INTERNATIONAL STANDARD

ISO/IEC 12785-1

First edition
2009-12-15

Information technology — Learning, education, and training — Content packaging

Part 1: Information model

Technologies de l'information — Apprentissage, éducation et formation — Paquetage du contenu

Partie 1: Modèle de l'information

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

Reference number
ISO/IEC 12785-1:2009(E)

© ISO/IEC 2009

PDF disclaimer

This PDF file may contain embedded typefaces. In accordance with Adobe's licensing policy, this file may be printed or viewed but shall not be edited unless the typefaces which are embedded are licensed to and installed on the computer performing the editing. In downloading this file, parties accept therein the responsibility of not infringing Adobe's licensing policy. The ISO Central Secretariat accepts no liability in this area.

Adobe is a trademark of Adobe Systems Incorporated.

Details of the software products used to create this PDF file can be found in the General Info relative to the file; the PDF-creation parameters were optimized for printing. Every care has been taken to ensure that the file is suitable for use by ISO member bodies. In the unlikely event that a problem relating to it is found, please inform the Central Secretariat at the address given below.

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

COPYRIGHT PROTECTED DOCUMENT

© ISO/IEC 2009

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either ISO at the address below or ISO's member body in the country of the requester.

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

 © ISO/IEC 2009 – All rights reserved

Contents

Page

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

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.

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

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

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.

ISO/IEC 12785-1 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 36, *Information technology for learning, education and training*.

ISO/IEC 12785 consists of the following parts, under the general title *Information technology — Learning, education and training — Content packaging*:

— *Part 1: Information model*

The Extensible Markup Language (XML) Schema binding for Content Packaging Information Model and associated namespace identifiers will be declared in Part 2. Practices related to the interpretation and implementation of the Information Model will be addressed in Part 3.

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

0 Introduction

0.1 Purpose and overview

ISO/IEC 12785 is derived from the IMS Global Learning Consortium (IMS GLC) Content Packaging version 1.2 Specification. IMS Content Packaging is probably the most widely used specification in support of learning technology around the world. IMS Content Packaging has been an integral foundation of Sharable Content Object Reference Model (SCORM) from its inception to the current version. But, most importantly, IMS Content Packaging has also been used widely outside of SCORM on a standalone basis. IMS Content Packaging is also used in many other high profile educational uses, such as archiving for MIT OpenCourseWare, distributing content packages that exclude runtime and metadata for the Learning Federation of Australia, and nationwide e-learning services for the Cyber Home Learning System in Korea.

The IMS Content Packaging Information Model that is the source and base specification for this part of ISO/IEC 12785 describes data structures that can be used to exchange data between systems that wish to import, export, aggregate, and disaggregate packages of learning, education and training (LET) content.

The IMS Content Packaging specification was initially conceived for the packaging of instructional content. The specification supports the description of content associated with a given learning activity, location of the content, and how these pieces of content can be organized for best instructional effect. As a result of wide adoption of the specification, millions of IMS content packages of instructional content are used in a variety of software applications.

Adopters of IMS Content Packaging have extended its use beyond just the packaging of instructional content. IMS Content Packaging is now referenced by other IMS Specifications to package and exchange other types of data.

Requests for major functional additions were not included in the IMS Content Packaging version 1.1.x series and were accumulated as practice matured around implementing IMS Content Packaging. Evaluation of these requests in 2006, combined with feedback from the wider adopter community, led to the decision to make a significant update and definitive release for this specification as an International Standard series.

The new functionality and clarifications incorporated in this Content Packaging specification are as follows.

a) The meanings of terms used within the specification have been clarified.

b) The use of (sub)manifests, now termed child-manifests, has been clarified and enhanced:

   1) Interpretation of an item pointing to a child-manifest has been clarified.

   2) New functionality allowing components of child-manifests to be precisely referenced and interpreted has been added.

   3) Support for external child-manifests has been added.

c) Support for external referenced metadata files has been added.

d) All internal vocabularies have been removed and are now maintained through the IMS vocabularies registration process (see http://www.imsglobal.org/vdex/index.html).

e) A new resource type of “stand-alone resource” has been added that allows another package to be used as a piece of LET content.

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

f) The syntax and usage of the Base, Parameter, 'IsVisible', and 'Href' Information Model classes has been clarified.

g) Support for variant resources has been added. This includes support for alternative resources for accessible LET content.

h) Support for Organization and Item titles in multiple languages has been added.

i) Support for interchange packages that contain only content and interchange packages that have no local content files has been clarified.

0.2 Compatibility

This part of ISO/IEC 12785 arises in an active implementation environment of ever increasing adoption of IMS Content Packaging. A primary goal of this part of ISO/IEC 12785 is to enable future growth while regularizing current practice. To that end, the following definition of backwards compatibility has guided the development of this Content Packaging Information Model:

a) From the perspective of the IMS Content Packaging Information Model, the IMS Content Packaging Information Model v1.1.4 is a proper subset of this Content Packaging Information Model.

b) The semantics of the Content Packaging Information Model components persists between versions, except where necessary to ensure disambiguation.

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

Information technology — Learning, education, and training — Content packaging

Part 1: Information model

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

1 Scope

This part of ISO/IEC 12785 defines the data structures that can be used to exchange language, education and training (LET) content among systems that wish to import, export, aggregate, and disaggregate packages of LET content.

It illustrates the conceptual structure of the Content Packaging Information Model and defines the structural relationships, data-type, value-space, and number of occurrences permitted for each kind of information object.

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 639-2:1998, *Codes for the representation of names of languages — Part 2: Alpha-3 code*

ISO 3166-1:1997, *Codes for the representation of names of countries and their subdivisions — Part 1: Country codes*

ISO/IEC 10646:2003, *Information technology — Universal Multiple-Octet Coded Character Set (UCS)*

IEEE 1484.12.1-2002, *Draft Standard for Learning Object Metadata*

IETF RFC 1951 (1996), *DEFLATE Compressed Data Format Specification version 1.3*

IETF RFC 2119 (1997), *Keywords for use in RFCs to Indicate Requirement Levels*

IETF RFC 2234 (1997), *Augmented BNF for Syntax Specifications: ABNF*

IETF RFC 2732 (1999), *Format for Literal IPv6 Addresses in URL's*

IETF RFC 3986 (2005), *Uniform Resource Identifier (URI): Generic Syntax*

© ISO/IEC 2009 – All rights reserved

3 Terms and definitions

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

3.1
child manifest
complete, subordinate manifest contained in parent manifest

NOTE 1 A manifest can contain one or more child manifests.

NOTE 2 A manifest can include a reference to a child manifest that is external to the interchange package.

NOTE 3 A child manifest describes a complete logical package that is part of the larger logical package defined by its parent manifest.

NOTE 4 A child manifest can be local or remote.

cf. interchange package, local, logical package, manifest, remote

3.2
content file
computer file(s) that embodies the LET content described by the manifest

NOTE 1 Content files can be local or remote.

NOTE 2 Content will often contain more than one content file. For example, web content is often instantiated by HTML, JPEG, and CSS files. Content files can be local or remote.

cf. local, logical package, remote

3.3
content organization
organization
logical relationships, such as a hierarchical tree, among units of LET content

NOTE 1 More than one logical organization can be described in a manifest.

NOTE 2 An organization is bound to files through the relationship among item components and their referenced resource components of a manifest.

NOTE 3 The rules governing the structuring and ordering of a hierarchical tree need to be specified.

cf. resource

3.4
control file
single computer file that governs the binding of the Content Packaging Information Model (CPIM) to make it suitable for machine processing

NOTE A software component can refer to a control file when assessing the validity of a bound instance of the information model or to guide the creation of a bound instance of the information model. For example, a file containing an XML schema can be used as a control file for an XML binding of a manifest.

3.5
interchange package
set of usable (reusable) LET content that is exchanged among computing systems used for information technology for learning, education and training (ITLET) purposes

NOTE An interchange package can be instantiated in a single compressed binary file (package interchange file) or as a collection of files on portable media (e.g., CD, DVD, USB memory device).

cf. logical package, package interchange file

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

3.6
launchable URI
representation of a Universal Resource Locator (URL) that may be included in a resource description and that is used to locate and access the content described by the resource

NOTE The launchable Uniform Resource Identifier (URI) is not meant to be resolved by a package reader.

cf. interchange package, package reader

3.7
LET content
content
logical unit to represent usable (and reusable) information contained in or related to learning, education, and training (LET) data in a formalized manner suitable for interpretation by human means

EXAMPLE In the instructional context, content can be web-based instructional materials.

NOTE 1 A logical unit of usable (and reusable) information can be described by a logical package.

NOTE 2 A logical package can contain one or more units of LET content.

cf. logical package

3.8
local
⟨component of the logical package⟩ contained within the interchange package

cf. logical package, interchange package

3.9
logical package
representation of one or more units of usable (and reusable) LET content

NOTE A logical package encompasses the full set of components described by the manifest and its child manifests, including the local components and the remote components included by reference.

cf. child manifest, local, manifest, remote

3.10
manifest
description of a complete instance of a logical package

NOTE 1 A manifest describes resources in the logical package, their organization and the locations of the associated content and control files.

NOTE 2 A manifest can contain references to components that are local or remote.

cf. local, logical package, remote

3.11
manifest document
manifest with contents that are structured according to various binding technologies

3.12
metadata
⟨content packaging⟩ descriptive information about logical packages, logical organizations, content, and files

NOTE 1 Metadata can be assigned to any of the core structures within the logical package, including the manifest.

NOTE 2 Any binding of a metadata object is permitted. Each object of metadata can be local or remote.

cf. local, logical package, remote

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

3.13
namespace
XML namespace identified by a URI reference

NOTE Namespace in Content Packaging follows W3C recommendation *Namespaces in XML 1.0 (Second Edition)*.

3.14
package
unit of usable (and reusable) LET content

NOTE 1 This can be part of a learning course that has instructional relevance outside of a LET content aggregation and can be delivered independently, as an entire learning course or as a collection of learning courses.

NOTE 2 A package is able to stand-alone; that is, it contains all the information needed to use the contents for learning, education, and training when it has been unpacked.

cf. interchange package, logical package

3.15
package interchange file
PIF
instantiation of an interchange package which is physically encapsulated as a compressed binary file conforming to IETF RFC 1951 (1996)

NOTE 1 An interchange package can be instantiated in a format other than a package interchange file (PIF).

NOTE 2 Usually the representation (binding) is expressed in XML.

EXAMPLE An interchange package can be instantiated as a collection of files on removable media, e.g. CD, DVD, USB memory device, or compressed using another format such as .zip, .tar, .jar, .cab.

cf. interchange package

3.16
package reader
software that reads a manifest and verifies the contents of an interchange package

NOTE A package reader can also process a logical package (retrieve and store information referenced by the manifest, unpack local files from a PIF, retrieve or log addresses of remote files, etc.) or delegate that task to another software typed process.

cf. interchange package, logical package

3.17
package writer
software that creates or modifies an instance of an interchange package and assembles content file(s) and other files declared local to the interchange package and writes them to the targeted interchange package binding, or delegates those tasks to another software typed process

cf. interchange package

3.18
referenced manifest
manifest or a component of a manifest that is referenced from within another manifest

NOTE 1 A manifest can reference an organization in another manifest.

NOTE 2 The manifest containing the component being referenced is called a referenced-manifest.

NOTE 3 A manifest can contain references to components that are local or remote.

cf. local, remote

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

3.19
relative reference
expression of a URI reference relative to the namespace of another hierarchical URI

NOTE 1 See IETF RFC 3986 (2005).

NOTE 2 The extension and the context are combined to create a target URI.

EXAMPLE A relative/path/to/resource.txt is a relative reference that is interpreted in terms of a context to be resolved. [The algorithm for resolving relative references in terms of contexts is defined in Section 5 of IETF RFC 3986 (2005).]

3.20
remote
⟨component of the logical package⟩ located outside the interchange package

cf. interchange package, logical package

3.21
resource
description of a collection of content files used by the logical package

NOTE 1 The description can include metadata about the collection of LET content and resource files, a description of each of the files, and information about variant forms of the collection of files.

cf. logical package

NOTE 2 A resource is often used to describe a unit of LET content. When this is the case, the resource can contain a launchable URI for the LET content.

NOTE 3 The files described by a resource can be local or remote.

cf. launchable URI, local, remote

3.22
stand-alone resource
resource that allows a manifest to declare a relationship to another manifest in such a way that the related manifests are processed as separate but related data sets

NOTE 1 The related manifests can be contained within a single content package or accessible as an external URI addressable resource.

NOTE 2 Each manifest represents a stand-alone learning resource which can be aggregated with other learning resources to create arbitrarily rich learning experiences.

3.23
uniform resource identifier
URI
compact sequence of characters that identifies an abstract or physical resource

NOTE See IETF RFC 3986 (2005).

3.24
uniform resource locator
URL
subset of URIs that provide a means of locating a resource by describing its primary access mechanism

NOTE See (IETF RFC 3986 (2005).

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

3.25
variant
container for referencing and describing somewhat different LET content

NOTE 1 A particular resource can have variants of different formats and for different purposes.

NOTE 2 The listing of variants within a resource identifies alternative collections of files for the resource.

NOTE 3 Metadata is used to describe the intended uses of the original resource and the intended uses of the variants.

EXAMPLES Lingual variants, visual or auditory variants, remediation variants, and platform delivery variants.

cf. metadata, resource

4 Acronyms and abbreviations

ABNF[1)]	Augmented Backus-Naur Form
ADL	Advanced Distributed Learning
CPCM	Content Packaging Conceptual Model
CPIM	Content Packaging Information Model
IETF	Internet Engineering Task Force
LET	Learning, Education, and Training
MPEG	Moving Picture Experts Group
MPEG-21	ISO/IEC 21000 (all parts)
PIF	Packaging Interchange File
PIM	Platform Independent Model
RFC	Request for Comments
SCORM	Sharable Content Object Reference Model
UML	Unified Modeling Language
URI	Uniform Resource Identifier (IETF RFC 3986)
URL	Uniform Resource Locator (IETF RFC 3986)
URN	Uniform Resource Name (IETF RFC 3986)
W3C	World Wide Web Consortium
XML	Extensible Markup Language (W3C XML)

1) “Augmented BNF for Syntax Specifications: ABNF,” D. Crocker (Ed.), P. Overell, Internet Engineering Task Force, Network Working Group, Request for Comments: 2234, November 1997.

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

5 The Content Packaging conceptual model (CPCM)

Content package is a unit of usable (and reusable) LET content as defined within the Content Package Information Model. A content package consists of a logical description of the package (the Manifest) and the physical resources.

Figure 1 is a conceptual diagram that illustrates the structure of the Content Packaging Information Model (CPIM).

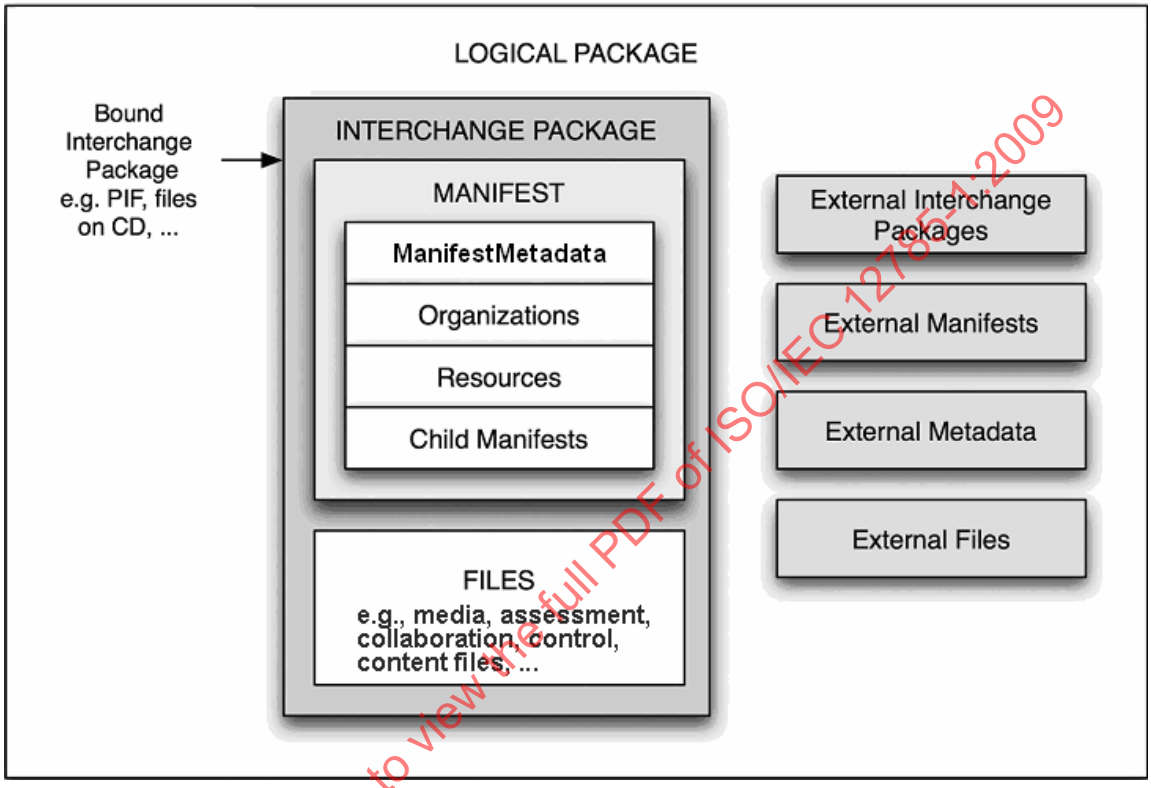


Figure 1 — Illustrative representation of the Content Packaging conceptual model (CPCM).

These core structural components are:

— **Logical package** – a representation of one or more units of usable (and reusable) LET content. The logical package encompasses the full set of components described by the manifest, including the local components and the remote components included by reference.

— **Interchange package** – the set of LET content related components that are to be exchanged among systems. An interchange package shall include a manifest and may include content files and control files. All of the files included in an interchange package shall be described in the manifest or a child manifest.

— **Manifest** – the component that describes a complete instance of a logical package. A manifest may contain references to components that are local or remote.

— **Organizations** – logical relationships among the units of LET content. More than one logical organization may be described.

— **Resources** – the description of LET content and resource files used by the logical package. The files may be local or remote.

— **Child manifests** – complete and subordinate manifests that are contained within or referenced from another manifest. This each describes a complete logical package that is part of the larger logical package. The child-manifests may be local or remote.

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

— **Files** – computer files that embody the LET content described by the logical package or govern the binding of other files to make them suitable for machine processing. Content files may be local or remote.

— **Metadata (in content packaging)** – descriptive information about content packages, logical organizations, content, or files. In this diagram the metadata box represents the set of local and/or remote metadata objects that are contained within the logical package. Any binding of a metadata object is permitted. Metadata may be assigned to any of the core structures within the logical package including the manifest.

The information model defines these core structural components for describing and organizing LET content for exchange. An important underpinning is support for extensibility. Implementers of this specification may use these extension mechanisms to define new vocabulary terms and structures.

6 Class description and relationship requirements

An informative overview of the entire Content Packaging Information Model (CPIM) is provided as a Platform Independent Model (PIM) expressed in UML constructs. All UML diagrams expressed as “Platform Independent Model” are non-normative. Normative tables defining the classes in this Information Model follow the informative UML diagrams.

A full definition of the UML Profile and the terms used in the normative tabular descriptions in this document to describe the PIM can be found in IMS UML Profile guideline (bibliography item [9]).

In the tables in this subclause the character sequence “n / a” is used to mark a field “not applicable.” Any field so marked is not relevant to the class being defined. Features so marked shall be ignored when binding a class defined by this Information Model

Augmented Backus-Naur form (ABNF) is used to define certain rules in the tables. Augmented Backus-Naur form is defined in IETF RFC 2234.

6.1 Key terms and concepts

Classes in this information model are of four abstract types in Content Packaging Information Model. These abstractions are bound to specific data structures for machine processing in the Content Packaging Part 2 - XML Binding. The four abstract class types are:

— **container class:** A container class may be a parent of one or more child classes.

— **value class:** A value class shall not be a parent. That is, it shall not be a composite of characteristic, container, value, or unspecified class types. A value class shall always be a child of a container class and shall have semantic value within the scope of its parent class’s semantic value.

— **characteristic class:** A characteristic class shall not be a parent. A characteristic class shall declare a trait or value that is an intrinsic feature or part of a container class. A characteristic class is tightly coupled to the container class it modifies or one of which facets it describes.

— **unspecified class:** An unspecified class may be a parent. An unspecified class serves as an extension point for this Information Model.

Where the numbers of elements are greater than one, the importance of the ordering of siblings is also indicated by appending either "ordered" or "unordered". "ordered" specifies a sequence of siblings as listed, "unordered" specifies a collection or bag of siblings. Order is not important.

Table 1 lists the class descriptors used to describe the abstract classes and definitions of the descriptors.

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

Table 1 — Class descriptors.

Descriptor	Definition
Class name	The name given to the class being described.
Class type	The abstract class type of this class.
Data type	For value and characteristic classes, the allowed structure for valid values for the class. Valid data types are: • **URI**: Any syntactically valid instance of a URI as defined in IETF RFC 3986. Note: Many of the foundational Specifications, Standards, and Recommendations referred to by this Information Model use IETF RFC 3986 and IETF RFC 2732 as the definitions of URI. These are made obsolete by IETF RFC 3986, but many of the foundational documents have not been updated to reference IETF RFC 3986. • **LUID**: An identifier that is locally unique within a manifest This will be based upon the String data-type that has a constrained value-space. • **LUIDref**: A reference to a LUID that has been defined elsewhere within a manifest. The value of the LUID and the LUIDref(s) that reference it shall be the same. • **Boolean**: The primitive, two-valued data type that uses the keywords “true” and “false” to indicate the logical state of an object. • **String**: A sequence of printable characters. • **Unspecified**: The data type is not known or is not important.
Value space	The range of valid values for this class. If the value space is unspecified, it is not known or is not important.
Multiplicity	A property of a class indicating the number of times it may be used or appear in a given parent context. The values of this property are expressed as a range or shorthand for a range using this notation: • . ‘**0..1**’ [optional; restricted] • . ‘**0..unbounded**’ [optional; unrestricted] • . ‘**1..1**’ [mandatory; restricted] • . ‘**1..unbounded**’ [mandatory; unrestricted] Multiplicities may also appear in short-hand notation in the UML models. The short-hand equivalents shall be (exclusive of bracketed comments): • . ‘*****’ [optional; unrestricted] • . ‘**1**’ [mandatory; restricted] • . ‘**1..***’ [mandatory; unrestricted] Where multiplicity is greater than one, the importance of the ordering of siblings is also indicated by appending either "ordered" or "unordered".
Characteristic classes	Lists the characteristic classes associated with this class in the form “{“ characteristic *”,“ characteristic ”}”. One or more characteristics may be expressed within curly braces. Each characteristic shall be separated by a comma. Where more than one characteristic is listed, the importance of the ordering of siblings is also indicated by appending either "ordered" or "unordered".
Parents	Lists classes that may be parents of this class.
Children	Lists the possible child classes of this class in the form “[” child *“,” child “]”. One or more child classes may be expressed within square brackets. Each child class shall be separated by a comma. Where more than one child is listed, the importance of the ordering of siblings is also indicated by appending either "ordered" or "unordered".
Description	Contains descriptions relating to the class and its values space.

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

6.2 Platform-independent model of the Content Packaging Information Model (CPIM)

Figure 2 is a visual summary of the structure of the Content Packaging Information Model (CPIM)

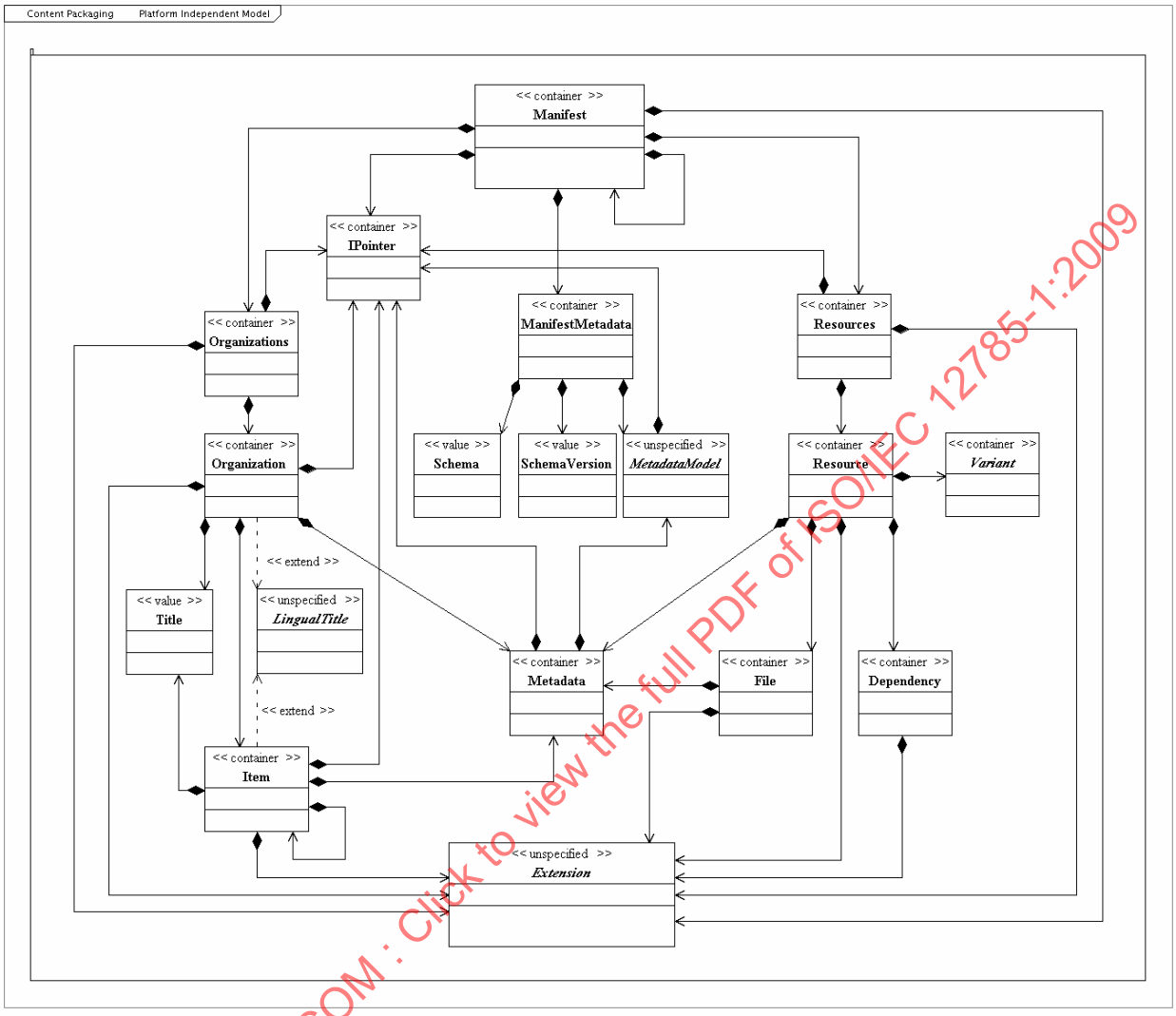


Figure 2 — Structural view of the Content Packaging PIM (overview).

The structural view in Figure 2 is focused on the relationships among the primary classes. It does not contain the details of the multiplicities, ordering, or attributes for the classes.

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

6.3 InterchangePackage class

The interchange package is identical to the logical package when all components in the logical package are local to the interchange package. The criteria for determining the components of a logical package that should be included in the interchange package is out of scope for this specification and shall be determined by the implementer. Influencing factors may include the size of the instantiation of the interchange package, architectural reasons, and business policies.

Table 2 defines the InterchangePackage class in the Content Packaging Information Model (CPIM).

© ISO/IEC 2009 – All rights reserved

Table 2 — InterchangePackage class definition.

Descriptor	Definition
Class name	InterchangePackage
Class type	container
Data type	n / a
Value space	n / a
Multiplicity	1..1
Characteristic classes	n / a
Parents	none
Children	[Manifest]
Description	An InterchangePackage object is the subset of a logical package that is exchanged between systems. An InterchangePackage shall meet the following conditions: a) The InterchangePackage shall include a Manifest object and may include content files and control files. b) Any file described in the Manifest using a URI that resolves to a location within the InterchangePackage shall be included in the InterchangePackage. c) All files included in an InterchangePackage shall be described in the Manifest. d) Files contained in the InterchangePackage may contain internal references to other files (e.g., a HTML content file may reference a JPEG content file, or an XML Schema control file may reference another XML Schema control file). When a file contained in an InterchangePackage references another file using a URI that resolves to a location within the InterchangePackage, the referenced file shall be described in the InterchangePackage's Manifest. A package interchange file (PIF) is a particular instantiation of an InterchangePackage. Package readers and writers are not obliged to support the reading or writing of PIFs. A PIF shall meet the following conditions: a) Physical encapsulation shall be as a compressed binary file conforming to IETF RFC 1951. b) A PIF shall contain a single manifest document object at the root of the PIF. That object shall be bound as an XML document named "imsmanifest.xml". This document is called the root manifest document for the PIF. c) Any control files included in the PIF shall be placed at the root of the PIF. d) All references from within the root manifest document to files contained in the PIF shall be via relative reference. The references shall be relative to the root of the PIF. e) A PIF shall not include any file with an absolute path (declared or resolved from a relative reference) higher than that of the manifest document object in the same hierarchical path, or when their absolute path is completely distinct from the location of a manifest document.

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

6.4 Manifest class family

This subclause defines the following classes in the Content Packaging Information Model (CPIM):

- Manifest
- ManifestMetadata
- Schema
- SchemaVersion
- Metadata Model

This subclause also defines relationships with the following classes defined in 6.5.1, 6.6.1, 6.9, and 6.10, respectively:

- Organizations
- Resources
- IPointer
- Extension

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

Figure 3 shows the Manifest class platform independent model (PIM).

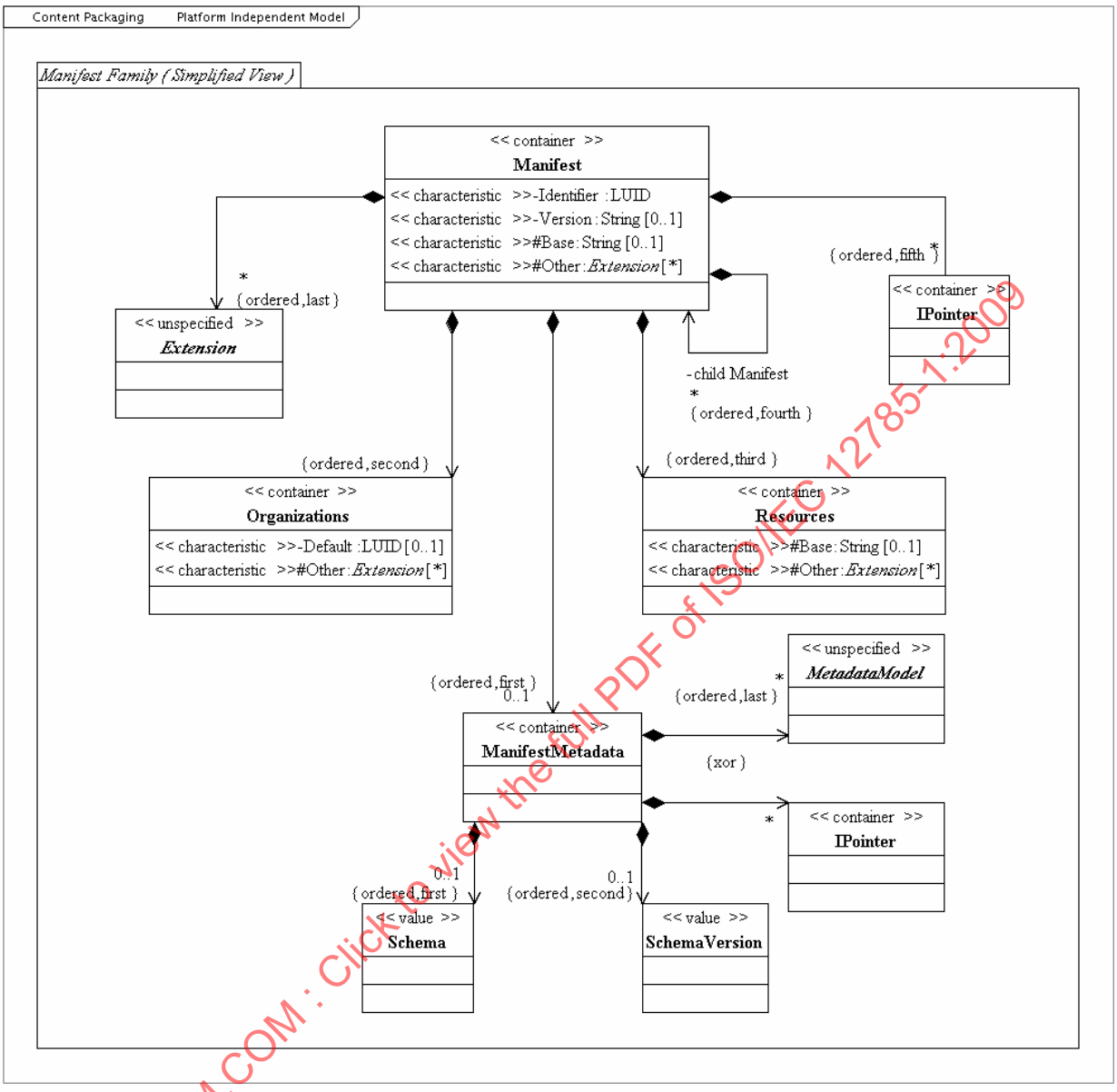


Figure 3 — Simplified view of the Manifest class PIM.

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

6.4.1 Manifest class

Table 3 defines the Manifest class in the Content Packaging Information Model (CPIM).

Table 3 — Manifest class definition.

Descriptor	Definition
Class name	Manifest
Class type	Container
Data type	n / a
Value space	n / a
Multiplicity	1..1 as child of InterchangePackage 0..unbounded as child of Manifest, unordered
Characteristic classes	{ Identifier, Version, Base, Other }, unordered
Parents	InterchangePackage Manifest
Children	[ManifestMetadata, Organizations, Resources, Manifest, IPointer, Extension], ordered
Description	A Manifest object is a container for the data structures that describe a complete instance of a logical package. A Manifest may contain child objects of the Manifest class (child-manifests). Child-manifests define complete instances of logical packages that are part of the larger logical package. A Manifest may also contain child objects of the IPointer class that reference child-manifests external to the Manifest. Any combination of child-manifests and externally referenced child-manifests are permitted and their order within the Manifest is not significant. Appropriate targets for an IPointer declared as a direct child of a Manifest are defined in 6.9. A Manifest may contain references to components that are local or remote to its ancestor InterchangePackage object. References are made via Resource, File, and IPointer child objects. Resource and File are used to reference content files and control files. IPointer is used to identify referenced-manifests. A Manifest shall contain File objects that describe all of the control files needed to interpret the Manifest.

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

6.4.2 ManifestMetadata class

Table 4 defines the ManifestMetadata class in the Content Packaging Information Model (CPIM).

Table 4 — ManifestMetadata class definition.

Descriptor	Definition
Class name	ManifestMetadata
Class type	container
Data type	n / a
Value space	n / a
Multiplicity	0..1
Characteristic classes	n / a
Parents	Manifest
Children	[Schema, Schema Version, IPointer, Metadata Model], ordered
Description	A ManifestMetadata object contains descriptive information about its parent Manifest object. The scope of ManifestMetadata is the entire logical package described by the parent Manifest. The Schema and SchemaVersion children of ManifestMetadata provide information about the specification or profile that governs the meaning of the parent Manifest. A Metadata Model child of ManifestMetadata serves as a placeholder for general descriptive information about its parent Manifest. Metadata Model is an extension point that allows metadata with an information structure that is defined in another namespace. Multiple, differing metadata models may be declared as extensions contained within a single ManifestMetadata object. If the metadata is defined in an external Metadata object, then the link to that object is achieved using an IPointer object. Any combination of Metadata Model and externally referenced metadata is permitted and their order within the Metadata object is not significant Appropriate targets for IPointer declared as a child of ManifestMetadata are defined in 6.9.

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

6.4.3 Schema class

Table 5 defines the Schema class in the Content Packaging Information Model (CPIM)

Table 5 — Schema class definition.

Descriptor	Definition
Class name	Schema
Class type	value
Data type	string
Value space	repertoire of printable characters from [UCS]
Multiplicity	0..1
Characteristic classes	n / a
Parents	ManifestMetadata
Children	n / a
Description	A Schema object declares the name of a specification or profile for its parent Manifest object. The value of Schema informs a package reader of the specification or profile that governs the meaning of the Manifest. This value should not be confused with the name or label assigned to a given metadata schema No requirements are placed on the contents or syntax of a string declared as the value of a Schema, except that the string shall not include any versioning information. The default value shall be “LET content”.

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

6.4.4 SchemaVersion class

Table 6 defines the SchemaVersion class in the Content Packaging Information Model (CPIM).

Table 6 — SchemaVersion class definition.

Descriptor	Definition
Class name	SchemaVersion
Class type	value
Data type	string
Value space	repertoire of printable characters from [UCS]
Multiplicity	0..1
Characteristic classes	n / a
Parents	ManifestMetadata
Children	n / a
Description	A SchemaVersion object declares the version of the specification or profile that is declared as a value for its sibling Schema object. This value informs a package reader of the version of the model or profile identified by a sibling Schema. A value for SchemaVersion shall include no other information. The default value shall be “ISO/IEC 12785:2009” when “LET content” is declared as the value for its sibling Schema and the manifest document is governed by a binding of this Information Model. Neither a syntax for version information nor a heuristic for applying any versioning syntax is specified by this Information Model.

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

6.4.5 MetadataModel class

Table 7 defines the MetadataModel class in the Content Packaging Information Model (CPIM).

Table 7 — Metadata Model class definition.

Descriptor	Definition
Class name	MetadataModel
Class type	unspecified
Data type	unspecified
Value space	unspecified
Multiplicity	0..unbounded, ordering also unspecified
Characteristic classes	unspecified, ordering also unspecified
Parents	ManifestMetadata Metadata
Children	unspecified, ordering also unspecified
Description	A Metadata Model object is a placeholder. It informs bindings of this Information Model as to the valid locations for the inclusion of metadata in an interchange package. The actual names of extending container or value class types comprising one or more metadata models will be known only when a binding of those classes is imported into a bound instance of this Information Model. Hence, the actual type of class is unspecified in this Information Model. Metadata Model is used by two metadata containers: ManifestMetadata and Metadata. MetadataModel is the only means for expressing metadata models and associated information in a bound instance of this Information Model. A package writer may express as many different metadata models as is needed to adequately describe the contents encapsulated by Metadata Model's parent object. Metadata Model shall be used only for declaring one or more metadata models and information associated with them. The semantics of Metadata Model shall stay within the semantics of its parent in this Information Model. The semantics of Metadata Model shall not override or redefine the semantics of its parent as defined in this Information Model.

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

6.5 Organizations class family

This subclause defines the following classes in the Content Packaging Information Model (CPIM):

- Organizations
- Organization
- Title
- Lingual Title

© ISO/IEC 2009 – All rights reserved

- Item

It also defines relationships with the following classes defined in 6.7, 6.9, and 6.10, respectively:

- Metadata
- IPointer
- Extension

Figure 4 shows the Organizations class platform independent model (PIM).

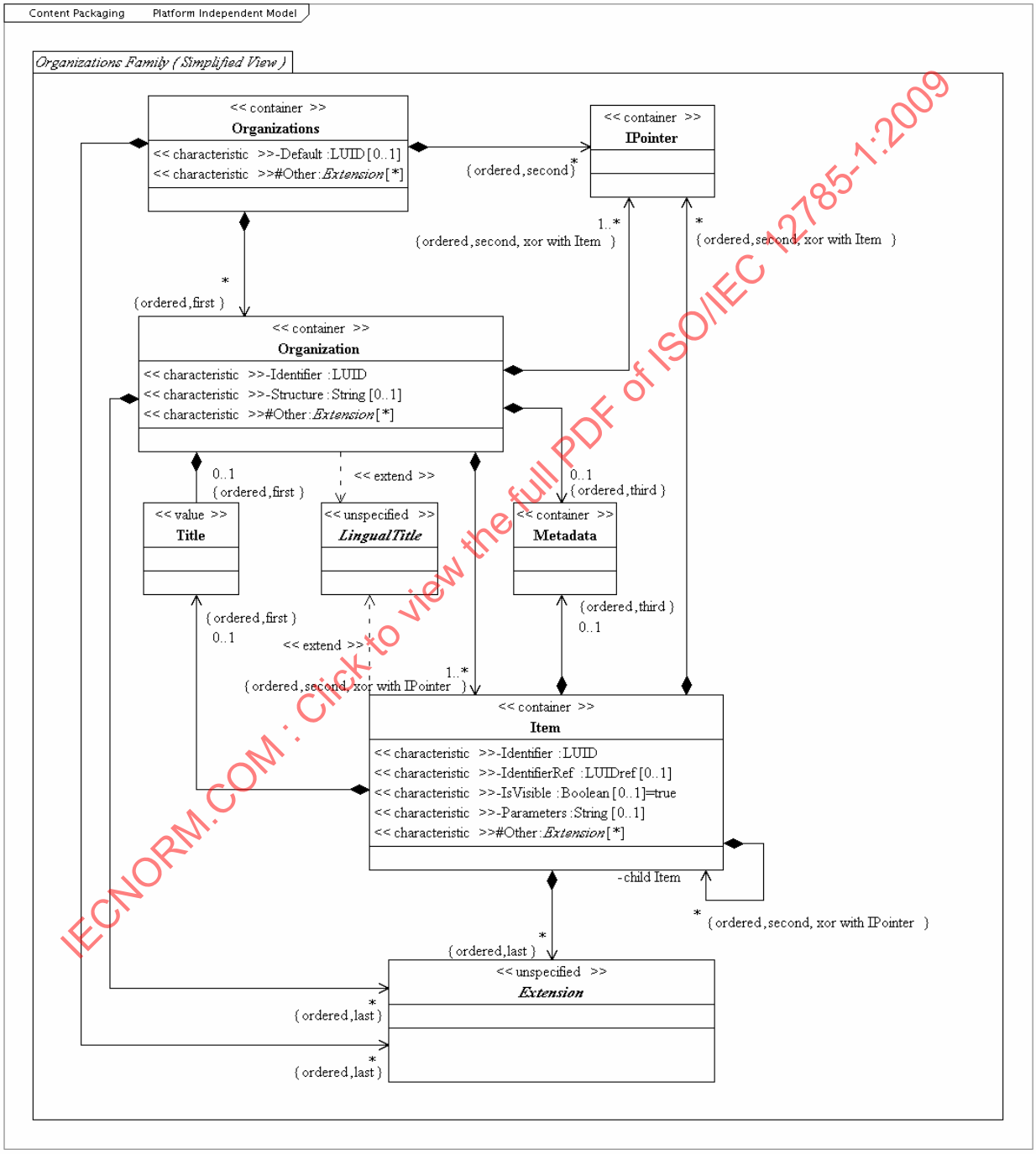


Figure 4 — Simplified view of the Organizations class PIM.

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

6.5.1 Organizations class

Table 8 defines the Organizations class in the Content Packaging Information Model (CPIM).

Table 8 — Organizations class definition.

Descriptor	Definition
Class name	Organizations
Class type	container
Data type	n / a
Value space	n / a
Multiplicity	1..1
Characteristic classes	{ Default, Other }, unordered
Parents	Manifest
Children	[Organization, IPointer, Extension], ordered
Description	An Organizations object is a container for classes describing logical relationships among Resources objects. More than one logical organization may be described using either Organization or IPointer children of Organizations. The order of Organization and IPointer objects is not significant. Organizations defined in referenced-manifests are identified using IPointer. Appropriate targets for IPointer declared as a child of Organizations are defined in 6.9.

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

6.5.2 Organization class

Table 9 defines the Organizations class in the Content Packaging Information Model (CPIM).

Table 9 — Organization class definition.

Descriptor	Definition
Class name	Organization
Class type	container
Data type	n / a
Value space	n / a
Multiplicity	0..unbounded, ordered
Characteristic classes	{ Identifier, Structure, Other }, unordered
Parents	Organizations
Children	[Title, LingualTitle, Item, IPointer, Metadata, Extension], ordered
Description	An Organization object is a container for classes describing a particular logical relationship between the Resource objects encapsulated by a Manifest object. Multiple Organization objects are equivalent in purpose. Each shows a way for structuring the same set of Resource objects within a given Manifest, though each Organization should show a different structure. Item child objects are used to represent structural nodes within an Organization. Item objects defined in referenced-manifests are identified using IPointer objects. Any combination of Item and IPointer objects is permitted and the order of the objects is significant. Appropriate targets for IPointer declared as a child of an Organization are defined in 6.9. At least one child Item or IPointer shall be declared for each Organization.

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

6.5.3 Title class

Table 10 defines the Title class in the Content Packaging Information Model (CPIM).

Table 10 — Title class definition.

Descriptor	Definition
Class name	Title
Class type	value
Data type	string
Value space	repertoire of printable characters from [ISO/IEC 10646:2003]
Multiplicity	0..1
Characteristic classes	n / a
Parents	Organization Item
Children	n / a
Description	A Title object contains a textual value applied to an Organization or Item object to name or label the structures each represents. The textual value has no language type. The value of a Title is a string of characters. Implementers that desire one or more language-sensitive representations of a string of characters for a title should use the LingualTitle class.

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

6.5.4 LingualTitle class

Table 11 defines the Lingual Title class in the Content Packaging Information Model (CPIM).

Table 11 — LingualTitle class definition.

Descriptor	Definition
Class name	LingualTitle
Class type	value
Data type	string
Value space	repertoire of printable characters from [ISO/IEC 10646:2003]
Multiplicity	0..unbounded, unordered
Characteristic classes	{ Language }
Parents	Organization Item
Children	n / a
Description	A LingualTitle object contains a language-specific textual value applied to an Organization or Item object to name or label the structures each represents. The language of LingualTitle is specified by a Language object.

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

6.5.5 Item class

Table 12 defines the Item class in the Content Packaging Information Model (CPIM).

Table 12 — Item class definition.

Descriptor	Definition
Class name	Item
Class type	container
Data type	n / a
Value space	n / a
Multiplicity	1..unbounded, ordered
Characteristic classes	{ Identifier, IdentifierRef, IsVisible, Parameters, Other }, unordered
Parents	Organization Item
Children	[Title, Item, IPointer Metadata, Extension], ordered
Description	An Item object is a container for representing a structural node in a particular Organization or in another Item object. Item objects may contain child Item or IPointer objects, each representing a unique structural node. Items defined in referenced-manifests are identified using the IPointer class. Any combination of Item and IPointer is permitted within an Item, and the order of the objects is significant. Appropriate targets for IPointer declared as a child of an Item are defined in 6.9. An Item may use an IdentifierRef object to express an internal reference to a Resource object that is to be associated with its structural role and position. An Item may use IdentifierRef to express an internal reference to a child-manifest object that is to be associated with its structural role and position. The result of the reference is as if: a) the referencing Item and all of its descendants are replaced with the collection of objects contained in the default Organization of the referenced child-manifest or the first Organization in the referenced child-manifest, if the referenced child-manifest does not have a default Organization; b) the sibling Item objects of the replaced referencing Item are displaced in the sequence of child objects by the indirectly referenced Item and its siblings. A package reader need not actually create a memory model or a data-storage model that mimics an actual splicing or joining of the two structures described above. However, the package reader shall interpret the reference as if the structure described above had been created.

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

6.6 Resources class family

This subclause defines the following classes in the Content Packaging Information Model (CPIM):

- Resources
- Resource
- File
- Dependency

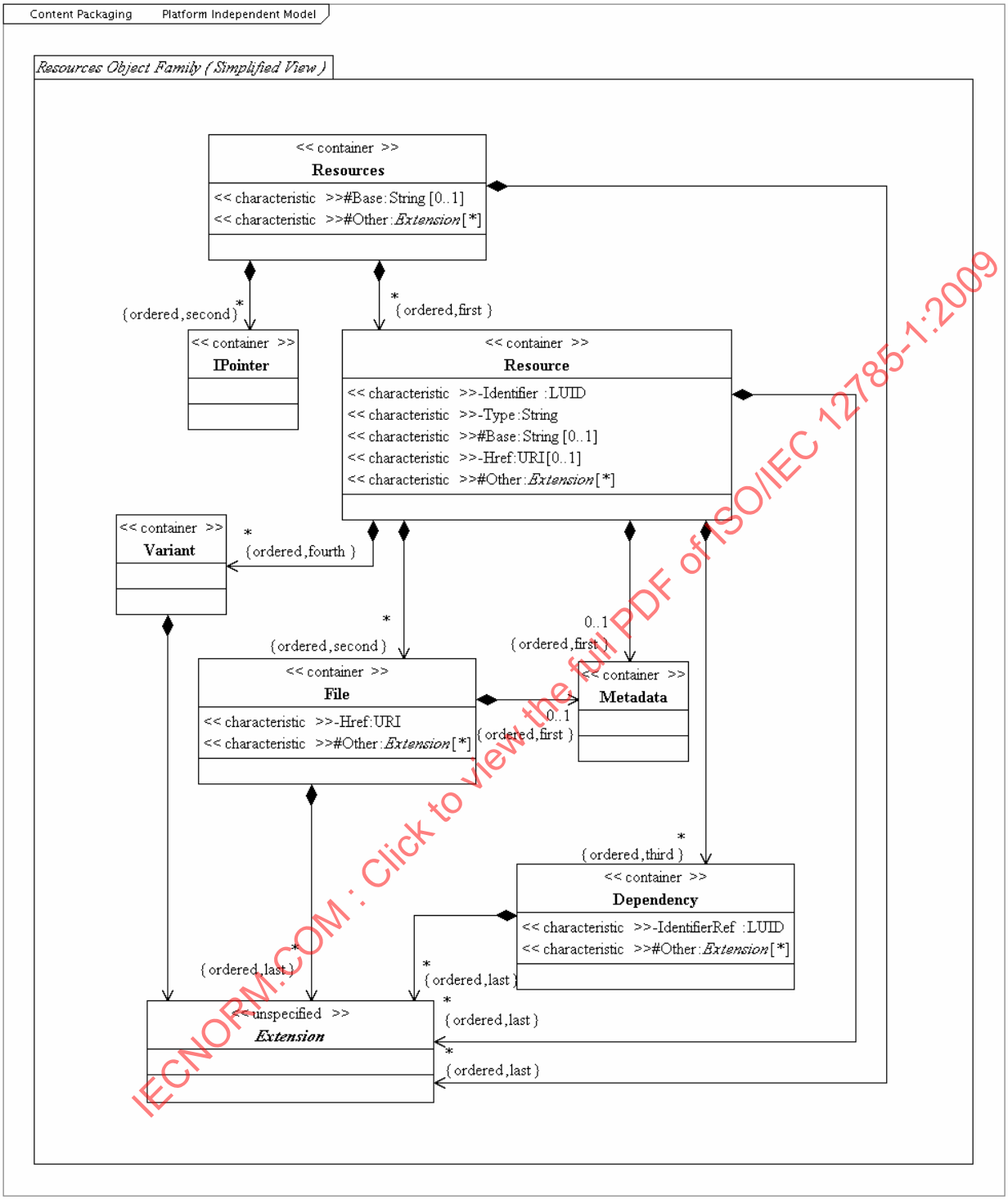
It also defines relationships with the following classes defined in 6.7, 6.8, 6.9, and 6.10, respectively:

- Metadata
- Variant
- IPointer
- Extension

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

Figure 5 shows the Resources class platform independent model (PIM).



IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

Figure 5 — Simplified view of the Resources class PIM.

© ISO/IEC 2009 – All rights reserved

6.6.1 Resources class

Table 13 defines the Resources class in the Content Packaging Information Model (CPIM).

Table 13 — Resources class definition.

Descriptor	Definition
Class name	Resources
Class type	container
Data type	n / a
Value space	n / a
Multiplicity	1..1
Characteristic classes	{ Base, Other }, unordered
Parents	Manifest
Children	[Resource, IPointer, Extension], ordered
Child grouping model	ordered
Description	A Resources object is a container for all information about files used by the parent Manifest object. The files may be local or remote. Resources describes files for its parent Manifest. Reference files in any other Manifest object, including Manifest objects that are siblings to, descendants of, or ancestors of the parent Manifest, are out of the Resources object's scope. The Resource and IPointer children of the Resources object are used to describe a particular collection of files. IPointer is used to identify a Resource in a referenced-manifest. The order of the child Resource and IPointer objects is not significant. Appropriate targets for an IPointer child of a Resources object are defined in 6.9.

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

6.6.2 Resource class

Table 14 defines the Resource class in the Content Packaging Information Model (CPIM).

Table 14 — Resource class definition.

Descriptor	Definition
Class name	Resource
Class type	container
Data type	n / a
Value space	n / a
Multiplicity	0..unbounded, unordered
Characteristic classes	{ Identifier, Type, Base, Href, Other }, unordered
Parents	Resources
Children	[Metadata, File, Dependency, Variant, Extension], ordered
Description	A Resource object is a container for information relating to a particular collection of files used by the ancestor Manifest object. Variant resources are identified using the Variant class. The relationship between the parent Resource and the set of Variant objects shall be described using metadata in the variant resources. A value declared by a Resource{ Href } object is a launchable URI. A file reference declared in a Resource{ Href } shall have an associated declaration in a File object in the same Resource. The Href of the associated File object shall reference the same file as the Resource{ Href }. The Resource{ Href } and associated File{ Href } URIs may differ in that the Resource{ Href } may contain "launch" parameters in the URI.

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

6.6.3 File class

Table 15 defines the File class in the Content Packaging Information Model (CPIM).

Table 15 — File class definition.

Descriptor	Definition
Class name	File
Class type	container
Data type	n / a
Value space	n / a
Multiplicity	0..unbounded, unordered
Characteristic classes	{ Href, Other }, unordered
Parents	Resource
Children	[Metadata, Extension], ordered
Description	A File object is a container for all information relating to a single computer file, encapsulated by its parent Resource object. A File holds metadata describing a computer file and a reference to the location of the file.

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

6.6.4 Dependency class

Table 16 defines the Dependency class in the Content Packaging Information Model (CPIM).

Table 16 — Dependency class definition.

Descriptor	Definition
Class name	Dependency
Class type	container
Data type	n / a
Value space	n / a
Multiplicity	0..unbounded, unordered
Characteristic classes	{ IdentifierRef, Other }, unordered
Parents	Resource
Children	[Extension]
Description	A Dependency object allows a Resource object to reference the collection of files described in a sibling Resource object. Dependency shall use an IdentifierRef object to reference a Resource or an IPointer object. The referenced Resource or IPointer shall be encapsulated by the Dependency's grandparent Resources object. If the reference is to an IPointer, then the IPointer shall reference a Resource within the referenced–manifest. The result of the reference is that the files included in the scope of a referenced Resource shall be considered within the scope of the referencing Dependency's parent Resource. The characteristic objects associated with the referenced Resource shall not be considered in scope of the referencing Dependency's parent Resource.

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

6.7 Metadata Class

This subclause defines the Metadata class abstraction that is part of the Content Packaging Information Model (CPIM). It also defines relationships with the following object defined in 6.9:

- IPointer

Figure 6 shows the Metadata class platform independent model (PIM).

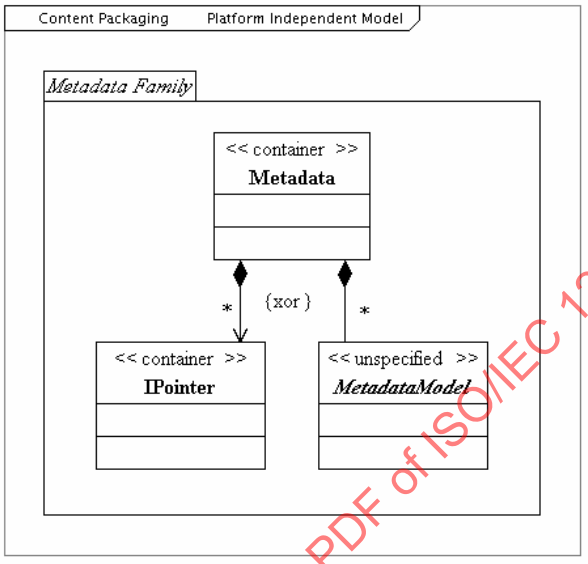


Figure 6 — Simplified view of the Metadata class PIM.

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

Table 17 defines the Metadata class in the Content Packaging Information Model (CPIM).

Table 17 — Metadata class definition.

Descriptor	Definition
Class name	Metadata
Class type	container
Data type	n / a
Value space	n / a
Multiplicity	0..1
Characteristic classes	none
Parents	Organization Item Resource File
Children	[IPointer] or [MetadataModel]
Description	A Metadata object contains descriptive information about its parent containers-class-type object. The scope of Metadata is the parent container class only. The MetadataModel child object serves as a container for general descriptive information about its parent. MetadataModel is an extension point that allows metadata with an information structure that is defined in another namespace. Multiple, differing metadata models may be declared as extensions contained within a single MetadataModel object. If the metadata is defined in an external object, then the link to that object is achieved using an IPointer object. Any combination of MetadataModel and externally referenced metadata is permitted, and their order within the Metadata object is not significant. Appropriate targets for IPointer declared as a child of Metadata are defined in 6.9.

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

6.8 Variant class

This subclause defines the Variant class that is part of the Content Packaging Information Model (CPIM). It also defines relationships with the following classes defined in 6.6.2 and 6.7, respectively:

- Resource
- Metadata

Figure 7 shows the Variant class platform independent model (PIM).

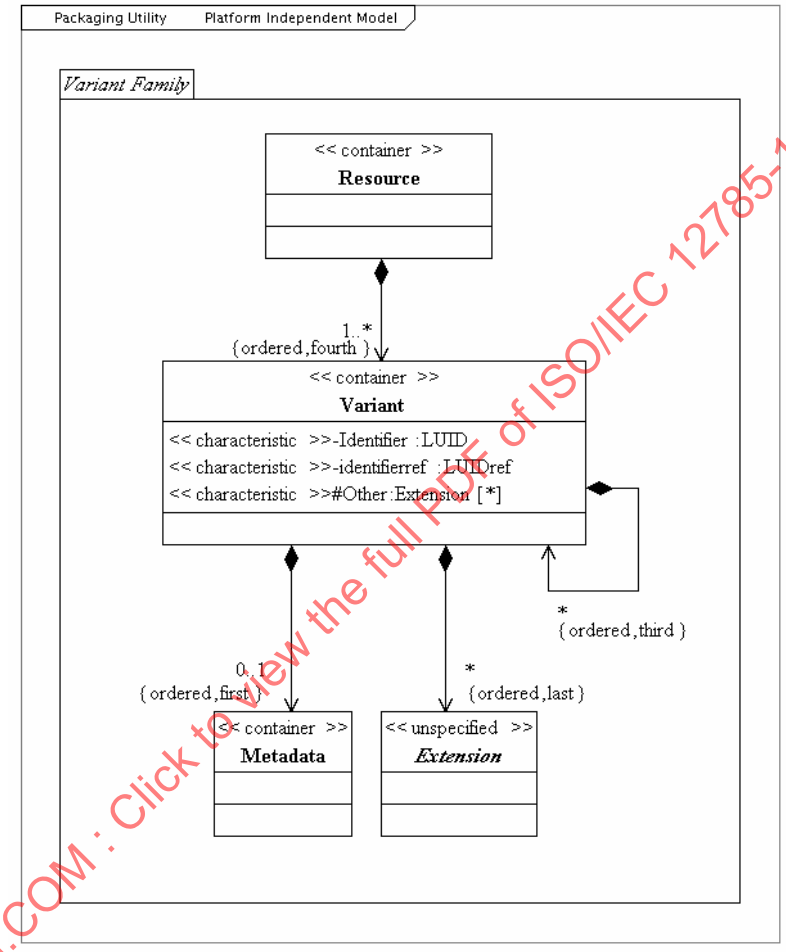


Figure 7 — Simplified view of the Variant class PIM.

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

Table 18 defines the Variant class in the Content Packaging Information Model (CPIM).

Table 18 — Variant class definition.

Descriptor	Definition
Class name	Variant
Class type	container
Data type	n / a
Value space	n / a
Multiplicity	1..unbounded, unordered
Characteristic classes	{ Identifier, IdentifierRef, Other }, unordered
Parents	Resource
Children	[Metadata, Extension], ordered
Description	A Variant object allows a Resource object to reference and describe a variant Resource object. A Variant shall use an IdentifierRef object to reference a Resource or an IPointer object. The referenced Resource or IPointer shall be encapsulated by the Variant's grandparent Resources object. If the reference is to an IPointer, then the IPointer shall reference a Resource within the referenced–manifest. The Metadata child object for the Variant should be used to describe the relationship between the Variant and the parent Resource.

6.9 IPointer class

This subclause defines the IPointer class that is part of the Content Packaging Information Model (CPIM). It also defines relationships with the following classes defined in 6.4.1, 6.4.2, 6.7, 6.5.1, 6.5.2, 6.5.5, and 6.6.1, respectively:

- Manifest
- ManifestMetadata
- Metadata
- Organizations
- Organization
- Item
- Resources

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

Figure 8 shows the IPointer class platform independent model (PIM).

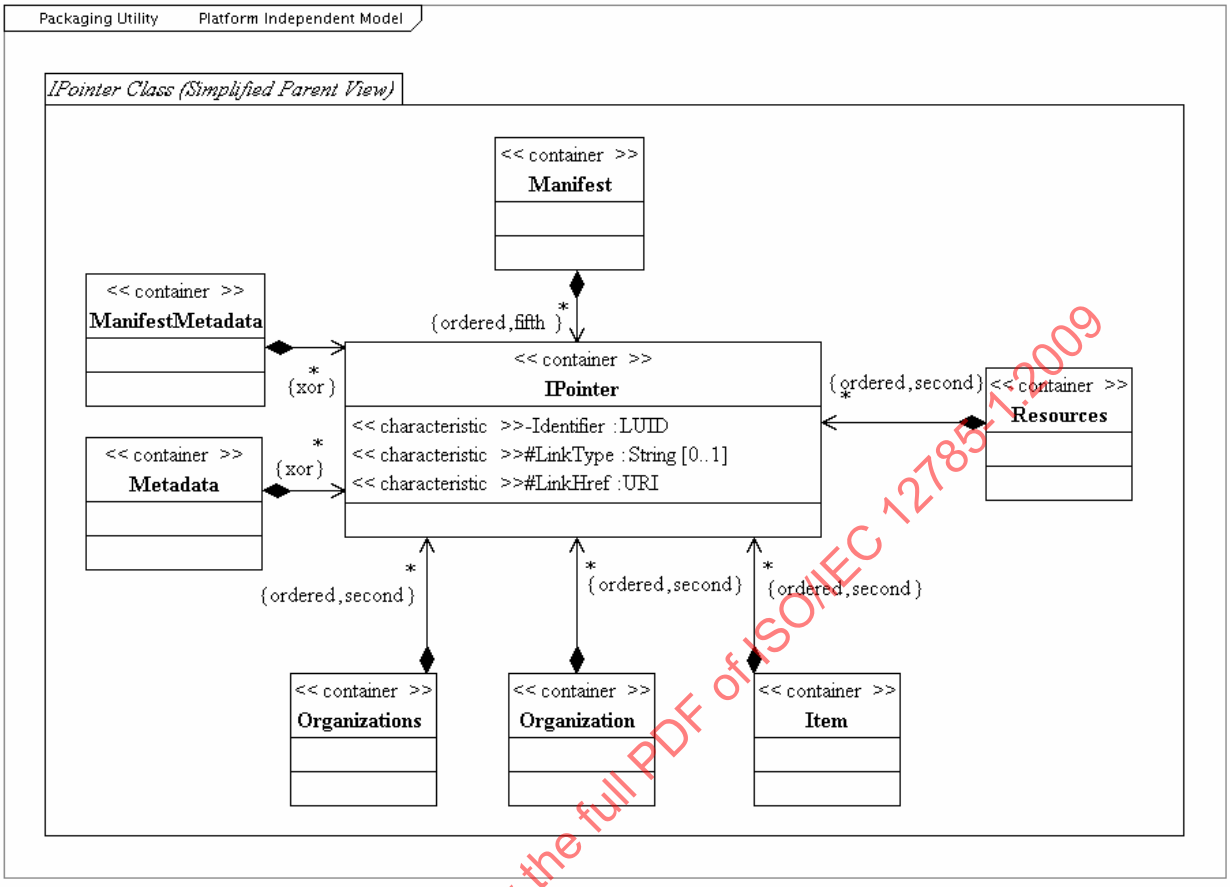


Figure 8 — Simplified view of the IPointer class PIM.

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

Table 19 defines the IPointer class in the Content Packaging Information Model (CPIM).

Table 19 — IPointer class definition.

Descriptor	Definition
Class name	IPointer
Class type	container
Data type	n / a
Value space	n / a
Multiplicity	0..unbounded, ordered
Characteristic classes	{ Identifier, LinkType, LinkHref, Other }, unordered
Parents	Manifest ManifestMetadata Metadata Organizations Organization Item Resources
Children	n / a
Description	An IPointer object is a linking object. Its purpose is to identify a node set in a manifest document and associate that node set with its parent. The source of the identified node set may be either be local or remote. The node set that is identified by the IPointer shall be a valid child of the IPointer's parent class (see Table 20).

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

Table 20 defines the permitted combinations of parent-class-to-target-class linking allowed using the IPointer class.

Table 20 — Permitted linking combinations of parent/target classes.

Parent class of the IPointer class	Valid target nodeset for the IPointer class
Manifest	Manifest
Organizations	Organization
Organization	Item
Item	Item
Resources	Resource
ManifestMetadata	MetadataModel
Metadata	MetadataModel

6.10 Extension class

This subclause defines the Extension class placeholder that is part of the Content Packaging Information Model (CPIM). Figure 9 shows the Extension class platform independent model (PIM).

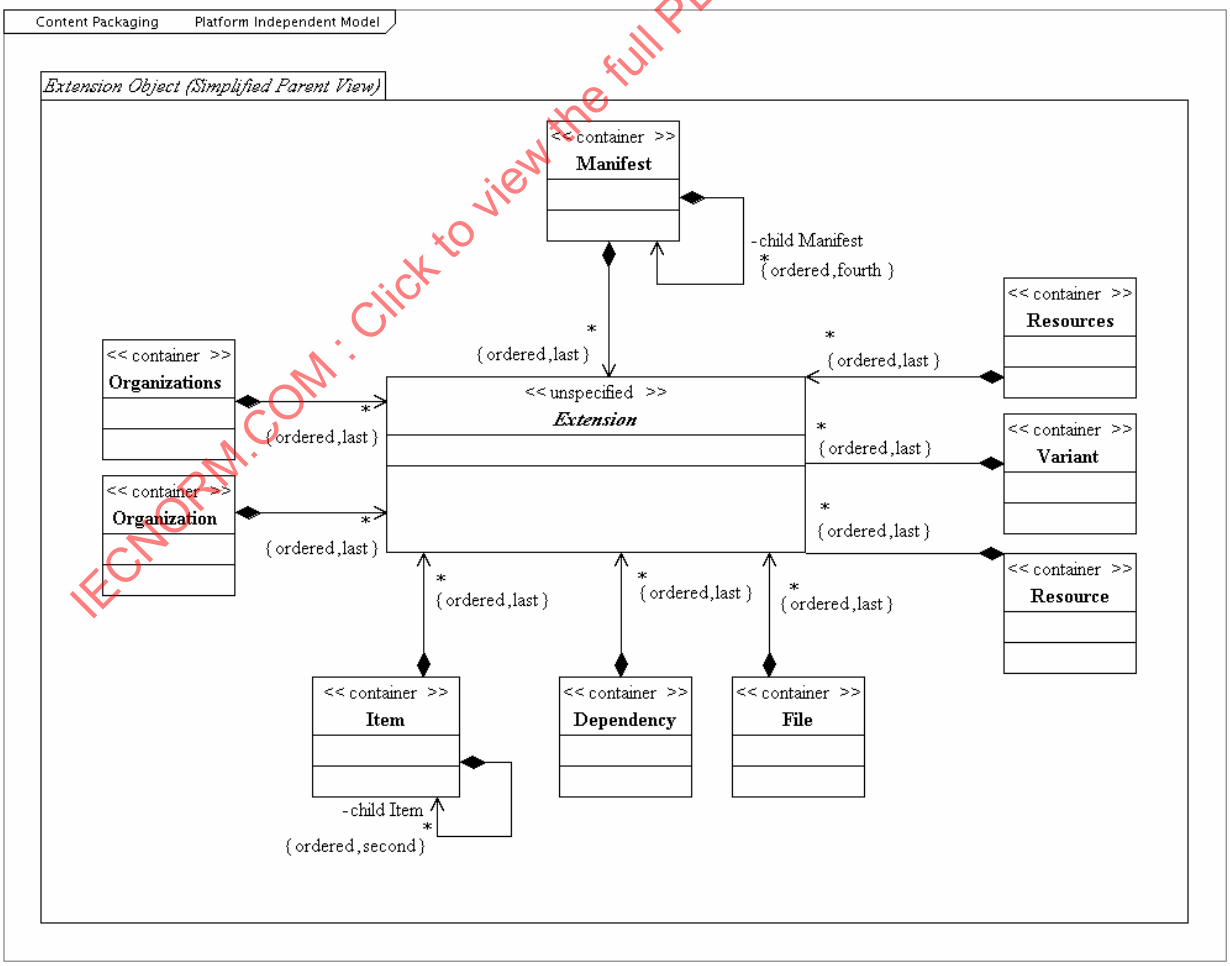


Figure 9 — Simplified view of the Extension class PIM.

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

Table 21 defines the Extension class in the Content Packaging Information Model (CPIM).

Table 21 — Extension class definition.

Descriptor	Definition
Class name	Extension
Class type	unspecified
Data type	unspecified
Value space	unspecified
Multiplicity	0..unbounded, ordered
Characteristic classes	unspecified, ordering also unspecified
Parents	Manifest Organizations Organization Item Resources Resource Variant File Dependency
Children	unspecified, ordering also unspecified
Description	An Extension object is a placeholder. It informs bindings of this Information Model as to the valid locations for the inclusion of value or container classes that extend any class of type container in this Information Model. The Extension class is one of two mechanisms for extending any class of type container in this Information Model. The second extension mechanism is the Other class. An Extension object is used to extend classes of type container. The actual name of an extending container or value class type used in place of an Extension object will be known only when a binding of that object is imported into a bound instance of this Information Model. Hence, the actual type of class is unspecified in this Information Model. The semantics of Extension shall stay within the scope of the semantics of its parent object in this Information Model. The semantics of Extension shall not override or redefine the semantics of any parent object defined in this Information Model.

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

6.11 Characteristic classes

6.11.1 Base class

Table 22 defines the Base class in the Content Packaging Information Model (CPIM).

Table 22 — Base class definition.

Descriptor	Definition
Class name	Base
Class type	characteristic
Data type	URI
Value space	Binding language dependent.
Multiplicity	0..1
Parents	Manifest Resources Resource
Description	A Base object is used to specify an object's base URI for the object with which Base is associated for the purpose of resolving relative references that appear in descendants of the associated object. The scope of Base's value shall apply to all descendants of the object with which Base is associated, unless the value declared for Base is replaced or amended by a subsequent Base object in a permitted descendant. All relative path segments expressed as a value for Base shall resolve to the manifest document containing those expressions. Relative path segments shall be constructed in such a way that the segment expressed Base can be appended to a relative path segment expressed in the nearest ancestor object expressing a value in its Base object and so on up to the root Manifest object. The resulting sequence of collected relative path segments shall then be appended to the location at which the manifest document is located or being processed to create an absolute path. Relative references appearing within the scope of a Base object shall then be interpreted according to the rules defined in IETF RFC 3986.

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

6.11.2 Default class

Table 23 defines the Default class in the Content Packaging Information Model (CPIM).

Table 23 — Default class definition.

Descriptor	Definition
Class name	Default
Class type	characteristic
Data type	Binding language dependent (the default is string).
Value space	the value of an Organizations{ Default }.Organization{ Identifier } object
Multiplicity	0..1
Parents	Organizations
Description	A Default object designates a single child Organization object of an Organizations object as the primary or default organizing structure for a given Manifest object. The designation of a default Organization shall be done by referencing the value of the Identifier object of the target Organization object. A target Organization shall be a child of an Organizations object that has a Default. No other reference by Default shall be permitted. When a Default is not declared for an Organizations object, the first defined Organization object in an Organizations object shall be considered the primary or default organizing structure.

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

6.11.3 Href class

Table 24 defines the Href class in the Content Packaging Information Model (CPIM).

Table 24 — Href class definition.

Descriptor	Definition
Class name	Href
Class type	characteristic
Data type	URI
Value space	as defined by IETF RFC 3986.
Multiplicity, by parent object	0..1 : Resource 1..1 : File
Parents	Resource File
Description	An Href object is used to locate a resource. A value declared for Href shall be a syntactically valid URI. This document makes no guarantee that a string so declared is a valid URI or will resolve to an actual resource or part of a resource at the location so identified. A value declared for Manifest.Resources.Resource{Href} represents a URL that can be used to locate and access the content described by the resource. A package reader is not required to resolve the URI. The URI is meant to be stored for later use by other software components after the contents of an interchange package are processed. A value declared for Manifest.Resources.Resource.File{Href}shall be a locator for a single digital resource, with no other implication as to the use of the resource so identified.

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

6.11.4 Identifier class

Table 25 defines the Identifier class in the Content Packaging Information Model (CPIM).

Table 25 — Identifier class definition.

Descriptor	Definition
Class name	Identifier
Class type	characteristic
Data type	Binding language dependent Locally Unique Identifier (LUID).
Value space	Binding language dependent.
Multiplicity	1..1
Parents	Manifest Organization Item Resource IPointer Variant
Description	An Identifier object uniquely identifies that Identifier object's parent within a Manifest object. That is, only one occurrence of a given value for an Identifier may occur within the same Manifest, including that Manifest's child-manifest objects. A value for Identifier may be used for internal referencing from another object using an IdentifierRef object.

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

6.11.5 IdentifierRef class

Table 26 defines the IdentifierRef class in the Content Packaging Information Model (CPIM).

Table 26 — IdentifierRef class definition.

Descriptor	Definition
Class name	IdentifierRef
Class type	Characteristic
Data type	Binding language dependent reference to an existing LUID (LUIDref).
Value space	A value for an Identifier object in the IdentifierRef object's immediately containing Manifest object, and as further constrained by internal referencing rules defined below.
Multiplicity, by parent objects	0..1 : Item 1..1 : Dependency 1..1 : Variant
Parents	Item Dependency Variant
Description	An IdentifierRef object shall duplicate exactly a value for an Identifier object that is a descendant of the IdentifierRef object's immediately containing Manifest object. An object's immediately containing Manifest is defined as the first encountered Manifest in the object's chain of ancestors. The referenced Identifier may be in a child-manifest of the immediately containing Manifest. In addition, IdentifierRef is limited by the following internal referencing rules: A. An IdentifierRef child of an Item object may reference one of: i) A Resource or IPointer object that is a descendant of the referencing Item's immediately containing Manifest object. ii) A Manifest or IPointer that is a descendant of the referencing Item's immediately containing Manifest iii) A Resource or IPointer that is contained within a child-manifest object that is a descendant of the referencing Item's immediately containing Manifest. No reference shall be allowed from an IdentifierRef of an Item in a child-manifest object to any object in an ancestor Manifest. B. An IdentifierRef child of a Dependency object: i) Shall reference only a Resource that is a sibling of the Dependency's parent Resource. ii) Shall not reference the Dependency's parent Resource. iii) Shall not reference any object in a Manifest that is a child or ancestor of the Dependency's immediately containing Manifest.

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

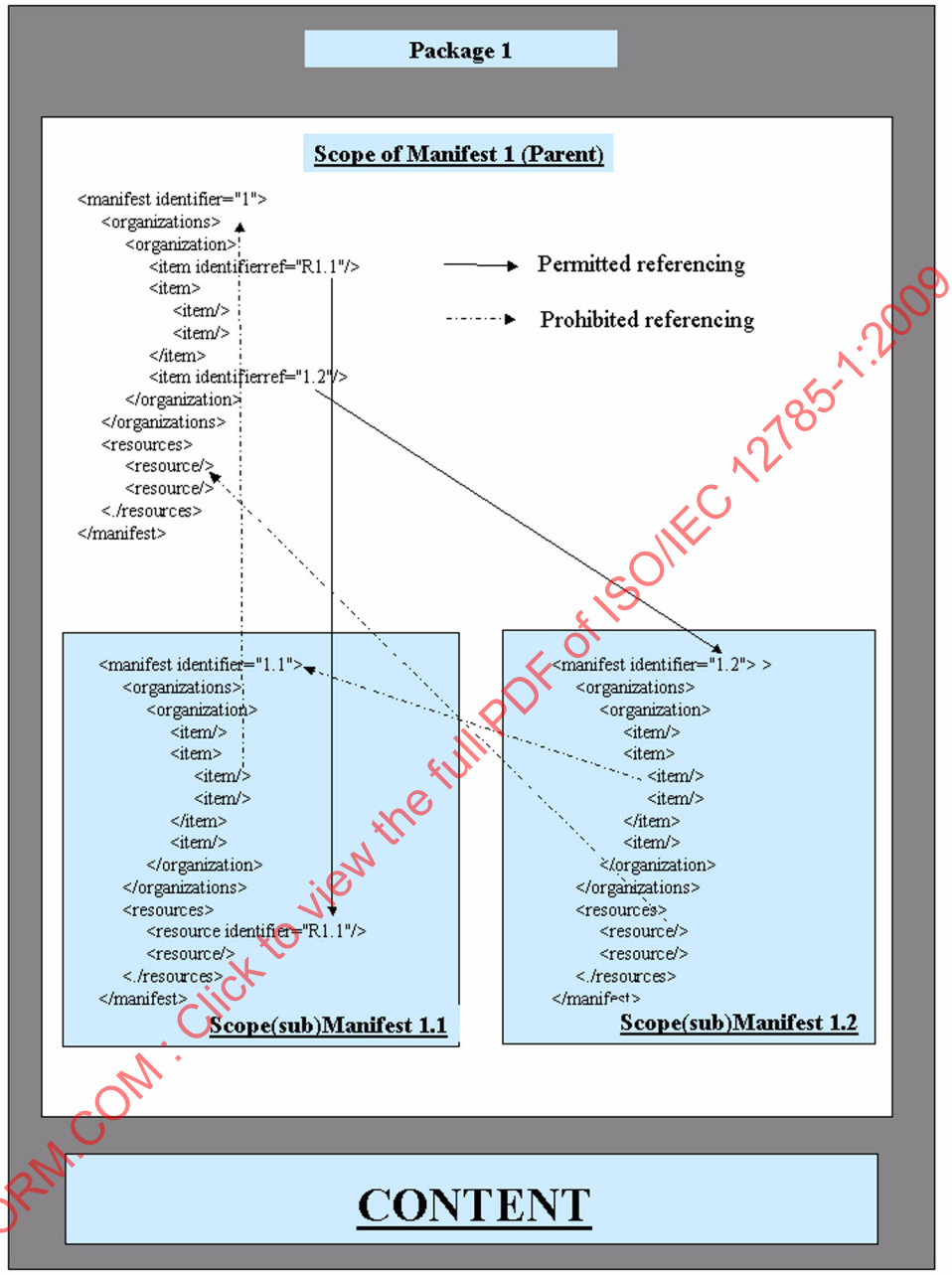
© ISO/IEC 2009 – All rights reserved

	C. An IdentifierRef child of a Variant object: i) Shall reference only a Resource or IPointer that is a sibling of the Variant's parent Resource. ii) Shall not reference the Variant's parent Resource. iii) Shall not reference any object in a Manifest that is a child or ancestor of the Variant's immediately containing Manifest.

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved

The scoping rules for manifests and (sub)manifests are shown in Figure 10.



IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

Figure 10 — Scoping rules for manifests and (sub)manifests.

© ISO/IEC 2009 – All rights reserved

6.11.6 IsVisible class

Table 27 defines the IsVisible class in the Content Packaging Information Model (CPIM).

Table 27 — IsVisible class definition.

Descriptor	Definition
Class name	IsVisible
Class type	Characteristic
Data type	Boolean
Value space	true (default) false
Multiplicity	0..1
Parents	Item
Description	An IsVisible object signals a rendering process of whether to display a text string declared in a sibling Title object or to visually indicate the existence of an Item object in any other way. No other behavior is imputed by this flag. There is no inheritance of visibility state declared by this IsVisible for any descendant Item of this IsVisible's parent Item. A value of "true" shall be the default value for IsVisible, even if it is not declared in a bound instance of an Item. That is, the absence of IsVisible for a parent Item is the same as if IsVisible with a value of "true" had been declared for the Item. A value of "true" shall be interpreted to mean that the contents of a sibling Title should be displayed by a rendering application (i.e., Item{ IsVisible }.Title). A value of "false" shall be interpreted to mean that the contents of a sibling Title should not be displayed by a rendering application (i.e., Item{ IsVisible }.Title).

IECNORM.COM : Click to view the full PDF of ISO/IEC 12785-1:2009

© ISO/IEC 2009 – All rights reserved