TECHNICAL SPECIFICATION

ISO/IEC TS 24748-6

First edition
2016-12-01

Systems and software engineering — Life cycle management —

Part 6: System integration engineering

Ingénierie des systèmes et du logiciel — Gestion du cycle de vie — Partie 6: Ingénierie de l'intégration du système

Reference number
ISO/IEC TS 24748-6:2016(E)

ISO copyright office
Ch. de Blandonnet 8 • CP 401
CH-1214 Vernier, Geneva, Switzerland
Tel. +41 22 749 01 11
Fax +41 22 749 09 47
copyright@iso.org
www.iso.org

Contents

Page

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 procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation on the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the WTO principles in the Technical Barriers to Trade (TBT) see the following URL: Foreword - Supplementary information

The committee responsible for this document is ISO/IEC JTC 1, *Information technology*, Subcommittee SC 7, *Systems and software engineering*.

A list of all parts of the ISO/IEC 24748 series can be found on the ISO website.

Introduction

This document was developed in response to a need for consistent terminology, definitions and guidance that elaborates the area of system integration, taking into account the context of use and the proven practices for the development of systems.

ISO/IEC/IEEE 15288 includes an integration process that focuses on physically assembling the implemented system elements composing a system to obtain an "integrated system". This process interfaces directly to other technical processes and indirectly to activities and tasks of other technical processes, in particular, the processes that define the system requirements, architecture and design.

The purpose of this document is to facilitate the usage of the integration process of the latest revision of ISO/IEC/IEEE 15288 by providing guidance on system integration.

This document describes the integration engineering activities dealing with planning, performing and managing the integration of a system, including the related activities of other technical processes, in particular, verification and validation processes. These are real practices in industry, i.e. the integration of a system is technically engineered and managed as a project (included in the system development project). Although these practices are performed, they were not formalized in a standard or a guide when this document was written.

Systems and software engineering — Life cycle management —

Part 6: System integration engineering

1 Scope

This document

— specifies activities and processes to be implemented for engineering the integration of systems-of-interest throughout the life cycle (systems made of products and/or services; see Note 1),

— provides guidance for the integration process and its relationships to other system life cycle processes as described in ISO/IEC/IEEE 15288,

— specifies the information items to be produced through the implementation of the integration engineering (integration process and its relationships to other system life cycle processes),

— specifies the contents of the information items, and

— provides guidelines for the format of the information items.

This document can be applied to

— those who use or plan to use ISO/IEC/IEEE 15288 on projects dealing with man-made systems, software-intensive systems, products and services related to those systems, regardless of project scope, methodology, size or complexity, and

— anyone performing integration engineering activities to aid in ensuring that the application of the integration process and its relationships to other system life cycle processes conform to ISO/IEC/IEEE 15288.

NOTE 1 Systems concerned within this document are those as defined in ISO/IEC/IEEE 15288, i.e. systems that are man-made and can be configured with one or more of the following: hardware, software, data, humans, processes (e.g. processes for providing service to users), procedures (e.g. operator instructions), facilities, materials and naturally occurring entities.

NOTE 2 This document is intended to be consistent with the other parts of ISO/IEC 24748.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC/IEEE 15288:2015, *Systems and software engineering — System life cycle processes*

3 Terms, definitions and abbreviated terms

3.1 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC/IEEE 15288 and the following apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

— IEC Electropedia: available at http://www.electropedia.org/

— ISO Online browsing platform: available at https://www.iso.org/obp/

3.1.1
acquirer
stakeholder that acquires or procures a product or service from a *supplier* (3.1.9)

[SOURCE: ISO/IEC/IEEE 15288:2015, 4.1.1]

3.1.2
aggregate
composition of several implemented *system elements* (3.1.11) that are *assembled* (3.1.3), on which a set of *verification actions* (3.1.17) and/or *validation actions* (3.1.15) is applied

3.1.3
assemble
activities for combining and connecting implemented *system elements* (3.1.11) or *aggregates* (3.1.2) to support specific goals, i.e. *integration* (3.1.5), *verification* (3.1.16), *validation* (3.1.14), manufacturing and production.

3.1.4
enabling system
system (3.1.10) that supports a *system-of-interest* (3.1.12) during its life cycle stages, but does not necessarily contribute directly to its function during operation

EXAMPLE When a system-of-interest enters the production stage, a production enabling system is required.

Note 1 to entry: Each enabling system has a life cycle of its own. ISO/IEC/IEEE 15288 is applicable to each enabling system when, in its own right, it is treated as a system-of-interest.

[SOURCE: ISO/IEC/IEEE 15288:2015, 4.1.18]

3.1.5
integration
activities of combining several implemented *system elements* (3.1.11) and activating the *interfaces* (3.1.8) to form a realized system (product or service) that enables interoperation between the system elements and with other *systems* (3.1.10) to satisfy system requirements, architecture characteristics and design properties

Note 1 to entry: In this document, the term "integration" is limited to the integration of the implemented system elements which compose a system and the necessary life cycle related activities. Integration may occur to connect a *system-of-interest* (3.1.12) with external interoperating systems and/or *enabling systems* (3.1.4).

Note 2 to entry: The process of combining software components, hardware components or both into an overall system (ISO/IEC/IEEE 24765).

3.1.6
integration engineering
set of activities that defines, analyzes and executes *integration* (3.1.5) across the life cycle, including interactions with other life cycle processes

Note 1 to entry: The application of the system life cycle processes and knowledge of the *system-of-interest* (3.1.12) are necessary in order to integrate a set of *system elements* (3.1.11) into a *system* (3.1.10).

3.1.7
integration management
set of activities that plans, assesses and controls the integration activities and all related activities

Note 1 to entry: It helps ensure that the process outcomes are achieved and that the integration related information items are identified, documented, maintained, communicated and traced throughout the life cycle of the concerned *system* (3.1.10).

3.1.8
interface
set of logical and/or physical characteristics required to exist at a common boundary or connection between *system elements* (3.1.11)

Note 1 to entry: As examples of interface definition, refer to ISO/IEC/IEEE 24765.

3.1.9
supplier
organization or individual that enters into an agreement with the *acquirer* (3.1.1) for the supply of a product or service

[SOURCE: ISO/IEC/IEEE 15288:2015, 4.1.45]

3.1.10
system
combination of interacting elements organized to achieve one or more stated purposes

Note 1 to entry: A system may be considered as a product or as the services it provides.

[SOURCE: ISO/IEC/IEEE 15288:2015, 4.1.46]

3.1.11
system element
member of a set of elements that constitute a *system* (3.1.10)

[SOURCE: ISO/IEC/IEEE 15288:2015, 4.1.47]

3.1.12
system-of-interest
system (3.1.10) whose life cycle is under consideration in the context of this document

[SOURCE: ISO/IEC/IEEE 15288:2015, 4.1.48]

3.1.13
user
individual or a group that benefits from a *system* (3.1.10) during its utilization

[SOURCE: ISO/IEC/IEEE 15288:2015, 4.1.52, modified]

3.1.14
validation
confirmation, through the provision of objective evidence, that the requirements for a specific intended use or application have been fulfilled

Note 1 to entry: A *system* (3.1.10) is able to accomplish its intended use, goals and objectives (i.e. meet stakeholder requirements) in the intended operational environment. The right system was built.

[SOURCE: ISO/IEC/IEEE 15288:2015, 4.1.53]

3.1.15
validation action
action that describes what is to be validated (the element as reference), on which item the action is performed, the expected result from the performance of the action, the validation technique to apply and at which level of decomposition of the *system-of-interest* (3.1.12)

3.1.16
verification
confirmation, through the provision of objective evidence, that specified requirements have been fulfilled

[SOURCE: ISO/IEC/IEEE 15288:2015, 4.1.54]

3.1.17
verification action
action that describes what is to be verified (the element as reference), on which item the action is performed, the expected result from the performance of the action, the verification technique to apply and at which level of decomposition of the *system-of-interest* (3.1.12)

3.2 Abbreviatied terms

IES	Integration Enabling System
NDI	Non-Developmental Item
SEI	Software Engineering Institute
SoI	System-of-Interest

4 Conformance

4.1 Intended usage

This document provides requirements and guidance for the execution of ISO/IEC/IEEE 15288 processes and activities that deal with integration engineering. This document also provides definition of the content and recommendations for the format of the information items or documentation that result from the implementation of the related processes.

4.2 Conformance to processes

This document provides requirements and recommendations for a number of integration engineering processes and related activities suitable for usage during the life cycle of a system (made of products and/or services).

The requirements and recommendations for processes in this document are contained in Clause 6.

4.3 Conformance to information item content

This document provides requirements and recommendations for a number of integration engineering information items to be produced during the life cycle of a system (made of products and/or services).

The requirements and recommendations for information items in this document are contained in Clause 7.

NOTE In this document, for simplicity of reference, each information item is described as if it were published as a separate document. However, information items are considered as conforming if they are unpublished, but available in a repository for reference, divided into separate documents or volumes or combined with other information items into one document. It is not required to treat every topic in this document in the same order, using the same wording as its title or with the same level of detail. This will depend on the nature of the system, implementation methods, life cycle model and scope of the project; for example, test and Integration plan, a system engineering management plan containing integration plan information.

4.4 Full conformance

A claim of full conformance to this document is equivalent to claiming conformance

— to the provisions contained in 6.2 and 6.3, and

— to the information items cited in Clause 7.

4.5 Tailored conformance

4.5.1 Processes

This document does not make provision for tailoring processes. ISO/IEC/IEEE 15288:2015, Annex A provides normative direction regarding the tailoring of system life cycle process.

4.5.2 Information items

Information items described in this document are requirements and recommendations that can be adapted depending on the type of system, implementation methods, life cycle model and scope of the project.

5 Concepts and principles

5.1 General

This clause presents concepts and principles that apply to integration engineering and to the information items generated at all levels of the system-of-interest during the execution of the concerned processes. They also apply to the processes used in the integration itself and to management of the integration of a system.

5.2 Integration fundamentals

5.2.1 Terms and approaches

5.2.1.1 Fundamental concept behind the term “integration”

As stated in 3.1.5, the term “integration” is defined as activities of combining several implemented system elements and activating the interfaces to form a realized system (product or service) that enables interoperation between the system elements and with other systems to satisfy system requirements, architecture characteristics and design properties. The execution of these activities provides something integrated that becomes a whole. Integration of the system involves its functions.

EXAMPLES

— The engine is integrated with chassis and suspension to form the vehicle.

— Integrated repository: a repository for storing all information pertinent to the System Engineering Plan (SEP) to include all data, schema, models, tools, technical management decisions, process analysis information, requirement changes, process and product metrics and trade-offs (IEEE 1220-2005, 3.1.18).

— Integrated team: group of people with complementary skills and expertise who are committed to delivering specified work products in timely collaboration (ISO/IEC/IEEE 24765).

— Integrated circuit: a small piece of semiconductive material that contains interconnected electronic elements (ISO/IEC 2382)

The opposite of the verb “to integrate” is “to segregate” which means to take apart (synonyms: disassemble, separate).

To make the link with the term “system”, it is notable that a system deals with a whole, without using explicitly the term whole in its ISO/IEC/IEEE 15288 definition (combination of interacting elements organized to achieve one or more stated purposes; see 3.1.10).

5.2.1.2 Integration engineering approach

The concept of wholeness is key to integration engineering. Holistically, a system is more than the sum of its parts. So too, integration is more than mere assembly.

Integration engineering encompasses all activities throughout the life cycle of a system-of-interest that are linked to the integration of the implemented system elements to form the system-of-interest. This includes the following:

— the definition, preparation and performance of the assembly of the implemented system elements to form aggregates until obtaining the defined system-of-interest;

— the definition and performance of actions applied to aggregates in order to check the assembly focusing in particular on interfaces;

— the definition and performance of verification and validation actions applied to aggregates in order to check conformance of the assembly to the system requirements, architecture and design;

— the integration of the formed system into its context of use (environmental context); integration is recursive and so applies to the integration of the system-of-interest into the next level of the system structure.

It also includes the activities of other technical processes that influence and/or constrain, guide, serve and enable these previous activities.

It also includes the integration of the system-of-interest with interoperating systems and enabling systems. It is related to the engineering of enabling systems (set of enabling products and/or services) that support the integration of the system-of-interest (see 5.3.2).

The management of all these activities is part of the overall project management of the system-of-interest.

NOTE The system-of-interest can be composed of layers of system elements. A system element can be considered as a system or as a non-decomposable element. The system-of-interest is the highest abstraction in the decomposition into levels of systems.

While the system-of-interest is the highest level of abstraction in the decomposition of a particular system structure, systems can be abstracted up with respect to their interactions as part of one or more systems-of-systems. This perspective is necessary to aid the integration of the system-of-interest with the interoperating systems and enabling systems of its environment.

A system element is a discrete part of a system that can be implemented to fulfil specified requirements (for example, hardware, software, data, humans, processes, procedures, facilities, materials or any combination thereof).

5.2.1.3 Integration versus mass production

System integration is a part of the effort related to the realization of prototypes or one-shot-systems. The integration activity is different from the mounting of end products on a production or manufacturing line.

For mass/series production, an assembly line does not necessarily use the same assembly order of implemented system elements as it is done for prototypes within the integration process. Integration of prototypes composes systems, through aggregates (see 5.2.2.1), in order to verify and possibly validate those aggregates and their interfaces, almost separately (see 5.2.4).

Mass/series production is not interested in systems, but rather in sets of components (implemented system elements) to optimize the time and production effort. Nevertheless, the integration of a prototype often provides pertinent lessons to engineer a production line that repeats the prototype, in particular, about the order of assembly of the implemented system elements.

5.2.2 Notions of aggregate and of interface

5.2.2.1 Aggregate

The integration of a system is based on the notion of "aggregate". An aggregate is an assembled set of two or several implemented system elements and their interfaces as they are defined in the system architecture and design. An aggregate has a functional consistency that allows the performance of verification actions and possibly validation actions. Each aggregate is characterized by a configuration that specifies the implemented system elements that are physically assembled and their configuration status.

NOTE An aggregate, in the context of integration, does not necessarily represent a system as defined in the physical view of architecture or in the hierarchy decomposition of the system-of-interest. For the purpose of efficient integration and validation of the system, different sets of aggregates may be temporarily considered depending on the integration techniques or methods (see 5.2.6.1) that have been selected to define the integration, verification and validation strategies (see 5.2.9). The validation strategy is of concern because the validation of the implementation of certain requirements is not possible using the complete system due to, for example, security, physical or economical constraints. This is addressed by forming temporary aggregates in order to exercise the concerned requirements.

5.2.2.2 Interface

An "interface" is a concept, in the sense that any system element that binds two system elements may be considered to be an interface from an architectural perspective. An interface generally includes two aspects:

— a logical aspect, i.e. the input-output flow and the function that carries it;

— a physical aspect, i.e. the physical link, made of technology, that transports the input-output flow.

A logical interface consists of an input flow, an output flow or a bi-directional flow (transactional flow) between two functions of the system so that they may exchange material, energy and/or information.

A physical interface is a physical link or port that binds two system elements within the system-of-interest or one system element of the system-of-interest with an element external to the system-of-interest. A physical interface may be considered a system element.

The definition of interfaces is an intrinsic part of the system architecture and design definition and is critical to the success of integration. Interfaces are common failure points in complex systems. They are the points where independent systems or system elements (not necessarily made of the same technology) meet and communicate with each other. This is the reason why architecture and design definition activities and decisions have to consider how the integration (assembly of system elements and verification of the assembly) will be performed.

5.2.3 Integration based on architecture and design

5.2.3.1 Integration versus architecture

The term “system integration” is used to describe the activities related to assembling the implemented system elements of a system to obtain the corresponding final product or service.

The integration of a system pre-supposes that architecture and design activities have been performed upstream defining these system elements and their interfaces. They may pre-exist, be re-used or may be specifically developed.

Based on stakeholder and system requirements, the definition of the architecture of the system deals with the structure and composition of the system elements and of their interfaces. The design of system elements deals with the detailed description of characteristics and enablers of the necessary technologies for their implementation and interfacing.

Scalability, interoperability, availability, portability, etc. are stakeholder concerns that are addressed during architecture definition to equip the architecture with related architectural characteristics such as modularity, standardized interfaces, encapsulation, etc.

So, the integration activities first consist of defining the order of assembling the system elements to the extent that the order is relevant and the verification actions based on architecture and design characteristics; then, performing the assembly of the implemented system elements and executing the verification and/or validation actions. However, note that particularly in software, it is common to develop, expand or refactor software units without defining their order of integration into the software product. The order of assembling the system elements may be flexible. Even for hardware items, different subassemblies (aggregates) can be integrated in various sequences, though there may be a critical path for supply and production of system elements. This critical path consideration for supply and production often identifies integration constraints that influence requirements, architecture or design, including interfaces. These identified constraints are incorporated in the system requirements, architecture or design.

Nevertheless, as indicated in 5.2.2.2, architecture and design definition activities and decisions have to consider how the integration (assembly of system elements and verification of the assembly) will be performed.

In any case, the integration of implemented system elements is based on the architecture and design definition, whatever the life cycle scenario.

5.2.3.2 Integration scenarios

Sometimes, the term “integration” is interpreted based on a restrictive definition consisting of integrating existing elements. This may be the case, for example, when dealing with industrial practices that consist of using existing products or NDI (non developmental items; that have not necessarily been initially defined to run with others) to be incorporated into a given system at a particular life cycle stage.

Three example cases are given below that demonstrate the importance of the system architecture definition, beyond considerations of physical integration.

— **Planned integration**. This is the classic approach used in the development stage of a system in which the system elements are defined up front to be physically integrated. Those system elements have been defined, or identified as NDI during architecture definition, to perform functions that are allocated to them to satisfy system requirements. These system elements may be developed specifically for the system-of-interest. They may be existing and re-used as is or they may be modified. They may also be NDI evolved by external parties. The selection of the system elements is an architectural and development management decision. Planned integration may be achieved at design and build-time and may also be achieved through self-configuration during operation.

— **Evolutionary integration (evolution/extension/adaptation)**. This is the approach used when an existing in-service system (i.e. in its utilisation stage) needs to interoperate with another

one, existing or not. The integration cannot occur without performing upstream definition tasks because this case may be considered as a request that includes specific requirements. In particular, some architecture and evaluation studies are performed including interoperability analysis (compatibility of protocols, interfacing capability), integratability analysis (feasibility of the functional exchanges and of the physical connection) and verification and validation feasibility. These analyses are necessary inputs to the architecture definition/modification in order to either do nothing, modify the interfacing system elements or add new system elements. Nevertheless, to perform these analyses, it is assumed that the engineering information items (documentation) of the concerned systems or system elements exist. Otherwise, a reverse engineering step has to be performed to characterize the interfacing systems or system elements.

— **Capability extension integrating existing system elements or NDI**. This case often corresponds to a decision to develop a new capability of an in-service system (i.e. in its utilisation stage) incorporating an existing system element or an NDI. This case is generally highly constrained by a fast, low effort and low cost implementation with a limited set of available options. As well, these system elements can be significantly complex. This case can be considered as an extension request that includes specific requirements. Architecture and evaluation studies need to be performed upstream taking into account these requirements. This would include, in particular, interfacing feasibility, interoperability analysis, integratability analysis (feasibility of the functional exchanges and of the physical connection) and verification and validation feasibility. An architecture based on a kind of "plug-in" technology principle, a strong modularity property or a service-oriented architecture may be able to partially or totally address this situation. With such architectural characteristics, the integration is facilitated, although this may not be immediately apparent.

NOTE 1 Adoption of NDI is driven by the time-to-market constraints and the higher level of technology readiness of the NDI products. The use of NDI constrains the architecture of a system. Architecture decisions about NDI have a significant impact on integration. Deciding early on in the system development to use NDI vs. a "built-in house" solution aids identification of the appropriate interface and performance of adequate verification and validation actions.

NOTE 2 A system-of-interest is not necessarily static in its environment. The environment, or context of use, can change over time, so the system-of-interest has to adapt to new operating conditions by adding or modifying capabilities. Any type of system is considered in this document, including evolving systems having dynamic self-configuration capability (refer to ISO 15704[16], ISO/IEC/IEEE 15288 and ISO/IEC/TS 24748-1[7]) explain that a system may evolve over time and that every life cycle process can be applied at any time during the system life cycle. This is the case, in particular, with the integration process and related processes.

5.2.4 Integration by layers of systems

The definition of a system (left side of Figure 1) is performed on successive layers of abstraction; each layer corresponds to a physical architecture of systems and non-decomposable system elements. The integration (right side of Figure 1) consists in following the opposite way of composition layer by layer.

On a given layer, integration of implemented system elements is generally done on the basis of the physical architecture as defined during the system definition. The logical architecture is also used as the basis on which to select the system elements to be integrated that together perform a function of the system.

The goal is to progressively validate the system-of-interest per specified system layer, by forming aggregates (see 5.2.2.1), such that certain critical system elements or specific characteristics requiring to be integrated are verified and validated early. In that case, the integration method called "criterion driven integration" or any other integration method may be used (see 5.2.6.1). Mixing these two ways (per layer using integration methods) makes the global integration and validation of a system-of interest efficient, not to say optimized.

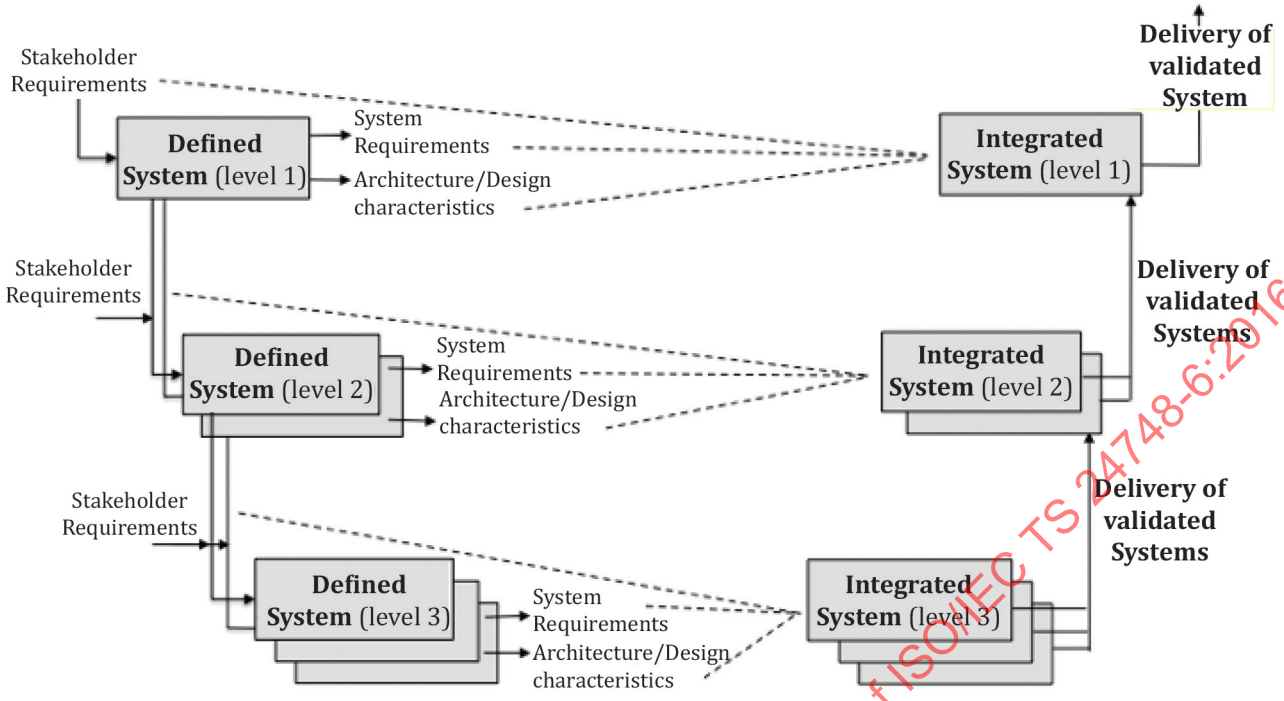


Figure 1 — Integration by layer of systems

NOTE Many integrated systems are systems-of-systems, and the integration starts with combination of existing validated (possibly NDI) systems. In this case, the system-of-systems is the system-of-interest, and each existing system is a system element within the system-of-interest.

5.2.5 Environmental context

Integration engineering also includes the integration of the system-of-interest in its environmental context which may include physical, organizational, social and legal environmental conditions. When the system-of-interest has been formed and functionally verified, it has to be integrated with the operational environment and transitioned into use.

The context of use (operational environment) can be seen as an upper level system in which the system-of-interest is nothing more than a system element. This incorporation follows the same approach as an integration step that consists of connecting the implemented system elements of the system-of-interest to the elements of the context and checking that the system-of-interest can operate under specified conditions.

The physical, organizational, social and legal environmental conditions, in which the system-of-interest is intended to be used, are identified and described as applicable Stakeholder Requirements and the impact of these is derived to the corresponding applicable System Requirements. These requirements are then taken into account as constraints and/or factors during architecture and design definition to endow the system-of-interest with architectural and/or design properties/characteristics. These properties/characteristics can have a significant effect on the effectiveness of the system-of-interest's mission. For example, the properties/characteristics may include thermal conditions, lighting, noise, spatial layout; the organizational and social aspects of the environment may include factors such as work practices, organizational structure and attitudes; safety aspects may include factors such as hazards to life and property; security aspects may include factors such as data integrity assurance and vulnerability to threats.

NOTE Elements for the context of use description are provided in ISO/IEC 25063, which focuses mainly on the usability of an interactive system (including human operators or human users).

Examples of systems or products and their environmental conditions of use:

— equipment for use in a cold store is designed to take into account the need for the workers to wear insulated protective gloves;

— a ticket machine, which is to be installed for use in an outdoor car park, is designed to accommodate the range of varying environmental conditions in which it will be used (e.g. darkness to bright sunlight).

Information regarding taking environmental factors into account can be found in ISO 8995-1, ISO 15265 for workplaces and in ISO 24500 for elderly and disabled persons.

These last considerations are more part of Operational Concept and System Requirements and so subject to validation activity. They could be seen as constraints to define and perform some integration activities (assembly and/or verification).

5.2.6 Integration strategy

The integration strategy defines which activities have to be performed where, when and how they are performed and who will perform them. Until obtaining the complete integrated system that conforms its architecture and design characteristics, the integration activities include

— the reception of each implemented system element,

— the integration (assembly through physical interfaces) of the implemented system elements to form aggregates, and

— the logical integration of aggregates (verification of logical interfaces between system elements through functional, behavioural, temporal and other characteristics as defined in the system architecture description).

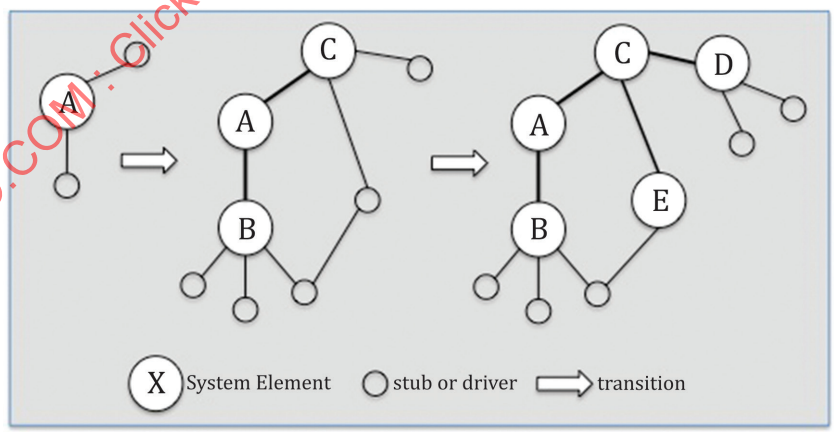
Before any execution of integration activities, an integration strategy has to be defined. The definition of the integration strategy is based on the architecture of the system and relies on the way the architecture of the system has been defined. The strategy is described in an Integration Plan that defines the configuration of expected aggregates, the order of assembly of these aggregates (using adequate integration techniques or methods; see 5.2.6.1) in order to carry out efficient verification and validation actions (for example, inspection and/or testing). The integration strategy is thus elaborated in coordination with the verification and validation strategies and the selected integration technique(s) or method(s).

Based on project objectives, such as delivery time, project cost, system security and/or safety, the integration strategy is defined taking into account criteria such as deadlines, integration cost, risk, availability of resources and skills. The integration strategy should be as flexible as possible, for example, in the case of large systems, the strategy is revised when an initial operation capability is required by stakeholders so that to reduce delays from needs to delivery before the final operation capability is available.

5.2.6.1 Integration techniques or methods

Some integration techniques or methods are summarized in Table 1.

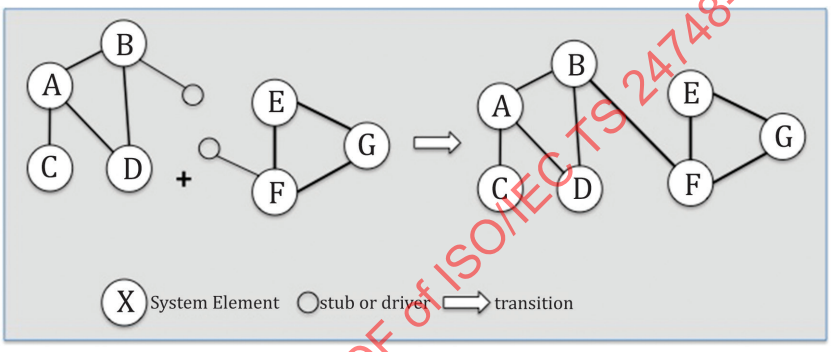
Table 1 — Integration techniques and methods

Integration technique/method	Brief description
Global integration	**Principle**: Also known as "big-bang integration"; all the delivered implemented system elements are assembled in only one step; after each implemented system element has been verified individually, the system elements are integrated concurrently. **Advantage**: No need for drivers/launchers[a] or stubs[b] to simulate absent system elements for verification purposes. **Disadvantage**: Interface errors are detected late; it may be difficult to detect, diagnose and locate remaining errors/faults/defects/flaws in the global system; expensive and inefficient to fix any issues. **Application**: Should be reserved for simple systems, with few interactions and few system elements, where technological risk is low.
Integration "with the stream"	**Principle**: The delivered implemented system elements are assembled as they become available, after each implemented system element has been verified individually. **Advantage**: Allows for quick start of the integration. **Disadvantage**: Necessitates a lot of drivers/launchers[a] or stubs[b] to simulate absent system elements; not efficient with respect to technical, means and cost aspects because it requires modification of the integration plan every time a system element is integrated. **Application**: Should be reserved for well-known and controlled systems without technological risk.
Incremental integration	**Principle**: In a predefined order, one or a very few implemented system elements are added to an already integrated increment of system elements. **Advantage**: Allows for early detection of interfaces errors, and facilitates locating remaining errors/faults/defects/flaws in the new increment. **Disadvantage**: Necessitates a certain number of drivers/launchers[a] or stubs[b] to simulate absent system elements. **Application**: Applies to any type of system architecture. 

[a] External dynamic component or simulator that activates the execution of functions of an aggregate of system elements.

[b] External static component or simulator that replaces missing system elements in an aggregate, providing predetermined results to enable the execution of functions of the aggregate.

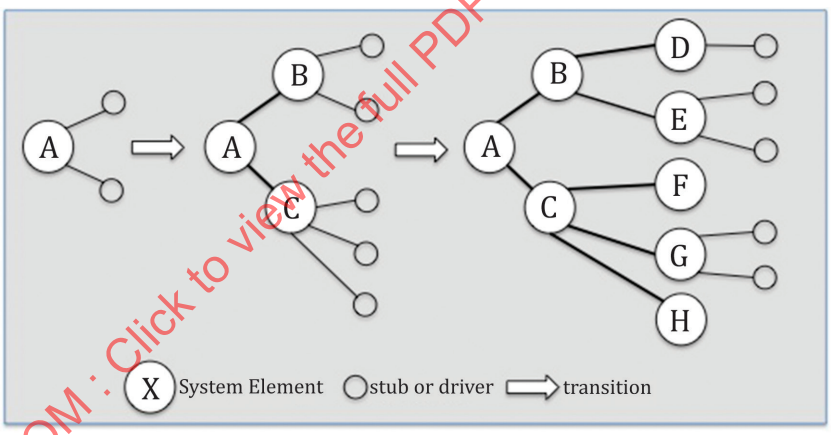
Table 1 *(continued)*

Integration technique/method	Brief description
Subsets integration (aggregates)	**Principle**: Implemented system elements are assembled by subsets (a subset is an aggregate), and then subsets are assembled together. **Advantage**: Parallel integration of aggregates or (sub) systems is possible; delivery of partial products or services is possible. **Constraint**: Necessitates a system architecture defined with (sub)systems; better suited to (sub)system with less complex interfaces. **Application**: Applies to system architectures composed of (sub)systems; applies well to products with functional or transactional chains. 

[a] External dynamic component or simulator that activates the execution of functions of an aggregate of system elements.

[b] External static component or simulator that replaces missing system elements in an aggregate, providing predetermined results to enable the execution of functions of the aggregate.

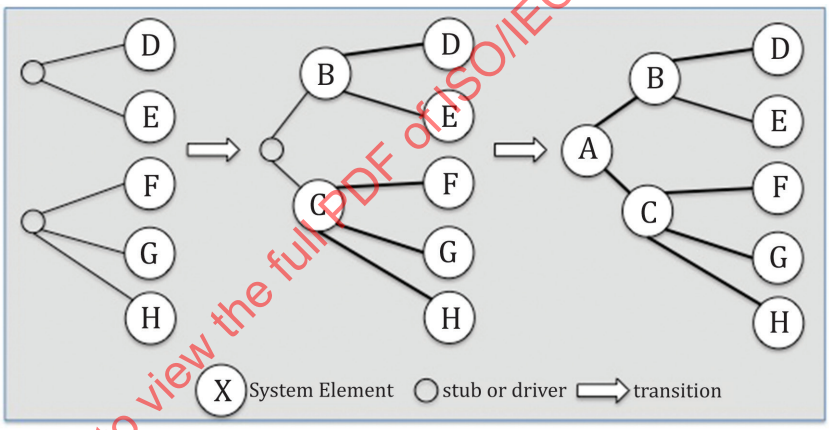
Table 1 *(continued)*

Integration technique/method	Brief description
Top-down integration	**Principle**: Starts with system elements related to the overarching framework of the system-of-interest (generally close to mission or users' vision and operations/commands, when there are users and operators in the system-of-interest); continues with system elements performing core or detailed functions/ transformations of the system; ends with system elements performing more detailed functions or close to acquisition (from sensors) and distribution or physical commands (onto actuators). **Advantage**: Early availability of a skeleton of the system that allows detecting and easily locating remaining errors/faults/defects/flaws; test cases for verification actions are close to reality; does not need many drivers. **Disadvantage**: A lot of stubs[b] to create; difficult to check efficiently and exhaustively at the lowest level system elements and interfaces with test cases from the operators'/users' level. **Application**: Mainly used in intensive software systems, or in systems with human operators or users that pilot or interface with command-control (sub) systems. NOTE A particular implementation of this method is the "onion ring approach": The integration starts from a kernel of the system or a backbone layer; then this kernel is used as a live stub or driver to integrate other system elements. The kernel may include one or several functionalities shared by a number of system elements. 

[a] External dynamic component or simulator that activates the execution of functions of an aggregate of system elements.

[b] External static component or simulator that replaces missing system elements in an aggregate, providing predetermined results to enable the execution of functions of the aggregate.

Table 1 *(continued)*

Integration technique/method	Brief description
Bottom-up integration	**Principle**: Starts with system elements performing very detailed functions or dealing with acquisition (from sensors) and distribution or physical commands (onto actuators); continues with system elements performing detailed or core functions/transformations of the system; ends with system elements related to overarching framework of the system-of-interest (i.e., close to mission or interfacing operators/users commands). **Advantage**: No or few stubs[b] to create; early detection of implementation errors/faults/defects/flaws; test cases to check interfaces and functions are easy to define. **Disadvantage**: Lower level system elements are solicited a lot by higher level test cases; absent system elements from the upper levels are replaced by drivers/launchers[a] to be created. **Application**: This is physically the only way to integrate hardware; also applies to software intensive systems. 
Criterion driven integration	**Principle**: The most critical implemented system elements with respect to the selected criterion are integrated first. Criteria come from system requirements and are generally related to technical risks; they address dependability, security, complexity, performances or effectiveness, usability, technological innovation, etc. The aggregates are defined around system elements that present technical risks. **Advantage**: Allows early and intensive testing of critical system elements; early check of design choices. **Disadvantage**: Drivers/launchers[a] and stubs[b] often difficult to define and realize. **Application**: Applies to any system architecture that includes critical system elements.

[a] External dynamic component or simulator that activates the execution of functions of an aggregate of system elements.

[b] External static component or simulator that replaces missing system elements in an aggregate, providing predetermined results to enable the execution of functions of the aggregate.

The selection of integration techniques or methods depends on several factors, including the type of system elements, their criticality, the delivery time, the order of delivery, technical risks, constraints, the verification strategy, the validation strategy, the type of project and organization, etc. Each integration technique or method has strengths and weaknesses that should be considered when defining the integration strategy.

Usually, a mix of integration techniques or methods is selected as a trade-off between the different techniques or methods listed above, in order to optimize work and to adapt the process to the system under development.

NOTE The integration activity is not a phase of the project fixed in time. Integration activity can also be performed dynamically during operation through self-configuration arrangements of the system-of-interest architecture. Top-down integration method can be used in this case; the presence of stubs enables to progressively build a system driven by, for example, availability, scalability and adaptability requirements.

5.2.6.2 Interface analysis

To make more efficient the integration strategy, the order to integrate the system elements can be done stepwise through interfaces analysis. Coupling matrix (or N square representation) is a typical useful interface analysis method.

Coupling matrix purpose:

N2 diagrams are used during definition to identify and define logical interfaces between functions. They normally concern functions, but can be used with system elements. They are tools commonly used by integrators for defining the order to integrate system elements and verify the interfaces. N2 diagrams and coupling matrices are equivalent representations.

Coupling matrices allow the definition of the aggregates of system elements and the verification of the interactions between system elements. When performing verification, they also facilitate the location of potential errors that remain in the system.

Coupling matrix description and example:

A coupling matrix presents the system elements on the diagonal of a square. Other surrounding cells of the table identify the presence or not of an interface between system elements; the upper right corner contains the interface direction from Ex to Ey, while the lower left corner contains the interface direction from Ey to Ex. An interface cell may simply indicate the fact that an interface exists or may contain the number of different interactions or interfaces binding the system elements. Figure 2 only indicates the presence (or not) of an interface. Inputs are identified by an “x” in the appropriate column, either above or below the item. For example, the inputs to E3 flow from E1, E2 and E6. Outputs are identified by an “x” in the appropriate row, either to the left or to the right of the item. For example, the outputs from E3 flow to E1 and E6. A blank indicates no interface between the elements.

Figure 2 presents two possibilities for the definition of aggregates. The arrangement Structure A on the left side has three aggregates S1, S2 and S3. These aggregates have many interfaces and interactions between them that necessitate defining at least 16 verification actions. In case of anomalies in the global functioning, it is difficult to locate the errors.

The arrangement Structure B on the right side has three aggregates P1, P2, P3. These aggregates each have only two interactions that conduct to two verification actions. The total number is six: two between P1 and P2, two between P2 and P3 and two between P1 and P3. In case of anomaly, the errors are easy to locate. Structure B increases the cohesiveness within the aggregates and reduces the coupling between them.

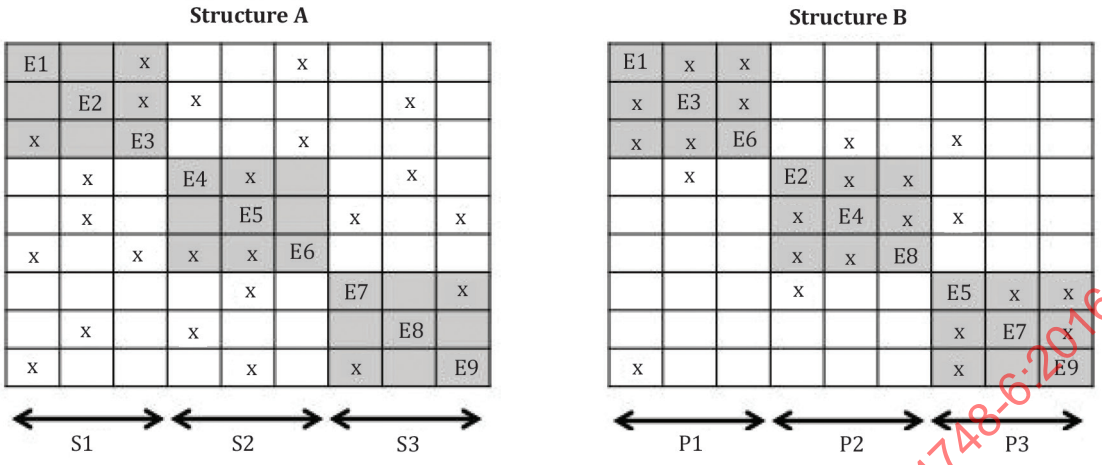


Figure 2 — Usage of coupling matrix to define aggregates of system elements

5.2.6.3 Virtual integration, simulation, mock-ups

As soon as the system elements design and the system architecture are defined, virtual environments, modelling activities, mock-ups or any other convenient tools can be used to model or simulate the assembly of the designed system elements. These activities provide an early check on the feasibility of the assembly, the definition of necessary checkpoints and an opportunity to simulate the interfaces operation/connection.

Performed before the integration, this technique avoids discovering integration mistakes late in development and to more easily mitigate risks. The result of this activity may lead to incorporating arrangements in the architecture and/or modifying the design of system elements and/or their interfaces.

5.2.7 Verification principles related to integration engineering

As indicated in 5.2.1.2, integration engineering includes the definition and performance of verification actions (e.g. inspections, tests) applied onto the aggregates in order to check the conformance of the assembly to the system requirements, architecture and design, focusing in particular on interfaces.

Integration activity is not limited to assembling implemented system elements. When an implemented system element is assembled with another, the interfaces are checked more or less simultaneously. Several verification actions are often necessarily performed before assembling or connecting the implemented system elements, otherwise accidents may happen. For example:

— inspection of dimensions of the physical faces of each system element to be connected,

— checking the voltage that is required between the delivered and used power,

— checking the position and/or polarity of pins,

— checking the compatibility of interface protocol in the software domain, and

— inspection of software code to check that interfaces are implemented identically in interfacing system elements.

So, the assembly activity is intimately linked to the verification process.

The verification process is quite extensively used by the system integration activities.

The purpose of verification is to detect the maximum of number of errors/faults/defects/flaws that the developed system may include before delivery. Because of the number of items to verify, time and cost could exceed expectations, so a verification strategy has to be established.

A verification strategy is based on the balance between what **should be checked** (architecture and design characteristics, system requirements) using relevant verification techniques (e.g. inspection, test), and **what can be checked** taking into account all constraints or limits (technical feasibility, cost, time, availability of verification means and qualified personnel, contractual constraints, etc.). The difference between what should be checked and what can be checked for the complete "end-to-end" system-of-interest reflects potential risks. The selection of verification actions should be made according to acceptable risks if verification actions were dropped out.

5.2.8 Validation principles related to integration engineering

As said in 5.2.2.1, the validation of the implementation of certain stakeholder and/or system requirements is not possible using the complete system due, for example, to security, safety, physical, economical constraints. This is addressed by forming temporary aggregates in order to exercise the concerned requirements.

The validation process is quite extensively used by the system integration activities. The purpose of validation is to obtain the maximum confidence in the use of the system before its final delivery (i.e. transfer of property from supplier to acquirer). Because of the number of requirements to validate, time and cost could exceed expectations, so a validation strategy has to be established.

A validation strategy is based on the balance between what **should be checked** (stakeholder requirements) using relevant validation techniques (e.g. demonstration, test) and what **can be checked** taking into account all constraints or limits (technical feasibility, cost, time, availability of validation means and qualified personnel, contractual constraints, etc.). The difference between what should be checked and what can be checked for the complete "end-to-end" system-of-interest reflects potential risks. The selection of validation actions should be made according to acceptable risks if validation actions were dropped out.

NOTE Concerning the integration of the system-of-interest in its environmental context or context of use, attributes to be validated include in particular the "end-to-end" performance, the usability, the interoperability with the other systems (existing or not) and the physical environment resistance thresholds (mechanical, climatic, chemical, bacteriological, electro-magnetic, etc.; (see ISO 9241-11). Operational scenarios, as defined during the concept definition activities early in the development stage, are used in particular to validate the operational concept on which the system-of-interest lies.

5.2.9 Efficiency of the integration strategy

The efficiency of the integration strategy involves minimizing workload, time and risk to obtain a complete integrated system operating in its context of use. It consists of the following:

— defining the appropriate order for assembling the implemented system elements using defined procedures;

— minimizing verification actions to check the physical interfaces before and during the assembly or connection of implemented system elements within an aggregate;

— minimizing verification actions to check the logical interfaces after the assembly or connection of implemented system elements, as well as the functionalities of the corresponding aggregate;

— minimizing validation actions to check certain system functionalities and requirements within the corresponding aggregates.

So, the efficiency of the integration activity of the system is obtained by considering or defining the strategies for integration, verification and validation together.

The efficiency of the verification strategy consists of defining and performing the minimum of required verification actions while obtaining the maximum of confidence that the system will work as intended

once integration is complete. The strategy definition includes studies of risks potentially generated if verification actions are dropped out.

The efficiency of the validation strategy consists of defining and performing the minimum of validation actions while obtaining the maximum of confidence in the use of the system. The strategy definition includes studies of risks potentially generated if validation actions are dropped out.

5.3 Practical considerations

5.3.1 Iteration and recursion of processes

Two forms of process application, iterative and recursive, are essential and useful for applying the processes defined in this document. Iteration and recursion notions of life cycle processes are explained in detail in ISO/IEC/TS 24748-1 and ISO/IEC/TR 24748-2.

5.3.1.1 Iterative application of processes

When the application of the same process or set of processes is repeated on the same system, the application is referred to as iterative.

The System integration process is used iteratively starting from a first aggregate of system elements until the completion of the system-of-interest; the last iteration of the process execution results in the entirely integrated system-of-interest.

The Verification process is called by the System integration process to perform the planned verification procedures related to the physical integration iteration.

The Validation process is used iteratively performing the planned validation procedures related to a set of validation actions.

5.3.1.2 Recursive application of processes

When the same set of processes or the same set of process activities are applied to successive levels of system elements within the system structure, the application method is referred to as recursive.

The System integration process is used by any system at any level of the decomposition of the system-of-interest.

5.3.2 Integration Enabling System

The integration engineering activities and the associated means can be viewed as an enabling system that enables or supports the system-of-interest to be integrated (see Note 1 and Note 2). This Integration Enabling System is a self-sufficient system in which its life cycle activities interact with those for engineering the system-of-interest to enable successful integration of the system-of-interest, as shown in Figure 3.

NOTE 1 The notion of “enabling system” is defined in ISO/IEC/IEEE 15288 and can apply in the case of integration. The Integration Enabling System has its own life cycle composed of generic stages and uses the generic system life cycle processes of ISO/IEC/IEEE 15288, but instantiated for each system-of-interest to be integrated.

NOTE 2 Other independent enabling systems can exist for validation and delivery purposes or combined with the Integration Enabling System.

The purpose of the Integration Enabling System is to provide a framework that serves to prepare and execute the integration of any system-of-interest. It can be viewed as a facility including the necessary services and the means and tools for integration.

In its relationships with a system-of-interest, the role of the Integration Enabling System is to identify, realize or obtain, then operate and maintain all the necessary resources, means and tools for integrating a particular system-of-interest.

The benefit of such an organization (i.e. separation between the system-of-interest and the Integration Enabling System) resides in the time saving achieved through parallelism of tasks under two different projects. It is known that the development of the Integration Enabling System may take a long time, sometimes longer than the system-of-interest development itself.

An Integration Enabling System may be implemented by an external company or as a separated project from the system-of-interest development. For a particular system-of-interest, a generic Integration Enabling System shall be instantiated to take into account the particularities of this system-of-interest.

Figure 3 represents some life cycle processes, as they may be instantiated and used concurrently for the system-of-interest and the Integration Enabling System, showing their main exchanges and how they fit together. In Figure 3, the arrows between the processes (boxes) represent some exchanged input and output flows. They do not represent the sequence of execution; iterations are not represented.

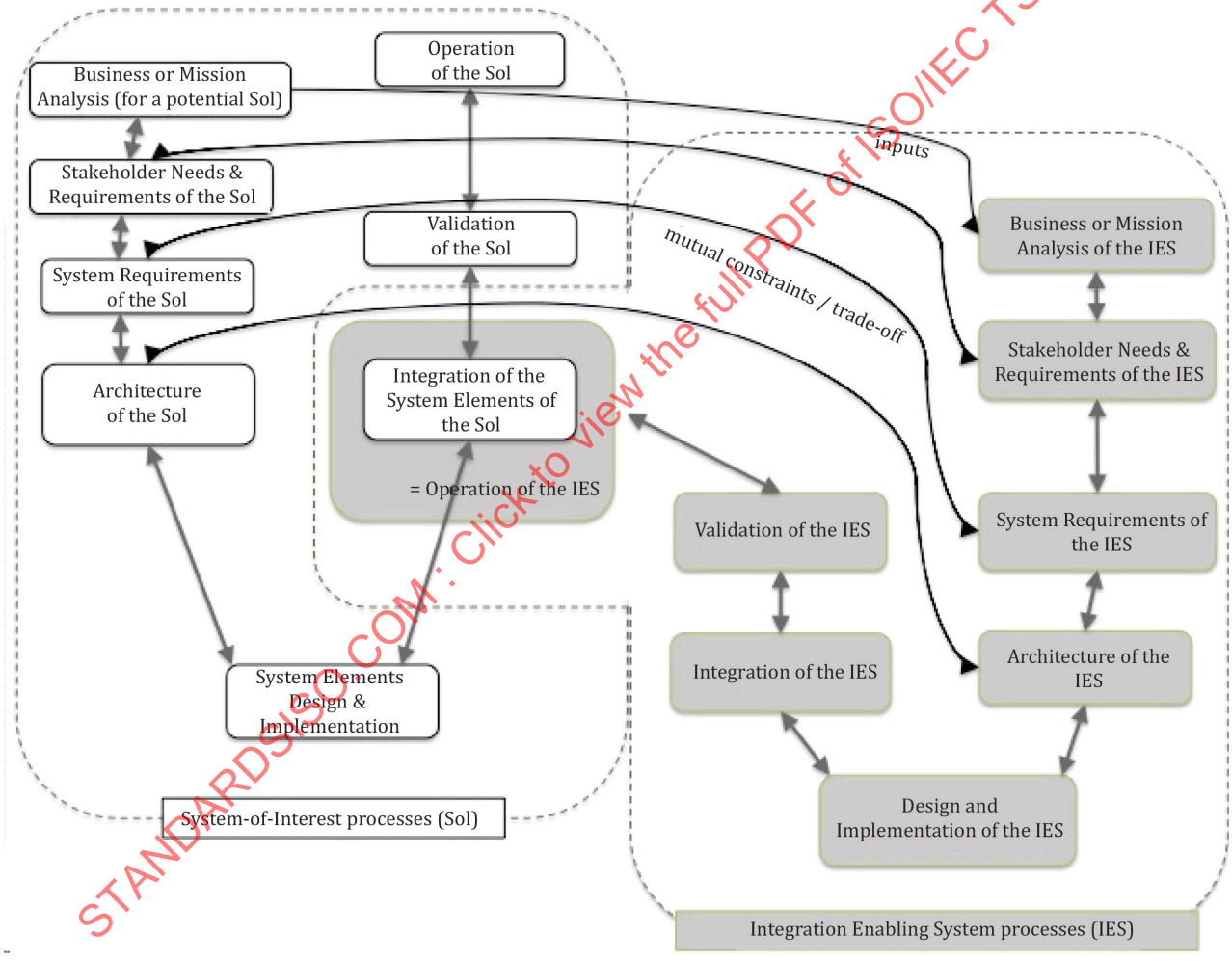


Figure 3 — Interactions between relating processes used by the system-of-interest and the Integration Enabling System

The technical life cycle processes applied and instantiated to the Integration Enabling System that are mentioned in Figure 3 are listed below.

— The Business or mission analysis process for the Integration Enabling System includes, in particular, the preliminary study of integration alternative strategies with respect to inputs from operational concept and operational scenarios of the system-of-interest. This study consists of the following:

  — analysing the operational concept and scenarios of the system-of-interest in order to identify and imagine the necessary integration concepts, principles and means to set up;

  — performing feasibility, effectiveness, cost and risk analyses concerning the integration of the system-of-interest.

— The Stakeholder needs and requirements definition process for the Integration Enabling System includes, in particular, the identification of stakeholders of the integration of the system-of-interest, their expectations and constraints and the definition of high-level objectives in terms of effectiveness, delivery time and cost. The resulting Stakeholder Requirements for integration means (tools, simulators, integration procedures, verification procedures, etc.), services and their utilization are deduced from the integration strategy of the system-of-interest.

— The System requirements definition process for the Integration Enabling System includes the definition of the requirements applicable to the integration means (tools, simulators, integration procedures, verification procedures, etc.), services and their utilization, the refinement of objectives (to define effectiveness requirements) and refinement of constraints (to define delivery timing and cost limits).

— The Architecture definition process for Integration Enabling System includes, in particular:

  — the participation in the refinement of the integration strategy (concerning mainly the relevance of aggregates definition, verification actions and applicable techniques or methods);

  — the development of a logical view of the Integration Enabling System (i.e. how assembly and verification tasks should be executed and synchronized: scheduling of delivery and assembly of the system elements of the system-of-interest, planning of the execution of the verification actions);

  — the projection/allocation of the logical view onto a physical view including adequate means (tools, simulators, integration procedures, verification procedures, etc.), resources (skills, personnel, etc.);

  — the initiation of acquisition/development of these various means through the definition of their respective requirements.

— The Design definition process and the Implementation process for the Integration Enabling System include the design characteristics definition and realization (either development, reuse, manufacturing or NDI buying) of every means (tools, simulators, integration procedures, verification procedures, etc.), as well as training of human resources providing suitably qualified and experienced operators executing the integration procedures, verification procedures, etc.

— The Integration process for the Integration Enabling System includes the acceptance and delivery of every means (tools, simulators, integration procedures, verification procedures, etc.), their assembly or connection and verification of their interfaces and functionalities.

— The Validation process for the Integration Enabling System includes the validation of these means (tools, simulators, integration procedures, verification procedures, etc.) together as a set ready for operation.

— The Operation process for the Integration Enabling System is nothing else than the execution of the integration of the system elements of the system-of-interest using the means (tools, simulators, integration procedures, verification procedures, etc.) defined, implemented, integrated and validated previously. As the whole validation is meant to be done on the system-of-interest, validation may

be an initial part of operation and sometimes, cannot be carried out on its own. Then, the final validation of the enabling system can be provided only thanks to the usage of the system-of-interest as an interface.

6 Processes

6.1 Integration engineering activities

From a process perspective, integration engineering, in particular, includes the activities of the Integration process (see 6.2), the activities of the processes it calls (Verification process and Validation process) and the activities of the other life cycle processes having relationships with integration. Activities of technical processes related to integration engineering are described in 6.3, in particular, the Verification process and the Validation process.

Guidelines for Processes:

The Integration process is described below on the basis of the process as described in ISO/IEC/IEEE 15288 with extended details.

In this document, the related processes are elaborated upon, in order to provide the user of the present document with additional planning and implementation guidance. The original ISO/IEC/IEEE 15288 tasks relevant to this document are highlighted in a box to show the reader the original text.

6.2 Integration Process

In case of a full conformance claim to this document, all provisions of 6.2 shall be applied.

6.2.1 Purpose

As stated in ISO/IEC/IEEE 15288:2015, 6.4.8.1:

The purpose of the Integration process is to synthesize a set of system elements into a realized system (product or service) that satisfies system requirements, architecture, and design.

This process assembles the implemented system elements. Interfaces are identified and activated to enable interoperation of the system elements as intended. This process integrates the enabling systems with the system-of-interest to facilitate interoperation.

NOTE 1 For a given level of the system hierarchy, this process iteratively combines implemented system elements to form complete or partial system configurations in order to build a product or service. It is used recursively for successive levels of the system hierarchy.

NOTE 2 The interfaces are defined by the Architecture definition and Design definition processes. This process coordinates with these other processes and checks to make sure the interface definitions are adequate and that they take into account the integration needs.

Additional expressions and discussions follow.

The goal of the Integration process is to assemble the implemented system elements in order to obtain the system (made of products and/or services) that is compliant with its architecture and design characteristics, properties and descriptions, its System Requirements, its Stakeholder Requirements and achieves the intended system purpose in its intended context of use.

Regarding interfaces, one of the tasks of this process is to check that the implemented system elements interface correctly. In other words, to check that the interfaces between the implemented system elements, correspond to their definition obtained during the execution of the Architecture and Design definition processes. In order to perform this check, the actual interfaces are first identified on the implemented system elements, and then activated through the assembly of the concerned system elements (physical aspect of the interfaces) and through the execution of verification actions (logical aspect of the interfaces). These checks have to demonstrate that the system elements interoperate as intended in the system definition.

Regarding enabling systems, the system-of-interest is obtained progressively through the assembly of implemented system elements to form aggregates (see 5.2.2.1) on which verification actions are executed. To facilitate the assembly and the execution of verification actions, different kinds of means (products, systems, services) are permanently or temporarily integrated to the concerned aggregate (see 5.3.2). This principle is extended to the integration of the complete system-of-interest in its operational environment, in which other necessary enabling systems are connected/integrated to the system-of-interest (for example, a maintenance system or a logistics support system) so that the system-of-interest can achieve its expected mission.

6.2.2 Outcomes

As stated in ISO/IEC/IEEE 15288:2015, 6.4.8.2:

As a result of the successful implementation of the Integration process:

a) Integration constraints that influence system requirements, architecture, or design, including interfaces, are identified.

b) Approach and checkpoints for the correct operation of the assembled interfaces and system functions are defined.

c) Any enabling systems or services needed for integration are available.

d) A system composed of implemented system elements is integrated.

e) The interfaces between the implemented system elements that compose the system are checked.

f) The interfaces between the system and the external environment are checked.

g) Integration results and anomalies are identified

h) Traceability of the integrated system elements is established.

Guidelines:

Regarding integration constraints, the technologies that support the system elements and the way to integrate these implemented system elements (how to assemble or connect them and what verification actions to execute) have impacts to the definition of the system. Those impacts have to be identified and translated into system requirements, corresponding architecture arrangements or characteristics and design properties, using the applicable corresponding processes. Most of the time, those impacts concern the interfaces.

Regarding the correct operation of interfaces and system functions, one has to identify the potential causes of any assembly errors that could alter the integrity of system elements, their interfaces and prevent system malfunctions. The assembly errors may happen because of wrong system elements, wrong configurations, wrong interfaces, several possible right or wrong ways to connect the interfaces of system elements. They may affect security, safety of operators, operability, interoperability, etc. To avoid such issues, an approach and checkpoints have to be defined early during the system definition stage.

Regarding enabling systems or services, they have to be available prior the execution of the integration steps. This, in particular, means that the enabling systems or services have been defined, developed, qualified/validated prior using them (this remark justifies the Integration Enabling System considerations in 5.3.2).

Regarding the traceability of the integrated system elements, the Integration process establishes the configuration status of each implemented system element that is integrated (i.e. which version is integrated, the history, rationale and impacts of modifications or changes) using the Configuration Management process, the results of integration execution are recorded as well.

6.2.3 Activities and tasks

Refer to ISO/IEC/IEEE 15288:2015, 6.4.8.3.

The project shall implement the following activities and tasks in accordance with applicable organization policies and procedures with respect to the Integration process.

a) **Prepare for integration.** This activity consists of the following tasks:

1) Identify and define checkpoints for the correct operation and integrity of the interfaces and the selected system functions.

NOTE 1 Detailed verification of the interfaces is performed using the Verification process.

NOTE 2 Refer to ISO/IEC/IEEE 15026 and the ISO/IEC 27000 series for information on assurance, integrity, and security. Consider anti-counterfeit, anti-tamper, system and software assurance and interoperability elements when identifying and defining checkpoints.

2) Define the integration strategy

NOTE 1 The integration is performed according to a predefined integration strategy that sequences the order for assembling the implemented system elements based on the priorities of the system requirements and architecture definition focusing on the interfaces, while minimizing integration time, cost, and risks.

NOTE 2 This strategy often provides for verification against a sequence of progressively more complete system element configurations. It is dependent on system element availability and is consistent with a fault isolation and diagnosis strategy. Wherever possible, an integrated configuration includes its human operators. Successive applications of the Integration process and the Verification process, and when appropriate the Validation process, are repeated for systems at successive levels until the system-of-interest has been realized.

Guidelines:

Generally, the system-of-interest cannot be obtained in a single round of assembly. As mentioned in 5.2.6, a strategy has to be defined describing the way the integration is to be prepared for and executed. The complete strategy is defined taking into account the following considerations and dispositions:

— analyze the system architecture description, in particular, physical and logical models, in order to
  — identify the system elements and their physical interfaces,
  — identify the system elements that perform the core functioning of the system,
  — identify the system elements that perform functionalities related to the core functioning of the system, and
  — identify the system elements that interface with the external elements to the system;
— analyse system requirements and architecture descriptions to identify assembly priorities for system elements;
— study alternative integration techniques or methods (see 5.2.6.1) that could be relevant with the system to be integrated;
— study alternative aggregates and set of aggregates that could predefine the order of assembly;
— analyze the Verification and Validation strategies to define which system elements and which aggregates of system elements are necessary to execute the selected verification actions. These strategies contain the list of verification and validation actions that will have to be executed; this list is the result of trade-offs between what should be verified and validated and what can be verified and validated taking into account multiple constraints such as cost, deadlines, feasibility, security, safety, etc. risks have to be minimized if verification actions are dropped;
— perform hazard analysis, security analysis, safety analysis and more generally risk analysis to prevent unintended or undesired consequences that could appear during (or be generated by)

integration performance. These consequences concern the integrity of the system itself, the safety of operators inside or outside the system, the integrators, the environmental conditions (material can be found in IEEE 1012-2012, Annex J[14]);

— assess and select one integration technique or method, or define a mix of these taking into account the preceding studies;

— define the order for assembling the system elements through the set of aggregates previously defined;

— consider project parameters and constraints to minimize integration time, cost and risk; synchronize integration tasks with the master schedule of the project.

The integration strategy may include early dispositions and tasks related to other development activities in order to facilitate the integration. In particular, consider using virtual environments during architecture and design activities to anticipate potential integration difficulties, hazards or threats, reduce integration time, cost and risk. These virtual environments may also be used to simulate the integration of implemented system elements in order to predefine the assembly procedures.

Another disposition may be studied within the strategy to facilitate the maintainability of the system-of-interest during integration (and subsequent activities such as validation, operation, maintenance) and/or to help efficient analysis causes of problems encountered during integration: consider to add functions in the system-of-interest, or in any enabling system, such as log data gathering and recording, help to diagnosis, etc.

The integration strategy is documented in the Integration Plan. The Integration Plan also indicates where (sites location), when (schedules), how (integration aggregate definition sheets) and who (skills) performs the integration activities.

3) Identify and plan for the necessary enabling systems or services needed to support integration. NOTE This includes identification of requirements and interfaces for the enabling systems. Enabling systems for integration include the integration facilities, assembly equipment, training systems, discrepancy reporting systems, simulators, measurement devices, and facility security. Changes needed for the enabling systems to support the integration tasks need to be identified and defined. The needs for these changes are provided to the stakeholders that govern the enabling systems.

Guidelines:

5.3.2 explains the concept of enabling system instantiated to the specific integration means and resources. It provides guidance on setting up Integration Enabling Systems and their relationships with the system-of-interest to be integrated.

4) Obtain or acquire access to the enabling systems or services, and materials to be used. NOTE The Validation process is used to objectively confirm that the Integration Enabling System achieves its intended use for its enabling functions.

Guidelines:

5.3.2 provides guidance on setting up integration of enabling systems and their relationships with the system-of-interest to be integrated, in particular, about the acquisition and validation of integration means.

5) Identify system constraints from integration to be incorporated in the system requirements, architecture or design. NOTE This includes requirements such as accessibility, safety for integrators and operators, required interconnections for sets of implemented system elements and for enablers, and interface constraints.

Guidelines:

As defined above, the integration strategy studies are conducted in order to define where (sites location), when (schedules), how (Integration Aggregate Definition Sheets) and who (skills and staffing) performs the integration activities.

These definitions generate constraints, some of which have to be incorporated into system requirements baselines, architecture description and design documents as feedbacks. The constraints to be incorporated in baselines influence the utilization or operation of the system (such as checkpoints to be incorporated in the system), the architecture and design characteristics (e.g. place or arrangement of implemented system elements, input-output flows of information or commands exchanged with operators or users, interfaces with harnesses or simulators, drivers/launchers and stubs).

b) **Perform integration**. Successively integrate system element configurations until the complete system is synthesized [assembled]. This activity consists of the following tasks:

1) Obtain implemented system elements in accordance with agreed schedules.

NOTE The implemented system elements are received from suppliers, the acquirer, or withdrawn from storage. System elements are handled in accordance with relevant health, safety, security and privacy considerations. As part of the acceptance of the implemented system elements, each [system] element is checked to ensure it has been verified and validated against acceptance criteria specified in an agreement. The delivered configuration, conformance, compatibility of interfaces, the presence of mandatory information items are checked. Implemented system elements that do not pass verification are identified as such and handled in accordance with defined procedures.

Guidelines:

The note added in the standard is sufficient to understand and perform this task.

2) Assemble the implemented system elements.

NOTE The assembly is performed to achieve system element configuration (complete or partial) connecting the implemented system elements as prescribed in the integration strategy, using the defined assembly procedures, interface control descriptions, and the related integration enabling systems.

Guidelines:

The assembly of the implemented system elements follows the integration strategy as previously defined and documented in the Integration Plan, in particular the order of assembly.

The integration performance is generally divided in several steps, assembling at each step some implemented system elements to form aggregates and integration configurations (composed of the aggregates and the necessary related enabling systems, tools or services); refer to 5.2.6.1. An integration step may start and end with two optional milestones (that can be called “Integration Step Readiness Review” and “Integration Step Results Review”) separating the effective execution of the assembly actions, verification actions and validation actions of each step.

Upstream of integration performance, for each integration step, an Integration Procedure is defined from the concerned System Integration Aggregate Sheets and documented (see 7.2). The Integration Procedure provides all necessary information and data to execute the concerned step (see 7.3). These information items are defined and deduced from the integration strategy. They may be optional depending on criteria such as, for example, safety, security and complexity of assembly operations.

In each step, the integration procedure is used to execute the assembly of the concerned aggregate(s). The Integration Procedure explains, in particular, how the system elements are handled and connected (physical interfaces) using the enabling means, tools, systems (harnesses, simulators, etc.) through actions to be directed or done by human operators. The integration operators should be skilled personnel trained to carry out assembly activities on simulators or mock-ups, prior to executing these activities onto the target implemented system elements. This disposition is optional depending on criteria such as, for example, safety, security and complexity of assembly operations.

When an implemented system element is assembled to another, the physical interfaces are checked more or less simultaneously. Several verification actions are often necessarily performed before assembling or connecting the implemented system elements, otherwise accidents may happen (see 5.2.7).

3) Perform check of the interfaces, selected functions, and critical quality characteristics. NOTE This is performed to ensure the operation of the interfaces (external and internal), functions, and quality characteristics. The interfaces are checked using the Verification process against the interface requirements.

Guidelines:

The objective of this task is to check the assembly of implemented system elements and the interfaces or interaction mechanisms between system elements and between the external components of the system. This is done in order to establish the conformance of the realized system (product, service) to the system architecture description and design descriptions.

Normally, the checks of the physical interfaces are performed before and/or during the assembly of system elements to form an aggregate (see previous task of the Integration process) using the Verification process.

In each integration step, when the implemented system elements have been assembled to form an aggregate, the aggregate is prepared in order to check the exchanged input-output flows (logical interfaces), the functions it has to perform and other architectural characteristics and design properties or quality characteristics. This preparation may consist of the following:

— adding and connecting to the system elements, through checkpoints, equipment such as probes, sensors, spies, trackers, specific analysers, etc. in order to observe the behaviour of the aggregate;

— adding and connecting drivers/launchers and stubs to simulate missing or absent implemented system elements or external components to the system-of-interest.

Then, the aggregate is operated using the Verification process and/or the Validation process (depending on the characteristics to be checked); measures, and data are collected during or at the end of the operation of the aggregate.

c) **Manage results of integration.** This activity consists of the following tasks: 1) Record integration results and any anomalies encountered. NOTE This includes anomalies due to the integration strategy, the Integration Enabling Systems, execution of the integration or incorrect system or [system] element definition. Where inconsistencies exist at the interface between the system, its operational environment and any systems that enable the utilization stage, the deviations lead to corrective actions or requirement changes. The Project assessment and control process is used to analyse the data to identify the root cause, enable corrective or improvement actions, and to record lessons learned.

Guidelines:

The note added in the standard is sufficient to understand and perform this task.

2) Maintain traceability of the integrated system elements. NOTE Bi-directional traceability is maintained between the integrated system elements and the integration strategy, system architecture, design, and system requirements including interface requirements and definitions that are necessary for integration.