TECHNICAL SPECIFICATION

ISO/TS 18876-1

First edition
2003-11-01

Industrial automation systems and integration — Integration of industrial data for exchange, access and sharing —

Part 1:
Architecture overview and description

Systèmes d'automatisation industrielle et intégration — Intégration des données industrielles pour l'échange, l'accès et le partage —

Partie 1: Vue d'ensemble et description de l'architecture

Reference number
ISO/TS 18876-1:2003(E)

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Contents

Page

Figures

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of technical committees is to prepare International Standards. Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75 % of the member bodies casting a vote.

In other circumstances, particularly when there is an urgent market requirement for such documents, a technical committee may decide to publish other types of normative document:

— an ISO Publicly Available Specification (ISO/PAS) represents an agreement between technical experts in an ISO working group and is accepted for publication if it is approved by more than 50 % of the members of the parent committee casting a vote;

— an ISO Technical Specification (ISO/TS) represents an agreement between the members of a technical committee and is accepted for publication if it is approved by 2/3 of the members of the committee casting a vote.

An ISO/PAS or ISO/TS is reviewed after three years in order to decide whether it will be confirmed for a further three years, revised to become an International Standard, or withdrawn. If the ISO/PAS or ISO/TS is confirmed, it is reviewed again after a further three years, at which time it must either be transformed into an International Standard or be withdrawn.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights.

ISO/TS 18876-1 was prepared by Technical Committee ISO/TC 184, *Industrial automation systems and integration*, Subcommittee SC 4, *Industrial data*.

This International Standard is organized as a series of parts, each published separately. The structure of this International Standard is described in this part of ISO 18876.

0 Introduction

0.1 Overview of ISO 18876

This International Standard establishes an architecture, a methodology, and other specifications for integrating industrial data for exchange, access, and sharing. It supports:

— data sharing and data integration;

— specification of mappings between models; and

— data transformation.

0.2 Organization of this part of ISO 18876

This part of ISO 18876 is organized as follows:

— Clause 1 specifies the scope and field of application of the International Standard and of this part of ISO 18876;

— Clause 2 identifies additional standards that, through references in this part of ISO 18876, constitute provisions of this part of ISO 18876;

— Clause 3 defines terms and abbreviations used in this part of ISO 18876;

— Clause 4 describes the organization of this International Standard;

— Clause 5 describes the fundamental concepts and assumptions on which this International Standard is based;

— Clause 6 provides an overview of the model integration process;

— Clause 7 identifies some components of the integration architecture;

— Clause 8 provides an overview of the processes of data mapping and consolidation;

— Clause 9 summarizes the relationships with other standards.

0.3 Target audiences

The target audiences for this part of ISO 18876 are as follows:

— technical managers wishing to determine whether ISO 18876 is appropriate for their business needs;

— implementers wishing to obtain an overview of its contents.

Industrial automation systems and integration — Integration of industrial data for exchange, access and sharing — Part 1: Architecture overview and description

1 Scope

This Technical Specification establishes an architecture, a methodology, and other specifications for integrating industrial data for exchange, access, and sharing. The following activities are supported:

- integrating data which may be:
  - from different sources or different contexts,
  - described by different models, or
  - defined in different modelling languages;
- sharing data among applications through systems integration architectures;
- resolving conflict between models developed with different objectives;
- translating data between different encodings;
- translating models between different modelling languages.

The following are within the scope of ISO 18876:

- integration models;
- methods for creating, extending, and updating integration models;
- methods for creating a mapping specification to map data instances between an integration model and an application model that falls within its scope;
- encoding and decoding of data and models with different formats, such as SGML [1], XML [7], EXPRESS [3], UML [6] and ISO 10303-21 [4];
- methods for consolidating data sets from different sources and different models;
- modelling and mapping specification languages.

The following is within the scope of this part of ISO 18876:

- the architecture and an outline of the methodology.

The following are outside the scope of this part of ISO 18876:

- integration models;

— detailed specifications of the methodology;

NOTE Such specifications can be found in other parts of ISO 18876 or in other standards.

— translating data between different encodings;

— encoding and decoding of data and models with different formats;

— modelling and mapping specification languages.

2 Normative references

The following referenced documents are indispensable for the application 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 8824-1:—[1)], *Information technology — Abstract Syntax Notation One (ASN.1) — Part 1: Specification of basic notation*

ISO 10303-1:1994, *Industrial automation systems and integration — Product data representation and exchange — Part 1: Overview and fundamental principles*

3 Terms, definitions, and abbreviations

3.1 Terms and definitions

For the purposes of this document, the following terms, definitions and abbreviations apply; those taken from ISO 10303-1 are repeated below for convenience.

NOTE 1 Definitions copied verbatim from other standards are followed by a reference to the standard in brackets, such as "[ISO 10303-1]". In these cases the definition in the referenced document is normative; its repetition here is informative and in the case of any discrepancy the definition in the referenced document has precedence. An explanatory note follows definitions that have been adapted from other standards. In these cases, the definition given here is normative for the purposes of this part of ISO 18876.

3.1.1
application model (AM)
model that represents information used for some particular purpose

NOTE Some application models are also integration models (see 3.1.12).

3.1.2
class
category or division of things

[1)] To be published. (Revision of ISO/IEC 8824-1:1998)

NOTE There are a number of ways that class can be defined. This definition is intended to be as broad as possible, and is broader than that used in ISO 15926-2.

EXAMPLE Pump, power station, engineer, and fictional space vehicle are examples of classes.

3.1.3
concept
general notion or idea of something

3.1.4
data
representation of information in a formal manner suitable for communication, interpretation, or processing by human beings or computers

[ISO 10303-1]

3.1.5
data model
set of constructs that provides the definition, structure, and format of data, whether physical or abstract in the sense of being bound to some recording medium

3.1.6
derived concept
concept in an integration model that is wholly defined in terms of primitive concepts

3.1.7
encoding transformation
transformation of the way data elements are represented for computer processing

EXAMPLE Conversion of data governed by an EXPRESS schema from an ISO 10303-21 file to an XML document is an example of an encoding transformation.

3.1.8
foundation concept
primitive concept that determines the underlying world viewpoint of an integration model

NOTE There can be a number of integration models. Each will have its own modelling paradigm which is characterised by the foundation concepts that it contains.

EXAMPLE The concepts of class and individual are foundation concepts for a general integration model.

3.1.9
general concept
primitive concept that has very wide applicability, but is a specialization of some foundation concept

NOTE A concept may be considered to be a foundation concept by one community, while it is considered to be a general concept by another.

3.1.10
individual
thing that exists in space and time

NOTE This includes things that actually exist, or have existed, and things that possibly exist (past, present, and future) in space and time.

EXAMPLE The pump with serial number ABC123, Battersea Power Station, Sir Joseph Whitworth, and the Starship "Enterprise" are examples of individuals.

3.1.11
information
facts, concepts, or instructions

[ISO 10303-1]

3.1.12
integration
activity that creates, modifies, or extends an integration model

3.1.13
integration model (IM)
application model that can represent the information that is represented by two or more application models

NOTE Being an integration model is about the role one model plays with respect to one or more application models.

3.1.14
mapping
correspondence between instances of one model and instances of another model that represent the same meaning

NOTE A mapping can be uni-directional or bi-directional.

3.1.15 mapping specification
definition of the transformations necessary to take information according to one data model and represent the same information according to another data model

NOTE 1 A mapping specification can include data structure transformations, data value transformations, data encoding transformations, and terminology transformations.

NOTE 2 Mapping specifications can be procedural, or declarative, or a combination of these.

3.1.16
model
limited information representation of something suitable for some purpose

3.1.17
model context
sum of implicit concepts and constraints that limit the possible extension of a model without changing any existing declarations

NOTE 1 The model context is therefore the class of all possible extensions to a model.

NOTE 2 This term is more general than application context as defined in ISO 10303-1.

3.1.18
model scope
range of information that an application model can describe

3.1.19
primitive concept
concept in an integration model that is not wholly defined in terms of other concepts

3.1.20
specific concept
primitive concept that is a specialization of some general concept and has a limited range of applicability

EXAMPLE Car, process plant, quark, purchase order, and XML document are examples of specific concepts.

NOTE The boundary between a general concept and a specific concept may be arbitrary; some concepts may be thought of as both general concepts and specific concepts.

3.1.21
structural transformation
type of mapping specification that is a transformation to the structure of data

NOTE The change in structure could be due to the rearranging of attributes, the splitting of attributes across entity types, or the creation of new attributes.

3.1.22
terminology transformation
transformation of the term used to refer to a thing

NOTE This could be between synonyms in one language, or between different languages.

3.1.23
transformation
change of form

3.1.24
view
constrained representation of a data model

3.2 Abbreviations

For the purposes of this part of ISO 18876, the following abbreviations apply:

AM application model

NOTE In ISO 10303 the abbreviation AM is used for Application Module. An Application Module is not the same as an Application Model.

IM integration model

4 Organization of ISO 18876

ISO 18876 is divided into a number of parts.

ISO/TS 18876-1, this part, provides an overview and specifies an architecture for the integration of industrial data.

ISO/TS 18876-2 specifies methods for integrating application models and for developing and extending integration models.

NOTE Other specifications may be developed to extend the capability of ISO 18876, such as:

— models designed to integrate two or more other models;

— models designed to meet the needs of a particular application;

— mapping specifications designed to specify how a data population of one model may be migrated to another model;

— mapping specifications designed to specify how a model in one language may be migrated to another language;

— methods and languages to support the definition of models and mapping specificatios between different modelling languages;

— methods and specifications for the encoding of models and transformation between encodings;

— the specification of services and interfaces to be provided by conforming implementations.

5 Fundamental concepts and assumptions

The following fundamental concepts and assumptions apply to this Technical Specification.

5.1 Integration models

5.1.1 Principles

The three-schema architecture for data models shows that for any data model it is possible to construct views on the original model.

NOTE 1 The three schema architecture is described in ISO/TR 9007 [2].

In this Technical Specification this principle is extended to cover other types of model and modelling languages. In the integration of models this process is reversed: a model is created for which the initial models are views. A model created in this way is an integration model with respect to the initial models in that it is capable of representing information with the scope of either or both of the original models. This is illustrated in Figure 1 below.

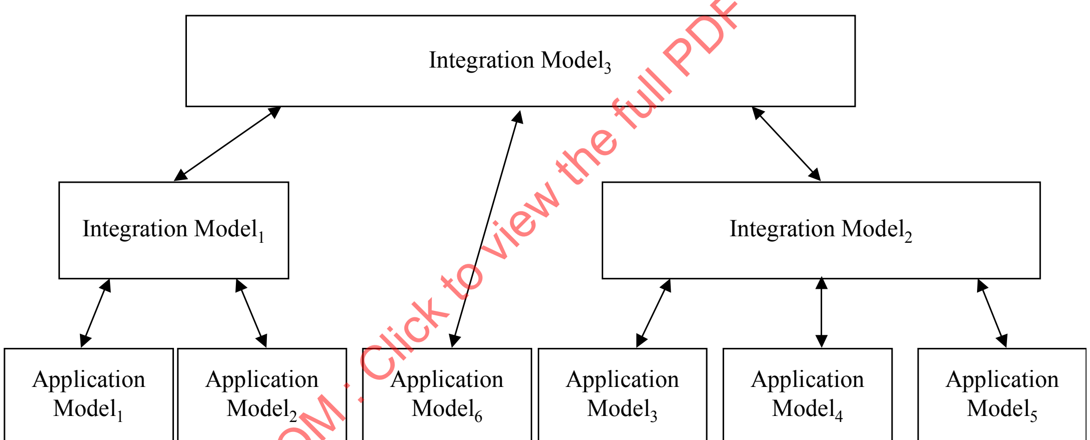

Figure 1 — Model integration

An integration model can be created if a common understanding of the application models to be integrated can be established.

NOTE 2 This has been shown by Barwise and Seligman [9].

NOTE 3 Difficulties in creating such a model point to a gap in human knowledge about the subject of the application models.

There may be more than one integration model to which an application model can be integrated, where the integration models support different ways of looking at the world. See Figure 2 below.

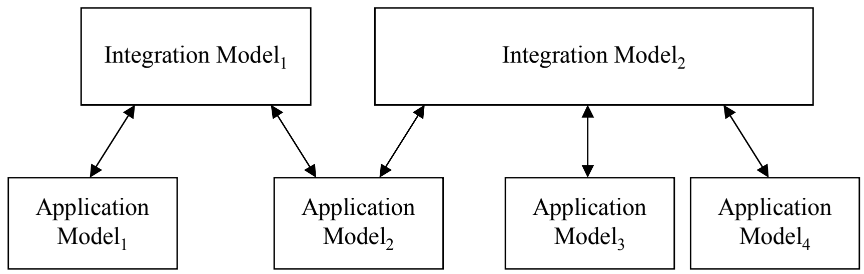

Figure 2 — Integration into more than one integration model

Models that have been created as integration models can themselves be integrated. This means that any arbitrary set of models can, in principle, be integrated at the cost of creating a new model; this is supported by the architecture defined in this Technical Specification.

NOTE 4 One possible use for the architecture defined here is the development of an integration model that is stable in the face of the integration of additional models. Here stable means that the existing integration model does not need to be changed as more models are integrated, though extensions of the integration model may be necessary.

Integration models sometimes represent concepts that are more generic than the models they integrate. This is necessarily the case when the models being integrated have conflicting constraints affecting the information that is to be represented. These constraints should be preserved by the integration process, and held in some form other than in the structure of the integration model.

NOTE 5 The constraints can be held in the mapping specification or as data within the integration model.

5.1.2 Scope and context

The scope of a model is what the model actually covers. The context of a model is the sum of constraints that limit the ability to extend the scope of the model without changing any existing declarations. These constraints include:

— activities that produce or use the data;

— organizations that produce or use the data;

— concepts that the model is “about” but that are not explicitly represented in the model;

— constraints inherent in the structure of the model.

Integration models can be created that consider two or more application models of interest – the scope and context of such an integration model can be no smaller than the combined scope and context of the application models being integrated. The relationship between such an integration model and the other application models that it integrates is shown in Figure 3 below.

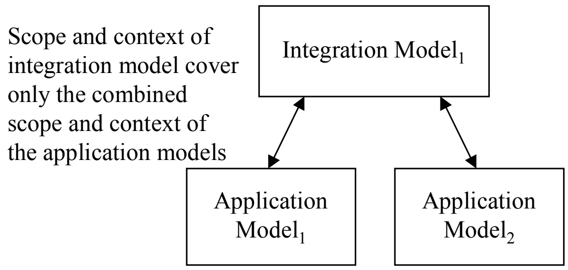

Figure 3 — A limited integration model

If requirements subsequently emerge to integrate additional application models with Application Model$_1$ and Application Model$_2$ (and hence with Integration Model$_1$) it is unlikely that the context of these further models will fit within that of Integration Model$_1$. This implies that Integration Model$_1$ cannot support the information represented by the further application models, and will itself have to be integrated with another integration model (created for this purpose or selected from candidate integration models). This is shown in Figure 4 below.

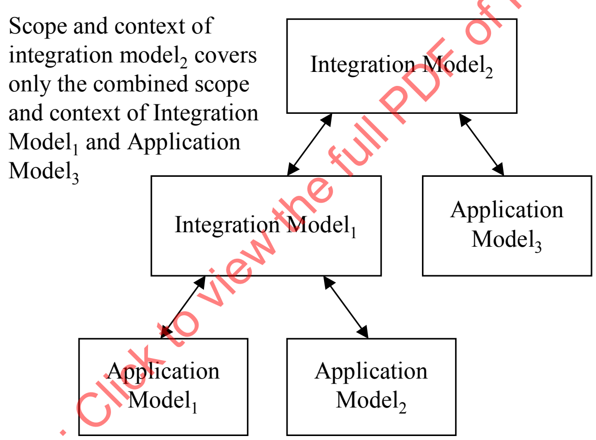

Figure 4 — Integrating an application model and a limited integration model

However, the initial integration model (IM) can be chosen to have a wide model context. This means that it can support the information needs of many different applications, even though its initial model scope is limited to that of the models that it integrates, as shown in Figure 5 below. A potential integration model can be tested by looking at the statements it makes and checking that these statements are true under all circumstances, not just in a particular context. Whilst this will make it less likely that an integration model would have to be changed, it is not possible to be certain without the requirements for the full range of application models to be eventually integrated.

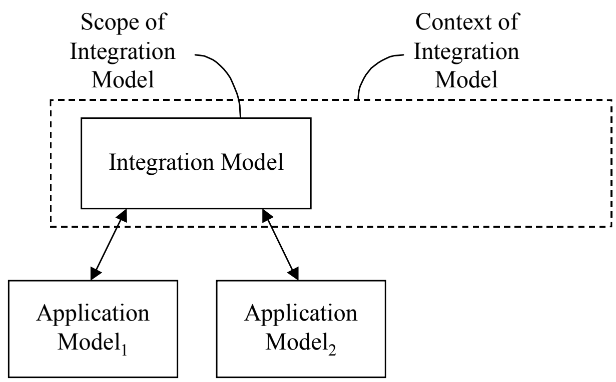

Figure 5 — Using an integration model with a wide model context

Integration of further application models can then be achieved through extension of the integration model – enlarging the model scope within the wide model context, as shown in Figure 6 below. In this case, the integration model can be extended (through the addition of constructs that represent specific and derived concepts) without changing its initial content or the mapping specifications to the initial application models AM_1 and AM_2.

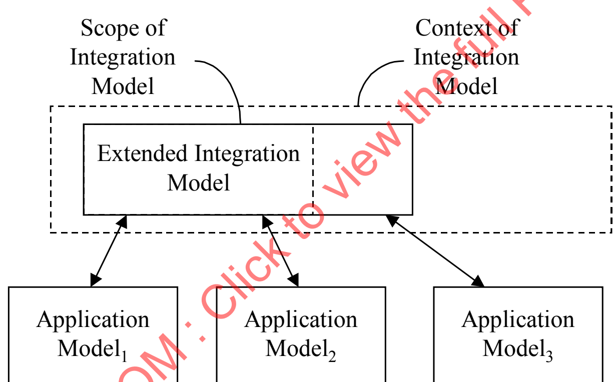

Figure 6 — Integrating additional application models

When an integration model is changed to accommodate a new application model, if the previous version of the integration model is not a subset of the new version, the previous mapping specifications to the integration model may no longer be valid. One model is a subset of another model when each declaration specified in the first model is also a declaration in the second model. The second model is then an extension to the first.

When the changes required to the integration model are not an extension, then a new integration model should be created together with a mapping specification to the original integration model to preserve any mapping specifications that may have been created to it.

A consequence of this is that integration models should be constructed such that integrating further application models results in an extension to the model. This is true for integration models that have a broad context. These have the following characteristics.

— Each object is modelled in terms of its underlying nature, based on a specific paradigm or view of the world being modelled.

NOTE 1 Modelling something in terms of its underlying nature means that the class that is selected is one that it is a member of for the whole of its life, not more and not less. Sometimes things are modelled in terms of a role played in a particular context, such as employee, however that which is an employee is not an employee for the whole of its life, but a person.

EXAMPLE A payrole system might have to be extended from dealing with employees to dealing with pensioners as well, who are not employees. The model would then have to take into account that employee and pensioner were different states of a person.

— The names chosen for the model constructs reflect the underlying nature of the subject of the model.

NOTE 2 When names are not chosen appropriately, then users of the model can be mislead as to what is represented.

— Definitions of model constructs are clear and unambiguous.

NOTE 3 Unclear or ambiguous definitions can lead to the model being misused.

— Only those constraints that are applicable for the whole model context and model scope are represented in the structure of the model.

NOTE 4 When constraints are incorporated into the structure of a model, then when an extension is required where that constraint no longer applies, then the model will have to be changed. This is undesirable.

— Classes are represented as nodes in a subclass/superclass (specialization/generalization) hierarchy that is rooted in a single class.

NOTE 5 The position in a subtype/supertype hierarchy is a key part of the formal definition of a concept. It also supports appropriate inheritance of characteristics and behaviour.

— The model represents and manages history and change.

NOTE 6 Managing change in a model can be complex, and there are many ways in which it can be done with varying effectiveness. For some purposes it will be necessary, so it is beneficial to decide early how this will be addressed.

— The modelling language chosen to represent the integration model is used consistently.

NOTE 7 Inconsistency in the use of a language introduces uncertainty as to how a processor should interpret constructs.

5.1.3 Integration model concepts

The concepts represented by an integration model can be classified as primitive concepts (see 3.1.19), and derived concepts (see 3.1.6). Primitive concepts are the building blocks for the definition of other concepts, and can be further classified as follows:

— foundation concepts (see 3.1.8);

— general concepts (see 3.1.9);

— specific concepts (see 3.1.20).

This is represented in Figure 7 below.

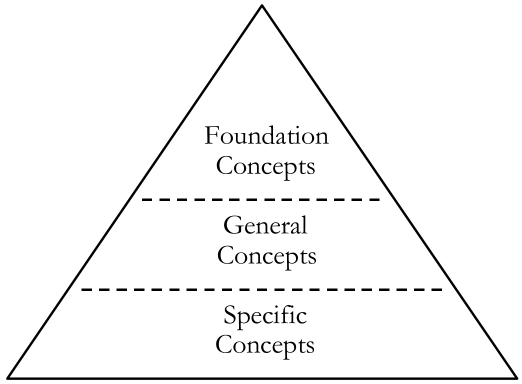

Figure 7 — Primitive Concepts

Specific concepts are dependent on general concepts that are dependent on foundation concepts, since all the lower concepts rely on the existence of one or more higher level concepts.

EXAMPLE At the top level, an integration model should have foundation concepts like classification, connection and composition. General concepts might include those of physics. Discipline specific concepts would be those that are limited in their range of application.

5.1.4 A full integration model

A full integration model, as shown in Figure 8, is more than just primitive concepts; it includes derived concepts – useful and valid combinations of primitive concepts. Only derived concepts that are of interest need be recorded.

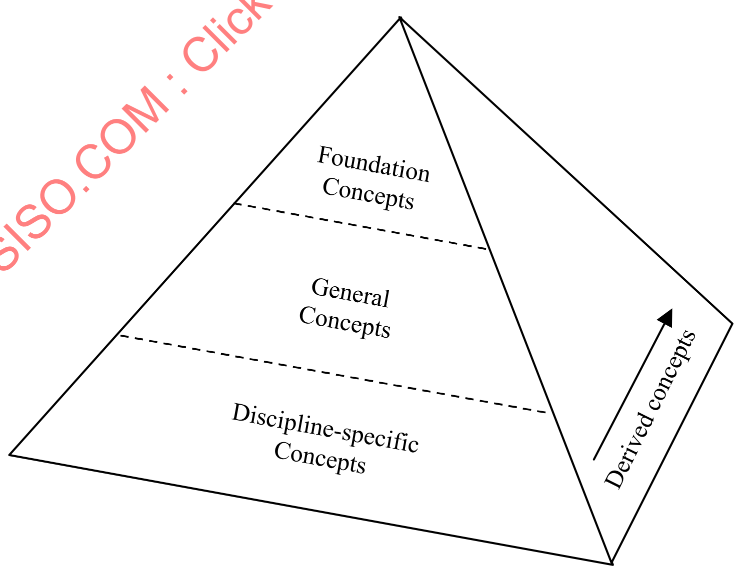

Figure 8 — A full integration model

This architecture does not require that primitive concepts remain primitive forever. If a concept that is initially thought to be a primitive concept turns out not to be, then the concepts it is derived from can be identified/

added and the derivation added, so that it becomes a derived concept away from the front face of the pyramid. This allows flexibility to reflect an improved knowledge of the world, rather than reflecting knowledge of the world that is constrained by a modeller's knowledge at a point in time. Therefore, an integration model will need to be maintained and extended, and a mechanism for maintenance and extension will be necessary.

5.2 Mapping specifications

Mapping specifications specify the transformations that determine the mapping of instances of one model to instances of another model. Mapping specifications are used in two ways, as follows.

a) The mapping specification can describe the mapping transformations between a subset of an integration model and a pre-existing application model that governs data that is separate from that governed by the integration model. In this case the mapping specification defines the transformations on a set of instances of one model to create an equivalent set of instances of the other.

b) The mapping specification can describe the mapping transformations between a subset of an integration model and an application model that is used as a view. In this case the mapping specification describes how instances in the application view are created from instances in the integration model.

The following fundamental concepts apply to the mapping specifications that are created during the integration process.

— New primitive concepts are not introduced in the mapping specification; mapping specifications are limited to specifying application model concepts in terms of integration model concepts, transformations of structure, terminology, and encoding.

— Mapping specifications are either declarative or procedural.

  — A declarative mapping specification identifies a pattern of instances of one model that corresponds to a pattern of instances in another model, hence the mapping specification is bi-directional.

  EXAMPLE 1 An informal declarative mapping specification is: "for each thing in model A that is classified as "red" and classified as a "car", there is in model B an object that is an instance of "red car".

  EXAMPLE 2 The mapping specification in the mapping table of ISO 10303-214 is a declarative mapping.

  — A procedural mapping specification defines the process by which an instance of the target model can be created from instance(s) of the source model. For procedural mapping specifications, a mapping specification in each direction is required.

  EXAMPLE 3 An informal procedural mapping specification in one direction is: "for each thing in Model A that is classified as "red car" create in model B an object that is an instance of the class "car" and an instance of the class "red".

  EXAMPLE 4 A mapping specification created with EXPRESS-X using the map construct is a procedural mapping specification.

6 Overview of the model integration process

The model integration process takes a number of application models and an integration model. It ensures that all the concepts of the application models are represented in the integration model, and develops a mapping specification between the integration model and each of the application models.

There are three possible cases for the integration process:

— the integration model and the application models exist before the integration process starts;

— the application models to be integrated exist before the integration process starts, but not the integration model;

— the integration model exists before the integration process starts, but the application model needs to be developed from some statement of requirements.

The first of these covers all the elements of the other two, and is described here in outline for one application model. The other two processes are described in more detail in ISO/TS 18876-2.

Integrating an application model with an integration model is illustrated in Figure 9 below. The goal of this integration process is to allow the same information that is represented in the application models to be represented in the integration model without loss of meaning, and to allow transformations between these representations. The result of the integration process is a mapping specification between the application model and a part of the integration model. In order to define this mapping specification it may be necessary to extend the integration model so that it precisely represents the concepts found in the application model.

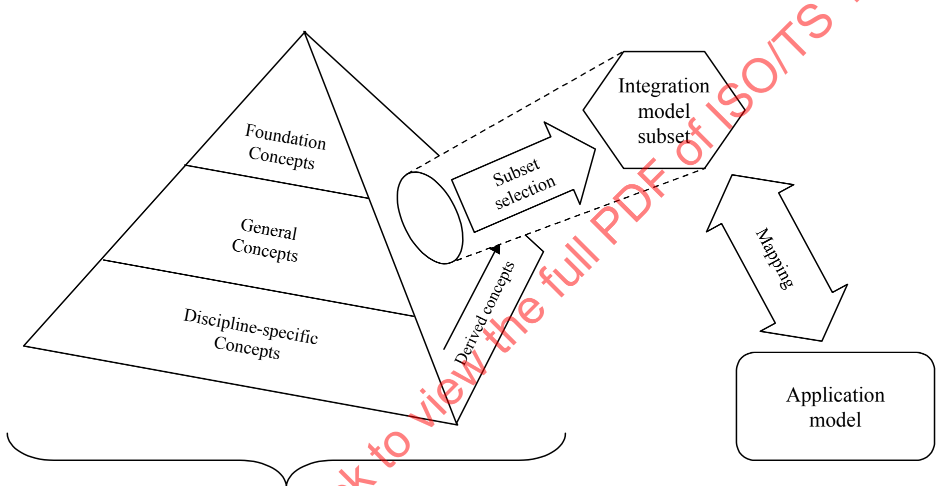

Figure 9 — Integrating application models with an integration model

For a model, being an integration model derives from the role that it plays with respect to other models. The fundamental characteristic of an integration model is that it integrates two or more application models.

The process of integrating an application model with an integration model is divided into a number of steps, as follows:

— analyse the application models and identify the equivalent concept of the integration model, including any constraints that apply (see Figure 10);

NOTE 1 Most application models have a model context within which the model has to be understood, but which is not explicit in the model itself. Usually it will be inappropriate to add this information explicitly to the application model. In this case these requirements should be captured in the mapping specification as part of the integration process.

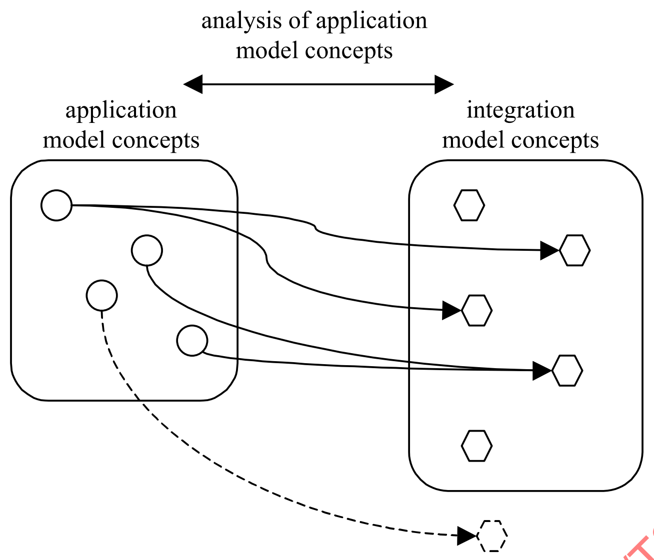

Figure 10 — Analyzing the application models

NOTE 2 Different symbols are used for the objects in the application model and the integration model to emphasise that these are different information objects, even if they refer to the same real world objects.

— if necessary, extend the integration model so that it includes all the concepts found in the application models (see Figure 11);

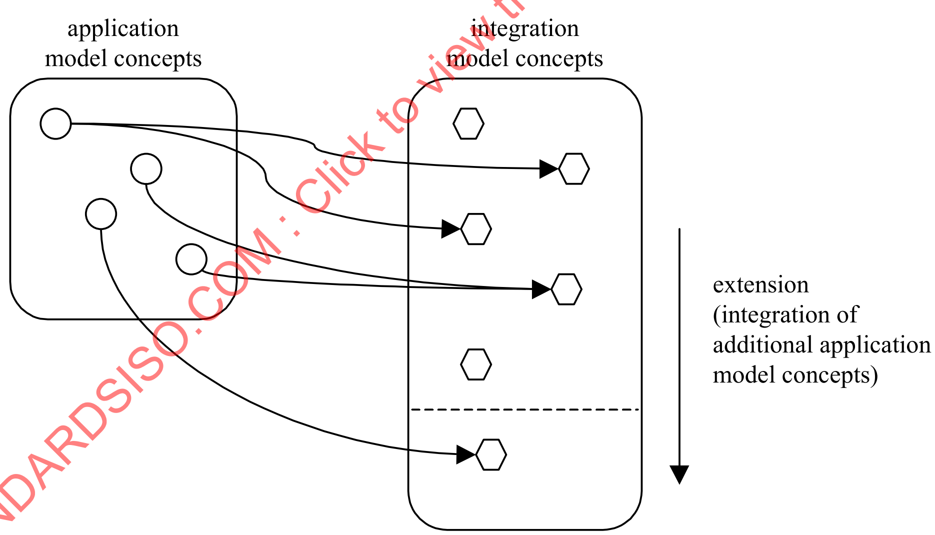

Figure 11 — Adding any missing concepts to the integration model

— identify the part of the integration model that represents the concepts in each application model (see Figure 12);

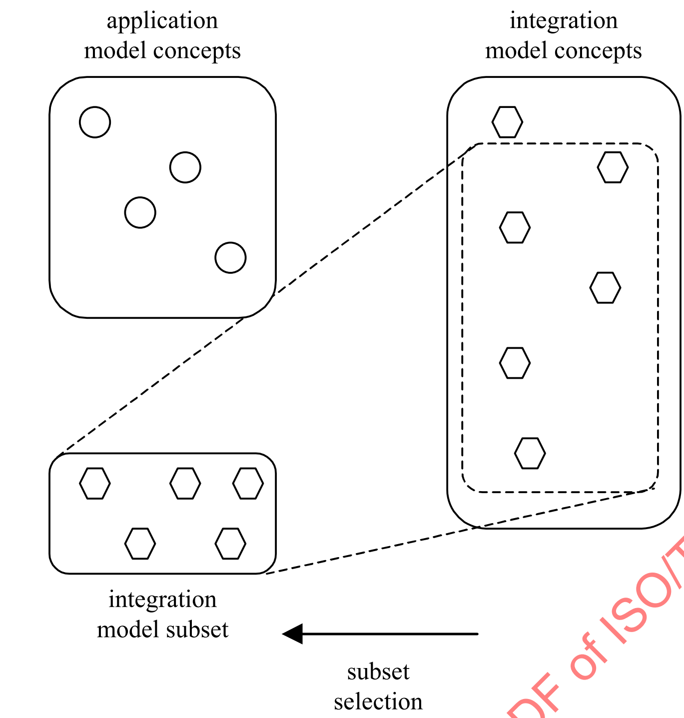

Figure 12 — Identifying the subset of the integration model

— create the mapping specification in each direction between each application model and the appropriate subset of the integration model (see Figure 13);

NOTE 3 The identification of an integration model subset is a by product of the integration process.

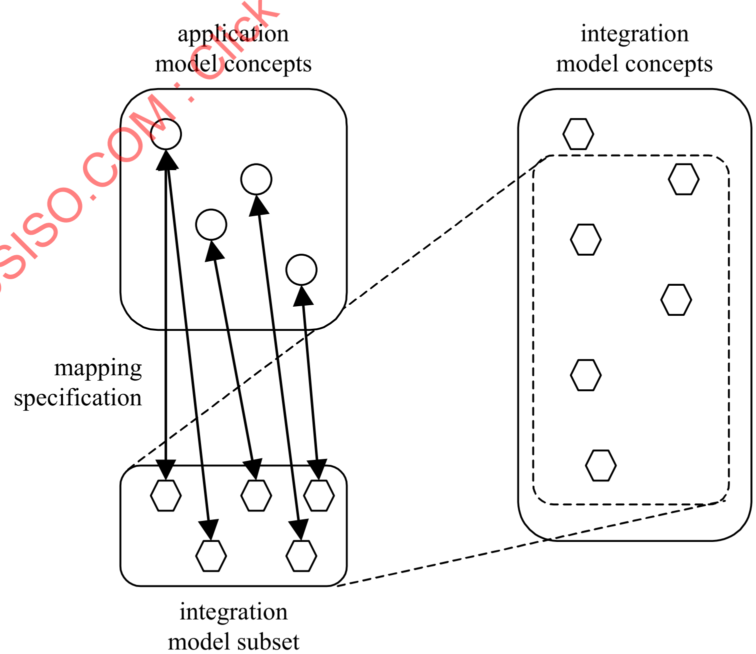

Figure 13 — Creating the mapping specification between the integration model subset and the application model

— specify any structural transformations, terminology transformations, or encoding transformations that apply within the mapping specification;

— specify any transformations that are necessary between model representations;

EXAMPLE 1 If an application model is specified in the XML Schema definition language and the integration model to which it is mapped is specifed in EXPRESS [3], a transformation between these languages will be necessary to map between different representations of the same concepts.

Most application models have a model context within which the model has to be understood, but which is not explicitly represented in the application model itself. Mapping successfully in both directions requires that both the explicit model and its model context be mapped into the integration model. The two way mapping specification is successful if all instances of the application model can be mapped to the integration model, and if the reverse mapping specification allows all and only those instances from the integration model that are appropriate in the context of the application model to be mapped in the reverse direction.

EXAMPLE 2 In a salary payment system may be an entity data type called **employee.** However, it is often implicit that each person represented by instances of this entity data type is an employee of the company that operates the system and legally eligible for employment under company and governmental policies.

7 Integration architecture components

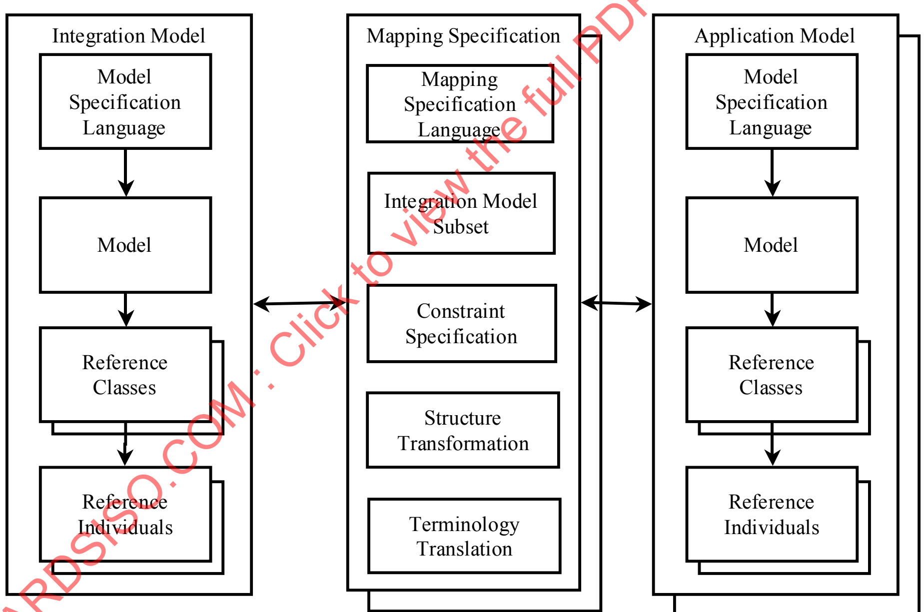

Figure 14 — Integration architecture components

Figure 14 shows the elements of Figure 9 for the case where an entity-attribute-relationship modelling language is used together with reference data. For each of the named boxes, standard instances can be created, as follows:

NOTE 1 Object oriented models are considered in this International Standard as entity attribute relationship models.

— integration models;

NOTE 2 This may be a single model with multiple levels of abstraction, using a suitable logic based language, such as KIF [16], or a layered model, using an entity-relationship language such as EXPRESS [3] with a data model and reference data libraries. Figure 14 illustrates the use of a model to define the structure of a reference data library that can hold reference classes and reference individuals.

