
**Industrial automation systems and
integration — Product data
representation and exchange —**

**Part 28:
Implementation methods: XML
representations of EXPRESS schemas
and data, using XML schemas**

*Systèmes d'automatisation industrielle et intégration — Représentation
et échange de données de produits —*

*Partie 28: Méthodes d'implémentation: représentations XML de
schémas et de données EXPRESS en utilisant des schémas XML*



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.



COPYRIGHT PROTECTED DOCUMENT

© ISO 2007

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

Contents	Page
1 Scope.....	1
2 Normative references.....	1
3 Terms, definitions, abbreviations, and conventions	2
3.1 Terms defined in ISO 10303-1.....	2
3.2 Terms defined in ISO 10303-11.....	2
3.3 Terms defined in the XML Standards.....	4
3.4 Other terms and definitions.....	5
3.5 Conflicting terminology.....	7
3.6 Abbreviations.....	7
3.7 Conventions	7
3.7.1 Text conventions.....	8
3.7.2 Namespace conventions.....	8
4 Conformance.....	8
4.1 Conformance of an XML document	9
4.1.1 Conformance of an iso-10303-28 document.....	9
4.1.2 Conformance of a uos document	10
4.1.3 Conformance of a configured document	10
4.2 Conformance of a derived XML schema document	10
4.3 Conformance of a configuration file.....	11
4.4 Conformance of a pre-processor.....	11
4.5 Conformance of a post-processor.....	12
4.6 Conformance of an XML schema generator.....	12
5 Document level elements.....	12
5.1 The iso-10303-28 document	13
5.2 Document and uos header information	14
5.2.1 The exp:header element.....	14
5.2.2 The name element.....	15
5.2.3 The time_stamp element.....	15
5.2.4 The author element	15
5.2.5 The organization element.....	15
5.2.6 The authorization element	15
5.2.7 The originating_system element.....	15
5.2.8 The preprocessor_version element	15
5.3 The schema_population element.....	15
5.4 The express element.....	17
5.4.1 By-reference representation of an EXPRESS schema.....	18
5.4.2 By-value representation of an EXPRESS schema.....	18
5.5 The configuration element.....	18
5.6 The unit of serialization element.....	18
5.7 The uos document	20
5.8 The configured document	20
5.9 Enterprise data objects	20
6 Derived XML Schema.....	20
6.1 Preconditions.....	21
6.2 Unmapped EXPRESS concepts	21
6.3 Abstract entity data types.....	21

7	Default XML Schema Binding	22
7.1	Naming conventions	22
7.1.1	Schema.....	22
7.1.2	EXPRESS identifiers	22
7.1.3	Data types	22
7.2	XML Schema data types corresponding to EXPRESS data types	22
7.2.1	EXPRESS simple data types.....	23
7.2.2	Aggregation data types	30
7.2.3	Constructed data types	44
7.2.4	Defined data types	45
7.2.5	ENTITY data types.....	45
7.3	XML Schema definitions and declarations for EXPRESS defined data types	45
7.3.1	Simple underlying types	46
7.3.2	Aggregate underlying types.....	47
7.3.3	ENUMERATION underlying types	48
7.3.4	SELECT underlying types.....	49
7.3.5	Defined data type underlying type.....	53
7.4	Instance elements corresponding to EXPRESS data types	54
7.4.1	Instance elements for simple data types.....	55
7.4.2	Instance elements for anonymous aggregation data types	59
7.4.3	Instance elements for defined data types	62
7.4.4	Instance elements for entity data types	63
7.4.5	Instance element attributes.....	63
7.4.6	Referenceable instances.....	64
7.5	XML Schema definitions and declarations for EXPRESS entity data types	64
7.5.1	Type graph associated with the EXPRESS entity data type	65
7.5.2	Complex entity instances.....	66
7.5.3	Base XML data types and elements for EXPRESS entity data types.....	66
7.5.4	XML data type definitions for entity data types	69
7.5.5	Instance elements corresponding to entity data types.....	71
7.5.6	XML groups corresponding to entity data types.....	72
7.5.7	Single entity value elements corresponding to entity data types	75
7.5.8	Proxy elements corresponding to entity data types.....	77
7.5.9	XML Uniqueness constraints for entity data types.....	78
7.6	XML Schema declarations for EXPRESS attributes	79
7.6.1	Accessor element and attribute naming	79
7.6.2	EXPRESS attributes mapped to XML schema	80
7.6.3	Accessor elements	81
7.7	XML Schema and namespaces for EXPRESS Schema	86
7.7.1	Namespace prefixes	86
7.7.2	URI for the target namespace of the derived XML schema	87
7.7.3	Namespace declarations for the derived XML schema	88
7.7.4	Import declarations for the derived XML schema.....	88
7.8	Context-schema specific unit of serialization	89
8	Configured XML Schema Binding.....	91
8.1	Naming conventions	91
8.1.1	Schema.....	91
8.1.2	EXPRESS identifiers	91
8.1.3	Data types	91
8.2	XML Schema data types corresponding to EXPRESS data types	92
8.2.1	EXPRESS simple data types.....	92
8.2.2	Aggregation data types	97
8.2.3	Constructed data types	108
8.2.4	Defined data types	108

8.2.5	ENTITY data types.....	108
8.3	XML Schema definitions and declarations for EXPRESS defined data types	108
8.3.1	Simple underlying types	109
8.3.2	Aggregate underlying types.....	111
8.3.3	ENUMERATION underlying types	112
8.3.4	SELECT underlying types.....	112
8.3.5	Defined data type underlying type.....	115
8.3.6	Defined data types mapped by map configuration directive.....	116
8.4	Instance elements corresponding to EXPRESS data types	116
8.4.1	Instance elements for simple data types.....	117
8.4.2	Instance elements for anonymous aggregation data types	120
8.4.3	Instance elements for defined data types	124
8.4.4	Instance elements for entity data types	124
8.4.5	Instance element attributes.....	124
8.4.6	XML identity-constraints for instance elements.....	125
8.4.7	Referenceable instances.....	128
8.5	XML Schema definitions and declarations for EXPRESS entity data types	129
8.5.1	Type graph associated with the EXPRESS entity data type	131
8.5.2	Complex entity instances.....	131
8.5.3	Base XML data types and elements for EXPRESS entity data types.....	132
8.5.4	XML data type definitions for entity data types	134
8.5.5	Instance elements corresponding to entity data types.....	146
8.5.6	XML groups corresponding to entity data types.....	147
8.5.7	Single entity value elements corresponding to entity data types	149
8.5.8	Proxy elements corresponding to entity data types.....	150
8.5.9	XML Identity constraints corresponding to entity data types.....	151
8.5.10	XML Uniqueness constraints for entity data types.....	154
8.5.11	Dynamic subtype elements corresponding to entity data types	155
8.6	XML Schema declarations for EXPRESS attributes	156
8.6.1	Accessor element and attribute naming	157
8.6.2	EXPRESS attributes mapped to XML schema.....	157
8.6.3	Accessor attributes	159
8.6.4	Accessor elements	163
8.6.5	Type-tagged attributes	170
8.6.6	No-tag attributes	173
8.7	XML Schema and namespaces for EXPRESS Schema	174
8.7.1	Namespace prefixes	175
8.7.2	URI for the target namespace of the derived XML schema	175
8.7.3	Namespace declarations for the derived XML schema	175
8.7.4	Import declarations for the derived XML schema.....	175
8.8	Context-schema specific unit of serialization	176
9	XML document creation.....	177
9.1	Preconditions.....	177
9.2	General XML document structure	177
9.2.1	Structure of an iso-10303-28 document.....	178
9.2.2	Structure of a uos document	178
9.2.3	Encoding of EXPRESS schemas	179
9.2.4	Encoding of configuration files	179
9.2.5	Encoding of population definitions.....	180
9.2.6	Encoding of data sets – the unit of serialization	180
9.3	Representation of EXPRESS entity instances.....	183
9.3.1	By-value representation of entity instances	184
9.3.2	External representation of EXPRESS entity instances	187
9.3.3	By-reference representation of EXPRESS entity instances.....	189

9.3.4	Complex entity representation of EXPRESS entity instances	190
9.4	Representation of an EXPRESS attribute	192
9.4.1	Determining by-reference or by-value representation	192
9.4.2	Representation of EXPRESS attribute value as accessor attribute	193
9.4.3	Attribute-tag representation of EXPRESS attribute value	194
9.4.4	Double-tag representation of EXPRESS attribute value	196
9.4.5	Type tag representation of EXPRESS attribute value	197
9.4.6	No-tag representation of entity instance as EXPRESS attribute value	198
9.4.7	No-tag-simple representation of entity instance as EXPRESS attribute value	198
9.5	Representation of simple values	198
9.5.1	Representation of BINARY values	199
9.5.2	Representation of BOOLEAN values	199
9.5.3	Representation of INTEGER values	199
9.5.4	Representation of LOGICAL values	200
9.5.5	Representation of NUMBER values	201
9.5.6	Representation of REAL values	201
9.5.7	Representation of STRING values	202
9.6	Representation of enumeration items	203
9.7	Representation of values of SELECT types	204
9.8	Representation of aggregate values	206
9.8.1	List-of-values representation of aggregate values	207
9.8.2	Sequence-of-elements representation of aggregate values	209
9.8.3	Indexed representation of aggregate values	210
9.8.4	List-of-references representation of aggregate values	211
9.8.5	Aggregates of aggregate values	212
9.8.6	Aggregates of values of defined data types	219
9.8.7	Instance elements for component values	219
9.9	Representation of values of defined data types	220
9.10	Representation of values in instance elements	221
9.10.1	By-value instance elements for non-entity data types	222
9.10.2	By-reference instance elements for non-entity data types	223
10	Configuration Language	223
10.1	The configuration element	225
10.1.1	By-reference representation of a configuration file	226
10.1.2	By-value representation of a configuration file	226
10.2	Configuration options	226
10.2.1	name	227
10.2.2	exp-type	227
10.2.3	content	227
10.2.4	aggregate-content	228
10.2.5	exp-attribute	228
10.2.6	entity-attribute	229
10.2.7	concrete-attribute	229
10.2.8	tagless	229
10.2.9	flatten	230
10.2.10	use-id	230
10.2.11	keep	231
10.2.12	keep-all	231
10.2.13	map	232
10.2.14	naming-convention	234
10.2.15	inheritance	234
10.2.16	notation	234
10.2.17	tag-source and tag-value	234
10.2.18	namespace	235

10.2.19	ref.....	236
10.2.20	use.....	236
10.2.21	implementation.....	236
10.2.22	facet.....	237
10.2.23	generate-keys.....	237
10.2.24	embed-schema-items.....	238
10.2.25	alias and prefix.....	238
10.2.26	select.....	238
10.3	Scoping elements.....	239
10.3.1	Option element.....	240
10.3.2	Type element.....	240
10.3.3	Entity element.....	241
10.3.4	Attribute element.....	246
10.3.5	Inverse element.....	248
10.3.6	Aggregate element.....	250
10.3.7	Schema element.....	251
10.3.8	UosElement element.....	255
10.3.9	UosEntity element.....	255
10.3.10	RootEntity element.....	255
10.4	Configuration attributes.....	256
10.5	Applicability of configuration directives.....	257
10.5.1	exp-attribute.....	257
10.5.2	content and use-id.....	259
10.5.3	exp-type.....	260
10.5.4	map.....	260
10.5.5	tagless.....	261
10.5.6	flatten.....	261
10.5.7	inheritance.....	262
10.5.8	notation.....	262
10.5.9	keep.....	262
10.5.10	ref.....	262
10.5.11	use.....	262
10.5.12	implementation.....	263
10.5.13	facet.....	263
Annex A (normative)	Universal Resource Names for bindings of EXPRESS schemas.....	264
Annex B (normative)	XML Schema for the configuration language.....	265
Annex C (normative)	Base XML Schema.....	272
Annex D (normative)	Document Schema.....	280
Annex E (normative)	Valid populations of EXPRESS entity instances.....	291
Annex F (normative)	Information object registration.....	302
Annex G (informative)	Configuration language examples.....	303
Bibliography.....		307
Index.....		308

Figures

Figure 1 -	Choice group.....	73
Figure 2 -	Choice group for inheritance mapping	148

Tables

Table 1	— Namespace prefixes.....	8
Table 2	— Subclause governing aggregation data type correspondence	30
Table 3	— Subclause governing aggregation data type correspondence	99
Table 4	— Instance elements for <code>STRING</code> data types mapped to XML data types.....	120
Table 5	— XML key names for anonymous <code>EXPRESS</code> data types.....	127
Table 6	— Representation of <code>EXPRESS</code> characters invalid in XML normalizedString.....	203
Table 7	— Subclause governing XML representation of aggregate value	207
Table 8	— Subclause governing XML representation of aggregates of aggregate values .	213
Table 9	— Pattern strings for select	239

Foreword

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

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

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

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

ISO 10303-28 was prepared by Technical Committee ISO TC184, *Industrial automation systems and integration*, Subcommittee SC4 *Industrial data*.

ISO 10303-28 constitutes a technical revision of ISO/TS 10303-28:2003, which is provisionally retained in order to support continued use and maintenance of implementations based on it, and to satisfy the normative references of other parts of ISO 10303.

ISO 10303 is organized as a series of parts, each published separately. The structure of ISO 10303 is described in ISO 10303-1.

Each part of ISO 10303 is a member of one of the following series: description methods, implementation methods, conformance testing methodology and framework, integrated generic resources, integrated application resources, application protocols, abstract test suites, application interpreted constructs, and application modules. This part of ISO 10303 is a member of the implementation methods series.

A complete list of parts of ISO 10303 is available from the following URL:

http://www.tc184-sc4.org/titles/STEP_Titles.htm

Introduction

ISO 10303 is an International Standard for the computer-interpretable representation of product information and for the exchange of product data. The objective is to provide a neutral mechanism capable of describing products throughout their life cycle. This mechanism is suitable not only for neutral file exchange, but also as a basis for implementing and sharing product databases and as a basis for archiving.

This part of ISO 10303 is a member of the implementation methods series. This part of ISO 10303 specifies means by which schemas specified using the EXPRESS language (defined in ISO 10303-11) and data governed by EXPRESS schemas can be represented as an XML document. This enables product data described in EXPRESS to be exchanged using XML and the many software tools developed to support XML technologies. It also permits product data sets so described to be readily incorporated into "electronic commerce" transactions represented in XML.

Readers of this part of ISO 10303 should have knowledge of the EXPRESS language, the XML Schema language, XML, and XML-related standards in order to understand its technical content.

For the representation of data corresponding to an EXPRESS schema, this part of ISO 10303 formally specifies the structure of conforming exchange documents using the XML Schema language. Some elements of those documents represent data sets conforming to EXPRESS schemas, and this part of ISO 10303 specifies the structure of those elements using XML Schema type definitions and element declarations that are derived from the EXPRESS schema declarations. This part of ISO 10303 also specifies the rules for encoding conforming data in XML to match the derived XML schema. In order to accommodate a number of conflicting requirements for the use of conforming exchange documents, this part of ISO 10303 also defines certain configuration directives that can be used to specify alternative structures in the derived XML schema and alternative encoding rules.

Several components of this part of ISO 10303 are available in electronic form. This access is provided through the specification of Universal Resource Locators (URLs) that identify the location of these files on the Internet. If there is difficulty accessing these files contact the ISO Central Secretariat, or contact the ISO TC 184/SC4 Secretariat directly at: sc4sec@tc184-sc4.org.

This part of ISO 10303 constitutes a technical revision of ISO/TS 10303-28:2003, which is provisionally retained to allow for the implementation of the Part 28 late binding. ISO/TS 10303-28:2003 was intended for trial use with emerging XML technologies, and although the fundamental capabilities remain the same, the underlying XML technologies have progressed, and this part of ISO 10303 uses technologies and features that were formerly unavailable or differently provided. This part of ISO 10303 therefore, is not "upwardly compatible" with ISO/TS 10303-28:2003. Neither a document nor a processor that conformed to ISO/TS 10303-28:2003 will conform to the specifications of this part of ISO 10303 without substantial modification.

The major technical changes from ISO/TS 10303-28:2003 are:

- This part of ISO 10303 specifies the structure of XML exchange documents using XML Schema, and specifies the mapping of the EXPRESS data model to an XML Schema data model. ISO/TS 10303-28:2003 specified the structure using Document Type Definitions (DTDs). It is expected that most future XML documents will be validated against an XML schema, instead of a DTD.

- The ISO/TS 10303-28:2003 mapping of EXPRESS schema text into a complex XML structure is no longer included in this part of ISO 10303; representation of an EXPRESS schema as a body of text is the only form retained.

- The ISO/TS 10303-28:2003 mapping of EXPRESS-modelled data sets designated the "late-binding" is not included in this part of ISO 10303.

— The ISO/TS 10303-28:2003 mapping of EXPRESS-modelled data sets designated the "ETEB" is not included as such. The major features of that binding, as they appear in the XML form of the data, are included in the "default mapping" in this part of ISO 10303. But this part of ISO 10303 makes use of an XML Schema feature that was not available in DTDs — context names — to simplify the names for the "accessor elements".

— The ISO/TS 10303-28:2003 mapping of EXPRESS-modelled data sets designated the "OSEB" is not included as such. The major features of that binding, as they appear in the XML form of the data, are included in the "attribute-content" mapping in this part of ISO 10303. But in this part of ISO 10303, there are several changes to the representation structures for aggregation data types and SELECT data types that provide for representation options and consistency across them.

— This part of ISO 10303 permits the use of XML Schema "simple list" structures to represent many values of aggregation data types, a feature that is not standardized in XML 1.0 and could not be specified in a DTD.

— Many separate options for re-configuring the EXPRESS schema, configuring the XML schema , and configuring the XML data are now supported by a configuration language, and support for that language is a mandatory feature of conforming processors.

Warning:

This part of ISO 10303 provides a specification intended to be implemented in software. Incompatibilities may result in machine-to-machine communication in the case of software developed on the basis of translations of this part of ISO 10303 into languages other than the official ISO languages. It is accordingly strongly recommended that any implementations be developed only on the basis of the texts in the official ISO languages.

Industrial automation systems and integration — Product data representation and exchange —

Part 28:

Implementation methods: XML representations of EXPRESS schemas and data, using XML schemas

1 Scope

This part of ISO 10303 specifies use of the Extensible Markup Language (XML) to represent schemas specified using the EXPRESS data specification language, ISO 10303-11, and data that is governed by EXPRESS schemas. This part of ISO 10303 formally specifies the XML representation by specifying an overall XML schema for the exchange document and additional XML schemas that correspond to the EXPRESS schemas that govern the exchange data sets.

The following are within the scope of this part of ISO 10303:

- specification of the form of XML documents containing EXPRESS schemas and data governed by EXPRESS schemas (see Clause 5);
- for an arbitrary EXPRESS schema, specification of an XML schema that corresponds to the EXPRESS schema and formally describes the XML representation of data governed by that schema (see Clause 6);
- specification of the representation of values of EXPRESS data types as XML element content and as XML attribute values (see Clause 9);
- specification of the set of configuration directives that may be used to specify options for the structure of the XML representation of data sets that conform to EXPRESS schemas (see Clause 10).

The following are outside the scope of this part of ISO 10303:

- specification of XML Schema declarations or definitions that depend on the semantic intent, as distinct from the EXPRESS language statements, of any particular EXPRESS schema;
- specification of mappings from the XML Schema language to the EXPRESS language;
- specification of the mapping to an EXPRESS schema from an XML schema that has been derived from an EXPRESS schema.

2 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 8824-1:2002, *Information technology – Abstract Syntax Notation One (ASN.1) – Part 1: Specification of basic notation*

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

ISO 10303-11:2004, *Industrial automation systems and integration – Product data representation and exchange – Part 11: Description methods: The EXPRESS language reference manual*

ISO 10303-22:1998, *Industrial automation systems and integration – Product data representation and exchange – Part 22: Implementation methods: Standard data access interface*

ISO 639-1:2002, *Codes for the representation of names of languages – Part 1: Alpha 2 code*

ISO 3166-1:2006, *Codes for the representation of names of countries and their subdivisions – Part 1: Country codes*

Uniform Resource Identifiers (URI): Generic Syntax. Internet Engineering Task Force RFC 2396 August 1998 [cited 2004-03-15]. Available from World Wide Web:
<<http://www.ietf.org/rfc/rfc2396.txt>>

Extensible Markup Language (XML) 1.0. World Wide Web Consortium Recommendation 4 February 2004 [cited 2004-03-15]. Available from World Wide Web:
<<http://www.w3.org/TR/REC-xml>>

Namespaces in XML. World Wide Web Consortium Recommendation 14 January 1999 [cited 2004-03-15]. Available from World Wide Web:
<www.w3.org/TR/REC-xml-names/>

XML Schema Part 1: Structures. World Wide Web Consortium Recommendation, 2 May 2001 [cited 2004-03-15]. Available from World Wide Web:
<<http://www.w3.org/TR/xmlschema-1/>>

XML Schema Part 2: Datatypes. World Wide Web Consortium Recommendation, 2 May 2001 [cited 2004-03-15]. Available from World Wide Web:
<<http://www.w3.org/TR/xmlschema-2/>>

Xpointer Framework Version 1.0. World Wide Web Consortium Recommendation 25 March 2003 [cited 2004-03-15]. Available from World Wide Web:
<<http://www.w3.org/TR/xptr-framework/>>

3 Terms, definitions, abbreviations, and conventions

3.1 Terms defined in ISO 10303-1

For the purposes of this document, the following terms defined in ISO 10303-1 apply.

- data;
- information.

3.2 Terms defined in ISO 10303-11

For the purposes of this document, the following terms defined in ISO 10303-11 apply.

3.2.1

data type

domain of values

NOTE Because two standards for specification of data types are used extensively in this part of ISO 10303, the term *data type* is always prefixed by "EXPRESS" or "XML Schema", to indicate the context in which each usage is to be understood.

3.2.2

EXPRESS attribute

property of an EXPRESS entity instance that is represented by a value of an EXPRESS data type and a name that indicates the role that value plays in characterizing the instance

3.2.3

EXPRESS data type

data type specified in the syntax of the EXPRESS language

3.2.4

EXPRESS entity instance ; entity instance

named unit of data that represents a unit of information within the domain defined by an entity data type

3.2.5

EXPRESS language element

concept in the EXPRESS language, and by extension, its syntactic representation

NOTE In general, the term "element" is used in this part of ISO 10303 to refer to the fundamental syntactic component of XML data structures.

3.2.6

fundamental type

EXPRESS data type used to determine the representation of values of a defined data type

NOTE Because a defined data type can be defined in terms of another defined data type, "fundamental type" is formally defined recursively as: "The fundamental type of a defined type is the fundamental type of the underlying type, and the fundamental type of a data type other than a defined type is the data type itself."

3.2.7

generalized data type

EXPRESS data type that is used to specify a generalization of certain other data types, and which can only be used in certain very specific contexts

3.2.8

independent entity instance

EXPRESS entity instance that appears in a schema instance and need not play a role in characterizing some other entity instance in the schema instance

NOTE A *dependent entity* instance is one that appears in a schema instance *because* it is (a component of) the value of an attribute of some other entity instance. An *independent entity* instance is one that does not have that rationale, although it may play such a role.

3.2.9

schema instance

set of EXPRESS entity instances, grouped for some purpose, that is governed by a single EXPRESS schema

3.2.10

underlying type

domain of values of a defined data type, as specified by the syntactic object `underlying_type` in the EXPRESS type declaration for the defined data type (cf. fundamental type)

NOTE Adapted from ISO 10303-11:2004.

3.3 Terms defined in the XML Standards

For the purposes of this document, the following terms and definitions apply. Terms defined in Recommendations of the World Wide Web Consortium and the Internet Engineering Task Force (IETF) are repeated below for convenience.

NOTE Definitions copied verbatim from other standards are followed by a reference to the source standard in brackets. Definitions that have been adapted from other standards are followed by an explanatory note.

3.3.1

element; XML element

bounded component of the logical structure of an XML document that has a type and that may have XML attributes and content [*XML 1.0 Recommendation*]

3.3.2

infoset

abstract data set providing a consistent set of definitions for use in specifications that need to refer to the information in an XML document [*XML Schema Part 1 – Structures Recommendation*]

3.3.3

qualified name

complete name of an XML element, attribute or data type, including the local name and a prefix that identifies the namespace in which the local name is defined/declared [*Namespaces in XML Recommendation*]

3.3.4

Uniform Resource Identifier (URI)

compact string of characters for identifying an abstract or physical resource [IETF RFC 2396: *Uniform Resource Identifiers (URI): Generic Syntax*]

3.3.5

XML attribute

name/value pair associated with an XML element [*XML 1.0 Recommendation*]

3.3.6

XML document

(text) data object that conforms to the XML requirements for being well-formed [*XML 1.0 Recommendation*]

3.3.7

XML schema

body of declarations and definitions in the XML Schema language that specify the structure of a class of XML documents [*XML Schema Part 1 -- Structures Recommendation*]

NOTE "schema" is the XML Schema term that denotes a body of declarations and definitions. But, to distinguish the XML Schema term "schema" from the EXPRESS term "schema", this part of ISO 10303 prefixes the XML Schema term with "XML" (see 3.5). Unfortunately, that means that the case of the the word "schema" is all that distinguishes the name of the XML Schema language from the "schema" concept. This part of ISO 10303, however, deals with two XML schemas in particular, and the text generally uses the specific terms "derived XML schema" and "Base XML Schema" to refer to them.

3.3.8**XML Schema component**

concept in the XML Schema language, and by extension, its syntactic representation [*XML Schema Part 1 -- Structures Recommendation*]

NOTE In general, the term "component" is used in this part of ISO 10303 to refer to the values that are members of the collection of values constituting an aggregate value.

3.3.9**XML Schema data type; XML data type**

data type specified in the syntax of the XML schema language [*XML Schema Part 2 -- Data types Recommendation*]

3.3.10**XML Schema language; XML Schema**

formal language used to specify the structure of XML documents, specified in the *XML Schema Part 1 -- Structures Recommendation*

3.4 Other terms and definitions

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

3.4.1**accessor attribute**

XML attribute (of an XML element representing an entity instance) that represents an EXPRESS attribute and its value (cf. accessor element)

3.4.2**accessor element**

XML element that represents an EXPRESS attribute and its value (cf. accessor attribute)

3.4.3**aggregate value**

value of an EXPRESS aggregation data type, or a value of a defined data type whose fundamental type is an EXPRESS aggregation data type

3.4.4**base-type**

for an EXPRESS aggregation data type, data type of the component values

3.4.5**by-reference instance element**

instance element that has no XML content but includes a reference to a by-value instance element whose content is a representation of the value

3.4.6**by-value instance element**

instance element whose content is the representation of a data value, or the complete representation of an EXPRESS entity instance

3.4.7**characterized entity instance**

EXPRESS entity instance for which there is one EXPRESS entity data type in the context schema that is declared to have or inherit all of the properties that are present in the instance

3.4.8

context schema; governing schema

for a `uos` element, EXPRESS schema that contains the model of the data contained in that element

3.4.9

deepest base-type

for an aggregation data type that contains nested aggregation data types, base-type at some level of nesting whose values are used as the component values in the XML representation of the (possibly multi-dimensional) aggregate values

3.4.10

deepest underlying type

for an aggregation data type that contains nested aggregation data types, first encountered base-type at some level of nesting that is not an anonymous aggregation type, a defined data type whose underlying type is an aggregation type, or a SELECT type whose working select list contains a single defined data type whose underlying type is an aggregation type

3.4.11

default mapping

for a given EXPRESS Schema, XML schema that is derived as specified in this Part of ISO 10303, using the default values of all configuration directives and no additional configuration directives

3.4.12

derived XML schema

XML schema derived from the EXPRESS schema corresponding to the given EXPRESS schema and the given configuration file

3.4.13

independent element

instance element that appears as an immediate child of the `uos` element

3.4.14

instance element

XML element that represents a value of an EXPRESS data type

NOTE An instance element may be a by-value instance element, representing the value directly, or a by-reference instance element, which does not contain a representation of the value, but refers to a by-value element that does.

3.4.15

list-of-values

XML representation of an aggregate value that consists of a single character string in which the encodings of the component values appear in order, separated by SPACE characters (cf. sequence-of-elements)

3.4.16

nested aggregation data type

EXPRESS aggregation data type that appears as the base-type of another aggregation data type

3.4.17

sequence-of-elements

XML representation of an aggregate value that consists of a sequence of XML elements in which the content of each element is the encoding of a single component value (cf. list-of-values)

3.4.18**uncharacterized entity instance**

entity instance for which no single entity data type in the context schema has all the properties needed to represent the entity instance

3.4.19**unit of serialization; uos**

XML representation of a schema instance — a collection of EXPRESS entity instances, together with all of their attribute values, that are described by a single exchange schema

3.5 Conflicting terminology

EXPRESS and XML (Schema) use similar or identical words for related concepts. In this part of ISO 10303, the qualifiers "XML" and "EXPRESS" are used to distinguish between the different contexts. These qualifiers are used with the following terms:

- attribute;
- data type;
- element;
- entity;
- schema.

Exception: The term "element" always refers to an XML "element", except when it is preceded by the qualifier "EXPRESS". In many cases, the word "element" is part of a term denoting a specific type of XML element that is defined in this part of ISO 10303.

NOTE The XML "entity" component is not used in this edition of this part of ISO 10303. The term "entity" always refers to an EXPRESS entity data type or entity instance, and the qualifier "EXPRESS" always appears on the first use in a paragraph.

3.6 Abbreviations

For the purposes of this document, the following abbreviations apply.

IANA	Internet Assigned Names Authority
IETF	Internet Engineering Task Force
MIME	Multi-purpose Internet Mail Extension
URI	Uniform Resource Identifier
URN	Uniform Resource Name
XML	Extensible Markup Language

3.7 Conventions

The following conventions are used in this document.

3.7.1 Text conventions

- EXPRESS language keywords are represented in SMALL CAPITALS;
- Verbatim XML Schema is represented by text in a Courier font;
- XML Schema parameters are represented by ***Courier*** text;
- Configuration directives are represented by **bold** text;
- Terms are highlighted by *italic* font where they are introduced and defined in the text.

When the ellipsis character "..." appears in normative text, it represents all the other members of the list.

When the ellipsis character "..." appears in an example, it indicates that the (EXPRESS or XML) schema component is incomplete.

3.7.2 Namespace conventions

For the sake of clarity, the namespace prefixes in Table 1 are used throughout this part of ISO 10303 to refer to the namespaces identified by the corresponding URIs in Table 1.

NOTE There is no requirement for conforming documents to use these prefixes for these namespaces, although the use of `xs:` and `xsi:` for the XML standard namespaces is recommended.

Table 1 — Namespace prefixes

Prefix	Associated namespace URI
<code>xs:</code>	<code>http://www.w3.org/2001/XMLSchema</code>
<code>xsi:</code>	<code>http://www.w3.org/2001/XMLSchema-instance</code>
<code>exp:</code>	<code>urn:iso:std:iso:10303:-28:ed-2:tech:XMLschema:common</code> (as defined in Annex C)
<code>cnf:</code>	<code>urn:iso:std:iso:10303:-28:ed-2:tech:XMLschema:configuration_language</code> (as defined in Annex B)
<code>doc:</code>	<code>urn:iso:std:iso:10303:-28:ed-2:tech:XMLschema:document</code> (as defined in Annex D)
<code>Tns:</code>	the target namespace associated with the EXPRESS context schema (see 7.7 or 8.7)

4 Conformance

This clause specifies the requirements that must be satisfied by an XML document or a document processor to claim conformance to this part of ISO 10303. Conformance is defined for the following items:

- an XML instance document;

- an XML schema derived from an EXPRESS schema;
- a configuration file;
- a pre-processor that produces XML instance documents;
- a post-processor that accepts XML instance documents;
- a processor that produces an XML schema from an EXPRESS schema.

4.1 Conformance of an XML document

An *XML instance document* is an XML document that conveys EXPRESS-defined data. There are three classes of XML instance documents that can conform to this part of ISO 10303:

- iso-10303-28 documents, as specified in 4.1.1,
- uos documents, as specified in 4.1.2, and
- configured documents, as specified in 4.1.2.

For the purposes of this subclause, the XML information set (infoset) for a document shall be considered *valid* if the validity property for each element and for each attribute in the document's post-schema-validation infoset is "valid", as defined in *XML schema Part 1: Structures*.

4.1.1 Conformance of an iso-10303-28 document

An XML document shall *conform as an iso-10303-28 document* if it satisfies all of the following conditions:

- The root element for the document is an `iso_10303_28` element, as specified in Clause 5.
- The XML infoset for the `iso_10303_28` element is valid with respect to the Document Schema defined in Annex D.
- Each data set encoded in the document, if any, conforms as specified in ISO 10303-11 to a given EXPRESS schema, designated the *governing schema* for that data set.
- Each data set is encoded using a substitute unit of serialization element for the `uos` element specified in the Base XML Schema (see Annex C).
- For each substitute unit of serialization element, the XML infoset is valid with respect to the XML schema that is derived as specified in Clause 6, using the EXPRESS schema and the configuration file specified by that unit of serialization element.

NOTE 1 The governing EXPRESS schema can be different for each data set.

NOTE 2 The substitute `uos` element corresponding to a given EXPRESS schema is described in 7.8 or 8.8.

NOTE 3 Each unit of serialization element specifies the governing EXPRESS schema and the configuration file to which it corresponds. See 5.6 and 9.2.

NOTE 4 There is no requirement for the data set to be a "valid population", as defined in ISO 10303-11. The relationship between encoded data sets and populations is specified in a given `iso_10303_28` element by `schema_population` elements. See 5.3.

4.1.2 Conformance of a uos document

An XML document shall *conform as a uos document* if it satisfies all of the following conditions:

- The root element for the document is a `uos` element.
- The data set encoded in the `uos` element conforms as specified in ISO 10303-11 to a given EXPRESS schema, designated the *governing schema* for that data set.
- The XML infoset for the `uos` element is valid with respect to the XML schema that is derived from the governing EXPRESS schema as specified in Clause 6.

NOTE In a `uos` document, the `uos` element is the one that corresponds to the default mapping of the governing schema, that is, the one corresponding to an empty configuration file.

4.1.3 Conformance of a configured document

Properly, the conformance of a configured document is defined by the specification that defines the configured XML schema. This subclause is intended solely for reference by such specifications. Conformance of a configured document in the absence of such a referencing specification is not defined by this Part of ISO 10303.

An XML document shall *conform as a configured document* for a given EXPRESS schema and a given configuration file if it satisfies all of the following conditions:

- The root element for the document is the unit of serialization element derived as described in 8.8 from the given EXPRESS schema using the given configuration file.
- The data set encoded in the unit of serialization element conforms to the given EXPRESS schema as specified in ISO 10303-11.
- The XML infoset for the unit of serialization element is valid with respect to the XML schema that is derived from the governing EXPRESS schema as specified in Clause 6 using the given configuration file.

NOTE For specifications on the creation of an XML instance document conforming to this part of ISO 10303, see Clause 9.

4.2 Conformance of a derived XML schema document

An XML document shall *conform as a derived XML schema document* for a given EXPRESS schema and a given configuration file if it satisfies all of the following conditions:

- The given EXPRESS schema conforms to ISO 10303-11.
- The given configuration file, if any, conforms as a configuration file for the given EXPRESS schema as specified in 4.3.
- The content of the document satisfies the schema validity constraints described in Clause 5 of *XML Schema: Part 1: Structures*.
- The XML schema definition contains exactly the definitions and declarations that are specified in Clause 7 or Clause 8, as indicated by Clause 6, to be derived from the given EXPRESS schema using the given configuration file. The XML schema definition may contain additional XML Schema annotation elements.

There are two conformance classes for derived XML schema documents:

- The XML schema document *conforms in class 1* if it includes all definitions and declarations that are specified in Clause 7 or Clause 8, as indicated by Clause 6, *except* those specified to be included only in conformance class 2. A derived XML schema that conforms in class 1 shall not include any definitions or declarations specified in Clause 8 to be included only in conformance class 2 (see 10.2.23).
- The XML schema document *conforms in class 2* if it includes all of the definitions and declarations that are specified in Clause 8, *including* all of those specified to be included only in conformance class 2.

NOTE In order for a document to conform in class 2, Clause 6 must specify that the derived XML Schema will be the Configured XML Schema binding, specified in Clause 8, and the configuration directive **generate-keys="true"** must apply.

4.3 Conformance of a configuration file

An XML document shall *conform as a configuration file* for a given EXPRESS schema if it satisfies all of the following conditions:

- The root element of the document is a `configuration` element.
- The `configuration` element is valid with respect to the Configuration Schema specified in Annex B.
- The configuration file conforms to the specifications given in Clause 10, where the context schema is the given EXPRESS schema.

NOTE If the configuration file contains no elements that are stated in Clause 10 to be schema-specific, the configuration file will conform for an arbitrary EXPRESS schema. But for the purposes of this part of ISO 10303, it is only useful to define conformance in the context of a given EXPRESS schema.

4.4 Conformance of a pre-processor

A software application is said to conform as a *pre-processor* if it produces XML instance documents that conform to this part of ISO 10303 as specified in 4.1. An application may conform as a pre-processor for iso-10303-28 documents only, or for uos documents only, or for both. An application may conform as a pre-processor for arbitrary EXPRESS schemas, or it may conform as a pre-processor for certain EXPRESS schemas only.

There are three conformance classes for pre-processors:

- A pre-processor *conforms in class 1* if it can produce uos documents conforming to the class 1 derived XML schema for some EXPRESS schemas.
- A pre-processor *conforms in class 2* if it can produce iso-10303-28 documents that contain uos elements, and independent uos documents, conforming to the class 1 derived XML schema for some EXPRESS schemas.
- A pre-processor *conforms in class 3* if it can produce iso-10303-28 documents that contain substitute uos elements conforming to arbitrary EXPRESS schemas with arbitrary configuration files.

Conformance of a pre-processor in any case shall be claimed based on the EXPRESS schemas to which its XML document production is limited.

NOTE Both class 1 and class 2 define pre-processor conformance as the ability to produce XML data sets corresponding to the default XML schemas. Class 3 requires the pre-processor to support arbitrary configuration files. If an application conforms only for configured documents corresponding to specific configured XML schemas, it is not said to conform as a pre-processor to this part of ISO 10303. Rather, it conforms as a pre-processor to the specifications that define the configured XML schemas.

4.5 Conformance of a post-processor

A software application is said to conform as a *post-processor* if it can accept and process XML instance documents that conform to this part of ISO 10303 as specified in 4.1. An application may conform as a post-processor for iso-10303-28 documents only, or for uos documents only, or for both. An application may conform as a post-processor for arbitrary EXPRESS schemas, or it may conform as a post-processor for certain EXPRESS schemas only.

There are three conformance classes for post-processors:

- A post-processor *conforms in class 1* if it can accept and process uos documents and/or iso-10303-28 documents that contain uos elements conforming to the class 1 derived XML schema for an arbitrary EXPRESS schema.
- A post-processor *conforms in class 2* if it can accept and process uos documents and/or iso-10303-28 documents that contain uos elements conforming to the class 2 derived XML schema for an arbitrary EXPRESS schema, and perform the corresponding XML validation.
- A post-processor *conforms in class 3* if it can accept and process iso-10303-28 documents that contain unit of serialization elements conforming to EXPRESS schemas with an arbitrary conforming configuration file.

In each of conformance classes 1 and 2, conformance of a post-processor shall be claimed based on the types of documents and the EXPRESS schemas to which its XML document processing is limited. In conformance class 3, conformance of a post-processor shall be claimed based on the EXPRESS schemas to which its XML document processing is limited.

NOTE If an application conforms only for configured documents corresponding to specific configured XML schemas, it is not said to conform as a post-processor to this part of ISO 10303. Rather, it conforms as a post-processor to the specifications that define the configured XML schemas.

4.6 Conformance of an XML schema generator

A software application *conforms as an XML schema generator* if it can process any valid EXPRESS schema and produce a conforming XML schema document as defined in 4.2.

An XML schema generator *conforms in class 1* if it can generate the default XML schema specified in Clause 7, that is, the derived schema that results from an empty configuration file.

An XML schema generator *conforms in class 2* if it can generate the derived configured XML schema corresponding to any EXPRESS schema and any conforming configuration file, as specified in Clause 8.

5 Document level elements

This clause specifies the XML elements that define the structure of an XML instance document conforming to this Part of ISO 10303. 5.1 defines the structure of an XML document that conforms as an iso-10303-28 document. 5.2 through 5.6 define the the content elements of the iso-10303-28

document root element. 5.7 defines the structure of an XML document that conforms as a uos document. 5.8 defines the structure of an XML document that conforms as a configured document. 5.9 defines an XML attribute contained in the the iso-10303-28 document root element.

This Part of ISO 10303 specifies in Annex C a collection of XML Schema declarations and definitions, called the Base XML Schema, that shall be part of the base schema for every conforming XML instance document. The XML element names, attribute names and data type names declared in that collection shall constitute the namespace designated `urn:iso:std:iso:10303:-28:ed-2:tech:XMLschema:common`. This Part of ISO 10303 also specifies in Annex D a collection of XML Schema declarations and definitions, called the Document Schema, that define the XML infoset for a conforming iso-10303-28 document. The XML element names, attribute names and data type names declared in that collection shall constitute the namespace designated `urn:iso:std:iso:10303:-28:ed-2:tech:XMLschema:document`.

5.1 The iso-10303-28 document

The root element of an iso-10303-28 document shall be an `iso_10303_28` element. The `iso_10303_28` element contains all other elements defined in this part of ISO 10303. These elements permit the document to contain descriptive information, EXPRESS schemas, XML schema configuration files, and data sets described by EXPRESS schemas. These declarations are so specified that a conforming document can contain EXPRESS schemas, configuration files, and data sets from other XML documents by reference.

This Part of ISO 10303 specifies in Annex D a collection of XML Schema declarations and definitions, called the Document Schema, that shall be part of the document schema for every XML instance document that conforms as an iso-10303-28 document. The XML element names, attribute names and data type names declared in that collection shall constitute the namespace specified in Annex A.3.

The element type declaration in the Document Schema is:

```
<xs:element name="iso_10303_28">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="exp:header" minOccurs="0"/>
      <xs:choice minOccurs="0" maxOccurs="unbounded">
        <xs:element ref="doc:schema_population"/>
        <xs:element ref="exp:uos"/>
        <xs:element ref="doc:express"/>
        <xs:element ref="doc:schema"/>
        <xs:element ref="cnf:configuration"/>
      </xs:choice>
    </xs:sequence>

    <xs:attribute name="version" type="xs:string" use="required"/>
    <xs:attribute name="edo" type="xs:anyURI" use="optional"/>
  </xs:complexType>
</xs:element>
```

The `version` XML attribute shall identify the edition of this part of ISO 10303 to which the XML document conforms. For documents conforming to this edition of this part of ISO 10303, the value of `version` shall be '2.0'.

If present, the `edo` attribute should contain the fixed global identifier for the document as an enterprise data object (see 5.9).

The content elements of the `iso_10303_28` element and their XML attributes are described in the following subclauses.

5.2 Document and uos header information

This subclause specifies elements to contain "provenance" information that describes an element that is an information resource. The resource may be the entire iso-10303-28 document element or a unit of serialization (see 5.6) that it contains. "Provenance" information is information about the source of the information contained in the resource.

5.2.1 The `exp:header` element

The `exp:header` element shall contain administrative information that characterizes the content of the entire XML resource – the element that contains the `exp:header` element. The resource that contains the header element may be an `iso_10303_28` element (representing the entire document) or a `uos` element instance document. The element type declaration in the Document Schema is:

```
<xs:complexType name="name_and_address">
  <xs:sequence>
    <xs:element name="name" type="xs:string" />
    <xs:element name="address">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="address_line" type="xs:string"
            minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
</xs:complexType>

<xs:element name="header">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="name" type="xs:string" minOccurs="0"/>
      <xs:element name="time_stamp" type="xs:dateTime" minOccurs="0"/>
      <xs:element name="author" type="exp:name_and_address"
        minOccurs="0"/>
      <xs:element name="organization" type="exp:name_and_address"
        minOccurs="0"/>
      <xs:element name="preprocessor_version" type="xs:string"
        minOccurs="0"/>
      <xs:element name="originating_system" type="xs:string"
        minOccurs="0"/>
      <xs:element name="authorization" type="xs:string" minOccurs="0"/>
      <xs:element name="documentation" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

NOTE This definition corresponds to the `FILE_NAME` entity defined in ISO10303-21 [1].

5.2.2 The name element

The `name` element shall provide a human readable identifier for the XML resource.

5.2.3 The time_stamp element

The `time_stamp` element shall specify the date and time when the XML resource was created.

5.2.4 The author element

The `author` element shall identify the person or group of persons who created the XML resource. This part of ISO 10303 places no further requirements on the content of the author element.

5.2.5 The organization element

The `organization` element shall identify the organization that created, or is responsible for, the XML resource.

5.2.6 The authorization element

The `authorization` element shall specify the release authorization for the XML resource and the signatory, where appropriate.

NOTE This may be distinct from the authorizations for various information units contained within the document.

5.2.7 The originating_system element

The `originating_system` element shall identify the software system that created or captured the information contained in the XML resource, including platform and version identifiers.

5.2.8 The preprocessor_version element

The `preprocessor_version` element shall identify the software system that created the XML resource itself, including platform and version identifiers.

NOTE The `preprocessor_version` will identify the system that was used to produce the XML resource. It may well be distinct from the software system that created or captured the original information.

5.3 The schema_population element

The `schema_population` element identifies a collection of EXPRESS entity instances that are represented in unit of serialization elements within the document as constituting a "population", which can be validated under the rules of an EXPRESS schema, as defined in ISO 10303-11. Given the specification of a population via a `schema_population` element, a conforming post-processor may, but is not required to, validate the population against the governing schema.

The element type declaration in the Document Schema is:

```
<xs:element name="schema_population">
  <xs:complexType>
    <xs:attribute name="governing_schema" type="xs:IDREF" use="required"/>
    <xs:attribute name="governed_sections" type="xs:IDREFS"
      use="optional"/>
    <xs:attribute name="determination_method"
      type="xs:normalizedString" default="section_boundary"/>
  </xs:complexType>
</xs:element>
```

```

    </xs:complexType>
  </xs:element>

```

The collection of entity instances constituting the population shall be determined by applying an algorithm identified by the XML attribute `determination_method` to the set of unit of serialization elements identified by the XML attribute `governed_sections`.

The value of the `governing_schema` XML attribute shall be the local XML identifier for the `express` element that represents the EXPRESS schema to be used to validate the population described by the `schema_population` element. The corresponding `express` element shall appear in the document.

NOTE 1 The referenced `express` element may contain a representation of the EXPRESS schema or an external reference for the EXPRESS schema.

The value of the `governed_sections` XML attribute shall be a sequence of local XML identifiers separated by spaces, each of which shall identify a unit of serialization element appearing in the document. If no value for the `governed_sections` attribute is provided, the specified list of unit of serialization elements shall comprise all unit of serialization elements appearing in the document. The specified unit of serialization elements shall be the input operands of the determination method.

NOTE 2 Possible relationships between the context schema for a unit of serialization element and the `governing_schema` for the population(s) to which its entity instances belong are discussed in Annex E.

The value of the `determination_method` attribute shall be a string of characters that identifies the algorithm to be used for selecting EXPRESS entity instances to comprise the population. Three possible values for this attribute and the corresponding selection algorithms are specified in Annex E. Other values may appear, and their interpretation is not specified in this Part of ISO 10303. If no value for this attribute is provided, the value "section_boundary" is assumed and the section boundary method described in Annex E shall be used.

A conforming `iso_10303_28` element may contain zero or more `schema_population` elements. Each `schema_population` element describes one complete population for purposes of validation as specified in ISO 10303-11. If no `schema_population` elements appear, the relationship of the EXPRESS entity instances represented in the document, if any, to any "valid population" is not specified, and validation of the contents of the `iso_10303_28` element against any EXPRESS schema is not possible without additional information.

A given unit of serialization element may be used explicitly, by a reference to the element in the `governed_sections` XML attribute of the `schema_population` element, in zero, one, or many populations. A given unit of serialization element may also be used implicitly in zero, one, or many populations, according to the determination method used for those populations, but such a use does not necessarily imply the use of all EXPRESS entity instances represented in that unit of serialization. If a unit of serialization element is not explicitly or implicitly referenced by any `schema_population` element, the relationship of the EXPRESS entity instances represented therein to any "valid population" is not specified.

NOTE 3 The following procedure describes an approach for checking the validity of a population that is consistent with the above:

For each `schema_population` element in the `iso_10303_28` element:

— find a set of EXPRESS entity instances by applying the determination method identified by the value of the `determination_method` attribute to the unit of serialization elements identified in the value of the

governed_sections attribute, or to all unit of serialization elements in the iso_10303_28 element if governed_sections is not specified;

— check this set of instances against the rules and constraints defined by the EXPRESS schema designated by the value of the governing_schema attribute.

5.4 The express element

The express element shall contain one EXPRESS schema, either "by-value" or "by-reference".

The element type declaration in the Document Schema is:

```
<xs:element name="express" nillable="true">
  <xs:complexType>
    <xs:simpleContent>
      <xs:extension base="xs:string">
        <xs:attribute name="schemaLocation" type="exp:Seq-anyURI"
          use="optional"/>
        <xs:attribute name="id" type="xs:ID" use="required"/>
        <xs:attribute name="schema_identifier"
          type="xs:normalizedString" use="optional"/>
        <xs:attribute name="schema_name" type="xs:normalizedString"
          use="optional"/>
        <xs:attribute name="schema_version" type="xs:normalizedString"
          use="optional"/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
</xs:element>
```

The Seq-anyURI type declaration in the Base XML Schema is:

```
<xs:simpleType name="Seq-anyURI">
  <xs:list itemType="xs:anyURI"/>
</xs:simpleType>
```

In either case:

— The value of the id XML attribute shall be a local XML identifier for the EXPRESS schema. This value provides the reference to the EXPRESS schema for schema_population elements and unit of serialization elements within the document.

— The schema_name XML attribute, if present, shall contain the EXPRESS identifier for the schema, converted as specified in 7.1.2 or 8.1.2, as specified by Clause 6.

— The schema_version XML attribute, if present, shall contain the version identifier for the schema.

— For an EXPRESS schema that is defined by a Part of ISO 10303, the schema_identifier attribute, if present, shall contain the ASN.1 identifier value associated with that schema by that Part of ISO 10303. In any other case, the use of the schema_identifier attribute is not specified by this Part of ISO 10303.

5.4.1 By-reference representation of an EXPRESS schema

When the EXPRESS schema is represented "by-reference", the `schemaLocation` XML attribute shall be present and its value shall identify one or more resources that contain the EXPRESS schema (text).

NOTE By convention, the value of `schemaLocation` is either a single URI, or a list of URIs of which the first is a URN that identifies the configuration file globally, and all of the others are possible locations of the text.

When the EXPRESS schema is represented "by-reference", the content of the `express` element shall be nil (empty).

5.4.2 By-value representation of an EXPRESS schema

When the EXPRESS schema is represented "by-value", the content of the `express` element shall be a text representation of the EXPRESS schema conforming to ISO 10303-11, using whatever character encoding is specified in the XML processing instruction(s). The content shall be one of the following;

- a valid XML CDATA section as specified in the *XML 1.0 Recommendation*, section 2.7;
- a valid XML text string in which the character '<' is replaced by '<'; and the character '&' is replaced by '&'.

If the character sequence ']]>' appears in the EXPRESS schema, the valid XML text string form shall be used to represent the schema.

When the EXPRESS schema is represented "by-value", the `schemaLocation` XML attribute shall not be present.

EXAMPLE The EXPRESS schema:

```
SCHEMA hello_world;
END_SCHEMA;
```

would appear as:

```
<express_schema id="s_01"><![CDATA[SCHEMA hello_world; END_SCHEMA;]]>
</express_schema>
```

as an XML CDATA section.

5.5 The configuration element

The `configuration` element contains a configuration file that controls the mapping of a given EXPRESS schema to an XML schema, and the XML representation of the data that is described by that EXPRESS schema. The structure and interpretation of the `configuration` element is specified in 10.1.

5.6 The unit of serialization element

A *schema instance* is a data set, grouped for some purpose, that is modeled by a single EXPRESS schema. The *unit of serialization element* is the XML element that represents one schema instance. The EXPRESS schema that models that schema instance is designated the *context schema* for that unit of serialization element. For each context schema, there is a specific unit of serialization element that

is declared in the namespace associated with the derived XML schema, as described in 7.8 or 8.8. Each unit of serialization element shall be a member of the substitution group for the `exp:uos` element, and its data type shall be an extension of the `exp:uos` data type, defined in the Base XML Schema as follows:

```
<xs:complexType name="uos">
  <xs:sequence>
    <xs:element ref="exp:header" minOccurs="0"/>
  </xs:sequence>

  <xs:attribute name="id" type="xs:ID" use="optional"/>
  <xs:attribute name="express" type="exp:Seq-anyURI"
    use="optional"/>
  <xs:attribute name="configuration" type="exp:Seq-anyURI"
    use="optional"/>
  <xs:attribute name="schemaLocation" type="exp:Seq-anyURI"
    use="optional"/>
  <xs:attribute name="edo" type="xs:anyURI" use="optional"/>
  <xs:attribute name="defaultLanguage" type="xs:language"
    use="optional"/>
</xs:complexType>

<xs:element name="uos" type="exp:uos" abstract="true" />
```

The `exp:header` element, if present, shall be as specified in 5.2.1.

When the unit of serialization is contained in an ISO 10303-28 document, the `id` XML attribute shall be present. The value of the `id` XML attribute shall be a local XML identifier for the "unit of serialization". This value provides the reference to the unit of serialization element for `schema_population` elements and serves as part of the unique identifier for the unit of serialization element when it is referenced as a resource from outside the ISO 10303-28 document. When the `uos` element is the root of the document (see 5.7), the `id` XML attribute should not be present.

The value of the `express` XML attribute shall identify one or more resources that contain the context schema for the schema instance encoded in the unit of serialization element. If this attribute is not present, the EXPRESS schema associated with the configuration file (see 10.1) is presumed to be the context schema.

The value of the `configuration` XML attribute shall identify one or more resources that contain the configuration file that was used to control the encoding of the EXPRESS schema instance in this unit of serialization element. If this attribute is not present, the default configuration for the EXPRESS schema specified by the `express` attribute is assumed.

The value of the `schemaLocation` XML attribute shall identify one or more resources that contain the governing XML schema for the unit of serialization element. If this attribute is not present, the governing XML schema shall be that derived from the specified EXPRESS schema using the specified configuration file.

For any of the `express`, `configuration`, and `schemaLocation` attributes, the URI may have the form: `uri#id`, where `uri` is a URI conforming to IETF RFC 2396 that locates the resource, and the fragment identifier `id` is an XML identifier local to that resource that identifies the schema element.

NOTE 1 When the URI has the form `#id` (with no URL), it refers to an XML element with that `id` in the current document. For example, it may refer to a `express` or `configuration` XML element in the current document.

NOTE 2 The use of the `configuration` and `express` attributes are described in more detail in 9.2.

If present, the `edo` attribute should contain the fixed global identifier for the unit of serialization as an enterprise data object (see 5.9).

The value of the `defaultLanguage` XML attribute shall identify the language used as default in the data set. Values of this XML attribute are defined by ISO-639, "Codes for the Representation of Names of Languages". Subcategories of these languages are identified by ISO-3166, "Codes for the Representation of Names of Countries".

EXAMPLE "fr-CA" for Canadian-french. If this attribute is specified, any string in the data set can be interpreted as belonging to the identified language. If this attribute is not present no language default can be assumed.

NOTE 3 A default language may be overwritten on the basis of an EXPRESS attribute, if the underlying EXPRESS schema and the configuration applied define so.

5.7 The uos document

A uos document represents a single schema instance, and a single unit of serialization, as defined in 5.6. The root element of a uos document shall be a schema-specific `uos` element conforming to the default XML schema for the EXPRESS schema that governs the schema instance, as specified in 7.8.

5.8 The configured document

A configured document represents a single schema instance, and a single unit of serialization, as defined in 5.6. The root element of a configured document shall be the unit of serialization element declared in the XML schema derived from the EXPRESS schema that governs the schema instance, according to the given configuration file, as specified in 8.8.

5.9 Enterprise data objects

An *enterprise data object* is a data resource that may be referred to in business transactions and other exchange documents. An enterprise data object has a fixed global identifier by which it is known. For consistency with XML practice, we assume that the fixed global identifier is a "reference URI" that is independent of location.

An enterprise data object can be an ISO 10303-28 document, a unit of serialization, or a single EXPRESS entity instance. In each case, the XML `edo` attribute is attached to the corresponding XML element to capture this property.

6 Derived XML Schema

The XML schema derived from the EXPRESS schema is said to be the *derived XML schema* corresponding to the given EXPRESS schema and an optional given configuration file.

The XML schema derived from the EXPRESS schema is said to be the *default XML schema* corresponding to the given EXPRESS schema if any of the following is true:

- no configuration file is given;
- the given configuration file contains no directives;
- all directives in the given configuration file specify only the default values.

When any of the above apply, the XML schema derived from the EXPRESS schema shall be the default XML schema binding, specified in Clause 7.

The XML schema derived from the EXPRESS schema is said to be the *configured XML schema* in all other cases. If the optional configuration file is included and it contains at least one directive that does not specify the default value for the given EXPRESS schema then the configured XML binding shall apply as specified in Clause 8.

The following subclauses specify the fundamental concepts and assumptions of the derived XML schema bindings.

6.1 Preconditions

The creation of XML Schema declarations and definitions from an EXPRESS schema assumes that the schema meets the following precondition:

- syntactically correct EXPRESS schema;
- the context schema is the complete schema for some collection of data sets to be exchanged.

6.2 Unmapped EXPRESS concepts

The following EXPRESS features are not mapped:

- RULE declarations;
- FUNCTION declarations and PROCEDURE declarations;
- CONSTANT declarations;
- local (WHERE) rules within ENTITY declarations;
- local (WHERE) rules within TYPE declarations;
- USE and REFERENCE statements (see 6.1);
- subtype constraints (SUPERTYPE clauses and SUBTYPE_CONSTRAINTS).

NOTE The subtype constraint itself is not mapped; however, if the subtype constraint declares an entity ABSTRACT, it affects the mapping to XMLSchema (see 7.5.3 or 8.5.3).

6.3 Abstract entity data types

An EXPRESS entity data type is *abstract* if every instance of that entity data type must be an instance of some subtype of it.

An EXPRESS entity datatype can be declared abstract in one of three ways:

- as an ABSTRACT ENTITY in the entity declaration,
- as an ABSTRACT SUPERTYPE in a supertype-expression in the entity declaration, or
- as ABSTRACT in a SUBTYPE_CONSTRAINT.

7 Default XML Schema Binding

This clause specifies a mapping from an EXPRESS schema to the default corresponding XML Schema declarations, definitions and representation. When the derived XML schema specified in Clause 6 is said to be the *default XML schema*, this clause shall apply.

7.1 Naming conventions

This clause specifies uniform conventions for creating XML names based on EXPRESS identifiers.

7.1.1 Schema

Every EXPRESS schema shall have an associated XML namespace URI. This URI shall designate the target namespace for the XML element and data type names that are schema-specific elements of the binding of that schema, as specified in 7.7.

7.1.2 EXPRESS identifiers

EXPRESS language identifiers are not case-sensitive. XML names are case-sensitive.

Every EXPRESS identifier that appears in the XML schema definitions and declarations shall be mapped to an XML name, or the local part of an XML name. The XML name shall be the same as the EXPRESS identifier except that the initial letter is upper case and all other letters are lower case.

Due to the XML naming restrictions specified in the XML Recommendation, any EXPRESS identifier beginning with the characters 'XML', regardless of case, shall map to an XML name beginning with the characters 'X-m-l'.

NOTE This subclause applies to most EXPRESS defined data type names, entity names, and attribute names. Exceptions for certain situations are specified in the subclauses dealing with particular kinds of EXPRESS identifiers.

7.1.3 Data types

Each EXPRESS data type identifier shall be mapped to a qualified name as defined in the Namespaces in XML Recommendation.

The local part of the qualified name shall be the EXPRESS data type identifier mapped as specified in 7.1.2. For every EXPRESS named data type declared in, or explicitly interfaced into, the context schema, the associated XML Schema namespace for the qualified name shall be that of the context schema.

For every data type that is implicitly interfaced into the context schema, the local part of the qualified name shall be derived as specified in 7.1.2 from the EXPRESS identifier for that data type, as it appears in the schema in which the data type is declared. The associated namespace shall be that of the schema in which the data type is declared.

NOTE For the non-local part of the URI, see Annex A for an approach to standardized namespace URIs for use with EXPRESS schemas defined in other parts of ISO 10303 and other ISO standards.

7.2 XML Schema data types corresponding to EXPRESS data types

For every EXPRESS data type appearing in the context schema, including named data types, built-in data types and anonymous constructed data types, there is a corresponding XML schema data type. This subclause specifies that correspondence.

NOTE This subclause does not specify any definitions or declarations to appear in the derived XML schema. It specifies XML schema components that may appear in definitions and declarations required by other subclauses.

7.2.1 EXPRESS simple data types

This subclause specifies the XML schema datatypes that correspond to the EXPRESS built-in simple data types.

7.2.1.1 BINARY data type

The EXPRESS BINARY data type is mapped to the `exp:hexBinary` data type. Exception: If the EXPRESS BINARY data type specifies a maximum or FIXED length, it shall be mapped as specified in 7.2.1.1.1

The `exp:hexBinary` data type is defined in the Base XML Schema (see Annex C) as follows:

```
<xs:complexType name="hexBinary" >
  <xs:simpleContent>
    <xs:extension base="xs:hexBinary">
      <xs:attribute name="extraBits" type="xs:integer"
        use="optional"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

EXAMPLE

```
ENTITY entity_with_binary_attribute;
  a_binary : BINARY;
END_ENTITY;
```

In XML Schema:

```
<xs:complexType name="Entity_with_binary_attribute">
  <xs:complexContent>
    <xs:extension base="exp:Entity">
      <xs:all>
        <xs:element type="exp:hexBinary" name="A_binary"/>
      </xs:all>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

XML instance document:

```
<Entity_with_binary_attribute>
  <A_binary>0FB7</A_binary>
</Entity_with_binary_attribute>
```

7.2.1.1.1 Constrained BINARY data types

An EXPRESS BINARY data type for which a maximum or FIXED length is specified is said to be a *constrained* BINARY data type. It shall be mapped to a `complexType` that is a restriction of the `exp:hexBinary` data type.

When the constrained BINARY data type is the underlying type in an EXPRESS type declaration, the corresponding XML data type shall be as specified in 7.3.1.1.

For a constrained `BINARY` data type, the derived XML schema shall contain a type definition for the XML data type that corresponds to the constrained `BINARY` data type. The type definition shall have the form:

```
<xs:complexType name="Binary.minBits.bits">
  <xs:simpleContent>
    <xs:restriction base="exp:hexBinary">
      <xs:minLength value="minLength"/>
      <xs:maxLength value="maxLength"/>
      <xs:attribute name="extraBits" type="xs:integer"
        fixed="remainder"/>
    </xs:restriction>
  </xs:simpleContent>
</xs:complexType>
```

where:

— **minBits** shall be the fixed length (in bits) specified for the EXPRESS `BINARY` data type, if it is declared `FIXED`. Otherwise, **bits** shall be "0".

— **bits** shall be the maximum or fixed length (in bits) specified for the EXPRESS `BINARY` data type.

— **minLength** shall be the octet value (see below) of the fixed length specified for the EXPRESS `BINARY` data type, if it is declared `FIXED`. Otherwise, the `xs:minLength` facet may be omitted.

— **maxLength** shall be the octet value of the maximum (or fixed) length specified for the EXPRESS `BINARY` data type.

— **remainder** shall be a decimal integer giving the value (**bits** modulo 8), where **bits** is as specified below, if the EXPRESS data type is declared `FIXED`. Otherwise, the redeclaration of the `extraBits` attribute shall be omitted.

Given **bits** = the maximum (or fixed) length in bits specified for the EXPRESS data type, the *octet value* shall be calculated as follows:

— If **bits** modulo 8 is greater than 0, $\text{bits} / 8 + 1$.

— If **bits** modulo 8 is equal to 0, $\text{bits} / 8$.

NOTE The values of **minBits** and **bits** are lengths in bits, while the values of **minLength** and **maxLength** are lengths in octets.

EXAMPLE

In EXPRESS:

```
ENTITY entity_with_fixed_binary_attribute;
  a_binary : BINARY(12) FIXED;
  b_binary : BINARY(8);
END_ENTITY;
```

In XML Schema:

```
<xs:complexType name="Binary.12.12">
  <xs:simpleContent>
    <xs:restriction base="exp:hexBinary">
```

```

        <xs:maxLength value="2"/>
        <xs:minLength value="2"/>
        <xs:attribute name="extrabits" type="xs:integer" fixed="4"/>
      </xs:restriction>
    </xs:simpleContent>
  </xs:complexType>

  <xs:complexType name="Binary.0.8">
    <xs:simpleContent>
      <xs:restriction base="exp:hexBinary">
        <xs:maxLength value="1"/>
      </xs:restriction>
    </xs:simpleContent>
  </xs:complexType>

  <xs:complexType name="Entity_with_fixed_binary_attribute">
    <xs:complexContent>
      <xs:extension base="exp:Entity">
        <xs:all>
          <xs:element name="A_binary" type="Tns:Binary.12.12"/>
          <xs:element name="B_binary" type="Tns:Binary.0.8"/>
        </xs:all>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>

```

7.2.1.2 BOOLEAN data type

The EXPRESS BOOLEAN data type is mapped to the `xs:boolean` data type.

EXAMPLE

```

ENTITY entity_with_boolean_attribute;
  a_boolean : BOOLEAN;
END_ENTITY;

```

In XML Schema:

```

<xs:complexType name="Entity_with_boolean_attribute">
  <xs:complexContent>
    <xs:extension base="exp:Entity">
      <xs:all>
        <xs:element type="xs:boolean" name="A_boolean"/>
      </xs:all>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

```

XML instance document:

```

<Entity_with_boolean_attribute>
  <A_boolean>true</A_boolean>
</Entity_with_boolean_attribute>

```

7.2.1.3 INTEGER data type

The EXPRESS INTEGER data type is mapped to the `xs:long` data type.

EXAMPLE

```

ENTITY entity_with_integer_attribute;
  an_integer : INTEGER;
END_ENTITY;

```

In XML Schema:

```

<xs:complexType name="Entity_with_integer_attribute">
  <xs:complexContent>
    <xs:extension base="exp:Entity">
      <xs:all>
        <xs:element type="xs:long" name="An_integer"/>
      </xs:all>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

```

XML instance document:

```

<Entity_with_integer_attribute>
  <An_integer>1234</An_integer>
</Entity_with_integer_attribute>

```

7.2.1.4 LOGICAL data type

The EXPRESS LOGICAL data type is mapped to the `exp:logical` data type.

The `exp:logical` data type is defined in the Base XML Schema (see Annex C) as follows:

```

<xs:simpleType name = "logical">
  <xs:restriction base = "xs:string">
    <xs:enumeration value = "false"/>
    <xs:enumeration value = "true"/>
    <xs:enumeration value = "unknown"/>
  </xs:restriction>
</xs:simpleType>

```

EXAMPLE

```

ENTITY entity_with_logical_attribute;
  a_logical : LOGICAL;
END_ENTITY;

```

In XML Schema:

```

<xs:complexType name="Entity_with_logical_attribute">
  <xs:complexContent>
    <xs:extension base="exp:Entity">
      <xs:all>
        <xs:element type="exp:logical" name="A_logical"/>
      </xs:all>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

```

XML instance document:

```

<Entity_with_logical_attribute>

```

```

    <A_logical>true</A_logical>
  </Entity_with_logical_attribute>

```

7.2.1.5 NUMBER data type

The EXPRESS NUMBER data type is mapped to the `xs:decimal` data type.

EXAMPLE

```

ENTITY entity_with_number_attribute;
  a_number    : NUMBER;
END_ENTITY;

```

In XML Schema:

```

<xs:complexType name="Entity_with_number_attribute">
  <xs:complexContent>
    <xs:extension base="exp:Entity">
      <xs:all>
        <xs:element type="xs:decimal" name="A_number"/>
      </xs:all>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

```

XML instance document:

```

<Entity_with_number_attribute>
  <A_number>1234.5678</A_number>
</Entity_with_number_attribute>

```

7.2.1.6 REAL data type

The EXPRESS REAL data type is mapped to the `xs:double` data type. EXPRESS REAL data type with precision shall be mapped without restriction of the precision.

EXAMPLE 1

```

ENTITY entity_with_real_attribute;
  a_real      : REAL;
END_ENTITY;

```

In XML Schema:

```

<xs:complexType name="Entity_with_real_attribute">
  <xs:complexContent>
    <xs:extension base="exp:Entity">
      <xs:all>
        <xs:element type="xs:double" name="A_real"/>
      </xs:all>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

```

XML instance document:

```

<Entity_with_real_attribute>
  <A_real>1.2E02</A_real>

```

```
</Entity_with_real_attribute>
```

EXAMPLE 2: Real with precision shall be mapped identically to example 1 above:

```
ENTITY entity_with_real_attribute;
a_real      : REAL(16);
END_ENTITY;
```

7.2.1.7 STRING data type

The EXPRESS STRING data type is mapped to the `xs:normalizedString` data type. Exception: If the EXPRESS data type is declared to have a maximum length or a `FIXED` length, the EXPRESS data type shall be mapped as specified in 7.2.1.7.1

EXAMPLE

```
ENTITY entity_with_string_attribute;
a_string     : STRING;
END_ENTITY;
```

In XML Schema:

```
<xs:complexType name="Entity_with_string_attribute">
  <xs:complexContent>
    <xs:extension base="exp:Entity">
      <xs:all>
        <xs:element type="xs:normalizedString" name="A_string"/>
      </xs:all>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

XML instance document:

```
<Entity_with_string_attribute>
  <A_string>Hallo</A_string>
</Entity_with_string_attribute>
```

7.2.1.7.1 Constrained STRING data types

An EXPRESS STRING data type for which a maximum or `FIXED` length is specified is said to be a *constrained* STRING data type. It shall be mapped to a `simpleType` that is a restriction of the `xs:normalizedString` data type.

When the constrained STRING data type is the underlying type in an EXPRESS type declaration, the corresponding XML data type shall be as specified in 7.3.1.3.

Otherwise, the derived XML schema shall contain a type definition for the XML data type corresponding to the constrained STRING data type. The type definition shall have the form:

```
<xs:simpleType name="String.minLength.maxLength">
  <xs:restriction base="xs:normalizedString">
    <xs:minLength value="minLength"/>
    <xs:maxLength value="maxLength"/>
  </xs:restriction>
</xs:simpleType>
```

where:

— **minLength** shall be the fixed length specified for the EXPRESS `STRING` data type, if it is declared `FIXED`. Otherwise, it shall be "0" and the `xs:minLength` facet may be omitted.

— **maxLength** shall be the maximum (or fixed) length specified for the EXPRESS `STRING` data type.

NOTE If the EXPRESS schema declares a defined data type for the constrained `STRING` data type, there will be an XML definition corresponding to the defined data type (see 7.3). The definition above is only required for constrained `STRING` data types that are anonymous.

EXAMPLE

```
ENTITY entity_with_string_types;
  a_bounded_string : STRING(16);
  a_fixed_length_string : STRING(8) FIXED;
END_ENTITY;
```

In XML Schema:

```
<xs:simpleType name="String.0.16">
  <xs:restriction base="xs:normalizedString">
    <xs:maxLength value="16"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="String.8.8">
  <xs:restriction base="xs:normalizedString">
    <xs:minLength value="8"/>
    <xs:maxLength value="8"/>
  </xs:restriction>
</xs:simpleType>

<xs:complexType name="Entity_with_string_types">
  <xs:complexContent>
    <xs:extension base="exp:Entity">
      <xs:all>
        <xs:element name="A bounded string" type="Tns:String.0.16"/>
        <xs:element name="A_fixed_length_string"
          type="Tns:String.8.8"/>
      </xs:all>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

XML instance document:

```
<Entity_with_string_types>
  <A_bounded_string>Public, John Q. </A_bounded_string>
  <A_fixed_length_string>jqpublic</A_fixed_length_string>
</Entity_with_string_types>
```

7.2.2 Aggregation data types

For an EXPRESS aggregation data type, that is, a SET, LIST, BAG or ARRAY data type, the *base-type* of the aggregate is the data type of the component values. The XML Schema data type corresponding to the aggregation data type depends on the base-type that applies to that use.

The XML data type that corresponds to a given use of the EXPRESS aggregation data type shall be as specified in Table 2 for the combination of aggregate type and base-type that applies to that use, as follows:

— The value for Base-type shall be as specified in the EXPRESS declaration for the aggregation data type. When the base type for the aggregation is a defined data type, the fundamental type of that defined data type shall be used as the Base-type value.

— *Simple* types are those whose XML representations have simpleTypes and contain no spaces, including EXPRESS types BOOLEAN, INTEGER, LOGICAL, REAL and NUMBER, and defined data types that are specializations of any of these.

— The value for Aggregate-type shall be as specified in the EXPRESS declaration for the aggregation data type. For the purposes of this table, ARRAY OF OPTIONAL base-type is considered a distinct aggregate-type.

— The applicable data type properties and the applicable values of the configuration directives should be ordered like the columns in Table 2, and compared one-for-one with the values in each row of the table.

— The applicable values will match all the values in exactly one row of the table. The corresponding XML data type shall be as specified in the subclause identified in the Subclause column of that row.

Table 2 — Subclause governing aggregation data type correspondence

Aggregate type	Base type	Subclause
ARRAY/LIST/SET/BAG	Simple	7.2.2.3
ARRAY/LIST/SET/BAG	STRING/BINARY	7.2.2.2
ARRAY/LIST/SET/BAG	ENUMERATION	7.2.2.5
ARRAY/LIST/SET/BAG	SELECT	7.2.2.6
ARRAY/LIST/SET/BAG	Entity	7.2.2.7
ARRAY/LIST/SET/BAG	Aggregation type	7.2.2.4
ARRAY OF OPTIONAL	Simple	7.2.2.2
ARRAY OF OPTIONAL	STRING/BINARY	7.2.2.2
ARRAY OF OPTIONAL	ENUMERATION	7.2.2.2
ARRAY OF OPTIONAL	SELECT	7.2.2.6
ARRAY OF OPTIONAL	Entity	7.2.2.7
ARRAY OF OPTIONAL	Aggregation type	7.2.2.4

7.2.2.1 XML attributes for aggregation data types

The XML schema data types corresponding to some EXPRESS aggregation data types are the Sequence-of-elements form having all of the following attributes:

```
<xs:attribute name="arraySize">
  <xs:simpleType>
    <xs:restriction>
      <xs:simpleType>
        <xs:list itemType="xs:integer"/>
      </xs:simpleType>
    </xs:restriction>
  </xs:simpleType>
</xs:attribute>
```

```

        <xs:minLength value="1"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>

  <xs:attribute name="itemType" type="xs:QName"/>

  <xs:attribute name="cType">
    <xs:simpleType>
      <xs:list itemType="exp:aggregateType"/>
    </xs:simpleType>
  </xs:attribute>

  <xs:simpleType name="aggregateType">
    <xs:restriction base="xs:normalizedString">
      <xs:enumeration value="array"/>
      <xs:enumeration value="list"/>
      <xs:enumeration value="set"/>
      <xs:enumeration value="bag"/>
      <xs:enumeration value="array-unique"/>
      <xs:enumeration value="array-optional"/>
      <xs:enumeration value="array-optional-unique"/>
      <xs:enumeration value="list-unique"/>
    </xs:restriction>
  </xs:simpleType>

```

The value of the `arraySize` attribute indicates the number of component elements in the aggregate value represented by a particular instance. For an `ARRAY`, the value is generally fixed and equal to $HiIndex - LoIndex + 1$ (see below- some exceptions apply). For a multi-dimensional array, the value is a sequence of integer values giving the number of component elements in each dimension (see 7.2.2.4 and 9.8.5).

The value of the `itemType` attribute indicates the base-type of the aggregate value — the type of the component elements. Each actual aggregation data type specifies a fixed value for `itemType` (see below).

The value of the `cType` attribute indicates the kind of EXPRESS aggregate — `SET`, `LIST`, `BAG`, `ARRAY`, `ARRAY OF UNIQUE` or `LIST OF UNIQUE`. Each actual aggregation data type specifies the required value of `cType` (see below and also 9.8).

7.2.2.2 Sequence-of-elements form of aggregates

When the *sequence-of-elements* form for representation of the EXPRESS aggregation data type is specified in Clause 7.2.2, the corresponding XML Schema data type shall be a `complexType` having the form:

```

<xs:complexType>
  <xs:sequence>
    <xs:element ref = "base-instance"
      minOccurs = "lower-bound" maxOccurs = "upper-bound" />
  </xs:sequence>
  <xs:attribute name="ref" type="xs:IDREF" use="optional"/>
  <xs:attribute ref="exp:arraySize" usage />
  <xs:attribute ref="exp:itemType" fixed="base-instance" />
  <xs:attribute ref="exp:cType" fixed="aggregate-type" />
</xs:complexType>

```

where:

— **base-instance** is the qualified name of the instance element corresponding to the base-type of the aggregation data type, as specified in 7.4 or 7.5.

— **lower-bound**: When the lower bound on the size of the collection is given and evaluates to a constant, and the EXPRESS aggregate-type is not ARRAY OF OPTIONAL, the value of `minOccurs` shall be the value of the lower bound on the size of the EXPRESS aggregate represented as a decimal integer. Otherwise the value of `minOccurs` shall be "0". If the lower bound evaluates to '1', the `minOccurs` attribute may be omitted.

— **upper-bound**: When the upper bound on the size of the collection is given and evaluates to a constant, the value of `maxOccurs` shall be the value of the upper bound on the size of the EXPRESS aggregate represented as a decimal integer. Otherwise the value of `maxOccurs` shall be "unbounded".

NOTE 1 If the EXPRESS aggregate is an ARRAY, the value of the lower and upper bounds is: $HiIndex(a) - LoIndex(a) + 1$, where a is the aggregation data type.

— **aggregate-type** is the keyword for the EXPRESS aggregate type, one of: "set" for SET, "bag" for BAG, "list" for LIST, "array" for ARRAY, "list-unique" for LIST OF UNIQUE, "array-unique" for ARRAY OF UNIQUE, array-optional for ARRAY OF OPTIONAL, and array-optional-unique for ARRAY OF OPTIONAL UNIQUE.

— **usage**: When the EXPRESS aggregate type is ARRAY (and it is not optional), and the upper and lower bounds on the array evaluate to constants, `arraySize` shall be declared `fixed="upper-bound"`, where **upper-bound** is as defined above. When the EXPRESS aggregate type is ARRAY, and the upper and lower bounds on the array do not evaluate to constants, `arraySize` shall be declared `use="required"`. Otherwise, `arraySize` shall be declared `use="optional"`.

NOTE 2 The values of this XML schema data type represent values of the EXPRESS aggregation data type as specified in 9.8.2 and 9.8.3.

NOTE 3 The above definition defines the body of the `complexType`, however the `complexType` element itself may or may not have a name attribute, as indicated in subsequent clauses.

EXAMPLE 1

In EXPRESS:

```
TYPE address = LIST [1:?] OF STRING;
END_TYPE;
```

In XML Schema:

```
<xs:complexType name="Address">
  <xs:sequence>
    <xs:element ref="exp:string-wrapper"
      minOccurs="1" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="ref" type="xs:IDREF" use="optional"/>
  <xs:attribute ref="exp:arraySize" use="optional"/>
  <xs:attribute ref="exp:itemType" fixed="exp:string-wrapper"/>
  <xs:attribute ref="exp:cType" fixed="list"/>
</xs:complexType>
```

EXAMPLE 2

In EXPRESS:

```

TYPE set_of_string_type = SET [1 : ?] OF defined_type_based_on_string;
END_TYPE;

TYPE defined_type_based_on_string = STRING;
END_TYPE;

```

In XML Schema:

```

<xs:complexType name="Set_of_string_type">
  <xs:sequence>
    <xs:element ref="Tns:Defined_type_based_on_string-wrapper"
      minOccurs="1" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="ref" type="xs:IDREF" use="optional"/>
  <xs:attribute ref="exp:arraySize" use="optional"/>
  <xs:attribute ref="exp:itemType"
    fixed="Tns:Defined_type_based_on_string-wrapper"/>
  <xs:attribute ref="exp:cType" fixed="set"/>
</xs:complexType>

```

7.2.2.3 List-of-values form of aggregates

When the *list-of-values* form for representation of the EXPRESS aggregation data type is specified in Clause 7.2.2, the corresponding XML Schema data type is based on a simple list data type, as specified in 7.2.2.3.1. The corresponding XML Schema data type is a restriction of the unrestricted `complexType` that supports aggregations of the EXPRESS base type, as specified in 7.2.2.3.2.

The XML data type corresponding to the EXPRESS aggregation data type shall have the form:

```

<xs:complexType>
  <xs:simpleContent>
    <xs:restriction base="unrestricted-type">
      <xs:simpleType>
        <xs:restriction base="list-type">
          <xs:minLength value="lower-bound" />
          <xs:maxLength value="upper-bound" />
        </xs:restriction>
      </xs:simpleType>
      <xs:attribute ref="exp:arraySize" usage />
      <xs:attribute ref="exp:cType" fixed="aggregate-type" />
    </xs:restriction>
  </xs:simpleContent>
</xs:complexType>

```

where:

— **unrestricted-type** shall be the qualified name of the unrestricted `complexType` that supports aggregations of the EXPRESS base-type, as specified in 7.2.2.3.2.

— **list-type** shall be the qualified name of the simple list data type that corresponds to the base type, as specified in 7.2.2.3.1.

— **lower-bound** is the lower bound on the EXPRESS aggregate. When the lower bound is given and evaluates to a constant, the value of `minLength` shall be the value of the lower bound, represented as a decimal integer. Otherwise, the value of `minLength` shall be '0'. If the value of `minLength` is '0', the `minLength` facet may be omitted.

— **upper-bound** is the upper bound on the EXPRESS aggregate. When the upper bound is given and evaluates to a constant, the `maxLength` attribute shall appear and shall have the value of the upper bound, represented as a decimal integer. Otherwise, the `maxLength` facet shall be omitted.

NOTE If the EXPRESS aggregation type is an ARRAY, the value of the lower and upper bounds is $HiIndex(a) - LoIndex(a) + 1$, where a is the aggregation data type.

— if the lower bound on the EXPRESS aggregation data type is 0 or unspecified and the upper bound is indeterminate or unspecified, the restriction on the simple list type may be omitted from the declaration, that is, only the values of the XML attributes are restricted.

— **aggregate-type** is the keyword for the EXPRESS aggregate type, one of: "set" for SET, "bag" for BAG, "list" for LIST, "array" for ARRAY, "list-unique" for LIST OF UNIQUE, "array-unique" for ARRAY OF UNIQUE. The list-of-values form is not used for ARRAY OF OPTIONAL.

— **usage**: When the EXPRESS aggregate type is ARRAY and the upper and lower bounds on the array evaluate to constants, `arraySize` shall be declared `fixed="upper-bound"`, where **upper-bound** is as defined above. When the EXPRESS aggregate type is ARRAY and the upper and lower bounds on the array do not evaluate to constants, `arraySize` shall be declared `use="required"`. Otherwise, `arraySize` shall be declared `use="optional"`.

EXAMPLE 1

In EXPRESS:

```
TYPE compoundplaneanglemeasure = ARRAY [1:3] OF REAL;
END_TYPE;
```

The simple type that supports aggregations of REAL must be defined as `Tns:List-double`, as shown in the example in 7.2.2.3.1. The unrestricted complexType that supports aggregations of REAL must be defined as `Tns:Seq-double`, as shown in the example in 7.2.2.3.2. The XML data type corresponding to `compoundplaneanglemeasure` is:

```
<xs:complexType name="Compoundplaneanglemeasure">
  <xs:simpleContent>
    <xs:restriction base="Tns:Seq-double">
      <xs:simpleType>
        <xs:restriction base="Tns:List-double">
          <xs:minLength value="3"/>
          <xs:maxLength value="3"/>
        </xs:restriction>
      </xs:simpleType>
      <xs:attribute ref="exp:arraySize" fixed="3" />
      <xs:attribute ref="exp:cType" fixed="array" />
    </xs:restriction>
  </xs:simpleContent>
</xs:complexType>
```

EXAMPLE 2

In EXPRESS

```
TYPE length_measure = REAL; END_TYPE;

TYPE measurements = LIST [1:?] OF length_measure;
END_TYPE;
```

The simple type that supports aggregations of `length_measure` must be defined as `Tns:List-Length_measure`, as shown in the example in 7.2.2.3.1. The unrestricted `complexType` that supports aggregations of `length_measure` must be defined in the derived XML schema as shown in the examples in 7.2.2.3.2.

And the XML data type corresponding to the EXPRESS type measurements is:

```
<xs:complexType name="Measurements">
  <xs:simpleContent>
    <xs:restriction base="Tns:Seq-Length_measure">
      <xs:simpleType>
        <xs:restriction base="Tns:List-Length_measure">
          <xs:minLength value="1"/>
        </xs:restriction>
      </xs:simpleType>
      <xs:attribute ref="exp:cType" fixed="list" />
      <xs:attribute ref="exp:arraySize" use="optional" />
    </xs:restriction>
  </xs:simpleContent>
</xs:complexType>
```

7.2.2.3.1 Simple list type supporting aggregations

When the *list-of-values* form for representation of the EXPRESS aggregation data type is specified in Clause 7.2.2, the corresponding XML Schema data type is based on a simple list data type. The simple list data type shall be defined in the derived XML schema by a `simpleType` definition of the form:

```
<xs:simpleType name="list-type" >
  <xs:list itemType="base-type" />
</xs:simpleType>
```

where:

- **list-type** shall be "List-", followed by the local part of the name of the XML schema data type that corresponds to the base-type of the aggregation data type, as specified in 7.2 or 7.3.
- **base-type** shall be the qualified name of the XML schema data type that corresponds to the base-type of the aggregation data type, as specified in 7.2 or 7.3.

Only one such definition is needed for all EXPRESS aggregation types having the same base-type.

EXAMPLE

In EXPRESS:

```
TYPE CompoundPlaneAngleMeasure = ARRAY [1:3] OF REAL;
END_TYPE;

TYPE length_measure = REAL; END_TYPE;

TYPE measurements = LIST [1:?] OF length_measure;
END_TYPE;
```

The corresponding simple types are defined in the derived XML schema as:

```
<xs:simpleType name="List-double">
  <xs:list itemType="xs:double"/>
```

```

</xs:simpleType>

<xs:simpleType name="List-Length_measure">
  <xs:list itemType="Tns:Length_measure"/>
</xs:simpleType>

```

7.2.2.3.2 Unrestricted complexType supporting aggregations

A single unrestricted `complexType` supports all aggregations (ARRAY, BAG, LIST, SET) of a given EXPRESS base-type. The XML data type that corresponds to each specific aggregation data type with that base-type is a restriction of the unrestricted `complexType`.

The derived XML schema shall contain a `complexType` definition for the unrestricted `complexType`. Only one such definition is needed for all EXPRESS aggregation types having the same base-type. The definition shall have the form:

```

<xs:complexType name="seq-type">
  <xs:simpleContent>
    <xs:extension base="Tns:list-type">
      <xs:attribute name="ref" type="xs:IDREF" use="optional"/>
      <xs:attribute ref="exp:arraySize" use="optional"/>
      <xs:attribute ref="exp:itemType" fixed="base-type" />
      <xs:attribute ref="exp:cType" default="set"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>

```

where:

- **Tns:list-type** shall be the qualified name for the simple *list-type* defined in 7.2.2.3.1 above.
- **seq-type** shall be "Seq-", followed by the local part of the name of the XML schema data type that corresponds to the base-type of the aggregation data type, as specified in 7.2 or 7.3.
- **base-type**, the `itemType` attribute value, shall be the qualified name of the XML schema data type that corresponds to the base-type of the aggregation data type, as specified in 7.2 or 7.3.

EXAMPLE 1

In EXPRESS:

```

TYPE CompoundPlaneAngleMeasure = ARRAY [1:3] OF REAL;
END_TYPE;

```

The unrestricted `complexType` for aggregations of REAL must be defined in the derived XML schema, as:

```

<xs:complexType name="Seq-double">
  <xs:simpleContent>
    <xs:extension base="Tns:List-double">
      <xs:attribute name="ref" type="xs:IDREF" use="optional"/>
      <xs:attribute ref="exp:arraySize" use="optional"/>
      <xs:attribute ref="exp:itemType" fixed="xs:double" />
      <xs:attribute ref="exp:cType" default="set"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>

```

This definition appears only once, even if the context schema contains several aggregation types whose base-type is REAL.

EXAMPLE 2

In EXPRESS:

```
TYPE length_measure = REAL; END_TYPE;

TYPE measurements = LIST [1:?] OF length_measure;
END_TYPE;
```

The unrestricted complexType for aggregations of length_measure must be defined in the derived XML schema, as:

```
<xs:complexType name="Seq-Length_measure">
  <xs:simpleContent>
    <xs:extension base="Tns:List-Length_measure">
      <xs:attribute name="ref" type="xs:IDREF" use="optional"/>
      <xs:attribute ref="exp:arraySize" use="optional"/>
      <xs:attribute ref="exp:itemType" fixed="Tns:Length_measure " />
      <xs:attribute ref="exp:cType" default="set"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

This definition appears only once, even if the context schema contains several aggregation types whose base-type is length_measure.

7.2.2.4 Aggregates of aggregation data types

When the base-type of the EXPRESS aggregation data type (called *type A* below) is another aggregation data type, or a defined data type whose fundamental type is another aggregation data type, the corresponding XML schema data type shall have the multi-dimensional array structure. The structure of the corresponding XML schema data type shall be as specified in this subclause.

An EXPRESS aggregation data type (SET, LIST, BAG or ARRAY) is said to be *nested*, if it appears in the context schema as the base-type of another aggregation data type.

Let the *deepest underlying type* of an aggregation type be determined as follows: If the base-type of the aggregation is an anonymous aggregation type, a defined data type whose underlying type is an aggregation type, or a SELECT type whose working select list contains a single defined data type whose underlying type is an aggregation type, the deepest underlying type is the deepest underlying type of that aggregation. Otherwise, the deepest underlying type is the base-type of the aggregation.

Type B shall be the deepest underlying type of type A.

The complexType corresponding to *type A* shall have a content model that only allows elements corresponding to *type B*. In addition, the complexType shall specify fixed values for the exp:itemType attribute, and the exp:cType attribute, as specified below.

The XML data type shall have the form:

```
<xs:complexType>
  <xs:sequence>
    <element-or-group ref="base-instance"
      minOccurs = "lower-bound" maxOccurs = "upper-bound" />
  </xs:sequence>
```

```

<xs:attribute name="ref" type="xs:IDREF" use="optional"/>
<xs:attribute ref="exp:arraySize"    usage/>
<xs:attribute ref="exp:itemType"    fixed="base-instance"/>
<xs:attribute ref="exp:cType"       fixed="aggregate-types"/>
</xs:complexType>

```

where:

— **base-instance**: If *type B* is an entity data type, **base-instance** shall be the qualified name of the corresponding complexEntity group (see 7.5.6.2). If *type B* is a SELECT data type, **base-instance** shall be the qualified name of the corresponding group or instance element (see 7.3.4). Otherwise, **base-instance** shall be the qualified name of the corresponding instance element (see 7.4).

— **element-or-group** is `xs:element`, if **base-instance** is an element, or `xs:group`, if **base-instance** is a group.

NOTE 1 In general, **base-instance** is an element. But **base-instance** is a group if *type B* is a SELECT data type (see 7.3.4) or an entity data type that uses the inheritance-free mapping (see 7.5).

— **aggregate-types** is a list of the keywords for the EXPRESS aggregation data types that are part of the multi-dimensional structure, in order, with the keyword for *type A* first and the keyword for the most nested aggregate last. Each keyword shall be one of: "set" for SET, "bag" for BAG, "list" for LIST, "array" for ARRAY, "list-unique" for LIST OF UNIQUE, "array-unique" for ARRAY OF UNIQUE, array-optional for ARRAY OF OPTIONAL, and array-optional-unique for ARRAY OF OPTIONAL UNIQUE. The items in the list shall be separated by whitespace.

— **lower-bound**: When none of the base-types are OPTIONAL, and the lower bound on the size of each collection is given and evaluates to a constant, the value of the `minOccurs` attribute shall be specified as the product of the lower bounds of each aggregation data type. Otherwise, the value of the `minOccurs` attribute shall be 0. If the lower bound evaluates to '1', the `minOccurs` attribute may be omitted.

— **upper-bound**: When the upper bound on the size of each collection is given and evaluates to a constant, the value of the `maxOccurs` attribute shall be specified as the product of the upper bounds of each aggregation data type. Otherwise, the value shall be "unbounded".

— **usage**: When type A and all of the nested aggregation data types are ARRAYS, and none of them are optional, and all of the index range bounds on type A and the nested ARRAYS evaluate to constants, `arraySize` shall be declared `fixed="size1 ... sizeN"`, where `size1 ... sizeN` are the sizes of *type A* and all the nested aggregates that are part of the multi-dimensional structure, in order with the size of *type A* first and the size of the most nested aggregate last. When type A and all of the nested aggregation data types are ARRAYS, and none of them are optional, and the index range bounds on type A or any of the nested ARRAYS does not evaluate to a constant, `arraySize` shall be declared `use="required"`. Otherwise, `arraySize` shall be declared `use="optional"`.

NOTE 2 If the aggregate is an ARRAY, what is given in the EXPRESS syntax is the LOINDEX and HIINDEX values — the subscript range. The size of the dimension is the difference between the index values plus 1.

NOTE 3 The multi-dimensional array form is a special case of the "sequence-of-elements" form specified in 7.2.2.2. It has the same structure, but the component instance elements represent values of the deepest underlying type instead of the base-type of *type A*. The proper interpretation of this structure depends on the values of `arraySize` and possibly `cType`.