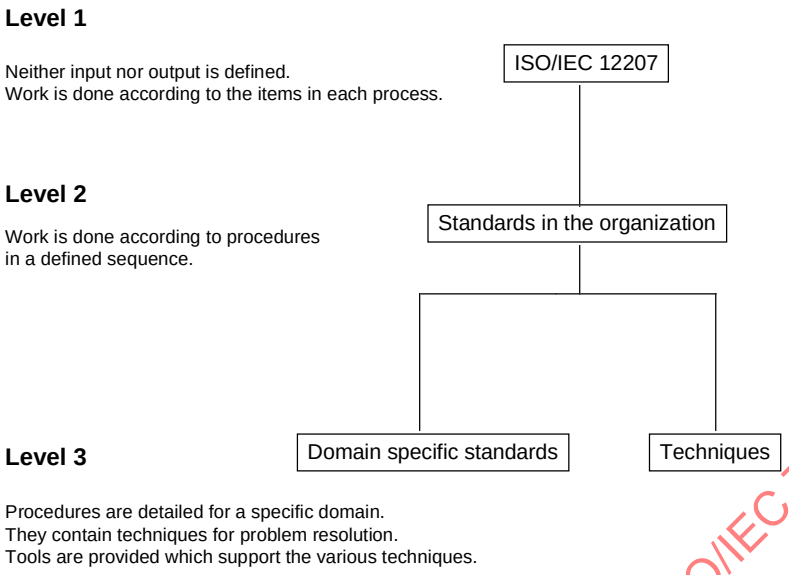

Figure 2 — Relationship with existing documents

4.6 Software and systems

4.6.1 Interface with systems engineering

ISO/IEC 12207 establishes a strong link between the system as a whole and the software. This is possible because ISO/IEC 12207 is based upon general systems engineering.

To a certain extent, ISO/IEC 12207 is designed to function within a systems engineering process. When the software is part of the total system, the software is isolated from the system, produced, and integrated back into the system. This feature of ISO/IEC 12207 is useful when there are no system-level standards available. When the software constitutes the entire scope of interest, the system-level tasks may be treated as useful guidance. In either case, ISO/IEC 12207 provides for the meaningful participation of software engineering in systems engineering.

4.6.2 Relation between software and the system

A system is a specific combination of hardware, computers, software, material, people, and facilities as illustrated in Figure 3. In reality, it is the system that must perform. In the parent system, processes such as business processes exist. Software serves by providing for the execution of certain functions of these processes in computers. The software could be resident in a computer, embedded in a piece of firmware, or integral to a hardware item. In any case, the acquisition, supply, development, operation, or maintenance of the software needs to be in coordination and harmony with those of the parent system.

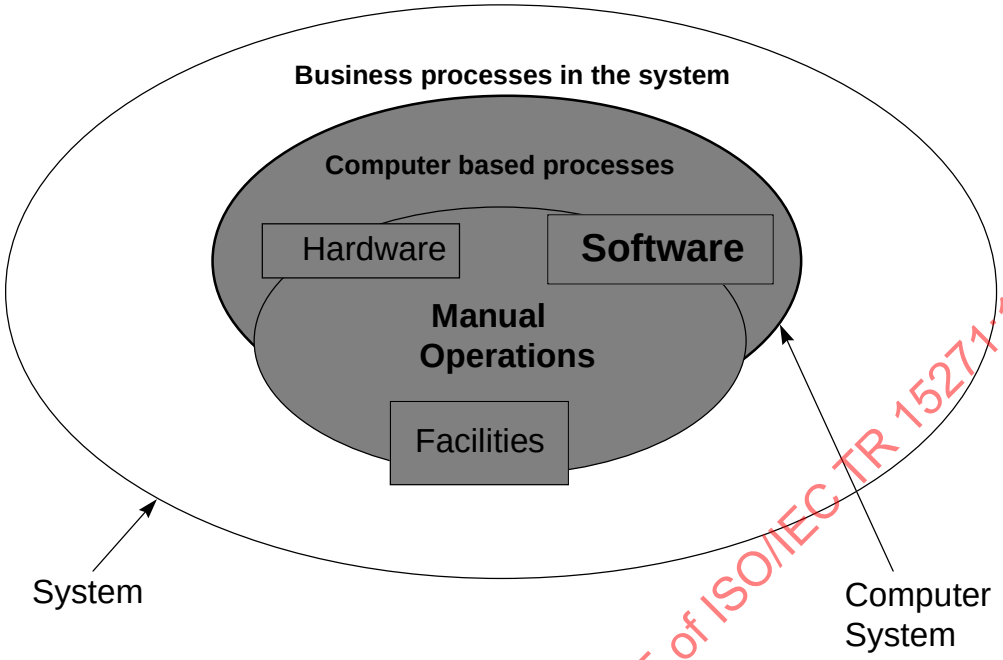

Figure 3 — Software in a system

Within an organization there may be a number of computer systems supporting the business processes as in Figure 4.

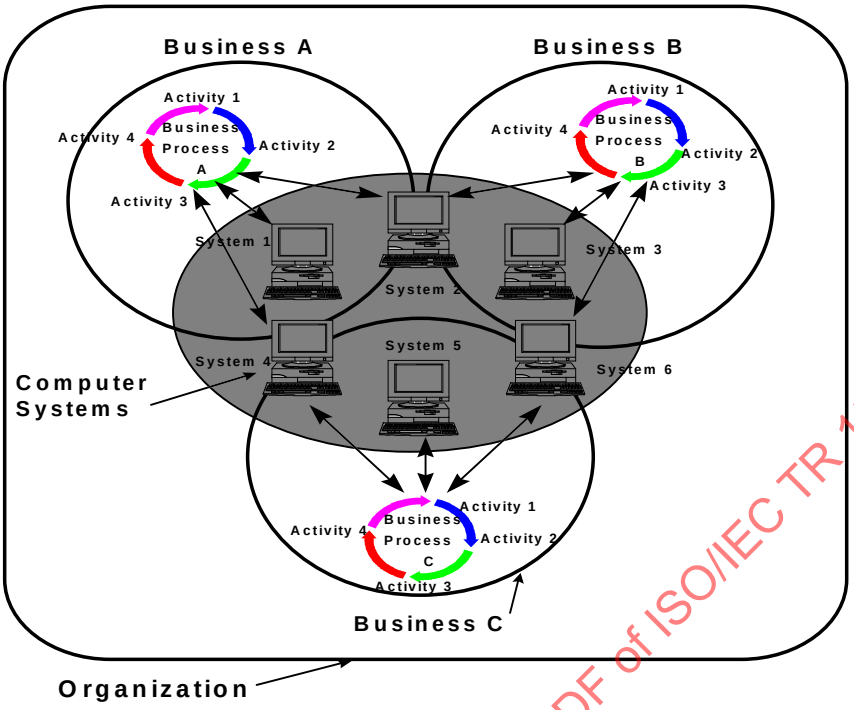

Figure 4 — Computer systems in an organization

4.6.3 Systems based on software

Although ISO/IEC 12207 defines the system, it only covers the life cycle process such as Development, Operation and Maintenance of the system focussed on the software. Therefore, there is no definition for the hardware life cycle process in ISO/IEC 12207.

4.6.4 Classification of system and software activities

Two types of activities are recognized in the Development process in ISO/IEC 12207, i.e. system and software. The scope for these activities is reflected in the name.

Figure 5 shows these activities divided into two groups based on the type and uses a V-presentation to illustrate the symmetry and correlation between the system and software activities.

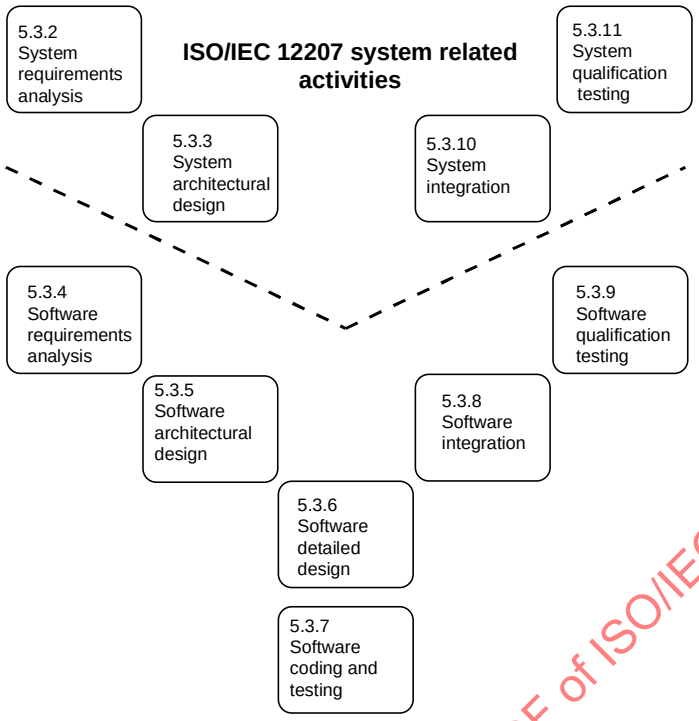

Figure 5 — ISO/IEC 12207 activity classification

As shown in Figure 5, the system activities in the ISO/IEC 12207 Development process begin with 5.3.2 System requirements analysis and terminate with 5.3.11 System qualification testing.

Clause 8 of this Technical Report describes how a system is a combination of hardware, software and manual operations. The division of a system into these elements being performed through the 5.3.3 System architectural design activity of ISO/IEC 12207. The software activities which evolve from this architectural design in turn begin with 5.3.4 Software requirements analysis and terminate with 5.3.9 Software qualification testing.

Once software development has been completed, hardware and manual operations are integrated through the ISO/IEC 12207 subclause 5.3.10 Software integration, and 5.3.11 System qualification testing is then performed. Based on the activities described above, it may be deduced that the system activities are a superset of the software activities.

4.7 Management and planning

For each of the Primary and Supporting processes, management of the process at the project level is done following instantiation of the Management process. It is through this Management process that planning, execution and control of all other planned events is achieved. The items which should be included in a plan are defined in subclause 7.1.2.1 of ISO/IEC 12207, while subclause 7.1.3.2 provides for progress reporting and subclause 7.1.3.3 caters for problem reporting.

4.7.1 Project management plan

In the Supply process, subclause 5.2.4.5 of ISO/IEC 12207 requires that a project management plan is produced and in subclause 5.2.5.1 this plan is executed and controlled. The Supply process in subclause 5.2.5.3 further provides for monitoring and reporting against technical performance, cost and schedules.

4.7.2 Subordinate plans

The items which are listed for consideration in subclause 5.2.4.5 of ISO/IEC 12207 include those relating to processes from the Supporting and Organizational process categories. A number of these processes require that plans be produced e.g. quality assurance, verification and training. Depending on the size and complexity of the project, as well as consideration of the need for subcontracting out some or all of this work, these plans may either be incorporated into the project management plan or developed as separate subordinate documents.

Where subcontractors are used, these should be managed via subclause 5.2.5.4 of ISO/IEC 12207 taking into account the need for increased emphasis on ensuring that appropriate interfaces are established so that the plans can be synchronised.

A summary of the set of subordinate plans may be taken from Table B.2 and Table B.3.

4.7.3 Document control

The requirements for managing documents including plans are described in the Documentation process.

4.8 Implementation of quality management principles

ISO/IEC 12207 implements quality management principles and it does so in three basic ways:

4.8.1 Integration of quality into the life cycle

ISO/IEC 12207 provides the requirements for a comprehensive, integrated set of processes covering the software life cycle. It provides each process with access to a "plan-do-check-act" cycle through the Improvement process. It treats all quality related activities as an integral part of the software life cycle and appropriates those activities to relevant processes of the life cycle. That is, each process and the personnel responsible for performing the process are assigned relevant process-internal quality related activities.

4.8.2 Quality Assurance process

The Quality Assurance process is dedicated to assuring compliance of products and services with their specified requirements and established plans. Persons responsible for this process are provided with the necessary organizational freedom and authority. Organizational freedom means the freedom from those who are directly responsible for producing the product, while authority means the authority to conduct evaluations and initiate corrective actions.

4.8.3 Improvement process

ISO/IEC 12207 contains an Improvement process for further improving quality organization-wide i.e. independently of contractual obligations.

4.9 Flexibility and responsiveness to evolving technology

ISO/IEC 12207 is flexible and responsive to the evolving software engineering discipline. It accomplishes this by providing a top-level, open architecture i.e. ISO/IEC 12207 is:

a) Useable with any:

- Life cycle model (e.g. waterfall, incremental or evolutionary);
- Software engineering method or technique (e.g. object oriented design, structured coding, top-down testing or prototyping);
- Programming language (e.g. COBOL, Ada or assembly).

These are dependent upon the project and state of the art of technology, and their selection is left to the user of ISO/IEC 12207.

b) Flexible from a top-level viewpoint i.e. the activities and tasks of a software life cycle process are “what-to-do” items not “how-to-do” items. In other words, a task might be “develop and document an architectural design”, but not “develop or document the architectural design using the top-down, functional-design method”. This scheme provides an acquirer an avenue to specify an end product or service and, at the same time, allows the vendor to be creative and to employ appropriate methods, techniques, and tools to produce the product or provide the service.

c) Adaptable to any local industrial practice (e.g. for military or commercial use), or for any national or organizational culture.

4.10 Processes and documentation

ISO/IEC 12207 is not a documentation standard i.e. even though ISO/IEC 12207 asks for documenting certain outputs from the processes, it does not specify the format or content of documents. Nor does it prescribe how to combine outputs of a similar nature, such as plans, specifications, or test requirements. See Annex B for details of documentation requirements.

4.11 Software metrics

ISO/IEC 12207 does not define or prescribe the attributes of software (such as reliability or maintainability) in terms of specific metrics and indicators. It does provide the means for specifying such software attributes but the details are left to the users of ISO/IEC 12207.

4.12 Compliance

ISO/IEC 12207 may be complied with in the following ways:

a) An organization declares publicly as a condition of trade, a set of processes, activities, and tasks from ISO/IEC 12207, with which suppliers to the organization comply;

b) A software project tailors appropriate processes, activities, and tasks, which are performed in accordance with contractually established criteria.

NOTE ISO/IEC 12207 contains a set of requirements in the form of "shall" statements. Users of ISO/IEC 12207 are not forced to comply with the document as a whole i.e. there is no "shall" statement in ISO/IEC 12207 requiring compliance with the whole standard. This would be on agreement between two parties. The parties can agree that ISO/IEC 12207 as it is will be the basis of an agreement or they can agree to tailor ISO/IEC 12207 to suit their particular requirements.

4.13 Summary

ISO/IEC 12207 is not a substitute for systematic, disciplined management and engineering of software systems. ISO/IEC 12207 provides a framework where the processes, activities, and tasks related to software can be reasonably identified, planned, and acted upon. ISO/IEC 12207 contains a set of well-defined building blocks (processes). The user of ISO/IEC 12207 has to select, tailor, and assemble them as is appropriate and cost-effective for the organization and the project. The architecture, intent and integrity of ISO/IEC 12207 should be preserved when it is tailored. For example, by inclusion of the element, but annotated as “Not applicable” plus the reason for the exemption.

5 Implementing ISO/IEC 12207

5.1 Overview

ISO/IEC 12207 may be implemented for a variety of reasons including:

— For use on a specific project, to define the software processes, activities and tasks required;

— By an organization as an initiative to improve its software processes;

— As a component within a larger System Life Cycle model process.

Whatever the reason for implementation of ISO/IEC 12207, a suggested implementation strategy consists of the following:

a) Plan the implementation;

b) Tailoring ISO/IEC 12207;

c) Conduct pilot project(s);

d) Formalize the approach;

e) Institutionalize the approach.

This strategy is typical of the approach that must be followed when introducing changes into an organization or project. The implementation strategy described above may be repeated several times within a project or across an organization as additional processes are addressed and/or improved.

When a project or organization is already in a steady state, i.e. where ISO/IEC 12207 processes have been established and institutionalized, then the implementation strategy could be shortened and would probably include the following:

a) Plan the implementation;

b) Tailoring ISO/IEC 12207 (for the risk level of the work);

c) Conduct the project(s).

5.2 Plan the implementation

Implementing ISO/IEC 12207 should be considered as a specific project and planned as such.

The following are examples of items to consider while planning the implementation project:

a) Define the scope of the project. Possibilities include:

— A single project either internal to an organization or as part of a two party contract;

— Concentration on some key processes or even a single process where there is expected to be some gain for an organization. This approach could be used where a weakness has been detected previously and could lead to a full implementation of ISO/IEC 12207 at some future point;

— Adoption of ISO/IEC 12207 across a range of projects with probably a staged introduction. Here the organization would probably have no or few defined processes and would be standardising on ISO/IEC 12207;

— Adoption of ISO/IEC 12207 across all projects and within all parts of the organization. It is unlikely that any organization except a very small one would take this approach. It would be relevant though for a new subsidiary of an existing organization that has adopted ISO/IEC 12207 into working practice previously.

b) Identify the project goals and determine how they fit into the organization-wide business goals. If no obvious link is established between this project and the organization's business focus, then lasting commitment to achieve the implementation project goals will be difficult if not impossible to maintain;

c) Identify roles and responsibilities of the project team/organization, assigning a single point of responsibility for each process. In many cases, one individual or organization may be responsible for more than one process, particularly in small projects or organizations;

d) Identify the resources available for the implementation of ISO/IEC 12207, such as time, money, people and equipment;

e) Create and document the project management plan for implementing ISO/IEC 12207.

5.3 Tailoring ISO/IEC 12207

When the point for tailoring is reached, the Tailoring process defined in Annex A of ISO/IEC 12207 is utilized. Advice on tailoring for specific uses of ISO/IEC 12207 is included in this Technical Report in clause 6 Application on projects, clause 7 Application in organizations and clause 8 Application using a system life cycle model.

Advice on tailoring in this Technical Report should be read in conjunction with the general tailoring advice contained in Annex B of ISO/IEC 12207, using the material which is relevant to the situation. For example, the use may be within an organization and also involve system life cycle model aspects.

Figure 6 of this Technical Report illustrates the sequence of events in the Tailoring process which is discussed in detail in Annex A of ISO/IEC 12207. Specific examples of tailoring are included in Annex D of this Technical Report. These examples:

— Use scenarios to define the project environment;

— Where necessary, define additional activities and tasks;

— Summarize the tailoring decisions and provide a rationale for tailoring.

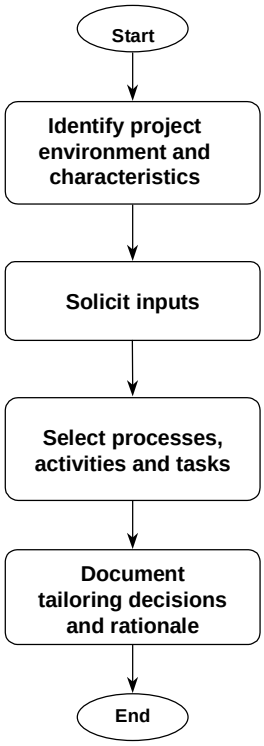

Figure 6 — Tailoring activities

5.3.1 Identify the project environment and characteristics

Organizational characteristics may be determined by considering such issues as:

— What processes, policies and procedures are already in place?
(It is vital to recognize the existing processes and practices that are already successful. These need to be incorporated into the overall suite of required processes.);

— Is this process fundamental to achieving the organization's goals?;

— Is there a high business risk involved?;

— Where are the problem areas?;

— What is the organizational culture (early adopters or adverse to change)?;

— What are the support requirements?.

The project characteristics may be determined by considering such issues as:

— What system or project life cycle model will be used?;

— What is the level of maturity for a particular process?;

— What are the technical risks?;

— Is this a safety critical system?;

— Is new technology being used?.

5.3.2 Solicit inputs

The requirements derived from the relevant business and contractual needs are a major driver in tailoring ISO/IEC 12207. For example, ISO/IEC 12207 could be tailored according to the contract between a supplier and the purchaser of the product. A customer may only require the design of software to be carried out and not full development of a software system. In another instance, if the customer requirements are for safety critical software instead of consumer software, the number of tasks from ISO/IEC 12207 could be quite different.

Affected parties should be involved in the tailoring decisions. These people can help ensure that the resulting tailored processes are feasible and useful. Where possible, include feedback from previous projects.

5.3.3 Select processes, activities and tasks

Identify and prioritize the processes or parts of process within ISO/IEC 12207 that will be implemented. It is usually preferable to start with those processes that will achieve the most significant returns, rather than attempting to implement all of ISO/IEC 12207 at once.

ISO/IEC 12207 does not define the sequencing of processes, activities and tasks and it does not prescribe any particular software life cycle model. Mapping the current processes, practices and/or methods to the processes, activities and tasks of ISO/IEC 12207 is useful at this stage.

The mapping may be used to verify the completeness of the approach i.e. to identify where gaps exist between the current situation and the target situation i.e. where processes from ISO/IEC 12207 are used.

5.3.4 Document the tailoring decisions and rationale

When applying ISO/IEC 12207, a mapping of the defined processes, activities and tasks onto the selected software life cycle model(s) should be documented together with the determined relationships and the reasons for adopting this approach. This documentation should be incorporated into the project management plan for implementing ISO/IEC 12207 as it provides a reference framework for evaluating the success or otherwise of the approach taken.

The examples in Annex D illustrate how a mapping may be documented.

5.4 Conduct pilot project(s)

When introducing ISO/IEC 12207 in an organization across many projects, some pilot use in key areas and for key processes will help to limit the exposure of the organization. A successful introduction would usually include such approaches as the following:

a) Identify pilot projects which can utilize the processes selected. These pilot projects should be chosen on the basis of high priority work, that will result in significant improvements, with a high probability of success, and that can be expected to provide quick, visible results;

b) Select a team of volunteers to conduct the pilot projects then publicize and reward their efforts;

c) Train all involved. Awareness can be aided by regular communication of progress in the implementation process, in addition to formal training classes;

d) Plan the pilot projects and identify critical success factors;

e) For each pilot project, incorporate the selected tailored process(es) into the project management plan. Reference or include as appropriate, the documentation described in subclause 5.3.4 of this Technical Report;

f) Execute the pilot project(s), tracking and documenting the performance against the critical success factors. Document lessons learned throughout the pilot project(s). And incorporate the lessons learned into revised processes.

5.5 Formalize the approach

Formalizing involves the introduction of a new process across several projects and/or across the organization. Issues such as training, documentation, provision of support tools for the process(es), and the tracking and oversight of the new process(es) use and acceptance. Planning for the transition to the new process(es) for any project which is already up and running should be addressed.

NOTE Improvements may be made within a project by monitoring at the project level. They may also be made by comparing one project against another to determine approaches which were successful and which should be incorporated into future projects.

5.6 Institutionalize the approach

Institutionalization focuses on what is involved in ensuring that a process is used consistently and automatically throughout the project or organization. This also involves measuring performance of the process, and implementing process improvement again as necessary. This could involve invoking the Improvement process described as clause 7.3 in ISO/IEC 12207.

6 Application on projects

This clause provides additional guidance on applying ISO/IEC 12207 for a project. However, the guidance is not exhaustive because of the variability which can occur from one project to another.

Aspects which can influence how the software would be acquired, developed, operated, or maintained include:

— Variations in organizational policies and procedures;

— Project-unique acquisition-supply strategies;

— Project size and complexity;

— System requirements.

ISO/IEC 12207 is written to accommodate such variations and for any project. Therefore, in the interest of cost reduction and quality improvement, ISO/IEC 12207 may need to be tailored for an individual project. All roles involved in the project should be given the opportunity to contribute to tailoring.

6.1 Factors in applying ISO/IEC 12207

The following subclauses examine the key factors which should be considered when applying ISO/IEC 12207 on a project. These factors are not exhaustive, represent current thinking and will interact with and affect each other depending upon the software project. When considering the project environment, it may be useful to group the factors into e.g.

— Organizational issues;

— Project risk;

— Capability/maturity of resources.

6.1.1 System life cycle model

The extent of tailoring and the use of ISO/IEC 12207 as a requirement or guide depends upon the relative position of the software project within the system life cycle model. (See Annex C of this Technical Report for a summary of typical life cycle models.)

Identify the current point in the life cycle model of the system of which the software would be a part (see subclause 8.1 of this Technical Report). This determination helps to:

— Demarcate whether the software is a part of the system or the sole scope of involvement;

— Identify whether ISO/IEC 12207 would be used as a method for conducting computer modelling and simulation;

— Identify whether ISO/IEC 12207 would be used for the development, operation, or maintenance of the software.

See the relevant subsequent parts of this clause of this Technical Report and ensure that the necessary interfaces have been established within the system life cycle model.

6.1.2 Organizational policies and procedures

Identify the relevant policies and procedures of the organizations involved, particularly of the acquirer and supplier, with which the software project needs to comply. Examples include policies and procedures related to:

— Security;

— Safety;

— Privacy;

— Risk management;

— Use of an independent verification and validation agent;

— Use of a specific computer programming language;

— Hardware resourcing.

Identify any pertinent laws and regulations including those related to environment, public safety and privacy that may impact the software project. It may be necessary to monitor such external sources on an ongoing basis to ensure that the system is compliant.

The above policies and procedures need to be considered during the development, operation, and maintenance of the software as appropriate. For example, if there are safety and security policies, the requirements analysis activities of the Development process and the system operation activity of the Operation process need to incorporate these policies.

6.1.3 System characteristics

Identify sub-systems and configuration items of the system to an appropriate level of detail. Identify system characteristics, especially those that are allocated to software. Include in the identification those characteristics that are critical to the operation of the system.

Examples of system-level characteristics (if allocated to software) that should be considered include:

— Inter-system and intra-system interfaces;

— User interfaces;

— Impact of software errors on system security and safety;

— Computer sizing and timing constraints;

— Presence of firmware;

— Availability of suitable computers.

If the system has many sub-systems or configuration items, the system-level activities in the Development process should be carefully performed for each sub-system and configuration item. All interface and integration requirements should be considered. The same rigour may not be necessary for a small system.

6.1.4 Software characteristics

Identify software-level characteristics. Examples include:

- Potential number of software configuration items;
- Types, size, and criticality of software;
- Technical risks;
- Types, extent, and medium of documentation;
- Whether new development, modification, or reuse;
- Aspects included in ISO/IEC 9126 such as reliability.

If the software has many software configuration items, components, and units, the software-level activities in the Development process should be carefully performed for each software configuration item. All interface and integration requirements should be considered. The same rigour may not be necessary for a small software application.

Determine the extent of management control and evaluation-related activities that would be needed for software by considering the critical system and software characteristics.

6.1.5 Software maintenance strategy

Identify considerations relating to the maintenance of the software. Typical considerations include:

- Expected period of the maintenance;
- Degree of change over time;
- The criticality of the application;
- Who will perform the maintenance;
- The extent of training;
- The environment needed for software maintenance, including testing.

All documentation requirements should be considered, especially if the software is expected to have a long maintenance period or is expected to change significantly. If appropriate, the documentation environment may need to be preserved to facilitate future retrieval.

It is useful to have the documentation available electronically throughout the maintenance period.

6.1.6 Life cycle model of the project

Select one or more appropriate life cycle models for the project. (See Annex C of this Technical Report.) Determine whether the software life cycle model is a sub-part of the life cycle model of the system or is the complete life cycle model.

Each life cycle model in Annex C contains certain processes and activities that may be performed sequentially, repeated, and combined. The activities of the processes in ISO/IEC 12207 should be mapped to the selected life cycle model. For incremental, evolutionary, build, and pre-planned product improvement approaches, the outputs of one life cycle model activity feed into the next. In these cases, the relevant documentation should be complete at the end of the activity i.e. before the next activity commences.

6.1.7 Diversity of the parties involved

Identify which parties are going to be involved in the project and for which processes they would be responsible. All the activities and tasks in ISO/IEC 12207, that relate to interfaces between the parties should be considered.

A large project involving many persons needs significant management supervision and control. For example, internal and independent evaluations, reviews, audits, inspections and report generation are important tools for a large project. For small projects, these controls may be excessive and should be applied carefully.

6.1.8 Software types

Determine which types of software are involved, as different types of software require different tailoring decisions. Examples of the types of software are:

a) New development;

b) Off-the-shelf;

c) Firmware;

d) Stand-alone;

e) Non-deliverable.

The following presents some basic guidance on the key types of software. The main point to remember is that different types of software enter the Development process at different points in the set of activities.

a) New development.

This software enters the Development process at the beginning. All of the requirements under the Development process should be considered.

b) Off-the-shelf.

— Use of off-the-shelf software exactly “as is”. This type of software has already been designed, coded and tested but more testing may be required depending on such factors as criticality and history of use. It enters the Development process no later than software qualification testing as the full Development process may be excessive. Performance, documentation, proprietary rights, and future support related to the software should be evaluated.

— Incorporation of off-the-shelf software without modification but where for example configuration of application parameters is required. An example would be where parameters are required to be set for such items as currency, date format or paper size. This type of software enters the Development process when software units are tested or integrated, as judged appropriate. The full Development process may be excessive. Performance, documentation, proprietary rights, and future support related to the software type should be evaluated.

— Modification of off-the-shelf software e.g. by adding more suitable reporting formats or when documentation is not available. Depending on the criticality and expected future changes, the Development process should be used via the Maintenance process. The Development process is entered at software coding and testing. Performance, documentation, proprietary rights, and future support related to the software should be evaluated.

c) Firmware

Software or firmware embedded in or integral to a system. Since such software is a part of a larger system, the system-related activities in the Development process should be considered. In the system-related activities, only one verb “perform” or “support” needs to be selected. If the software or firmware is not likely to be modified in the future, the extent of documentation needs should be carefully examined.

d) Stand-alone

Software that is stand-alone. Since such software is not a part of the final system, the system-related activities in the Development process need not be required. Documentation needs, particularly for maintenance, should be carefully examined.

e) Non-deliverable.

As no items are being acquired, supplied, or developed, no clause of ISO/IEC 12207 except clause 5.3.1.5 of the Development process should be considered. However, if the acquirer decides to acquire a piece of such software for future operation and maintenance, then this software should be treated as in b) through d) above.

6.1.9 Project size

A large project involving tens or hundreds of people represents a significant management problem compared with a project of, say, three people. Large projects, or projects with subcontractors require careful management supervision and control. Some projects achieve this through joint reviews, audits, verification, validation and quality assurance. For small projects, all these controls may be excessive.

6.1.10 Project criticality

The more dependent the system is upon the software operating correctly and being finished on time, the more visibility and control are needed. Conversely, extreme overseeing and control of non-critical software is probably not cost-effective. (See clause 6.4.1.1 of ISO/IEC 12207 for an explanation of criticality.)

6.1.11 Technical risk

Development of software may have technical risks. If the software technology used is not mature, the software being developed is unprecedented or complex, or the software contains safety, security, or other critical requirements, then rigorous specification, design, testing, and evaluations may be needed. Independent verification and validation may be important.

7 Application in organizations

7.1 Considerations and techniques

Organizations would generally use ISO/IEC 12207 as part of an effort to improve software related processes. This may be through standalone use or in conjunction with available process assessment and capability determination methods such as ISO/IEC TR 15504.

Application of ISO/IEC 12207 within an organization is based upon the same approaches as are used on projects. Consideration is given to the issues raised in clause 6, and the strategies described in clause 5 of this Technical Report are followed by organizations when using ISO/IEC 12207.

7.2 Application opportunities

The reasons for applying ISO/IEC 12207 internally within an organization may include such situations as:

— Verifying the thoroughness of an existing method. This would usually be more relevant where the method was developed inhouse or adopted and extensively modified;

— Adapting an existing method to cater for the risks associated with moves into new market sectors where more rigour is required because of perceived risks;

— Developing a new method e.g. to meet the needs of a new organization. This includes organizations created through mergers or business alliances. It may be necessary to maintain several process models to suit particular activities;

— Managing the introduction of new technology. Examples include the automation of existing manual processes, or a change in the techniques used to implement a software product. ISO/IEC 12207 defines criteria which can be used to benchmark the completeness of the method before and after the technology is changed;

— Evaluating an internal capability of a party to meet agreed criteria e.g. as part of a tender review process;

— Establishing a benchmark upon which improvement programmes can be developed e.g. audit against ISO/IEC 12207 and use of the Improvement process itself.

7.3 Management commitment

As with any programme which results in changes to work practices, it is essential that the management within the affected organization is visibly committed to implementing and supporting the changes. In a two-party situation this would be initiated by a contract and then, as for general organizational use, it would be normal practice to establish policies covering the relevant parties.

8 Application using a system life cycle model

This clause illustrates a generic life cycle model of a system or project and describes how ISO/IEC 12207 can be applied within that system life cycle model. See Annex C of this Technical Report for a summary of typical life cycle models.

8.1 System life cycle model

A typical system life cycle model begins with the conception of an idea or a need, runs through the development, production, operation, and maintenance of the system, and ends with its retirement. The life cycle model is normally divided using e.g. periods of time or milestones. Each division represents major, distinct activities and tasks to be performed and may require an authorisation at a transition.

For example, the generic system life cycle model is divided as follows, with each division being adaptable to a particular system life cycle model:

a) Needs determination;

b) Concept exploration and definition;

c) Demonstration and validation;

d) Engineering/development;

e) Production/manufacturing;

f) Deployment/sales;

g) Operations;

h) Maintenance and support;

i) Retirement.

8.2 Software life cycle model

A typical software life cycle model consists of several activities. It begins with an idea or a concept for a software product or service, continues through systems engineering and software engineering, and advances to operation, maintenance and support, and ends in retirement. ISO/IEC 12207 organizes these and related activities into the primary, supporting, and organizational processes that make up the software life cycle model.

8.3 Example of ISO/IEC 12207 in a generic system life cycle model

Figure 7 of this Technical Report presents highlights of the use of ISO/IEC 12207 in a generic example system life cycle model. Its basic purpose is stated briefly, followed by the mode of use of ISO/IEC 12207. Also presented is Table 2 containing system life cycle activities and the applicable software life cycle processes.

In any activity or across the total life cycle model, an organization may use ISO/IEC 12207 internally or may retain a supplier to provide the products or services in whole or part.

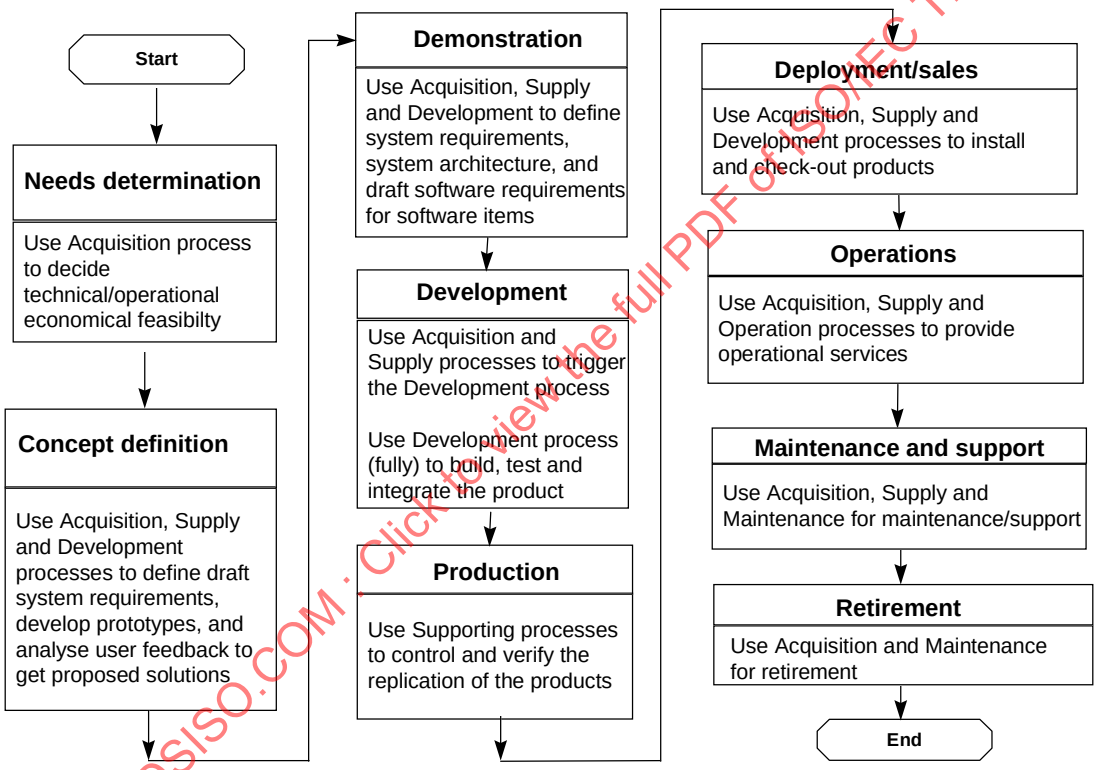

Figure 7 — Using ISO/IEC 12207 to support a system life cycle model

8.4 Needs determination activity

During this activity, an idea or a need for a new or improved system is identified and determined. A top-level statement of need is developed, along with a review of such items as the cost, criticality, and feasibility of the intended system.

The Acquisition process may be used to help decide whether the need is technologically or operationally feasible before proceeding with further studies, development, and commitment. Further, the Development process may be used to develop software, or the methods or models employed in making the decisions.

8.5 Concept exploration and definition activity

This activity is the initial planning period when technical, strategic, and economic, market bases are assessed through comprehensive studies, experimental development, and concept evaluation. Solutions proposed to meet an identified need may be refined, or alternative solutions developed through feasibility evaluations, estimates (such as cost, schedule, marketing, intelligence and logistics), trade-off studies, and analyses. Outputs of this activity are generally preliminary system requirements, and possibly software for prototypes, that feed into the next activity.

The Acquisition, Supply, and Development processes may be used to:

- Help determine preliminary system requirements;
- Develop prototypes;
- Analyse and incorporate user's feedback to the proposed solutions.

The Development process itself may be used as a method to develop software of the analytical/simulation models needed in studies and decision making and definition of solutions.

8.6 Demonstration and validation activity

During this activity, system characteristics, concepts, and solutions, including computer resources, are further defined through systems engineering, development of preliminary equipment and prototype software, and testing and evaluation. The system characteristics and preferred concepts and solutions are validated to demonstrate that the system, including its computer hardware and software resources, is suitable for engineering development. System requirements are baselined and are allocated initially to its components, such as hardware, computers, software, and personnel. These outputs feed into the next activity.

The Acquisition, Supply, and Development processes may be used for analysing and defining system requirements, top-level system design solutions, and preliminary requirements for system components, including software. The Development process may be used as a method for analysis, demonstration, validation, testing, prototyping of requirements and design solutions.

8.7 Engineering/development activity

This activity is the period when the system hardware, computers, software, facilities, personnel sub-systems, training, and support items are designed, fabricated, integrated, tested, and evaluated. The output is a system which closely approximates the production item, the documentation necessary for the next activity, and the test results that qualify the system to be produced.

ISO/IEC 12207 is designed to be fully applicable during this activity. The processes, activities, and tasks of the Acquisition, Supply, and Development processes should be selected, tailored appropriately, and applied to perform software development or upgrades. This activity may include one or more iterations of the Development process executed in coordination with other system components. Outputs are baselined software requirements, design, and code.

If the software to be developed is to be a part of the system, then all the activities of the Development process should be required. In addition, it needs to be clarified whether the developer would perform or support the system-related activities. If the software to be developed is to be stand-alone and not a part of the system, then the system-related activities should not be required, but should be considered.

8.8 Production/manufacturing activity

During this activity, the engineered and developed system undergoes production for the acquirer (users) or manufacturing for the market (consumers). The production period contains approval to produce and production, until the system is delivered and accepted. The objective is to efficiently produce and deliver the working and supported system to the acquirer (users). The manufacturing period is from the approval to manufacture and manufacturing until the system is redesigned or retired. The objective is to efficiently manufacture and deliver working and supported systems to the consumers.

For software, production/manufacturing is minimal when compared to hardware. It involves copying the developed software and documentation onto appropriate media for various users/consumers. (There are no explicit tasks for this in ISO/IEC 12207.) Local industry practices and government regulations may be applicable. The Release Management and Delivery activity of the Configuration Management process may be used for the control of these related tasks. Other activities such as Integration Verification may be used where appropriate.

8.9 Deployment/sales activity

During this activity, the system undergoes deployment for the acquirer (users) or sales to consumers. The deployment period commences with delivery of the first operational system to the acquirer (users and maintainer). The sales period is from the distribution of the first batch of systems to the consumers until the system is withdrawn from the market.

The Acquisition, Supply, and Development processes may be used for installation and check-out of the developed or modified software.

8.10 Operations activity

This activity covers operation, execution, or use of the system by the users and consumers and ends when the system is removed from operations.

The Acquisition, Supply, and Operation processes may be used for the operation of the software and for providing operational support to the associated users.

8.11 Maintenance and support activity

During this activity the system is modified due to errors, deficiencies, problems, user requests, or the organizational need for adaptation or improvement. This activity includes provision of logistics, technical, and repair support to users (or consumers).

The Acquisition, Supply, and Maintenance processes may be used for the maintenance of the software and for providing support services to the organization, users, and consumers.

All interfaces with the Development process need to be determined. Depending on the impact of the work, the activities from the Development process which are needed may vary taking into account the applicable situation.

8.12 Retirement activity

During this period, the system is retired from normal services. It includes archiving the retiring system and providing limited support to its users for a given period.

The Acquisition process and the Retirement Activity of the Maintenance process may be used for retiring the software and for providing support services to the organization, users, and consumers for a specific period.

8.13 Software life cycle processes in a generic system life cycle model

Table 2 shows an example of the placement of the software life cycle processes in system life cycle divisions. Only primary processes of ISO/IEC 12207 are shown. Use of a supporting or organizational process would be through a primary process. An “S” indicates use of a process in ISO/IEC 12207 and an “M” indicates use of a method. An “(S)” or “(M)” indicates a possible use.

Table 2 — Software life cycle processes in a generic system life cycle model

		Software life cycle processes				
		Acquisition	**Supply**	**Development**	**Operation**	**Maintenance**
System life cycle divisions	Needs determination	S		(M)		
	Concept exploration and definition	S	(S)	(S), M		
	Demonstration and validation	S	S	S , M		
	Engineering and development	S	S	S , M		
	Production/manufacturing					
	Deployment/sales	S	S	S		
	Operations	S	S		S	
	Maintenance and support	S	S			S
	Retirement	S				S

Annex A

Quality processes and evaluation requirements

The Supporting life cycle processes that are associated with quality are shown grouped in the structure diagram in Figure 1 in ISO/IEC 12207. These processes are:

- Quality Assurance;
- Verification;
- Validation;
- Joint Review;
- Audit.

Within each of the Primary processes, specific tasks invoke these processes to perform evaluation or assessment actions, but there are additional evaluation tasks that are the responsibility of the person carrying out the task. These additional tasks are intended to progressively improve the quality of the tasks, activities and processes. For some projects, this approach will lead to duplication of effort or more effort than is needed to produce a quality product. For other projects, such as mission critical ones, all evaluation processes, activities and tasks will be needed. A key aspect of using ISO/IEC 12207 is therefore the tailoring of the quality related processes to suit the project being undertaken, and the mapping of specific quality processes to the roles that will conduct the project. ISO/IEC 12207 recognises this important task as the preparation of a quality plan, supported where necessary by other related plans, such as a Verification and Validation plan.

The tailoring of quality related tasks may lead to combining specific tasks and delegating responsibility to others. For example, in a small project to develop a management information system for use internally by the company, the quality plan could allow for verification and validation to be performed by the project team and a simple management review process. For a large mission critical system development for a customer, independent verification and validation teams and joint reviews and audits could be planned for the project. Here the drivers are the size and complexity of the project together with the integrity level of the application or system that is being produced.

Table A.1 of this Technical Report shows the requirements related to evaluations of products, services and processes. In an activity of the life cycle of a project or a process, an evaluator evaluates either the organization's own or another's software product or service. ISO/IEC 12207 groups these evaluations into the five types which are listed below. The first four evaluation types are at the project level and the last one is at the organizational level. These evaluations should be selected and tailored proportionally to the scope, magnitude, complexity, and criticality of the project, or of the policies and needs of the organization. The problem, non-conformance, and improvement reports from these evaluations feed into the Problem Resolution process.

Table A.1 — Evaluation requirements for products, services and processes

Evaluation type	ISO/IEC 12207 reference	Purpose	Conducted by	Notes
Process-internal evaluations	5.1 to 5.5	Day-to-day activities	Personnel performing assigned tasks within the process	—
Quality Assurance	6.3	Independently assure conformance of software products and services with the contract requirements and adherence to the established plans	Personnel organizationally independent of those responsible for developing the software	May use the results from other activities as inputs. May coordinate these activities with other evaluation activities
Verification and validation	6.4 and 6.5	Verify and validate the products in varying depth depending on the project	Acquirer, supplier, developer, operator, maintainer or an independent or third party	May not duplicate or replace other evaluations i.e. they are supplementary
Joint reviews and audits	6.6 and 6.7	Evaluate status and compliance of products and activities on a pre-agreed schedule	Evaluating party (reviewer or auditor) and evaluated party (reviewee or auditee) jointly	—
Assessment and improvement	7.3	Efficient management and self-improvement	Management	—

Annex B

Process output categorization

This annex identifies the outputs from the processes, which ISO/IEC 12207 requires or recommends to be documented. Only those ISO/IEC 12207 subclauses that require outputs are listed. The documentation of these outputs should be selected and scaled proportionally to the scope, magnitude, complexity, and criticality of the project or of the organization. In Table B.1, entries under “Outputs” are not necessarily documentation names or titles.

Table B.1 — Primary life cycle process outputs

Process	Sub-clause	Outputs	Output Type
Acquisition	5.1.1.8	Acquisition plan	Plan
	5.1.1.9	Acceptance strategy and conditions	Description
	5.1.2.1	Request-for-proposal [-tender]	Description
	5.1.2.1	Acquisition documentation	Description
	5.1.3.1	Supplier selection procedure	Procedure
	5.1.3.4	Contract	Contract
Supply	5.2.2.1	Proposal	Proposal
	5.2.4.5	Project management plan(s)	Plan
	5.2.6.4	Evaluation, review, audits, testing, and problem resolution reports	Report
Development	5.3.1.2	Problems and non-conformance records	Record
	5.3.1.4	Development plans	Plan
	5.3.2.1	System requirements specification	Description
	5.3.3.1	System architecture document	Description
	5.3.3.1	System allocation document	Description
	5.3.4.1	Software specification	Description
	5.3.4.2	Software specification evaluation	Record
	5.3.5.1	Software Configuration Item	Software
	5.3.5.1	Architecture specification	Description
	5.3.5.2	Interface software specification	Description
	5.3.5.3	Database top-level design	Description

Process	Sub-clause	Outputs	Output Type
	5.3.5.4	User's manual(s)	Manual
	5.3.5.5	Test requirements	Description
	5.3.5.6	Design review	Record
	5.3.6.1	Detailed design	Description
	5.3.6.2	Detailed interface software specification	Description
	5.3.6.3	Detailed database design	Description
	5.3.6.5	Software unit test requirements	Description
	5.3.6.7	Detailed design review	Record
	5.3.7.1	Software units and databases	Software
	5.3.7.1	Test procedure	Procedure
	5.3.7.2	Software unit test results	Record
	5.3.7.5	Software code and test results review	Record
	5.3.8.1	Software integration plan	Plan
	5.3.8.2	Software integration and test results	Record
	5.3.8.5	Software integration plan and documentation review	Record
	5.3.9.1	Software configuration item qualification test results	Record
	5.3.9.3	Software Integration Review	Record
	5.3.9.4	Software Integration Audit	Record
	5.3.10.1	System integration and test results	Record
	5.3.10.2	System qualification test requirements	Description
	5.3.10.3	System qualification test review	Record
	5.3.11.1	System qualification test results	Record
	5.3.11.3	System audit results	Record
	5.3.12.1	Software installation plan	Plan
	5.3.12.2	Software installation events and results	Record
	5.3.13.1	Software acceptance review and testing	Record

Process	Sub-clause	Outputs	Output Type
Operation	5.4.1.1	Operation plan	Plan
	5.4.1.2	Problem reporting procedure	Procedure
	5.4.1.2	Problem reporting	Record
	5.4.1.3	Test procedure for operational environment	Procedure
Maintenance	5.5.1.1	Maintenance plan	Plan
	5.5.1.1	Maintenance procedure	Procedure
	5.5.1.2	Problem and modification request procedures	Procedure
	5.5.2.4	Problem/modification request record	Record
	5.5.3.1	Modification records	Record
	5.5.3.2	Modification test results	Record
	5.5.5.2	Migration plan	Plan
	5.5.6.1	Retirement plan	Plan

Table B.2 — Supporting life cycle process outputs

Process	Sub-clause	Outputs	Output Type
Documentation	6.1.1.1	Documentation plan	Plan
Configuration Management	6.2.1.1	Configuration management plan	Plan
	6.2.4.1	Configuration Management reports and records	Record
Quality Assurance	6.3.1.3	Quality assurance plan	Plan
	6.3.1.4	Quality assurance records	Record
Verification	6.4.1.5	Verification plan	Plan
Validation	6.5.1.4	Validation plan	Plan
Joint Review	6.6.1.4	Joint review results	Record
Audit	6.7.1.5	Audit results	Record
Problem Resolution	6.8.1.1	Problem report	Record

Table B.3 — Organizational life cycle process outputs

Process	Sub-clause	Outputs	Output Type
Management	7.1.2.1	Management plan	Plan
	7.1.3.3	Problem analysis	Report
Infrastructure	7.2.1.2	Infrastructure plan	Plan
	7.2.2.1	Infrastructure configuration	Description
Improvement	7.3.1.1	Organizational processes procedures	Procedure
	7.3.2.1	Process assessment procedure	Procedure
Training	7.4.1.1	Training plan	Plan
	7.4.3.1	Training records	Record

The Tailoring process in Annex A of ISO/IEC 12207 is provided as a normative reference. When invoked, the following outputs are required.

Table B.4 — Tailoring process outputs

Process	Annex	Outputs	Output Type
Tailoring	A.4.1	Tailoring decisions and rationale	Record

Annex C

Life cycle models

There are many software life cycle models. However, there are three fundamental ones. These fundamental life cycle models are:

- Waterfall;
- Incremental;
- Evolutionary.

Each of these life cycle models can be used as is, or they can be combined to create another hybrid life cycle model. It is through the life cycle model chosen that the processes, activities and tasks of ISO/IEC 12207 are linked, and their precedence relationships are defined.

This annex describes the three fundamental life cycle models along with their associated risk factors (reasons against) and opportunities (benefits). These risk factors and opportunities should be considered when deciding upon a life cycle model for a project.

C.1 Waterfall

The waterfall life cycle model is essentially a once-through, do-each-activity-once approach i.e. in simple terms:

- Determine user needs;
- Define requirements;
- Design the system;
- Fabricate the system;
- Test;
- Correct;
- Deliver or use.

In this approach, as each software item is developed, the activities and tasks in the Development process are usually employed in serial. However, they may be employed partially in parallel when consecutive activities overlap.

When several software items are developed concurrently, the activities and tasks in the Development process may be employed in parallel over all of the software items. The Maintenance and Operation processes are usually employed after the Development process. Acquisition, Supply, Supporting and Organizational processes are usually employed in parallel with the Development process.

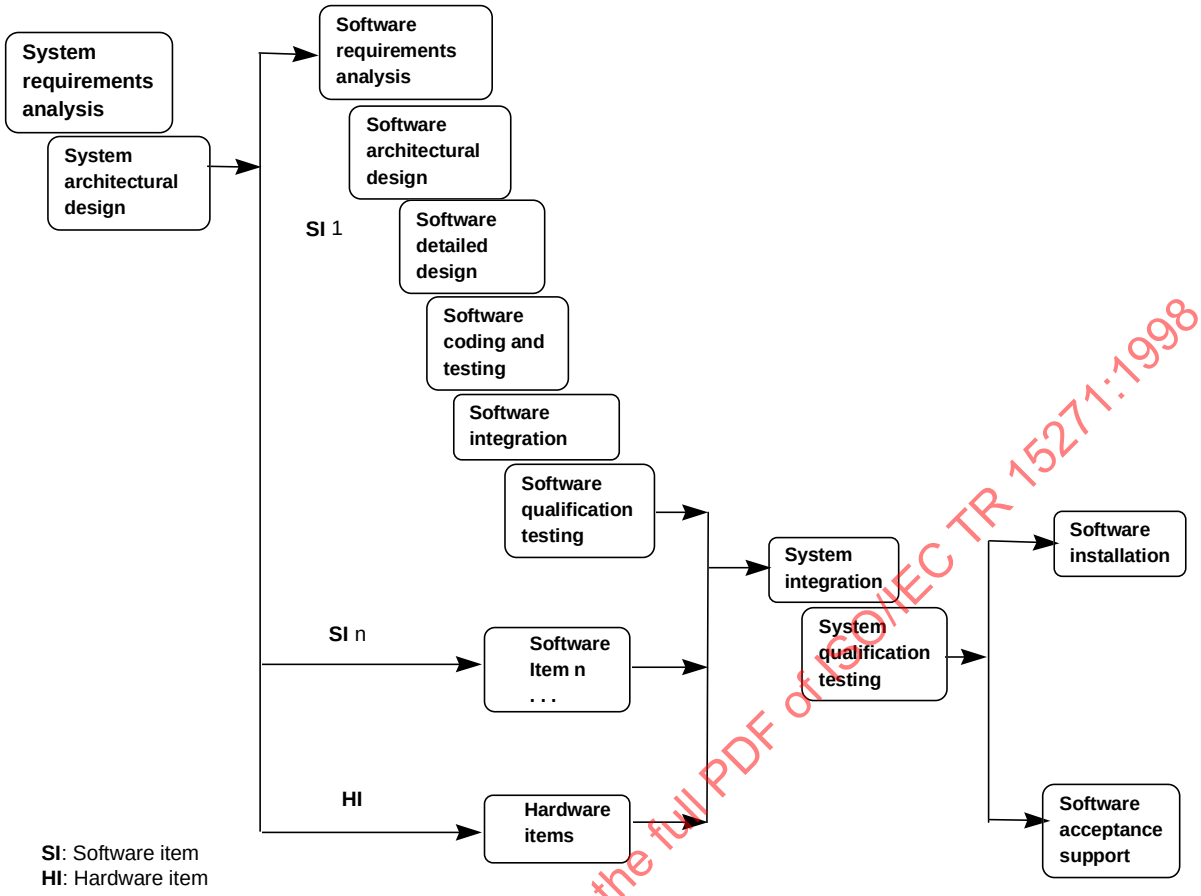

Figure C.1 — Waterfall example

C.1.1 Risks

There are associated risk factors that need to be considered when evaluating this approach:

a) Requirements are not well understood;
b) System too large to do all at once;
c) Rapid changes in technology anticipated;
d) Rapid changes in requirements;
e) Limited resources e.g. staff/funds now;
f) An interim product may not be available for use.

C.1.2 Opportunities

There are opportunities available with this approach i.e. when it is preferred that:

a) All system capabilities be delivered at once;
b) It is necessary to phase out an old system all at once.

C.2 Incremental

The incremental life cycle model, also called pre-planned product improvement, starts with a given set of requirements and performs the development in a sequence of builds. The first build incorporates a part of the requirements, the next build adds more requirements, and so on, until the system is complete. At each build the necessary processes, activities and tasks are performed e.g. requirements analysis and architectural may be performed once, while software detailed design, software coding and testing, software integration and software qualification testing are performed during each build.

In this approach, as each build is developed, the activities and tasks in the Development process are employed in serial or partially in parallel with overlapping. When consecutive builds are developed with partial concurrency, the activities and tasks in the Development process may be employed in parallel over the builds.

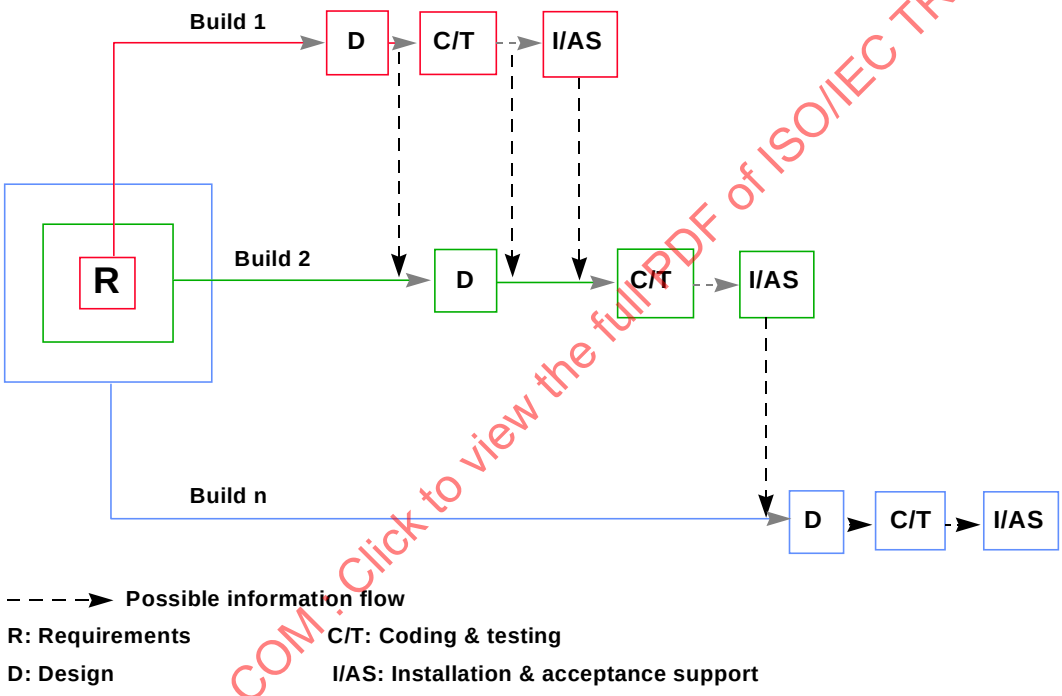

Figure C.2 — Incremental example

The activities and tasks in the Development process are usually employed repeatedly in the same sequence through all the builds. The Maintenance and Operation processes may be employed in parallel with the Development process. Acquisition, Supply, Supporting and Organizational processes are usually employed in parallel with the Development process.

C.2.1 Risks

There are associated risk factors that need to be considered when evaluating this approach:

a) Requirements are not well understood;
b) Capabilities preferred all at once;
c) Rapid changes in technology anticipated;
d) Rapid changes in requirements;
e) Limited resource commitment (staff/funds) in the long term.

C.2.2 Opportunities

There are opportunities available with this approach i.e. when it is preferred that:

a) Early capability is needed;
b) An interim product is available for use;
c) The system is partitioned naturally into increments;
d) Staffing/funding will be incremental.

C.3 Evolutionary

The evolutionary life cycle model also develops a system in builds, but differs from the incremental life cycle model in acknowledging that the requirements are not fully understood and cannot be defined initially. In this approach, the requirements are partially defined up front, then are refined in each successive build.

In this approach, as each build is developed, the activities and tasks in the Development process are employed in serial or parallel with partial overlapping.

The activities and tasks in the Development process are usually employed repeatedly in the same sequence for all the builds. The Maintenance and Operation processes may be employed in parallel with the Development process. Acquisition, Supply, Supporting and Organizational processes are usually employed in parallel with the Development process.

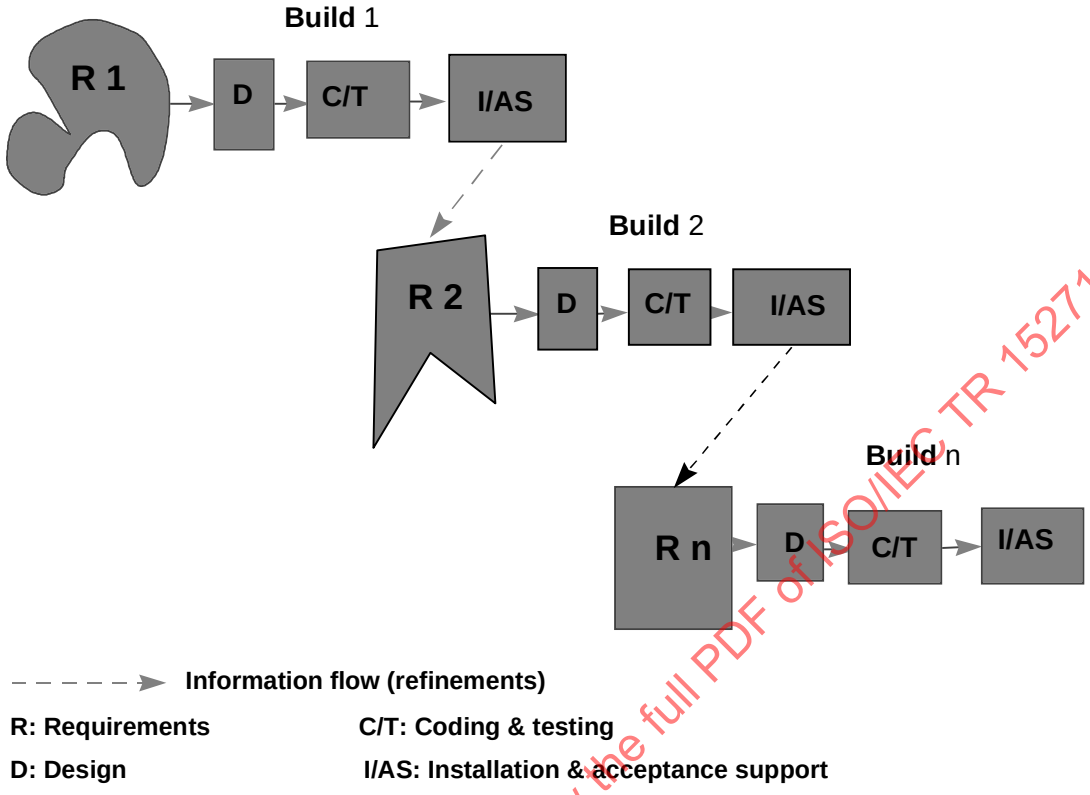

Figure C.3 — Evolutionary example

Table C.1 illustrates how software life cycle model processes may be mapped to produce a life cycle model for an evolutionary life cycle. Only the Development process has been considered. The marked entries indicate an instantiation of an activity or task and the horizontal axis represents a time scale. If required, the mapping may be produced at a greater level of detail.

Table C.1 — Example evolutionary development mapping

Process / Activity / Task	Time scale																								
Process implementation	✓	✓							✓	✓						✓									
System and software requirements analysis		✓	✓	✓						✓	✓						✓								
System and software architectural design			✓	✓	✓						✓	✓						✓							
Software detailed design				✓	✓	✓						✓	✓						✓						
Software coding and testing					✓	✓	✓						✓	✓						✓					
Software integration and qualification testing						✓	✓	✓						✓	✓						✓				
System integration and qualification testing							✓	✓	✓						✓	✓						✓			
Software installation and acceptance support								✓	✓	✓						✓	✓						✓		
Operation process										✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Maintenance process										✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Documentation process	✓	✓					✓	✓	✓	✓	✓			✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓
Configuration management process								✓			✓				✓	✓	✓	✓	✓	✓	✓	✓	✓		✓
Quality assurance process						✓	✓	✓							✓	✓		✓	✓	✓	✓	✓	✓		✓

Process / Activity / Task	Time scale																								
Verification process						✓	✓	✓	✓						✓	✓	✓		✓	✓	✓	✓	✓	✓			✓
Validation process						✓	✓	✓							✓	✓	✓		✓	✓	✓	✓	✓	✓			✓
Joint review process	✓	✓						✓	✓	✓						✓	✓					✓	✓	✓			✓
Audit process								✓								✓				✓		✓		✓			
Problem resolution process				✓				✓				✓		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Management of Development process	✓							✓	✓		✓				✓	✓	✓	✓				✓	✓	✓	✓	✓	✓
Management of Maintenance process								✓	✓	✓					✓	✓	✓	✓				✓	✓	✓	✓	✓	✓
Infrastructure process	✓	✓	✓	✓				✓	✓	✓				✓	✓	✓					✓	✓	✓	✓			
Training process	✓	✓	✓	✓					✓	✓	✓					✓											
Improvement process	✓								✓							✓											✓

C.3.1 Risks

There are associated risk factors that need to be considered when evaluating the evolutionary approach:

a) Capabilities preferred all at once;

b) Limited resources commitment (staff/funds) in the long term.

C.3.2 Opportunities

There are opportunities available with this approach i.e. when it is preferred that:

a) Early capability is needed;

b) An interim product may be available for use;

c) The system is partitioned naturally into increments;

d) Staffing/funding will be incremental;

e) User feedback needed to understand full requirements;

f) Monitoring of technology changes facilitated.