DEUTSCHE NORM September 2006

DIN EN ISO 19118

ICS 35.240.70

Geoinformation – Kodierung (ISO 19118:2005); Englische Fassung EN ISO 19118:2006

Geographic information – Encoding (ISO 19118:2005); English version EN ISO 19118:2006

Information géographique – Codage (ISO 19118:2005); Version anglaise EN ISO 19118:2006

Gesamtumfang 115 Seiten

Normenausschuss Bauwesen (NABau) im DIN

Documents
Document Sharing Hub

Full document details: DIN EN ISO 19118:2006-09

Nationales Vorwort

Der Text von ISO 19118:2005 wurde vom Technischen Komitee ISO/TC 211 „Geoinformation/Geomatik“ erarbeitet und als EN ISO 19118:2006 vom Technischen Komitee CEN/TC 287 „Geoinformation“ übernommen, dessen Sekretariat vom NEN (Niederlande) gehalten wird.

Der Text von ISO 19118:2005 wurde von CEN als EN ISO 19118:2006 ohne Änderungen angenommen.

Der für die deutsche Mitarbeit zuständige Arbeitsausschuss im DIN Deutsches Institut für Normung e. V. ist der als Spiegelausschuss zum CEN/TC 287 und ISO/TC 211 eingesetzte Arbeitsausschuss NA 005-03-03 AA „Kartographie und Geoinformation“ des Normenausschusses Bauwesen (NABau).

Das Präsidium des DIN hat mit seinem Beschluss 1/2004 festgelegt, dass von dem in den Regeln der europäischen Normungsarbeit von CEN/CENELEC verankerten Grundsatz, wonach Europäische Normen in den drei offiziellen Sprachen Deutsch, Englisch und Französisch veröffentlicht werden, in begründeten Ausnahmefällen abgewichen und auf die deutsche Sprachfassung verzichtet werden kann.

Die Genehmigung dafür hat die DIN-Geschäftsleitung entsprechend ihren in Anlage 1 zu dem DIN-Rundschreiben A 5/2004 festgelegten Kriterien für die vorliegende Norm auf Antrag des NABau-FB 03 „Vermessungswesen; Geoinformation“ erteilt.

Allgemeines

DIN EN ISO 19118 legt ein Regelwerk für die Beschreibung von Kodierungsregeln für geographische Daten im Rahmen der Normenreihe ISO 19100 fest. Eine Kodierungsregel erlaubt es, Daten zu einem Anwendungsschema als systemunabhängige Datenstrukturen für den Datentransport und die Datenspeicherung zu kodieren.

Diese Norm ist in drei logische Bereiche aufgeteilt:

— Die Anforderungen für die Festlegung von Kodierungsregeln auf der Basis von UML Schemata sind in den Abschnitten 6 bis 8 spezifiziert.

— Die Anforderungen für die Festlegung Kodierungsdiensten sind in Abschnitt 9 spezifiziert.

— Im Anhang A wird in Ergänzung zu dem allgemeinen Regelwerk eine, in dieser Version informative, allgemeine Kodierungsregel für XML-basierte Daten spezifiziert. Diese Beschreibung ist offen gestaltet und lässt Raum für konkretere Ausprägungen für XML-basierte Kodierungsregeln (siehe ISO/DIS 19136 für XML-Kodierungsregeln für Objektarten in Anwendungsschemata und ISO/FDTS 19139 für XML-Kodierungsregeln für Metadaten).

AVA Documents
Document Sharing Hub

Full document details: DIN EN ISO 19118:2006-09

Nationaler Anhang NA
(informativ)

Englisch-deutsche Fachwörterliste

Application schema	Anwendungsschema
Character	Zeichen
Code	Kode
Conceptual model	Konzeptuelles Modell
Conceptual schema	Konzeptuelles Schema
Conceptual schema language	Modellierungsprache
Conversion rule	Konvertierungsregel
Data	Daten
Data element	Datenelement
Data interchange	Datenaustausch
Data transfer	Datenübertragung
Data type	Datentyp
Dataset	Datenbestand
Encoding	Kodierung
Encoding rule	Kodierungsregel
Encoding service	Kodierungssdienst
Feature	Fachobjekt
File	Datei
Identifier	Identifikator
Instance model	Instanzenmodell
Interface	Schnittstelle
Medium	Medium
Model	Modell
Schema	Schema
Schema model	Schemamodell
Stereotype	Stereotyp
Transfer protocol	Übertragungsprotokoll
Universe of discourse	Gegenstandsbereich
Value domain	Wertebereich

Documents
Document Sharing Hub

Full document details: DIN EN ISO 19118:2006-09

Nationaler Anhang NB
(informativ)

Literaturhinweise

ISO/DIS 19136, *Geographic information — Geography Markup Language (GML)*

ISO/FDTS 19139, *Geographic information — Metadata — XML schema implementation*

Documents
Document Sharing Hub

Full document details: DIN EN ISO 19118:2006-09

EUROPEAN STANDARD

NORME EUROPÉENNE

EUROPÄISCHE NORM

EN ISO 19118

July 2006

ICS 35.240.70

English Version

Geographic information – Encoding (ISO 19118:2005)

Information géographique – Codage (ISO 19118:2005)

Geoinformation – Kodierung (ISO 19118:2005)

This European Standard was approved by CEN on 16 June 2006.

CEN members are bound to comply with the CEN/CENELEC Internal Regulations which stipulate the conditions for giving this European Standard the status of a national standard without any alteration. Up-to-date lists and bibliographical references concerning such national standards may be obtained on application to the Central Secretariat or to any CEN member.

This European Standard exists in three official versions (English, French, German). A version in any other language made by translation under the responsibility of a CEN member into its own language and notified to the Central Secretariat has the same status as the official versions.

CEN members are the national standards bodies of Austria, Belgium, Cyprus, Czech Republic, Denmark, Estonia, Finland, France, Germany, Greece, Hungary, Iceland, Ireland, Italy, Latvia, Lithuania, Luxembourg, Malta, Netherlands, Norway, Poland, Portugal, Romania, Slovakia, Slovenia, Spain, Sweden, Switzerland and United Kingdom.

EUROPEAN COMMITTEE FOR STANDARDIZATION
COMITÉ EUROPÉEN DE NORMALISATION
EUROPÄISCHES KOMITEE FÜR NORMUNG

Management Centre: rue de Stassart, 36 B-1050 Brussels

© 2006 CEN All rights of exploitation in any form and by any means reserved worldwide for CEN national Members.

Ref. No. EN ISO 19118:2006: E

Documents
Document Sharing Hub

Full document details: DIN EN ISO 19118:2006-09

Contents

Page

Foreword 5
Introduction 6
1 Scope 7
2 Conformance 7
3 Normative references 7
4 Terms and definitions 7
5 Symbols and abbreviated terms 11
6 Fundamental concepts and assumptions 12
6.1 Concepts 12
6.2 Data interchange 12
6.3 Application schema 13
6.4 Encoding rule 13
6.5 Encoding service 14
6.6 Transfer service 15
7 Character repertoire 15
8 Encoding rules 16
8.1 Introduction 16
8.2 General encoding requirements 17
8.3 Input data structure 19
8.4 Output data structure 19
8.5 Conversion rules 19
8.6 Examples 20
9 Encoding service 20
Annex A (informative) XML based encoding rule 22
A.1 Introduction 22
A.2 General encoding requirements 23
A.3 Input data structure 25
A.4 Output data structure 29
A.5 Schema conversion rules 29
A.6 Instance conversion rules 58
A.7 Abstract test suite 60
A.8 Level 2 conformance 61
Annex B (normative) Abstract test suite 62
B.1 Introduction 62
B.2 Level 1 conformance — General encoding requirements 62
B.3 Level 2 conformance — Interface 62
Annex C (informative) Extensible Markup Language (XML) 63
C.1 Introduction 63
C.2 Extensible Markup Language 63
Annex D (informative) Character repertoire 72
Annex E (informative) Examples 75
E.1 Introduction 75
E.2 Standard types 75
E.3 Simple-Road-Map 85
E.4 Property-Building-Loan 87
E.5 Property-Building-Updating 94
Bibliography 110

AVA Documents
Document Sharing Hub

Full document details: DIN EN ISO 19118:2006-09

Page

Figures

Figure 1 — Overview of data interchange between two systems 12
Figure 2 — The encoding rule defines conversion rules from input concepts to output concepts 14
Figure 3 — Overview of the encoding process 15
Figure 4 — Example of exchange metadata 18
Figure 5 — Conversion rules 20
Figure 6 — Example encoding service interface 21
Figure A.1 — XML based conversion rules 22
Figure A.2 — Instance model: Dataset, object and property 26
Figure A.3 — Instance model: Value types 26
Figure A.4 — Example application schema 28
Figure A.5 — Example data 29
Figure A.6 — Units of Measure 34
Figure A.7 — Measure types 35
Figure A.8 — Example of <<Enumeration>> 36
Figure A.9 — Example of <<CodeList>> 37
Figure A.10 — Example of <<Union>> 37
Figure A.11 — Record types 38
Figure A.12 — Example of bounded template type 39
Figure A.13 — Example of single inheritance 40
Figure A.14 — Example of multiple inheritance 42
Figure A.15 — Example attribute of a supertype 43
Figure A.16 — Example attribute 45
Figure A.17 — Example association 46
Figure A.18 — Example aggregation 47
Figure A.19 — Example composition 47
Figure A.20 — Document structure 48
Figure A.21 — Dataset contains objects 49
Figure A.22 — Exchange metadata 50
Figure A.23 — Update primitives 51
Figure A.24 — Configuration file: top elements 54
Figure A.25 — Configuration file: structured types 55
Figure A.26 — Configuration file: bounded template types 55
Figure A.27 — Configuration file: codelist, enumeration and external type 55
Figure D.1 — UCS-4 structure 72
Figure E.1 — Geometric primitives 76
Figure E.2 — Coordinate geometry 77
Figure E.3 — Geometric complexes 77
Figure E.4 — Topology 78
Figure E.5 — SRM application schema 85
Figure E.6 — Simple map according to the SRM application schema 86
Figure E.7 — PBL Application schema 87
Figure E.8 — PBL example data 88

Documents
Document Sharing Hub

Full document details: DIN EN ISO 19118:2006-09

Page

Figure E.9 — Property-Building-Updating application schema 95
Figure E.10 — Feature types by inheritance 95
Figure E.11 — Example data 96

Tables

Table A.1 — Stereotypes on classes 24
Table A.2 — Summary of relationship between UML and the instance model 27
Table A.3 — Mapping of attributes with multiplicity and collection type 28
Table A.4 — Multiplicity mapping for attributes 45
Table A.5 — Multiplicity mapping for content elements 45
Table C.1 — DTD attribute types 67
Table C.2 — Two special purpose XML attributes 67
Table C.3 — XLink attributes 68
Table D.1 — UTF8 byte sequences to represent a character 73

Documents
Document Sharing Hub

Full document details: DIN EN ISO 19118:2006-09

Foreword

The text of ISO 19118:2005 has been prepared by Technical Committee ISO/TC 211 "Geographic information/Geomatics" of the International Organization for Standardization (ISO) and has been taken over as EN ISO 19118:2006 by Technical Committee CEN/TC 287 "Geographic Information", the secretariat of which is held by NEN.

This European Standard shall be given the status of a national standard, either by publication of an identical text or by endorsement, at the latest by January 2007, and conflicting national standards shall be withdrawn at the latest by January 2007.

According to the CEN/CENELEC Internal Regulations, the national standards organizations of the following countries are bound to implement this European Standard: Austria, Belgium, Cyprus, Czech Republic, Denmark, Estonia, Finland, France, Germany, Greece, Hungary, Iceland, Ireland, Italy, Latvia, Lithuania, Luxembourg, Malta, Netherlands, Norway, Poland, Portugal, Romania, Slovakia, Slovenia, Spain, Sweden, Switzerland and United Kingdom.

Endorsement notice

The text of ISO 19118:2005 has been approved by CEN as EN ISO 19118:2006 without any modifications.

AVA Documents
Document Sharing Hub

Full document details: DIN EN ISO 19118:2006-09

Introduction

This International Standard specifies the requirements for defining encoding rules to be used for interchange of geographic data within the ISO 19100 series of International Standards. An encoding rule allows geographic information defined by application schemas and standardized schemas to be coded into a system-independent data structure suitable for transport and storage. The encoding rule specifies the types of data to be coded and the syntax, structure and coding schemes used in the resulting data structure. The resulting data structure may be stored on digital media or transferred using transfer protocols. It is intended to be read and interpreted by computers, but may be in a form that is human readable.

The choice of one encoding rule for neutral data interchange does not exclude application domains and individual nations from defining and using their own encoding rules that can be platform dependent or more effective with regards to data size or processing complexity. XML is a subset of ISO/IEC 8879 and has been chosen because it is independent of computing platform and interoperable with the World Wide Web.

This International Standard is divided into three logical sections. The requirements for creating encoding rules based on UML schemas are specified in Clauses 6 to 8. The requirements for creating encoding service are specified in Clause 9, and the XML based encoding rule is specified in Annex A.

The XML based encoding rule is intended to be used for neutral data interchange. It relies on the Extensible Markup Language (XML) and the ISO/IEC 10646 character set standards. Introductions to XML and ISO/IEC 10646 are given in Annexes C and D, respectively. Annex E contains examples of the application of this International Standard.

The geographic information standards are organized in the ISO 19100 series of International Standards. The background, the overall structure of this series of International Standards and the fundamental description techniques are defined in ISO 19101, ISO/TS 19103 and ISO 19104.

Users of this International Standard will develop application schemas to capture the semantics of geographic information. An application schema is compiled by integrating elements from a set of standardized conceptual schemas developed in ISO 19107, ISO 19108, ISO 19110, ISO 19111, ISO 19112, ISO 19113, ISO 19115 and ISO 19117, including eventually new standardized conceptual schemas. How this integration will take place is described in ISO 19109. The ISO 19100 series of International Standards also defines a set of common services that shall be available when developing geographic information applications. The common services are generally defined in ISO 19119 and will cover access to and processing of geographic information according to the common information model. Two service areas are defined more closely in ISO 19116 and ISO 19117. ISO 19105, ISO 19106, ISO 19114 and this International Standard cover implementation issues.

Documents
Document Sharing Hub

Full document details: DIN EN ISO 19118:2006-09

1 Scope

This International Standard specifies the requirements for defining encoding rules to be used for interchange of geographic data within the ISO 19100 series of International Standards.

This International Standard specifies

— requirements for creating encoding rules based on UML schemas,

— requirements for creating encoding services,

— an informative XML based encoding rule for neutral interchange of geographic data.

This International Standard does not specify any digital media, it does not define any transfer services or transfer protocols, nor does it specify how to encode inline large images.

2 Conformance

Two conformance levels are defined for this International Standard. The conformance levels are defined in the abstract test suite in Annex B.

3 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 10646, *Information technology — Universal Multiple-Octet Coded Character Set (UCS)*

ISO 8601, *Data elements and interchange formats — Information interchange — Representation of dates and times*

ISO/TS 19103:2005, *Geographic information — Conceptual schema language*

ISO 19109:2005, *Geographic information — Rules for application schema*

Extensible Markup Language (XML) 1.0 (Second Edition), W3C Recommendation 6 October 2000. Available at <http://www.w3.org/TR/REC-xml>

XML Schema Part 1: Structures, W3C Recommendation 2, May 2001. Available at <http://www.w3.org/TR/xmlschema-1/>

XML Schema Part 2: Datatypes, W3C Recommendation 2, May 2001. Available at <http://www.w3.org/TR/xmlschema-2/>

4 Terms and definitions

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

4.1
application schema
conceptual schema for **data** required by one or more applications

[ISO 19101]

Documents
Document Sharing Hub

Full document details: DIN EN ISO 19118:2006-09

NOTE An application schema describes the content, the structure and the constraints applicable to information in a specific application domain.

4.2
character
member of a set of elements that is used for the representation, organization, or control of **data**

[ISO/IEC 2382-1]

4.3
code
representation of a label according to a specified scheme

4.4
conceptual model
model that defines concepts of a **universe of discourse**

[ISO 19101]

4.5
conceptual schema
formal description of a **conceptual model**

[ISO 19101]

4.6
conceptual schema language
formal language based on a conceptual formalism for the purpose of representing **conceptual schemas**

[ISO 19101]

EXAMPLE UML, EXPRESS, IDEF1X.

NOTE A conceptual schema language may be lexical or graphical.

4.7
conversion rule
rule for converting instances in the input **data** structure to instances in the output **data** structure

4.8
data
reinterpretable representation of information in a formalized manner suitable for communication, interpretation, or processing

[ISO/IEC 2382-1]

4.9
data element
unit of **data** that, in a certain context, is considered indivisible

4.10
data interchange
delivery, receipt and interpretation of **data**

4.11
data transfer
movement of **data** from one point to another over a **medium**

NOTE Transfer of information implies transfer of data.

Documents
Document Sharing Hub

Full document details: DIN EN ISO 19118:2006-09

4.12
data type
specification of a **value domain** with operations allowed on values in this domain

[ISO/TS 19103]

EXAMPLE Integer, Real, Boolean, String and Date.

NOTE A data type is identified by a term, e.g. Integer. Values of the data types are of the specified value domain, e.g. all integer numbers between –65537 and 65536. The set of operations can be +, –, ÷ and × and is semantically well defined. A data type can be simple or complex. A simple data type defines a value domain where values are considered atomic in a certain context, e.g. Integer. A complex data type is a collection of data types which are grouped together. A complex data type may represent an object and can thus have identity.

4.13
dataset
identifiable collection of **data**

[ISO 19115]

4.14
encoding
conversion of **data** into a series of **codes**

4.15
encoding rule
identifiable collection of **conversion rules** that define the **encoding** for a particular **data** structure

EXAMPLE XML, ISO 10303-21, ISO/IEC 8211.

NOTE An encoding rule specifies the types of data to be converted as well as the syntax, structure and codes used in the resulting data structure.

4.16
encoding service
software component that has an **encoding rule** implemented

4.17
feature
abstraction of real world phenomena

[ISO 19101]

NOTE A feature may occur as a type or an instance. Feature type or feature instance is used when only one is meant.

4.18
file
named set of records stored or processed as a unit

[ISO/IEC 2382-1]

4.19
geographic data
data with implicit or explicit reference to a location relative to the Earth

[ISO 19109]

Documents
Document Sharing Hub
Full document details: DIN EN ISO 19118:2006-09

4.20
geographic information
information concerning phenomena implicitly or explicitly associated with a location relative to the Earth

[ISO 19101]

4.21
identifier
label that uniquely identifies an item or group of items

4.22
information
knowledge concerning objects, such as facts, events, things, processes, or ideas, including concepts, that within a certain context has a particular meaning

[ISO/IEC 2382-1]

4.23
instance model
representation **model** for storing **data** according to an **application schema**

4.24
interface <UML>
named set of operations that characterize the behaviour of an element

[ISO/IEC 19501]

4.25
interoperability
capability to communicate, execute programs, or transfer **data** among various functional units in a manner that requires the user to have little or no knowledge of the unique characteristics of those units

[ISO/IEC 2382-1]

4.26
medium
substance or agency for storing or transmitting **data**

EXAMPLE Compact disc, internet [1], radio waves, etc.

4.27
model
abstraction of some aspects of reality

[ISO 19109]

4.28
schema
formal description of a **model**

[ISO 19101]

4.29
schema model
representation **model** for storing **schemas**

EXAMPLE Representation model for a schema repository.

Documents
Document Sharing Hub

Full document details: DIN EN ISO 19118:2006-09

4.30
stereotype <UML>
new type of modelling element that extends the semantics of the metamodel

[ISO/IEC 19501]

NOTE Stereotypes must be based on certain existing types or classes in the metamodel. Stereotypes may extend the semantics, but not the structure of pre-existing types and classes. Certain stereotypes are predefined in the UML, others may be user defined. Stereotypes are one of three extensibility mechanisms in UML; the others are constraint and tagged value.

4.31
transfer protocol
common set of rules for defining interactions between distributed systems

4.32
universe of discourse
view of the real or hypothetical world that includes everything of interest

[ISO 19101]

4.33
value domain
set of accepted values

[ISO/TS 19103]

EXAMPLE The range 3-28, all integers, any ASCII character, enumeration of all accepted values (green, blue, white).

5 Symbols and abbreviated terms

API	Application Programming Interface
DTD	Document Type Definition
ID	Identifier
IDREF	An XML ID reference type
OMG	Object Management Group
UCS	Universal Multiple-Octet Coded Character Set
UML	Unified Modelling language
URI	Uniform Resource Identifier
UTF	UCS Transfer Format
UUID	Universal Unique Identifier
XMI	XML Metadata Interchange
XML	Extensible Markup Language
XPointer	XML Pointer language
XSD	XML Schema Document

Documents
Document Sharing Hub
Full document details: DIN EN ISO 19118:2006-09

6 Fundamental concepts and assumptions

6.1 Concepts

The purpose of the ISO 19100 series of International Standards is to enable interoperability between heterogeneous geographic information systems. To achieve interoperability between heterogeneous systems two fundamental issues need to be determined. The first issue is to define the semantics of the content and the logical structures of geographic data. This shall be done in an application schema. The second issue is to define a system and platform-independent data structure that can represent data corresponding to the application schema.

Subclauses 6.2 to 6.6 describe the fundamental concepts of data interchange, i.e. the procedure based on the application schema for encoding, delivery, receipt and interpretation of geographic data. 6.2 describes an overview of the data interchange process; 6.3 introduces application schemas that allow interpretation of geographic data; 6.4 describes the importance of the encoding rule for producing system-independent data structures; 6.5 describes a software component, called the encoding service, for executing the encoding rule; and 6.6 describes the procedure for delivery and receipt called the transfer service.

6.2 Data interchange

An overview of a data interchange is shown in Figure 1. System A wants to send a dataset to system B. To ensure a successful interchange, A and B must decide on three things, i.e. a common application schema I, which encoding rule R to apply, and what kind of transfer protocol to use. The application schema is the basis of a successful data transfer and defines the possible content and structure of the transferred data, whereas the encoding rule defines the conversion rules for how to code the data into a system-independent data structure.

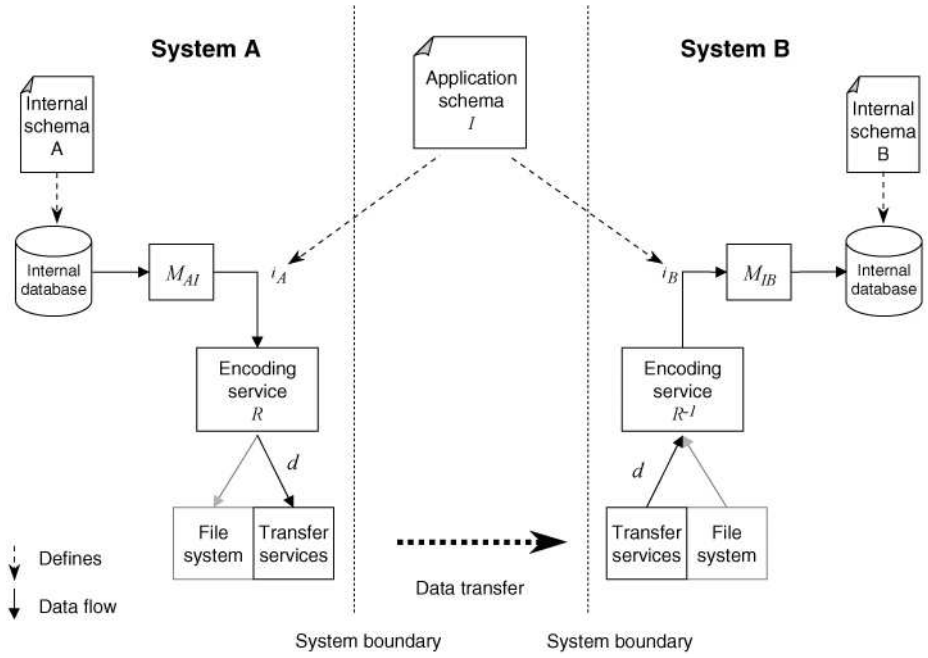

Figure 1 — Overview of data interchange between two systems

Both systems, A and B, store data in an internal database according to an internal schema, but the schemas are usually different, i.e. schema A ≠ schema B. The following logical steps must be taken in order to transfer a dataset from A's internal database to B's internal database.

a) The first step for system A is to translate its internal data into a data structure that is according to the common application schema I. Here this is done by defining a mapping from the concepts of the internal schema to the concepts defined in the application schema and by writing appropriate mapping software

Documents
Document Sharing Hub
Full document details: DIN EN ISO 19118:2006-09

to translate the data instances. In Figure 1 this mapping is denoted M_{AI}. The result is an application schema specific data structure i_A. The data structure is stored in memory or on an intermediate file and is system dependent and thus not suitable for transfer.

b) The next step is to use an encoding service, which applies the encoding rule R to create a data structure that is system independent and therefore suitable for transfer. This encoded dataset is called d and may be stored in a file system or transferred using a transfer service.

c) System A then invokes a transfer service to send the encoded dataset d to system B. The transfer service follows a transfer protocol for how to do packaging and how the actual transportation over an on-line or off-line communication medium should take place. Both parties must agree upon the transfer protocol used.

d) The transfer service on system B receives the transferred dataset, and according to the protocol the dataset is unpacked and stored as an encoded dataset d, e.g. on an intermediate file.

e) In order to get an application schema specific data structure i_B, system B applies the inverse encoding rule R^{-1} to interpret the encoded data.

f) To use the dataset, B must translate the application schema specific data structure i_B into its internal database. This is done by defining a mapping from the application schema into its internal schema and by writing software that does the actual translation. In Figure 1 this mapping is denoted M_{IB}.

This International Standard only specifies the requirements for creating encoding rules and the encoding services and not the whole data interchange process. Thus, only steps b) and e) are standardized. Steps c) and d) use general information technology transfer services.

6.3 Application schema

An application schema is a conceptual schema for applications with similar data requirements. The application schema is the basis of a successful data interchange and defines the possible content and structure of the data. It is also the basis for implementing application schema specific data structures for local storage of data.

The application schema used for encoding in compliance with this International Standard shall be written in the UML conceptual schema language, in accordance with ISO/TS 19103 and ISO 19109. These two standards specify a framework for how to write application schemas. The rules include specifications on how to use standardized schemas to define feature types. Both a sender and a receiver of data must have access to the application schema.

Implementation of application schema specific data structures may be semi-automated, e.g. by using a compiler for the lexical part or by using a graphical modelling tool with support for code generation.

The application schema shall be accessible to both ends of a data interchange to ensure a successful result. The application schema has to be transferred before data interchange takes place so that both the receiver and sender can prepare their systems by implementing mappings and data structures according to the application schema. It may be transferred together with the dataset, or it may be stored in a public place and referenced from the dataset.

The application schema may be interchanged by paper or electronic based methods. The XML Metadata Interchange (XMI) developed by the Object Management Group (OMG) is the recommended electronic method for data interchange of an application schema.

6.4 Encoding rule

6.4.1 Concept

An encoding rule is an identifiable collection of conversion rules that defines the encoding for a particular data structure. The encoding rule specifies the data types to be converted, as well as the syntax, structure and

Documents
Document Sharing Hub
Full document details: DIN EN ISO 19118:2006-09

coding schemes used in the resulting data structure. An encoding rule is applied to application schema specific data structures to produce system-independent data structures suitable for transport or storage. In order to define an encoding rule three important aspects must be specified, i.e. the input data structure, the output data structure and the conversion rules between the elements of the input and the output data structures. Both the input and output data structures are written using a conceptual schema language and the concepts in the languages are used to define the encoding rule.

6.4.2 Input data structure

The input data structure is an application schema specific data structure. The data structure can be thought of as a set of data instances, i.e. $\boldsymbol{i} = \{ i_1, ..., i_p \}$, see Figure 1. Each data instance i_k is an instance of a concept I_l defined in an application schema. The application schema defines a set of concepts defined in the application schema $\boldsymbol{I} = \{ I_1, ..., I_m \}$.

The application schema is a conceptual schema c written in a conceptual schema language C. The conceptual schema defines a set of concepts $\boldsymbol{c} = \{ c_1, ..., c_m \}$ by instantiating the concepts of the conceptual schema language $C = \{ C_1, ..., C_r \}$. Since the application schema is a conceptual schema $\boldsymbol{c} = \boldsymbol{I}$.

6.4.3 Output data structure

The output data structure is defined by a schema $\boldsymbol{D} = \{ D_1, ..., D_s \}$. D is the schema for the output structure and is not shown in Figure 1. The output data structure can be thought of as a set of data instances, i.e. $\boldsymbol{d} = \{ d_1, ..., d_q \}$ where each data instance d_k is an instance of a concept D_l.

The schema D defines the syntax, structure and coding schemes of the output data structure.

6.4.4 Conversion rules

A conversion rule specifies how a data instance in the input data structure shall be converted to zero, one, or more instances in the output data structure. The conversion rules are defined and based on the concepts of the conceptual schema language C and on the concepts of the output data structure schema D. We need to specify a conversion rule R_i for each of the legal combinations of concepts in the conceptual schema language. The set of conversion rules are $\boldsymbol{R} = \{ R_1, ..., R_n \}$, where R_i is the i-th conversion rule and C_i is the i-th legal combination of instances from the schema language. A conversion table for all possible C_i can be set up, where each C_i maps to a production of instances in the output data structure D. Figure 2 shows the relationship between the input and output conceptual schema language and the encoding rule.

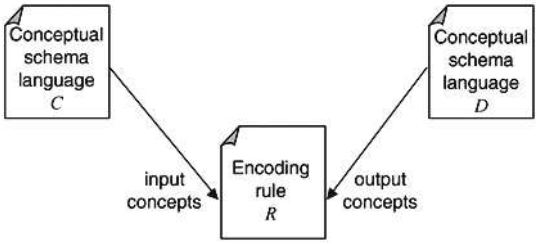

Figure 2 — The encoding rule defines conversion rules from input concepts to output concepts

NOTE The conversion rules are defined based on the two schema languages and not on any particular application schema. This is a generic approach that allows developers to write application schema independent encoding services, which can be used for different application schemas as long as the schemas are defined in the same conceptual schema language.

6.5 Encoding service

An encoding service is a software component that has implemented the encoding rule and provides an interface to encoding and decoding functionality. It is an integrated part of data interchange.

Documents
Document Sharing Hub
Full document details: DIN EN ISO 19118:2006-09

Figure 3 presents the details of an encoding service and its relationships to important specification schemas. The encoding service shall be able to read the input data structure and convert the instances to an output data structure and vice versa. It shall also be able to read the application schema declarations and write the corresponding output data structure schema. The input data structure is defined by an application schema. The application schema is defined using concepts of the conceptual schema language. The output data structure is also described with a schema, called the data structure schema, which defines the possible content, structure and coding schemes of the output data structure. The data structure schema is described with a schema language. The encoding rule specifies conversion rules at two levels, the first is at the schema level and the second is at the instance level. At the schema level, the conversion rules define a mapping for each of the concepts defined in the application schema to corresponding concepts in the data structure schema. At the instance level the conversion rules define a mapping for each of the instances in the input data structure to corresponding instances in the output data structure. The instance conversion rules are normally deduced from the schema conversion rules.

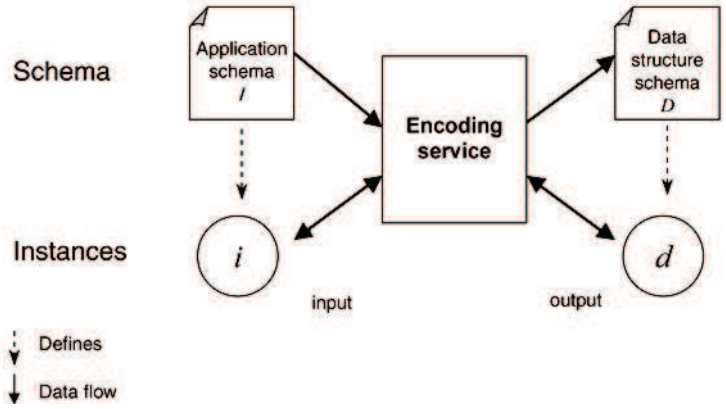

Figure 3 — Overview of the encoding process

An encoding service shall at least provide interfaces for encoding and decoding functionality. Examples of such interfaces are for encoding $d = encode\ (i, I)$, and for decoding: $i = decode\ (d, I)$. Here, i is a reference to an application schema specific data structure, I is a reference to the application schema and d is a reference to the system independent data structure.

6.6 Transfer service

A transfer service is a software component that has implemented one or more transfer protocols, which allows data transfer between distributed information systems over off-line or on-line communication media. To successfully transfer data between two systems, the sender and receiver need to agree on the transfer protocol to be used.

Different transfer protocols can be defined. One example is off-line transfer protocols where data is stored on optical or magnetic media and delivered using postal services or other dedicated delivery services. Another example is on-line transfer protocols where data is compressed and included as an email attachment, delivered using a file transfer protocol or transferred using other distributed information technology services which rely on an underlying network service.

This International Standard does not prescribe any preferred transfer protocols.

7 Character repertoire

This International Standard adopts the international character set standards defined in ISO/IEC 10646. ISO/IEC 10646 defines an internationally recognized repertoire of characters called the Universal Character Set (UCS) and their character-encoding schemes.

AVA Documents Document Sharing Hub

Full document details: DIN EN ISO 19118:2006-09

The character-encoding schemes that may be supported by international profiles of this International Standard are the following:

a) 8-bit variable size UCS Transfer Format UTF-8;

b) 16-bit variable size UCS Transfer Format UTF-16;

c) 16-bit fixed size Universal Character Set UCS-2;

d) 32-bit fixed size Universal Character Set UCS-4.

International encoding rules that claim conformance with this International Standard shall support one or more of these character-encoding schemes. Within national profiles and system implementations different character-encoding schemes may be used. The fixed size character-encoding schemes are often used in database implementations and the variable size is often used for data interchange purposes.

The character set standards adopted specify only the repertoire of characters and give no indication of which language is actually used. In cases where it is important to distinguish between different languages in text strings special mechanisms to indicate the language used shall be used. International profiles of this International Standard may support language-tagging mechanisms as defined in ISO/IEC 11179-3.

ISO/IEC 10646 defines mechanisms for creating composite characters. Composite characters are characters produced by superimposing one or more additional characters on a base character. ISO/IEC 10646 defines a set of precomposed characters and their defined decomposition. Since mixing composite characters with their precomposed equivalents may lead to interpretation problems, the use of a composite character if a precomposed character exists is deprecated, i.e. the precomposed character shall always be used.

To summarize an encoding rule shall

— support one or more character-encoding schemes;

— support language tagging mechanisms if necessary;

— not use composite characters if equivalent precomposed characters exist.

EXAMPLE The precomposed character ö has the defined decomposition o¨.

NOTE For a more detailed description of these standards, see Annex D.

8 Encoding rules

8.1 Introduction

Subclauses 8.2 to 8.6 define the requirements for specifying encoding rules. An encoding rule describes conversion rules for transforming data from an input data structure to an output data structure. Schemas shall be described for both the input and output data structure. The schema for the input data structure is called an application schema.

An encoding rule shall in general specify the following.

a) General encoding requirements (8.2):

   1) application schemas and schema language;

   2) international character repertoire and languages supported;

   3) necessary exchange metadata;

Documents
Document Sharing Hub

Full document details: DIN EN ISO 19118:2006-09

4) dataset and object identification scheme; and

5) possible update mechanism.

b) Input data structure (8.3):

1) data structure used to store data according to an application schema, called the instance model; and

2) how the instance model is related to the application schema.

c) Output data structure used, called the exchange format (8.4).

d) Conversion rules, called the mapping, for converting data in the instance model to the exchange format (8.5):

1) conversion rules for encoding;

2) if necessary, conversion rules for decoding.

e) Sufficient examples of abstract data, application of conversion rules and encoded data (8.6).

8.2 General encoding requirements

8.2.1 Application schema and schema language

The encoding rule shall specify the schema language used to define application schemas and describe how an application schema is organized.

8.2.2 Character repertoire and languages

The character repertoire defines the characters that are allowed. If more than one language is required, a language tagging mechanism should be considered to identify the actual language of encoded text.

The encoding rule shall specify the character repertoire used, the languages and language identification mechanisms supported.

8.2.3 Exchange metadata

Exchange metadata are metadata about the encoded data structure. They may describe the originator of the dataset, refer to metadata information about the dataset, refer to the application schema and provide information about the encoding rule applied.

The encoding rule shall specify the exchange metadata and how it accompanies the encoded data structure.

Figure 4 illustrates an example of an ExchangeMetadata class and a corresponding EncodingRule class. The CI_Citation class is defined in ISO 19115. An exchange file normally contains one instance of the ExchangeMetadata class. The "datasetCitation" attribute contains information about the originator of the dataset, the "metadataCitation" attribute may contain information about where to find metadata about the dataset, the "applicationSchemaCitation" attribute may hold a reference to the application schema. The EncodingRule's "encodingRuleCitation" attribute identifies the encoding rule applied and a description of how the rule was applied in this particular case. It may also contain information about the encoding service tool used to encode the dataset, where "toolName" and "toolVersion" indicate the name of the encoding service tool and its version number.

Documents
Document Sharing Hub
Full document details: DIN EN ISO 19118:2006-09

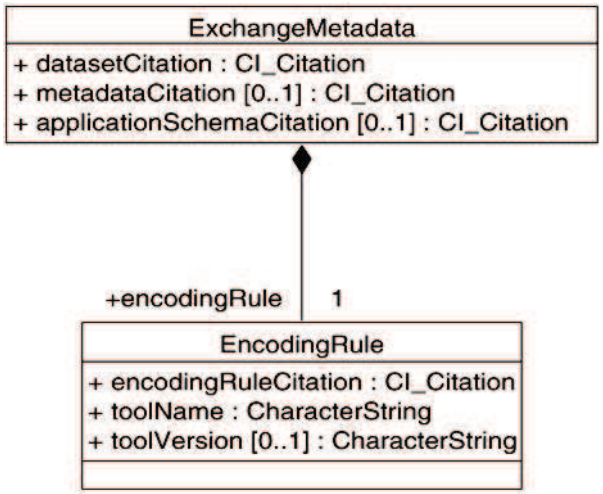

Figure 4 — Example of exchange metadata

8.2.4 Dataset

8.2.4.1 Granularity and structure

It is important to define the granularity and the structure of a dataset so that it can be encoded and decoded efficiently. An object is here considered as the basic unit of information and different types of objects must be identified. The objects in a dataset may be structured sequentially and/or hierarchically. An object may be internally structured as a sequence of attributes, it may contain references to other objects and it may also be composed of other objects in a hierarchical manner.

The encoding rule shall specify

— what an object is and the different types of object,

— the structure of an object,

— the structure of a dataset.

8.2.4.2 Object identification

Objects may be assigned identifiers that allow them to be uniquely identified within a particular context. Two different contexts should be considered.

a) The first context is a dataset. Here the object identifiers (IDs) are unique within a particular dataset. These identifiers allow objects to refer to other objects within the context of a dataset. The identifiers are usually assigned to the objects when they are inserted into the dataset and are transient in nature.

b) The second context is an application domain. An application domain defines a universe and an identification scheme called universal unique identifiers (UUIDs). A UUID is assigned to an object when it is created and is stable over the entire life span of the object. A UUID of a deleted object cannot be used again. UUIDs are required for long-term distributed data management and for realizing update mechanisms. These identifiers are also called persistent identifiers. A special name server may be used to resolve persistent identifiers. The identifiers are unique within a well-defined limited universe defined by an application domain.

The encoding rules shall specify

— the different object identification mechanisms used, and

Documents
Document Sharing Hub
Full document details: DIN EN ISO 19118:2006-09

— their internal structure.

NOTE There are several ways of defining universal unique identifiers. The International Hydrographic Organization uses a Universal Unique Identifier as defined in IHO S-57 [ISO/TR 19120] to identify all hydrographic objects in order to permit the continuous dynamic update of Electronic Nautical Charts. The Open Group CAE Specification, DCE 1.1: Remote Procedure Call also defines a UUID structure, see [2].

8.2.5 Update mechanism

An update mechanism allows previously exchanged data to be brought up to date without the need for reissuing a complete new dataset. Policies and procedures for update shall be defined by the specific application. Three basic update primitives are usually defined: *add*, *modify* and *delete*. These primitives work on the object level, but may also be defined to work on the attribute or association level. Any object that has previously been transmitted with a UUID may be modified or deleted. An update dataset contains an ordered sequence of update primitives. The basic primitives are described as follows.

a) *add*: A new object has been added to the source dataset and shall be added into the target dataset. An add primitive shall contain information about the new object to be added and may contain information about where it is inserted in the target dataset.

b) *modify*: An existing object has been modified in the source dataset and shall be modified in the target dataset. A modify primitive shall contain information that identifies the target object and the actual modifications. Examples of modification information could range from a complete object to just an updated attribute.

c) *delete*: An existing object has been deleted in the source dataset and shall therefore be deleted in the target dataset. The delete primitive shall contain information that identifies the target object to be deleted.

The encoding rule shall specify whether an update mechanism is required or not. If an update mechanism is required the encoding rule shall specify

— policies and procedures for updates,

— the different update primitives supported and how they refer to existing objects by universal unique identifiers (UUIDs) and convey update information.

8.3 Input data structure

The input data structure is called the instance model and it is an instrument for reasoning about application data, for defining conversion rules and for expressing examples. The instance model shall be capable of representing data according to the specification in the application schema. It may be specific to a particular application schema or capable of representing data according to any schema. It may be abstract in that it does not have to be implemented in order to realize an encoding service.
The encoding rule shall specify the instance model and its relation to the application schema.

8.4 Output data structure

The output data structure defines how data is structured and represented in an exchange file. A schema may accompany the output data structure.

The encoding rule shall specify the output data structure and if present the output data structure schema.

8.5 Conversion rules

A conversion rule specifies how a data instance in an input data structure is converted to a data instance in the output data structure. Two sets of conversion rules may exist. The first one is the set of schema conversion rules, which define a mapping from the UML schema to the schema of the output data structure.

AVA Documents Document Sharing Hub

Full document details: DIN EN ISO 19118:2006-09

The second one is the set of instance conversion rules, which defines a mapping from instances in the instance model to instances in the resulting data model. Figure 5 shows the different conversion rules.

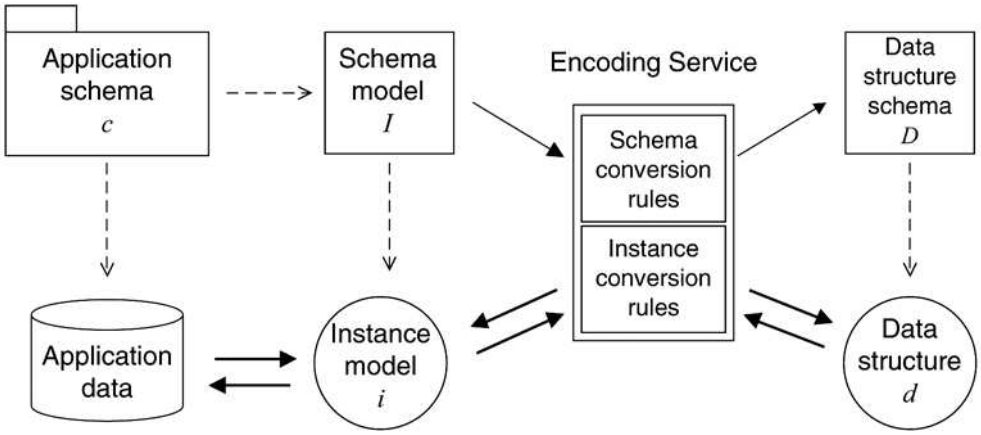

Figure 5 — Conversion rules

The encoding rule shall specify

— schema conversion rules;

— instance conversion rules.

8.6 Examples

Examples are important for understanding the conversion rules and for testing encoding services.

The encoding rule shall provide

— examples that illustrate conversion rules;

— test data that can be used in implementation of encoding services.

9 Encoding service

An encoding service is a software component that implements the encoding rule and provides an interface to its functionality.

An encoding service shall

— provide an interface to its functionality through one or more interface specifications according to ISO 19119;

— have capabilities to encode and decode data according to the instance conversion rules;

— be capable of creating an output data structure schema according to the schema conversion rules.

Figure 6 is an example of an XML based encoding service.

Documents
Document Sharing Hub

Full document details: DIN EN ISO 19118:2006-09

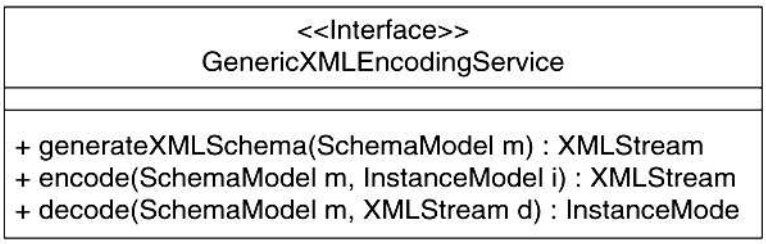

Figure 6 — Example encoding service interface

The interface of the encoding service is shown in Figure 6. It supports three operations: generateXMLSchema, encode and decode.

The "generateXMLSchema" operation can be used to generate an XML Schema file. This operation takes a schema model as input parameter and produces an XMLStream object as result. The "encode" operation can be used to generate an XML document. It takes a schema model and an instance model as input parameter and returns an XMLStream object. The "decode" operation can be used to interpret an XML document. It takes a schema model and an XMLStream object as input and returns an instance model.

Documents
Document Sharing Hub
Full document details: DIN EN ISO 19118:2006-09

Annex A
(informative)

XML based encoding rule [1)]

A.1 Introduction

This annex specifies the XML based encoding rule that shall be used for neutral data interchange. The encoding rule is compatible with the Unified Modelling Language (UML) and defines an encoding rule based on the Extensible Markup Language (XML). The annex follows the requirements in Clause 8 and specifies the

a) general encoding requirements in A.2;

b) input data structure in A.3;

c) output data structure in A.4;

d) conversion rules in A.5 and A.6;

e) examples are given in A.5.8, A.6.4 and Annex E.

The conversion rules are based on the idea that the class definitions in the application schema are mapped to type declarations in XML Schema, and that the objects in the instance model are mapped to corresponding element structures in the XML document. Figure A.1 depicts the two types of conversion rule.

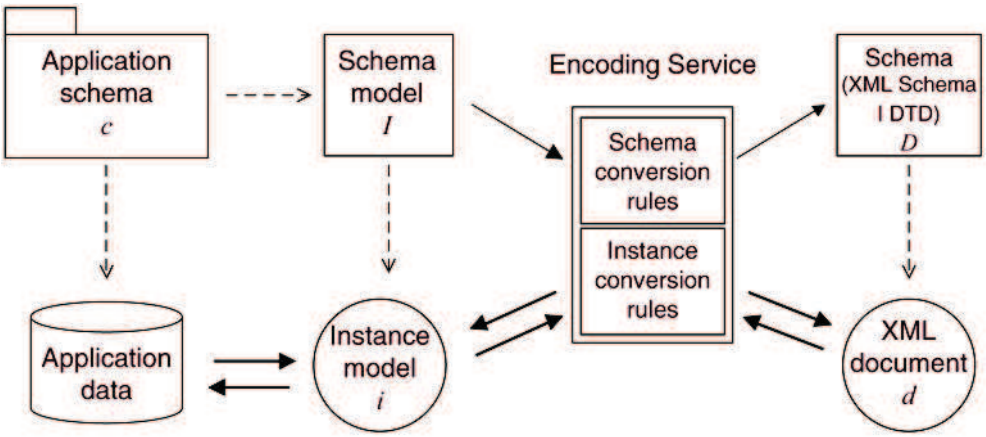

Figure A.1 — XML based conversion rules

NOTE An introduction to XML is given in Annex C.

1) This annex will be revised and made normative in a later amendment, harmonized with ISO 19136.

Documents
Document Sharing Hub

Full document details: DIN EN ISO 19118:2006-09

A.2 General encoding requirements

A.2.1 Application schema

A.2.1.1 Introduction

The application schema shall be expressed in UML schema language according to the rules specified in ISO/TS 19103 and ISO 19109. An application schema consists of application-defined concepts expressed as classes and associations. Some of these classes may be imported from the standardized schemas in other standards.

A.2.1.2 Class

The fundamental modelling concept in UML is the class. It is identified by a name and may have a stereotype. The class may have attributes, operations and constraints and participate in associations. A class defines a type that can be used as a building block to define other classes. Types are the fundamental building blocks out of which all forms of data can be composed and encoded, e.g. numbers, coordinates, text strings, dates, and objects. A type defines the legal value domain and the operations on values of that domain.

Two main categories of types are the simple data types and the complex data types. A type is a simple data type if there is a canonical encoding defined for the type. This canonical encoding may define how to represent values of the type as bits in a memory location or as characters in a textual encoding. Examples of simple types are integer, float and string. A type is of a complex data type when there is no canonical encoding defined for the type. Examples are object types, structured types, records, and collections. A complex type consists of a structured collection of basic and complex attributes that can be encoded using a combination of basic types and special structuring primitives. A third category is the external data types that are defined outside the ISO 19100 series of International Standards.

a) Simple data types – Fundamental types for representing values:

   1) Basic data types: CharacterString, Integer, Binary, Boolean, Date, Time, etc.

   2) Enumerated data types and code lists: A list of legal values, where each value is a word or a code with associated semantics.

b) Complex data types – Types for representing more complex collections of values:

   1) Collection data types: Template types for representing multiple occurrences of other types – Set, Bag, Sequence, Dictionary, etc.

   2) Structured data types: Types that define attribute groups.

   3) Object types: Types whose instances are objects. Often defined in application schemas or standardized schemas: GM_Point, Building, etc.

   4) Interfaces: Types whose instances are service components.

c) External data types – Basic or complex types with a well-defined encoding that are not defined within the ISO 19100 series of International Standards; examples are image formats such as NASA MODIS, TIFF, etc. Special referencing mechanisms shall be specified that allow references, usually stored in separate files, to external data types.

ISO/TS 19103 defines the modelling concepts applied, including a number of simple and complex data types together with their semantics. The application developers are free to specify user defined data types and object types using the stereotype extension mechanism of UML. Other International Standards within the ISO 19100 series of International Standards define more specialized data types. Application schema developers may occasionally need to use external data types. External data types are basic or structured data types that are defined outside this series of International Standards. They must have a well-defined encoding, examples are image formats such as NASA MODIS and TIFF. Objects of external data types can be stored in

Documents
Document Sharing Hub
Full document details: DIN EN ISO 19118:2006-09

separate files and special referencing mechanisms that allow references to external data types shall be specified.

The stereotypes allowed on classes are described in Table A.1.

Table A.1 — Stereotypes on classes

Stereotype	Description
<<BasicType>>	Defines a basic data type that has defined a canonical encoding.
<<DataType>>	Defines a structured data type. The instances are not considered objects and shall therefore not have an identity.
<<Union>>	Defines a union data type.
<<Enumeration>>	Defines an enumerated data type.
<<CodeList>>	Defines an extendable enumerated data type, consisting of code and value pairs.
<<Interface>>	Defines a service interface and shall not be encoded.
<<Type>>	Defines an object type. Instances shall have identity.
NONE	Defines an object type. Instances shall have identity.

An object is considered the fundamental unit of interchange. Only aspects that are essential for capturing an object's state shall be considered for data interchange purposes. Attributes and associations shall be encoded. Operations and constraints shall not be considered further. An encoding rule shall specify how instances of classes are represented, including how attributes and associations are structured and represented.

A.2.1.3 Attributes

An attribute is identified by a name. It may have a multiplicity statement and shall always have a type. The multiplicity statement shall indicate the number of legal value occurrences of a particular attribute.

Combinations of multiplicity and use of collection data types, allows nesting of values.

There shall be mechanisms to handle null values. The type shall define how a null value is represented. See ISO/TS 19103 for the definition of the basic and collection data types.

A.2.1.4 Associations

Associations define relationships between classes that involve connections among their instances. An instantiation of an association is called a link. A link contains an ordered list of references to objects. UML defines three different types of associations called association, aggregation and composition.

— Association defines general relationships between classes.

— Aggregation defines weak part-whole relationships between classes.

— Composition defines strong part-whole relationships between classes.

The three association types have different semantics. ISO/TS 19103 gives further details.

The end of an association is identified by its role name, its target class, and is further described by a multiplicity statement. The role name shall be used to represent a link. If the role name is missing then there will be no link.

Documents
Document Sharing Hub

Full document details: DIN EN ISO 19118:2006-09

A.2.2 Exchange structure

The exchange structure shall be divided into three parts. The first part is the exchange metadata, which shall be described according to 8.2.3. The second part is the dataset that shall contain XML elements that correspond to independent objects. The third part is optional and shall contain an update section of the update primitives as described in 8.2.5.

A.2.3 Character repertoire and language

ISO/IEC 10646 shall be used as the character repertoire of this encoding rule. The underlying exchange structure shall support the language and character repertoire.

A.2.4 Dataset and object identification

The two ways of specifying object identification as described in 8.2.4.2 shall be

— unique dataset identifiers according to XML's ID mechanism;

— universal unique identifiers according to the application domain's specification.

EXAMPLE An application domain may want to use a two component identifier. The first component is the domain name and the next component is an integer instance number. The components can be separated by colon ":". The instance numbers may be encoded in hex. There are no restrictions on the size of the instance number. Examples of two UUIDs in a domain called "example" are: "example:F23C30" and "example:FFFFFF12345A".

A.2.5 Update mechanism

Three update primitives are specified. The users may extend the list of primitives.

A.3 Input data structure

A.3.1 A generic instance model

A generic instance model is defined in this subclause. The instance model is capable of representing data described by application schemas expressed in UML. The instance model represents the application schema specific data structure defined in Clause 6. The instance model consists of a dataset (IM_Dataset) that contains a sequence of objects (IM_Object), where an object consists of a sequence of properties (IM_Property). Each property is encoded according to its data type. The instance model is shown in Figure A.2 and Figure A.3.

The application schema defines a number of classes and their attributes and associations and it is the basis for generating data representations. A data representation (dataset) contains one or more objects that are structured and encoded according to their class definitions. This clause describes the principles of how to represent objects, their attributes and associations between objects.

The basic unit of information in a dataset is the object. An object shall be an instance of a single concrete class. Thus, abstract classes and classes stereotyped as interfaces shall not be encoded. The class shall have a unique name within the application schema. The application schema may refer to or use classes defined in standardized schemas. The declaration of these classes shall either be included in the application schema itself or accompany the application schema as a separate file.

An object shall contain an ordered list of property values. The object's class defines the properties and they can either be inherited through the class' supertypes or defined within the class itself. There shall be no difference between inherited properties and locally declared properties. In order to differentiate between the different properties the properties shall have names that are unique within its class. The property's data type governs the possible values and the multiplicity statement indicates the number of occurrences of the value in an instantiated object.

AVA Documents Document Sharing Hub

Full document details: DIN EN ISO 19118:2006-09

An object has a corresponding class, defined in an application schema, which defines the possible attributes and associations that are necessary to represent the state of the object. An IM_Object refers to its class by the "class" attribute, it shall be identified within the context of a dataset by its unique identifier “id”, and may be universally uniquely identified within a defined universe, application domain or name space, by its “uuid” attribute.

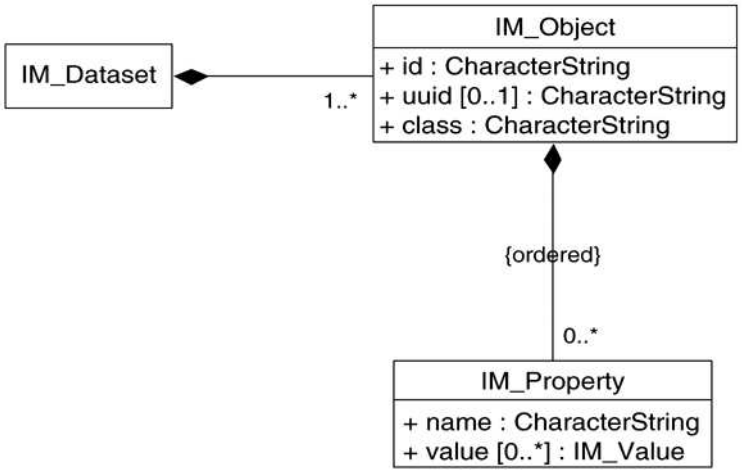

Figure A.2 — Instance model: Dataset, object and property

The attributes and associations defined by the class are mapped to an ordered sequence of properties. A property (IM_Property) represents a name-value pair. It can represent an attribute, association, aggregation or composition. The property name shall correspond to the attribute name or the target role name of an association. A value (IM_Value) represents a property value.

Null values may be given either explicitly or implicitly. An explicit null value shall be indicated by an instance of the corresponding IM_Property with an empty value. An implicit null value is indicated if the corresponding IM_Property instance is missing.

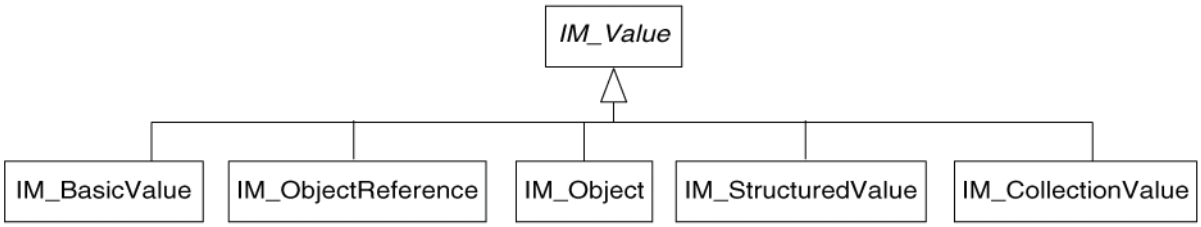

Figure A.3 — Instance model: Value types

The five value types are defined as follows.

— IM_BasicValue represents a value of a basic type. For example an integer or a character string.

— IM_ObjectReference represents a link or reference to a target object. The target object may be located in the same or other datasets. A unique identifier (ID) targets an object located within same dataset. A universal unique identifier (UUID) targets an object located within the context of an application domain.

— IM_Object represents an object.

— IM_StructuredValue represents an attribute group as a sequence of properties (IM_Property).

— IM_CollectionValue represents collection types, such as Set, Sequence, Bag and Dictionary. An IM_CollectionValue represent values of structured types that have no named elements. Its elements may

Documents
Document Sharing Hub
Full document details: DIN EN ISO 19118:2006-09

be of both basic and structured values. Examples of collection structured types are Dictionary<KeyType,ValueType> and Sequence<Sequence<Integer>>.

An object may through its associations be linked (or refer) to one or more objects. UML defines three different types of associations called association, aggregation and composition with different semantics, see ISO/TS 19103 for further details. There are in general two representing strategies for associations; either encapsulated as a part of the objects or detached from the objects as separate association objects or relational tables.

The encapsulated representation strategy splits an association into a source object property and a target object property. These two link properties then point to the other object respectively. The link property contains references to its target objects or in the case of a composition the target objects themselves. A link property is identified by the role name close to the target class and has a corresponding multiplicity. If the role name is missing or if the association is not navigable from the source object then there shall be no link property. The two link property values should be consistent in that the referential integrity constraint is enforced. That is if a source object is referring to another target object through a link property and the target object has a bi-directional association to the source object, the target object shall have a corresponding link property that is referring back to the source object.

A.3.2 Relation between UML and the instance model

Tables A.2 and A.3 give a summary of the relation between UML and the instance model. N/A stands for not applicable. Thus an abstract class shall not be instantiated.

Objects based on classes that have supertypes shall contain all the properties, associations and compositions of their class and of their supertypes. Thus, all attributes and associations shall be copied from the supertypes and are considered to be a part of the object. Attribute and association names shall be the way of accessing the values of the attributes and they shall therefore be unique within the class.

Operations and constraints shall not be mapped to the instance model.

Table A.2 — Summary of relationship between UML and the instance model

UML concept	Instance model
Package	N/A
Class Stereotype <<Interface>> <<BasicType>> <<DataType>> <<Union>> <<Enumeration>> <<CodeList>> <<Type>> or NONE Abstract class subclass	 N/A IM_BasicValue IM_StructuredValue IM_StructuredValue IM_BasicValue IM_BasicValue IM_Object N/A attributes and associations shall be copied from super classes
Attribute	IM_Property with IM_Value according to data type. Either (IM_BasicValue, IM_Object or IM_StructuredValue)
Association	IM_Property with IM_Value of IM_ObjectReference
Aggregation	IM_Property with IM_Value according to either IM_Object or IM_ObjectReference
Composition	IM_Property with IM_Value of IM_Object
Operation	N/A
Constraint	N/A

Documents
Document Sharing Hub
Full document details: DIN EN ISO 19118:2006-09

Table A.3 — Mapping of attributes with multiplicity and collection type

Attribute	Instance model
a1 [0..*] : Integer is the same as a1 : Sequence<Integer> [a]	IM_Property with multiple value occurrences.
a2 : Sequence<T>	IM_Property with multiple occurrences of value type T.
a3 : Dictionary<T1, T2>	IM_Composition with multiple occurrences of IM_Collection with two elements of value type T1 and T2.
a4 [0..*] : Sequence<Integer>	IM_Composition with multiple occurrences of IM_Collection with integer elements.
[a] This also is valid for any other basic type.	

NOTE 1 Attribute and association name clashes can cause problems when using inheritance. A simple way to avoid this is that all attributes and associations shall be prefixed with their appropriate class name; alternatively, the method of avoiding name classes is left to the user.

NOTE 2 Attribute and association redeclaration can also cause problems when using inheritance. Redeclaration happens when an attribute or association declared in a supertype gets redeclared in a subtype with a new or restricted type. Many object-oriented programming languages cannot handle redeclarations and it should be carefully considered whether redeclarations should be deprecated.

A.3.3 Example: Application schema and instance model

A.3.3.1 Application schema

This example defines an application schema, gives some data and shows the mapping from the data into the instance model.

Figure A.4 defines an example application schema, which defines four classes and their relationships. Here class C0 has three attributes. Attribute “a1” has a multiplicity of zero or one, which means it is optional, and a basic data type “Real”. “a2” has a multiplicity of zero or many. Attribute “a3” is of a structured data type. The association between “c0” and “c1” has two role names. “role1” belongs to C0 and names the association to C1 in the context of C0. Class C1 has an association to C0 which is called “main”. Notice that class C2 has an attribute named “pos” with a class data type Point.

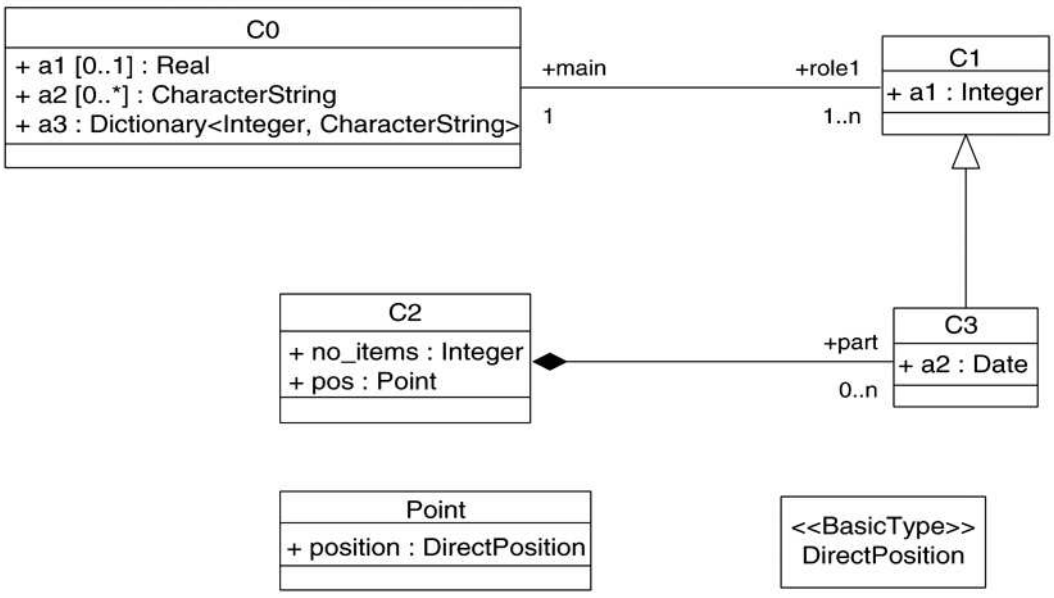

Figure A.4 — Example application schema

Documents
Document Sharing Hub
Full document details: DIN EN ISO 19118:2006-09

A.3.3.2 Example data

The example data is shown in Figure A.5 in an object diagram.

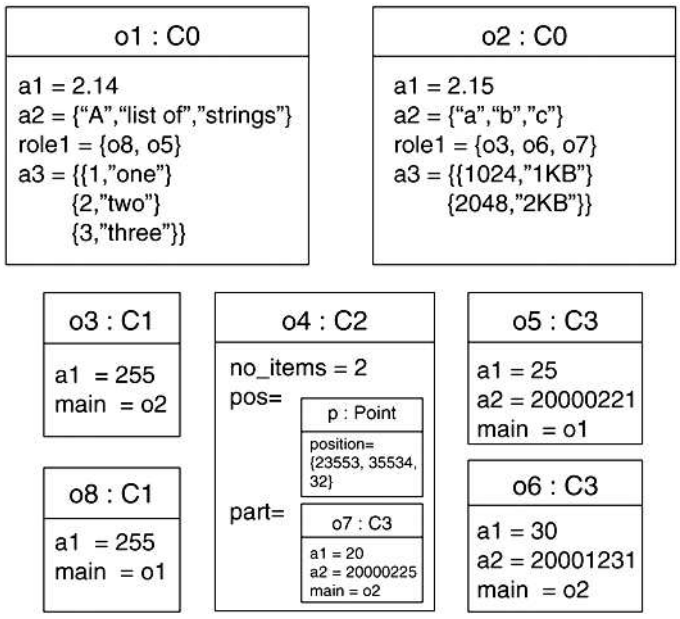

Figure A.5 — Example data

A.4 Output data structure

A.4.1 XML document

This encoding rule is based on the XML Recommendation 1.0. XML is a text format and the values of all data types have to be character encoded. Data shall be encoded using XML elements and the rules given in the XML recommendation. The basic units of encoding in XML are XML elements. An element may have attributes and content. This enables a hierarchical structure and combined with XML's linking facilities a network structure can be created. The exchange structure shall consist of a sequence of elements corresponding to the objects in the instance model.

The schema for the output data structure that governs the structure of the exchange format shall

— be a valid XML Schema.

A.4.2 XML Schema (XSD)

An XML Schema document (XSD) defines a number of complex types, simple types and element declarations that define the allowable structure and data instances of an XML document. The XML Schema conversion rules are defined in A.5.

A.5 Schema conversion rules

A.5.1 XML Schema

The schema conversion rules define how to produce an XML Schema Document (XSD) according to an application schema expressed in UML. The main purpose of the XSD is to ensure that XML documents produced using the data conversion rules are valid.

Documents
Document Sharing Hub
Full document details: DIN EN ISO 19118:2006-09

The XSD shall contain type definitions and attribute and element declarations that correspond to the classes defined in the application schema. The elements shall be organized in an exchange structure. The XSD shall adhere to XML Schema Part 1: Structures and Part 2: Datatypes.

The XSD may physically be represented in a single schema document or divided into several separate (sub) schema documents. Logically it shall be referred to as a single schema utilising the import or include mechanisms of XML Schema. There are no restrictions on the use of namespaces in an XSD.

A number of general rules are defined in the following clauses. A class shall in general be converted to a type definition according to A.5.2, it may be converted to an element declaration according to A.5.4 and it may be a member of the exchange structure.

Exceptions to the general rules are allowed as long as they are documented.

NOTE In the following the namespace "xs:" is used to refer to the namespace of XML Schema, which is "http://www.w3.org/2001/XMLSchema".

A.5.2 Types

A.5.2.1 <<BasicType>>

A.5.2.1.1 General rule

A class stereotyped <<BasicType>> shall be converted to a simpleType declaration in XML Schema. Any of the data types defined in XML Schema can be used as building blocks to define user defined basic types. The encoding of the basic types shall follow the canonical representation defined in XML Schema Part 2: Datatypes.

Modelling rules for basic types are defined in ISO/TS 19103. The basic types defined in ISO/TS 19103 are converted in the following clauses.

NOTE 1 Users are permitted to restrict the basic types further using the restriction mechanisms defined in XML Schema.

NOTE 2 The different types are not clearly defined in ISO/TS 19103 and neither is the <<BasicType>> stereotype used. The following declarations therefore follow a subset of the data type definitions in XML Schema Part 2: Datatypes.

A.5.2.1.2 Number

A Number is in ISO/TS 19103 defined as an abstract type and a supertype of Integer, Real and Decimal. However, it is much used in the different standardized schemas. XML Schema does not define an abstract number type, but defines decimal to be a supertype of integer.

NOTE A number data type is declared as a simple data type based on the decimal data type.

```
<xs:simpleType name="Number">
    <xs:restriction base="xs:decimal"/>
</xs:simpleType>
```

A.5.2.1.3 Integer

An Integer shall be based on the XML Schema integer data type. The value domain may be restricted.

```
<xs:simpleType name="Integer">
    <xs:restriction base="xs:integer"/>
</xs:simpleType>
```


Documents
Document Sharing Hub

Full document details: DIN EN ISO 19118:2006-09

A.5.2.1.4 Decimal and Real

The Decimal and Real types are both based on the XML Schema decimal type.

```
<xs:simpleType name="Decimal">
    <xs:restriction base="xs:decimal"/>
</xs:simpleType>

<xs:simpleType name="Real">
    <xs:restriction base="xs:decimal"/>
</xs:simpleType>
```

NOTE ISO/TS 19103 does not define the conceptual difference between a Decimal and a Real.

A.5.2.1.5 Vector

A Vector is defined as a sequence of numbers. The list construct defines a list of decimal values.

```
<xs:simpleType name="Vector">
    <xs:list itemType="xs:decimal"/>
</xs:simpleType>
```

A.5.2.1.6 Character

A Character is represented as an XML Schema string restricted to contain only one character.

```
<xs:simpleType name="Character">
    <xs:restriction base="xs:string">
        <xs:length value="1" fixed="true"/>
    </xs:restriction>
</xs:simpleType>
```

A.5.2.1.7 CharacterString

CharacterString is based on XML Schema string which can represent any ISO/IEC 10646 string.

```
<xs:simpleType name="CharacterString">
    <xs:restriction base="xs:string"/>
</xs:simpleType>
```

NOTE For language identification see A.5.6.

A.5.2.1.8 Date

Date is based on XML Schema date which has a canonical encoding according to ISO 8601.

```
<xs:simpleType name="Date">
    <xs:restriction base="xs:date"/>
</xs:simpleType>
```


Documents
Document Sharing Hub
Full document details: DIN EN ISO 19118:2006-09