ISO/IEC/IEEE FDIS 24641:~~2022~~2023(E)

ISO/IEC/JTC1/SC7/WG4 N1584

Date: ~~2022-11-20~~2023-01-17

Systems and software engineering – Methods and tools for model-based systems and software engineering

ISO copyright office	IEC Central Office	Institute of Electrical and Electronics Engineers, Inc.
CP 401 • Ch. de Blandonnet 8	3. Rue de Varembé	3 Park Avenue, New York
CH-1214 Vernier, Geneva	CH-1211 Geneva 20	NY 10016-5997, USA`
Tel. +41 22 749 01 11	Switzerland	E-mail stds.ipr@ieee.org
Fax +41 22 749 09 47	E-mail inmail@iec.ch	Web www.ieee.org
E-mail: copyright@iso.org	Web www.iec.ch	
Web www.iso.org		

Published in Switzerland

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.

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 ISO/IEC documents should be noted. This document was drafted in accordance with the rules given in the ISO/IEC Directives, Part 2 (see www.iso.org/directives or www.iec.ch/members_experts/refdocs).

IEEE Standards documents are developed within the IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Association (IEEE-SA) Standards Board. The IEEE develops its standards through a consensus development process, approved by the American National Standards Institute, which brings together volunteers representing varied viewpoints and interests to achieve the final product. Volunteers are not necessarily members of the Institute and serve without compensation. While the IEEE administers the process and establishes rules to promote fairness in the consensus development process, the IEEE does not independently evaluate, test, or verify the accuracy of any of the information contained in its standards.

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) or the IEC list of patent declarations received (see https://patents.iec.ch).

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

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT), see www.iso.org/iso/foreword.html. In the IEC, see www.iec.ch/understanding-standards.

ISO/IEC/IEEE 24641 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology,* Subcommittee SC 7, *Software and systems engineering,* in cooperation with the IEEE Computer Society Systems and Software Engineering Standards Committee, under the Partner Standards Development Organization cooperation agreement between ISO and IEEE.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html and www.iec.ch/national-committees.

Introduction

As systems grow in scale and complexity, some in the ~~Systems Engineering~~systems engineering community turn to ~~Model-Based Systems~~model-based systems and ~~Software Engineering~~software engineering (MBSSE~~),~~) to, among other objectives, manage complexity, maintain consistency, and help ensure traceability during system development. With an MBSSE approach, the systems and software engineering activities rely on evolving models that serve as the main or major source of knowledge about the system-of-interest and its life cycle processes, which could be any entity subject to a system model such as a program, project, product, or company.

MBSSE benefits differ significantly from 'engineering with models', which has been a common practice among the engineering disciplines for decades and that is mainly based on independent discipline-specific models that, even if very useful for each discipline and system analysis contribution, do not provide an overall understanding of the architecture of the system sharable among stakeholders, e.g. ~~Computer Aided Design~~computer-aided design (CAD) for ~~Mechanical Engineering~~mechanical engineering, aerodynamics models, control loop simulations~~…)~~. In addition, due to the diversity of approaches and terminologies (e.g. ~~Model Driven Development~~model-driven development or MDD~~)~~), MBSSE usually falls within the context of a specific engineering discipline (e.g. MDD for the software engineering community).

MBSSE is the formalized application of modelling to support systems engineering or software engineering activities. Faced with the issues and challenges linked to the growing complexity of the systems to be developed, document-centric approaches are less and less suitable. The MBSSE approach makes it possible to develop logically consistent multi-view architecture description. These serve as a bridge to enable the traceable, verifiable and dynamic correlation of the system-of-interest and/or software-of-interest models cross multidiscipline and throughout its entire life cycle, and to drive the system ~~engineering process, activities and tasks at all levels of its hierarchy from system-of-systems to system element across multiple engineering disciplines and throughout all stages of its life cycle~~

~~MBSSE drives the system~~ and software engineering processes, activities and tasks at all levels of its hierarchy from system-of-systems to system element across multiple engineering disciplines and throughout all stages of ~~the~~its life cycle~~.~~

From MBSSE perspective, other engineering disciplines (mechanical, thermal, electronic, electrical, etc.) are also considered.

Thus, a need exists to specify the considerations necessary for undertaking the application of MBSSE within an organization. An organization needs to address the considerations necessary for supporting the establishment of each project environment within its overall ecosystem, and the exchange of models between stakeholder organizations.

This document addresses MBSSE-related processes by categorizing them into four process groups:

— Plan MBSSE

— Build ~~Models~~models

— Perform MBSSE

— Support ~~Models~~models

Each process is defined in terms of purpose, inputs, outcomes, and supporting tasks. The task descriptions include tool and method guidance and the recommended capabilities needed to successfully implement them. The relationships among the four process groups in this document, the four process groups in ISO/IEC/IEEE 15288 and ISO/IEC/IEEE 12207, and the life cycle model and stages in ISO/IEC/IEEE 24748-1 are described in Annex A.

This document is intended to benefit those who acquire, supply, develop, operate, and maintain MBSSE tools and methods. It can be used by:

a) organizations that need to implement or build models – to understand, adopt, and enact the MBSSE processes, tools, and methods (it also helps to evaluate and select relevant tools and methods based on business- and user-related criteria~~).~~);

b) tool vendors who facilitate or leverage MBSSE practices – to provide a set of recommended tool capabilities for planning MBSSE, building models, MBSSE performance, and support.

Systems of systems are considered in this document to benefit from the same processes, methods and tool capabilities as any system.

The relationships between this document and other standards are described in Annex E.

NOTE 1 This document prescribes a way to engineer systems and software based on models thanks to a reference model and four process groups; however, other particular uses of models which are out of the scope of this document are used in "model engineering" in other ways: For example, in model-driven modernization [also called architecture-driven modernization (ADM) in object management group (OMG) terms], models are (automatically) generated from the existing code and artefacts of a running system in order to represent it and then build a new system in a different platform. Another usage scenario of models occurs in what is called "models@runtime" whereby the models are used to change the system and evolve with it; these are normally used in self-adaptive systems to achieve the required system self-adaptation features.

NOTE 2 The reference model does not take into account the system evolution (and that of its related models) as a fundamental phase of systems or software engineering in the maintenance and evolution of the system and its models.

NOTE 3 The design within the different domains, for example, mechanical, hydraulics, electrical, electronics, control algorithms, and software, has been performed using model-based techniques for decades. However, each domain uses specialized languages and tool chains for its modelling activities. The guideline to propose how the methods, modelling languages and tools apply in these domains is outside of the scope of this document. However, the interfaces of the engineering models and the system models are crucial and essential for applying MBSSE.

In this document, the following verbal forms are used:

— "shall" indicates a requirement;

— "should" indicates a recommendation;

— "may" indicates a permission;

Systems and software engineering – Methods and tools for model-based systems and software engineering

1 Scope

This document deals with the tool capabilities and methods for model-based systems and software engineering (MBSSE). This document:

- ~~provides terms and definitions related to MBSSE;~~
- specifies a reference model for the overall structure and processes of MBSSE-specific processes, and ~~describe~~describes how the components of the reference model fit together;
- specifies interrelationships between the components of the reference model;
- specifies MBSSE-specific processes for model-based systems and software engineering; the processes are described in terms of purpose, inputs, outcomes and tasks;
- specifies methods to support the defined tasks of each process;
- specifies tool capabilities to automate or semi-automate tasks or methods.

This document does not bring any additional life cycle processes for system and software but specifies an MBSSE reference model considered as activities, not only from the life cycle perspectives of systems engineering problem solving and the system-of-interest evolution, but also from the cognitive perspectives of modelling and model management, which can sustain and facilitate the system and software life cycle processes during digital transformation and in the digital age. ~~The relationships among the four process groups in this document, the four process groups in ISO/IEC/IEEE 15288 and ISO/IEC/IEEE 12207, and the life cycle model and stages in ISO/IEC/IEEE 24748-1 are described in Annex A (informative).~~.

The processes defined in this document are applicable for a single project, as well as for an organization performing multiple projects or an enterprise. These processes are applicable for managing and performing the systems and software engineering activities based on models within any stage in the life cycle of a system-of-interest.

~~Systems of systems are considered in this document to benefit from the same processes, methods and tool capabilities as any system.~~

~~The relationships between this document and other standards are described in Annex E (informative).~~

~~NOTE 1 This document prescribes a way to engineer systems and software based on models thanks to a reference model and four process groups; however, other particular uses of models which are out of the scope of this standard are used in "Model Engineering" in other ways: For example, in Model-Driven Modernization (also called Architecture-driven modernization (ADM) in Object Management Group (OMG) terms), models are (automatically) generated from the existing code and artefacts of a running system in order to represent it and then build a new system in a different platform. Another usage scenario of models occurs in what is called "models@runtime" whereby the models are used to change the system and evolve with it; these are normally used in self-adaptive systems to achieve the required system self-adaptation features.~~

NOTE 2 The reference model does not take into account the system evolution (and that of its related models) as a fundamental phase of systems or software engineering in the maintenance and evolution of the system and its models.

NOTE 3 The design within the different domains e.g. mechanical, hydraulics, electrical, electronics, control algorithms, and software has been performed using model-based techniques for decades. However, each domain uses specialized languages and tool chains for their modelling activities. The guideline to propose how the methods, modelling languages and tools apply in these domains is outside of the scope of this document. However, the interfaces of the engineering models and the system models are crucial and essential for applying MBSSE.

62 Normative references

ISO/IEC Directives, Part 1, Procedures for the technical work

ISO/IEC Directives, Part 1, Consolidated ISO Supplement

ISO/IEC Directives, Supplement – Procedures specific to IEC

There are no normative references in this document.

73 Terms, definitions, and abbreviated terms

7.13.1 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

ISO, IEC, and IEEE maintain terminology databases for use in standardization at the following addresses:

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

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

— IEEE Standards Dictionary Online: available at http://dictionary.ieee.org

NOTE For additional terms and definitions in the field of systems and software engineering, see ISO/IEC/IEEE 24765, which is published periodically as a “snapshot” of the SEVOCAB (Systems and Software Engineering Vocabulary) database and is publicly accessible at computer.org/sevocab.

3.1.1
analytical model
model (3.1.15) that describes mathematical relationships, such as differential equations that support quantifiable analysis about the *system* (3.1.35) parameters

Note 1 to entry: Analytical models can be further classified into dynamic and *static* model.*models* (3.1.34).

3.1.2
asset
item, thing, or entity that has potential, or actual value to an organization

Note 1 to the entry: Assets can be classified from different perspectives, such as tangible assets, intangible assets; moveable assets, immoveable assets. Intangible assets can be classified into digital assets and non-digital intangible assets.

Note 2 to the entry: Cognitive assets refer to intangible assets generated by an organization in the course of its operations. Data, information, knowledge, wisdom, and modelling assets are all belong to cognitive assets..

[SOURCE: ISO/IEC 19770-1:2017, 3.1, modified — The original three notes ~~are cancelled,~~to entry have been replaced by two ~~or three~~ new notes ~~are added~~to entry.]

3.1.3
capability
ability to do something useful under a particular set of conditions

~~NOTE —~~Note 1 to entry: Generally, different kinds of capabilities exist: organizational capability, *system* (3.1.35) capability, and operational capability. Organizational capabilities relate through the work practices that are adopted by the organizations. New systems (with new~~,~~ or enhanced system capabilities) are developed to enhance enterprise operational capability in response to ~~stakeholder's~~*stakeholders'* (3.1.32) *concerns* (3.1.5) about a problem situation. Operational capabilities provide operational services that are enabled by system capabilities. These system capabilities are inherent in the system that is conceived, developed, created, and/or operated by an enterprise. Enterprise SE concentrates its efforts on maximizing operational value for various stakeholders, some of whom ~~may~~can be interested in the improvement of some problem situation.

3.1.4
concept of operations
~~user-definition of how the overall organization will operate to satisfy its mission~~

verbal and graphic statement, in broad outline, of an organization's assumptions or intent in regard to an operation or series of operations of new, modified, or existing organizational *systems* (3.1.35)

Note 1 to entry: The concept of operations frequently is embodied in long-range strategic plans and annual operational plans. In the latter case, the concept of operations in the plan covers a series of connected operations to be carried out simultaneously or in succession to achieve an organizational performance objective. See also *operational concept* (3.1.24).

Note 2 to entry: The concept of operations provides the basis for bounding the operating space, system capabilities, interfaces, and operating environment.

[SOURCE: ISO/IEC/IEEE 15288~~:2015~~:—, 3.9]

3.1.5
concern
matter of interest or importance to a *stakeholder* (3.1.32)

EXAMPLE Affordability, agility, availability, dependability, flexibility, maintainability, *reliability*~~,~~ (3.1.27), *resilience*~~,~~ (3.1.28), usability and viability are examples of concerns. Survivability, depletion, degradation, loss, obsolescence are examples of concerns. The PESTEL mnemonic is a reminder of possible areas of concern: political, economic, social, technological, environmental, and legal.

[SOURCE: ISO/IEC/IEEE 42020:2019, 3.8~~]~~.]

3.1.6
descriptive model
model (3.1.15) that shows an interconnected set of *model elements* (3.1.19) which represent key *system* (3.1.35) aspects including its structure, behaviour, parametric, and requirements

Note 1 to entry: Descriptive models are complementary to analysis and design models of a system.

3.1.7
discipline-specific model

representation of a *system*~~,~~ (3.1.35), or system elements from the perspective of a discipline addressing domain-specific *concerns* (3.1.5) where the *model elements* ~~comes~~(3.1.19) come from a specific discipline

3.1.8
maturity
degree to which a *system*~~,~~ (3.1.35), product, or component meets needs for *reliability* (3.1.27) under normal operation

Note 1 to entry: The degree of maturity of a system, product or component can be associated with trust in, and good knowledge of, the behaviour of the system.

Note 2 to entry: The system, product or component is considered immature if there are still flaws~~,~~ or missing parts that prevent users from benefitting from the item. The item is considered mature if there are no flaws~~,~~ or missing parts that prevent users from benefitting from the item.

[SOURCE: ISO/IEC 25010:2011, 4.2.5.1, modified — The original note to entry has been replaced by two new notes ~~1 and 2 are added]~~to entry.]

3.1.9
maturity level
degree of achievement to which all goals have been attained

~~[SOURCE: ISO/IEC 33001:2015, 3.4.1, modified]~~

3.1.10
measure of effectiveness
~~**MoE**~~
MOE
operational measure of success that is closely related to the achievement of the operational objective being evaluated~~,~~ in the intended operational environment under a specified set of conditions

[SOURCE: ISO/IEC/IEEE 24748-~~2:2018~~4:2016, 4.7]

3.1.11
measure of performance
~~**MoP**~~
MOP
engineering parameter that provides critical performance requirements to satisfy a *measure of effectiveness* (~~MoE~~*MOE)* (3.1.10)

Note 1 to entry: An ~~MoP~~MOP typically characterizes physical or functional attributes relating to the *system* (3.1.35) operation.

[SOURCE: ISO/IEC/IEEE 24748-4:2016, 4.8~~]~~.]

3.1.12
meta-model
~~**metamodel**~~
~~a~~ special kind of model that specifies the abstract syntax of a modelling language~~.~~

Note 1 to entry: The typical role of a meta-model is to define the semantics for how *model elements* (3.1.19) in a *model* (3.1.15) get instantiated. A model typically contains model elements. These are created by instantiating model elements from a meta-model (i.e.~~,~~, meta-model elements).

[SOURCE: ISO/IEC 19506:2012 ~~Information technology — Object Management Group Architecture-Driven Modernization (ADM) — Knowledge Discovery Meta-Model (KDM),~~ Clause 4~~]~~.]

3.1.13
mission
important operational job or duty assigned to a *resource* (3.1.29) or a group of resources or certain groups of people

Note 1 to entry: A resource can be a human resource or a technical resource including *systems* (3.1.35) and products.

3.1.14
mode
definition of the expected behaviour of the *system* (3.1.35) (or of its actors, or of its components) in situations foreseen at design time

Note 1 to entry: Each mode is mainly characterized by the expected functional content of the system in this mode. A mode can reflect various concepts, such as:

— ~~Phases~~*phases* (3.1.25) of a *mission*~~,~~ (3.1.13) or of a flight for example (taxiing, taking-off, cruising, landing, etc~~.;~~.);

— ~~Specific~~specific required functioning of the system under certain conditions (connected, autonomous, etc.);

— ~~Specific~~specific conditions where the system is used; test, training, maintenance, etc~~.~~.

The transition from one mode to another is in general the result of a decision, such as a change in the way the system operates, in order to adapt to new needs, or new contexts; it is therefore conditioned by the choices of the system, its users, or of external actors.

3.1.15
model
abstract representation of an entity or collection of entities that provides the ability to portray, understand or predict the properties or characteristics of the entity or collection under conditions or situations of interest

Note 1 to entry: A model can use a formalism that could be based on mathematical or scientific principles and concepts. A model can be generated using an established ~~metamodel. Metamodels~~meta-model. Meta-models are often used to facilitate development of accurate, complete, consistent and understandable models.

Note 2 to entry: A model can be used to construct or express architecture views of the entity. *Descriptive models* (3.1.6) and analytic models are two kinds of models. A model ~~can~~should be governed by a model kind in accordance with ISO/IEC/IEEE 42010 ~~Systems and software engineering - Architecture description~~.

Note 3 to entry: A reference model can be used to capture a general case that is used as the basis for creating special case models for particular conditions or situations. A reference model can be used to encourage and enforce uniformity of architectures and architecture elements.

Note 4 to entry: The model can be an architecture model, architecture entity model, concept model or reference model, as the case may be.

Note 5 to entry: A physical model is a concrete representation that is distinguished from the mathematical and logical models, both of which are more abstract representations of the *system*~~.~~ (3.1.35). The abstract model can be further classified as descriptive (similar to logical) or analytical (similar to mathematical).

Note 6 to entry: (modelling and simulation) An approximation, representation, or idealization of selected aspects of the structure, behaviour, operation, or other characteristics of a real-world process, concept, or system. Models can have other models as components (Authoritative Dictionary of IEEE Standards Terms).

Note 7 to entry: An example of classification of models and relationships in MBSSE is given (see Annex C).

[SOURCE: ISO/IEC/IEEE 42020:2019, 3.13, notemodified — notes 5, 6 and 7 areto entry have been added].]

3.1.16
model-based systems and software engineering
MBSSE
formalized applications of modelling to support *systems* (3.1.35) and software engineering

3.1.17
model baseline
immutable set of *model configuration items* (3.1.18) with their associated versions and variants

3.1.18
model configuration item
MCI
logical part of the *model* (3.1.15) that is maintained in a controlled fashion, having a trackable revision history

Note 1 to entry: A model as well as any model element and its references can be part of a MCI. A MCI can be defined in different granularities, from a set of model elements, to the entire Model.EXAMPLE: MCIs are managed to maintain the integrity of the models.

Examples:

— Model (what is developed within a particular project)

— Each main package under the model root

— Catalogues which can be reusable as Libraries (e.g. Functions, Services, Measurements)

Note 1 to entry: A model as well as any *model element* (3.1.19) and its references can be part of a MCI. A MCI can be defined in different granularities, from a set of model elements, to the entire model. MCIs are managed to maintain the integrity of the models.

3.1.19
model element
atomic (elementary) itemsitem that representrepresents an individual components, actions, states, messages, properties, relationships, and other itemscomponent, action, *state* (3.1.33), message, property, relationship, or another item that describedescribes the composition, characteristics, or behaviour of a *system* (3.1.35)

3.1.20
model element library
set or catalogue of non-modifiable *model elements* (3.1.19) usable within any project, packaged in a single artefact

3.1.21
model pattern

general, reusable *model* (3.1.15) or model part that can be used as a solution to a commonly occurring problem within a given context in *system* (3.1.35) or software design

3.1.22
model repository
means to store different *models* (3.1.15) at different levels of abstraction and to facilitate understanding and ~~co-operation~~cooperation between *stakeholders* (3.1.32) and practitioners at different levels

~~SOURCE: Adapted from TOGAF 9.2 (The Open Group Architecture Framework).~~

[SOURCE: Adapted from TOGAF 9.2 (The Open Group Architecture Framework)]

3.1.23
ontology
logical structure of the terms used to describe a domain of knowledge, including both the definitions of the applicable terms and their relationships

[SOURCE: ISO/IEC/IEEE 24765:2017, 3.2691]

3.1.24
operational concept
verbal and graphic statement of an organization's assumptions or intent in regard to an operation or series of operations of a specific *system* (3.1.35) or a related set of specific new, existing or modified systems

Note 1 to entry: The operational concept is designed to give an overall picture of the operations using one or more specific systems, or set of related systems, in the organization's operational environment from the users' and operators' perspectives. See also *concept of operations* (3.1.4).

Note 2 to entry: The operational concept is about systems, while a concept of operations typically refers to organizations.

[SOURCE: ISO/IEC/IEEE 15288~~:2015~~:—, 3.23]

3.1.25
phase
period of time in the life cycle during which activities are performed that enable achievement of objectives for that phase

[SOURCE: ISO/IEC/IEEE 42020:2019, 3.15]

3.1.26
reference framework
structure for understanding significant relationships among the entities of some environment, and for the development of consistent standards or specifications supporting that environment

~~[SOURCE: ISO/IEC 20013:2020, 3.5]~~

Note 1 to entry: A reference framework provides a common backplane for consistency, collaboration, sharing, and reuse.

[SOURCE: ISO/IEC 20013:2020, 3.5, modified — The domain "<e-Portfolio>" has been removed; the original note to entry has been replaced by a new one.]