

Geographic information — Data description — Transfer

ICS 07.040; 35.240.70

National foreword

This Draft for Development is the English language version of ENV 12658:1998.

This publication is not to be regarded as a British Standard.

It is being issued in the Draft for Development series of publications and is of a provisional nature due to the limited duration of the European Prestandard. It should be applied on this provisional basis, so that information and experience of its practical application may be obtained.

Comments arising from the use of this Draft for Development are requested so that UK experience can be reported to the European organization responsible for its conversion into a European Standard. A review of this publication will be initiated 2 years after its publication by the European organization so that a decision can be taken on its status at the end of its three-year life. The commencement of the review period will be notified by an announcement in *Update Standards*.

According to the replies received by the end of the review period, the responsible BSI Committee will decide whether to support the conversion into a European Standard, to extend the life of the prestandard or to withdraw it. Comments should be sent in writing to the Secretary of BSI Technical Committee IST/36, Geographic information, at 389 Chiswick High Road, London W4 4AL, giving the document reference and clause number and proposing, where possible, an appropriate revision of the text.

A list of organizations represented on this committee can be obtained on request to its secretary.

Cross-references

The British Standards which implement international or European publications referred to in this document may be found in the BSI Standards Catalogue under the section entitled “International Standards Correspondence Index”, or by using the “Find” facility of the BSI Standards Electronic Catalogue.

Summary of pages

This document comprises a front cover, an inside front cover, the ENV title page, pages 2 to 100, an inside back cover and a back cover.

The BSI copyright notice displayed in this document indicates when the document was last issued.

This Draft for Development, having been prepared under the direction of the DISC Board, was published under the authority of the Standards Committee and comes into effect on 15 October 1999

© BSI 10-1999

ISBN 0 580 35124 6

Amendments issued since publication

Amd. No.	Date	Comments

ICS 07.040; 35.240.70

Descriptors: geographic information, data, data processing, data transfer, description

English version

Geographic information - Data description - Transfer

Information géographique - Description des données -
Transfert

Geoinformation - Datenbeschreibung - Übertragung

This European Prestandard (ENV) was approved by CEN on 9 October 1998 as a prospective standard for provisional application.

The period of validity of this ENV is limited initially to three years. After two years the members of CEN will be requested to submit their comments, particularly on the question whether the ENV can be converted into a European Standard.

CEN members are required to announce the existence of this ENV in the same way as for an EN and to make the ENV available promptly at national level in an appropriate form. It is permissible to keep conflicting national standards in force (in parallel to the ENV) until the final decision about the possible conversion of the ENV into an EN is reached.

CEN members are the national standards bodies of Austria, Belgium, Czech Republic, Denmark, Finland, France, Germany, Greece, Iceland, Ireland, Italy, Luxembourg, Netherlands, Norway, Portugal, Spain, Sweden, Switzerland and United Kingdom.



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

Central Secretariat: rue de Stassart, 36 B-1050 Brussels

Contents

Foreword	3
Introduction.....	4
1 Scope.....	6
2 Normative references.....	6
3 Definitions.....	7
4 Overview	10
4.1 Relationships of family of European Prestandards used in data transfer	10
4.2 Data transfer.....	10
4.3 Role of EXPRESS language	12
4.4 Role of ENV ISO 10303-21:1994	12
5 Application schema	12
5.1 General.....	12
5.2 Use of EXPRESS language	13
5.2.1 General.....	13
5.2.2 Interface specification.....	14
5.2.3 SUPERTYPEs and SUBTYPEs	14
5.3 Identification of application schemas	14
5.4 Transfer of sound, video and other binary data	14
6 Data transfer content	15
6.1 General.....	15
6.2 Encoded application schema	15
6.3 Geographic Data	16
6.3.1 Data_encoding schema.....	16
6.3.2 Data type length limits	18
7 Data transfer	18
7.1 General.....	18
7.2 File transfer.....	18
7.2.1 General.....	18
7.2.2 File datasets	18
7.2.3 Header section	18
7.2.4 Data section.....	19
7.2.5 Short names	19
7.2.6 Character Set	19
7.3 Message handling	20
Annex A (normative) "application_encoding" schema	21
Annex B (normative) "data_encoding" schema	32
Annex C (normative) "file_transfer" schema.....	37
Annex D (normative) "messaging" schema.....	38
Annex E (normative) "external_files" schema.....	39
Annex F (normative) Table of short names.....	40
Annex G (informative) Data users' guide	42
Annex H (informative) Application developers' guide	53
Annex I (informative) Bibliography.....	100

Foreword

This European Prestandard has been prepared by Technical Committee CEN/TC 287 "Geographic Information", the secretariat of which is held by AFNOR.

According to the CEN/CENELEC Internal Regulations, the national standards organizations of the following countries are bound to announce this European Prestandard: Austria, Belgium, Czech Republic, Denmark, Finland, France, Germany, Greece, Iceland, Ireland, Italy, Luxembourg, Netherlands, Norway, Portugal, Spain, Sweden, Switzerland and the United Kingdom.

Introduction

This European Prestandard refers to the transfer of geographic data and consists of the following clauses and annexes :

- clause 1 contains the scope of this European Prestandard ;
- clause 2 contains the normative references ;
- clause 3 contains the definitions of terms ;
- clause 4 contains an overview of this European Prestandard ;
- clause 5 concerns the creation of an application schema ;
- clause 6 defines the structure of the transfer content ;
- clause 7 defines the transfer schemas, the encoding rules and the implementations of transfer ;
- annex A contains the normative EXPRESS schema "application_encoding" ;
- annex B contains the normative EXPRESS schema "data_encoding" ;
- annex C contains the normative EXPRESS schema "file_transfer" ;
- annex D contains the normative EXPRESS schema "messaging" ;
- annex E contains the normative EXPRESS schema "external_files" ;
- annex F contains the normative short names for ENV ISO 10303-21:1994 encoding rules ;
- annex G contains a complete annotated example of a file transfer, as a guide for users ;
- annex H contains an application developers' guide for this European Prestandard ;
- annex I contains a bibliography.

The requirements imposed by this European Prestandard are contained in the numbered clauses, with their supporting detail in the normative Annexes A to F. These clauses and annexes provide the reference information for users of this European Prestandard. Annexes A to E contain schemas written in the EXPRESS language, whose names are given above. Whenever these schema names are used in the text of this European Prestandard, then they are enclosed in double quotes ". Whenever entity names or attribute names from these schemas are used in the text, they are also enclosed in double quotes ". Whenever keywords from the EXPRESS language are used, they are written in CAPITAL LETTERS.

Annex G gives an annotated example of a file transfer that conforms with this European Prestandard as an aid to understanding the overall structure imposed by this European Prestandard. This annex is suitable as a starting point for those who wish to understand the overall working of this European Prestandard, and uses information from all the clauses and normative annexes.

Annex H contains many small examples of the use of this European Prestandard, with explanatory comments. Many of the examples in Annex H are partial, each aiming to illustrate a specific point. This annex also includes two complete examples. This annex will be useful for developing software to use this European Prestandard, since it illustrates some of the situations that may arise.

Annex I provides informative detail only.

1 Scope

This European Prestandard specifies a method for the transfer of geographic data and their metadata. It defines the transfer schemas, implementation mechanisms and encoding rules for the transfer of geographic data. This European Prestandard supports transfer both for data defined by standardised schemas and for data defined by application specific schemas.

The implementation mechanisms are based on the formal data description language EXPRESS (ENV ISO 10303-11:1994) and are intended to cover file transfer and message handling communication services. The data are to be encoded in ENV ISO 10303-21:1994 clear text encoding. Decoding rules are not explicitly defined, since decoding is the reverse process of encoding.

The component parts of a transfer are described in detail in other European Prestandards for Geographic Information.

2 Normative references

This European Prestandard incorporates by dated or undated reference, provisions from other publications. These normative references are cited at the appropriate places in the text and the publications are listed hereafter. For dated references, subsequent amendments to or revisions of any of these publications apply to this European Prestandard only when incorporated in it by amendment or revision. For undated references the latest edition of the publication referred to applies.

ENV 12009:1997, *Geographic Information - Reference Model*.

ENV 12160:1997, *Geographic Information - Data Description - Spatial schema*.

ENV 12656:1998, *Geographic Information - Data Description - Quality*.

ENV 12657:1998, *Geographic Information - Data Description - Metadata*.

prENV 12762:1998, *Geographic Information - Referencing - Direct position*.

prCR 12660:1998, *Geographic Information - Processing - Query and Update* *spatial aspects*

ENV ISO 10303-1:1995, *Industrial automation systems and integration - Product data representation and exchange - Part 1 : Overview and fundamental principles (ISO 10303-1:1994)*

ENV ISO 10303-11:1994, *Industrial automation systems and integration - Product data representation and exchange - Part 11 : Description methods - The EXPRESS language reference manual (ISO 10303-11:1994)*

ENV ISO 10303-21:1994, *Industrial automation systems and integration - Product data representation and exchange - Part 21 : Clear text encoding of the exchange structure (ISO/DIS 10303-21:1993)*

ISO 8859-1: 1987, *Information processing - 8-bit single-byte coded graphic character sets - Part 1 : Latin alphabet No. 1*.

ISO 8859-2:1987, *Information processing - 8-bit single-byte coded graphic character sets - Part 2 : Latin alphabet No. 2*.

ISO 8859-5:1988, *Information processing - 8-bit single-byte coded graphic character sets - Part 5 : Latin / Cyrillic alphabet*.

ISO 8859-7:1987, *Information processing - 8-bit single-byte coded graphic character sets - Part 7 : Latin / Greek alphabet*.

ISO/IEC 10646-1:1993, *Information technology - Universal multiple-octet coded character set (UCS) - Part 1 : Architecture and basic multilingual plane*.

3 Definitions

For the purposes of this European Prestandard, the following definitions apply :

NOTE The use of terms within quotation marks in this European Prestandard is used to denote entities and attributes within EXPRESS schemas, as explained in the Introduction to this European Prestandard. Therefore explanations of such terms are not included in this clause.

3.1

aggregated attribute

Collection of attributes treated as a unit

3.2

application schema

Conceptual schema for a specific field of interest within the field of geographic information

[ENV 12009]

3.3

attribute

Representation of an essential trait, quality or property of an object or entity

3.4

clear text encoding

Encoding of information that only uses 8-bit byte values corresponding to the set of characters G(02/00) to G(07/14) of ISO 8859-1

[ENV ISO 10303-21:1994]

3.5

data transfer

Movement of data between systems, which may comprise file transfer or message handling

3.6

data type

Domain of values.

[ENV ISO 10303-11:1994]

3.7

dataset

Identifiable collection of data

3.8

domain

Set of permissible values that an attribute may take

3.9

encoding rules

Rules defining possible data structures, data types and symbols which are used in the representation of data

3.10

entity

Class of information defined by common properties

[ENV ISO 10303-11:1994]

3.11

exchange structure

Computer-interpretable format used for storing, accessing, transferring and archiving data

[ENV ISO 10303-1:1994]

3.12

external file

File whose encoding is not defined by this European Prestandard

3.13

file transfer

Data transfer using files, through a data network or unconnected media

3.14

geographic data

Computer treatable form of geographic information

[ENV 12009]

3.15

geographic dataset

Identifiable collection of geographic data

[ENV 12656]

3.16

geographic information

Information concerning phenomena directly or indirectly associated with a location relative to the earth

[ENV 12009]

3.17

geometric primitive

Description, partial or total, of the spatial aspects of an object by means of coordinates and mathematical functions

[ENV 12160]

3.18

grid

Point distribution defining a regular pattern, derived from the corners of the tessellation of a specific frame or a limited part of it

[ENV 12160]

3.19

instance

Named value

[ENV ISO 10303-11:1994]

3.20

interface specification

Mechanism used in the EXPRESS language for relating two schemas written in that language to each other

(As used in ENV ISO 10303-11:1994)

3.21

message handling

Data transfer using messages through a data network

3.22

metadata

Data about a geographic dataset or geographic datasets

[ENV 12657]

3.23

metadata schema

Schema describing metadata

[ENV 12009]

3.24

object

Single phenomenon existing in the real world

[ENV 12160]

3.25

plain attribute

Attribute that has a domain that is defined as a simple data type

NOTE Simple data type is used in ENV ISO 10303-11:1994.

3.26

quality

Totality of characteristics of a product that bear on its ability to satisfy stated and implied needs

[EN ISO 8402]

3.27

quality schema

Conceptual schema for the quality aspects of geographic data

[ENV 12009]

3.28

query

Operation to select and retrieve data from a database without changing their contents

[CR 12660]

3.29

raster band

2-dimensional geometric primitive which is a limited rectangular part of a specific 2-dimensional frame

[ENV 12160]

3.30

sequential file

File that can only be accessed in a sequential manner

[ENV ISO 10303-21:1994]

3.31

spatial schema

Conceptual schema for the spatial (geometric and topological) aspects of geographic data

[ENV 12009]

3.32

topological primitive

Description, partial or total, of the topological aspects of an object

[ENV 12160]

3.33

transfer schema

Conceptual schema for rules and operators for transferring geographic data and metadata

[ENV 12009]

4 Overview

4.1 Relationships of family of European Prestandards used in data transfer

This European Prestandard is part of the family of European Prestandards for geographic information that are each published separately and together provide for the description, structure, implementation, application and transfer of data. The content of this European Prestandard is mostly self-explanatory and the requirements for structuring and organising data are defined in the "file_transfer" and "messaging" schemas (clause 7).

4.2 Data transfer

This European Prestandard defines an overall structure of geographic data and their metadata to be transferred. Data transfer may be done by file transfer or message handling.

NOTE 1 For file transfer using a network, the network connection exists while the transfer is in progress and is then closed. File transfer between the source and the destination can be controlled by a third party, who initiated the transfer request. Real time data transfer and bulk data transfer are the most important types of file transfer. File transfer can also be carried out on unconnected media such as optical disk, magnetic tape or diskette. This can often happen with very large amounts of data.

The contents of a data transfer may consist of :

- metadata ;
- application schema description ;
- geographic data, collected in datasets ;
- quality data.

NOTE 2 Updates are treated in the same way as any other data transfer. They are not discussed as a separate case in this European Prestandard. Update data can be transferred if modelled in the application schema.

Figure 1 shows the framework for preparing data for transfer, and also an overview of the related clauses of this European Prestandard.

The application schema allows suppliers to define geographic entities using standardised entities defined in other European Prestandards for geographic information. In particular, the geographic entities are described in the application schema with reference to the spatial schema and to the quality schema. The application data shall be in conformance with the entities defined in the application schema. They form an application dataset.

NOTE 3 For example, if the application schema defines that a "tree" only has one attribute which is "date_of_planting", then in the application data each "tree" can only have an attribute that is "date_of_planting" and nothing else.

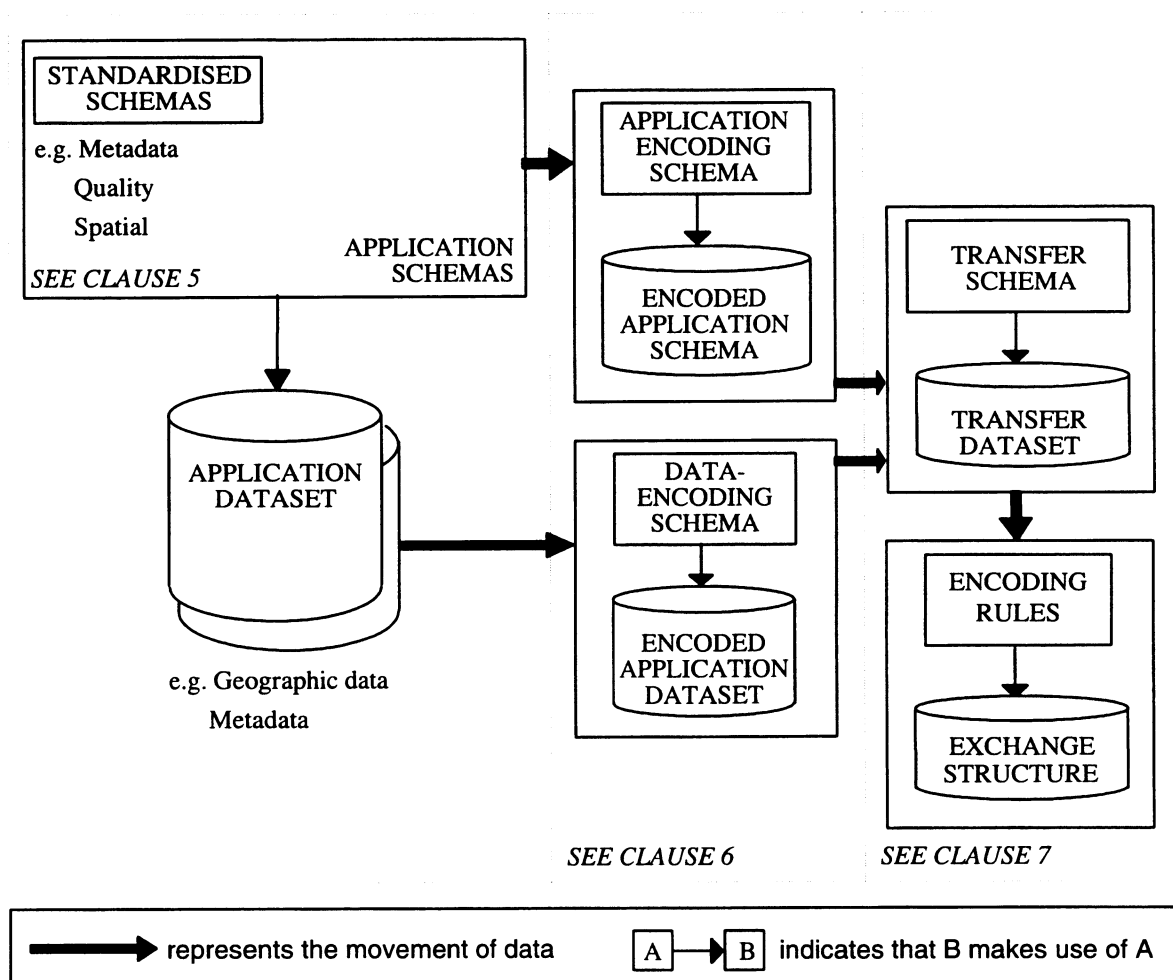


Figure 1 - Diagram of preparation of data for transfer

The application schema shall be translated into an encoded application schema in accordance with the schema "application_encoding" (as defined in clause 6). The related application dataset shall be translated into an encoded dataset, together with the encoded application schema, in accordance with the schema "data_encoding" (also defined in clause 6).

NOTE 4 The use of the schema "data_encoding" differs from the approach usually taken in the use of ENV ISO 10303-11:1994 and ENV ISO 10303-21:1994. This schema has been written for two main reasons. Firstly it provides a greater degree of independence from the national language of the supplier, and secondly it ensures that different data transfers will be similar in appearance to each other, by controlling the instances that can appear in an exchange structure.

The data transfer may contain one or more datasets. The exchange structure of a data transfer (file transfer or message handling) shall be produced by applying the encoding rules defined in this European Prestandard (as defined in clause 7).

For file transfer, additional rules shall be applied. If metadata are to be transferred, the metadata set shall be collected eventually together with encoded geographic data to form a "file_dataset". Several "file_dataset"s may be collected in a "transfer_set" in accordance with the schema "file_transfer".

For message handling, a "message_set" contains a "message_dataset".

4.3 Role of EXPRESS language

This European Prestandard uses the EXPRESS language (as defined in ENV ISO 10303-11:1994) as its formal description technique.

NOTE 1 The EXPRESS language has been designed to enable partitions within the field of data transfer, and schemas are the tools to allow this partitioning. The primary intent of the EXPRESS language is to define entities. An entity is described by means of its attributes, which in turn are defined by values in a domain. Attributes have names, which explain their significance, and have as their domain a simple data type (e.g., a string), or another entity.

The EXPRESS schemas of this European Prestandard refer to EXPRESS schemas defined in other European Prestandards for geographic information. The EXPRESS schemas of this European Prestandard are normative and are given in Annexes A to E.

Suppliers shall use the EXPRESS language when creating an application schema (as defined in clause 6), to describe the information to be transferred. This schema shall be sent to the user to permit interpretation of the data.

NOTE 2 A supplier can choose not to use the EXPRESS language to describe their data within the source system. However, to complete a data transfer, an application schema written in EXPRESS is required. The use of EXPRESS ensures an unambiguous statement of relationships between entities being transferred.

4.4 Role of ENV ISO 10303-21:1994

This European Prestandard allows data representing geographic information, whose semantics are described in an application schema using the EXPRESS language, to be transferred between one computer system and another. The exchange structure (as defined in clause 7) is encoded using ENV ISO 10303-21:1994.

ENV ISO 10303-21:1994 defines an exchange structure for data whose semantics have been described in an EXPRESS schema. The exchange structure uses a clear text encoding and is implemented as a sequential file. ENV ISO 10303-21:1994 defines the encoding rules to be used to create the exchange structure for instances of entities described in EXPRESS schemas. This European Prestandard uses these encoding rules.

5 Application schema

5.1 General

An application schema is a formal description of a particular field of interest within the field of geographic information. For data transfer to take place, the supplier and user both need to understand the nature of the objects being represented and their relationships. The application schema is the way in which the objects and relationships are rigorously described. This rigorous description is used to govern both the appearance of the data and its description, within the data transfer.

NOTE 1 This European Prestandard does not define any application schemas. It defines the rules for creating an application schema for transferring geographic data. This approach of not defining application schemas, but instead providing rules for suppliers to define their own application schemas, is deliberate. It is taken since users of European Prestandards for geographic information have different requirements of their geographic applications and of the geographic data managed by applications.

Although the application schema is not standardised, the method of its description is standardised. This European Prestandard defines constraints on the use of the EXPRESS language. Application schemas may import basic entities defined in other European Prestandards for geographic information.

This clause gives rules on how to use the EXPRESS language to construct an application schema. For this European Prestandard the application schema shall be written to conform to ENV ISO 10303-11:1994, and to the additional rules that appear in this clause.

NOTE 2 The application schema describes in more detail the data_definition_metadata schema and classification_metadata schema defined in ENV 12657.

5.2 Use of EXPRESS language

5.2.1 General

The EXPRESS language elements used in an application schema shall be drawn from the following list of EXPRESS language elements, which are listed together with the clause of ENV ISO 10303-11:1994 where they are described. If any other parts of the EXPRESS language are used to construct an application schema, then they cannot be transferred to the user. Application schemas for transfer shall therefore not use any other parts of the EXPRESS language.

EXPRESS element	ENV ISO 10303-11:1994 clause
Data types	8
Simple data types	8.1
Real data type	8.1.2
Integer data type	8.1.3
Logical data type	8.1.4
Boolean data type	8.1.5
String data type	8.1.6
Binary data type	8.1.7
Aggregation data types	8.2
Array data type	8.2.1
List data type	8.2.2
Bag data type	8.2.3
Set data type	8.2.4
Defined data type	8.3.2
Enumeration data type	8.4.1
Select data type	8.4.2
Domain rule	9.1
Entity declaration	9.2
Attribute	9.2.1.1
Derived attribute	9.2.1.2
Inverse attribute	9.2.1.3
Uniqueness rule	9.2.2.1
Domain rule	9.2.2.2
Subtype	9.2.3
Supertype	9.2.3
Schema declaration	9.3
Constant declaration	9.4
Function declaration	9.5.1
Procedure declaration	9.5.2
Rule declaration	9.6
Reference interface specification	11.2

In accordance with ENV ISO 10303-11:1994, identifiers are case-insensitive, they start with a letter, and they are made up of letters, "_" and digits only. A user-defined identifier shall not be a standard EXPRESS keyword.

NOTE It is suggested that user-defined identifiers within an application schema start with a lower-case letter, and be made up of lower-case letters, "_" and digits only. This suggestion is made since using lower-case visually distinguishes user-defined identifiers from standard EXPRESS language keywords that are conventionally written in upper-case in the application schema. Within the application schema user-defined identifiers are given to the schema, and also to entities, attributes, types, constants, functions, procedures, and rules.

5.2.2 Interface specification

This clause defines the use of the EXPRESS interface specifications. The REFERENCE interface specification shall only be used in an application schema to interface with schemas defined in European Prestandards for geographic information. Specifically, this specification shall only be used to incorporate the spatial and quality schemas that conform to ENV 12160 and ENV 12656 respectively and the "external_files" schema (see 5.6). REFERENCE may be used to interface with a whole schema or named constructs within a schema.

NOTE 1 As a result, if an application schema has been constructed in interfaced parts (excluding schemas that conform to European Prestandards for geographic information), then these parts are to be combined to give a single application schema that contains all user-defined entities and types.

The USE interface specification shall not be used within an application schema. However, standardised schemas defined in European Prestandards for geographic information may use the USE interface specification.

NOTE 2 This rule exists to prevent (for example) the inclusion in a transfer of geometric primitives that are not associated with an instance of an entity. When one schema interfaces with another by using REFERENCE, then the entities of the REFERENCED schema can only appear as attributes of entities defined in the first schema.

5.2.3 SUPERTYPEs and SUBTYPEs

Only explicit supertyping shall be used.

NOTE 1 Explicit supertyping is used in EXPRESS to mean that if ENTITY A states that it is a SUBTYPE of ENTITY B, then ENTITY B also states that it is a SUPERTYPE of ENTITY A. As a result, a supplier cannot create an entity that is a SUBTYPE of an entity in the standardised spatial schemas. This agrees with the rules laid down by ENV 12160.

A SUPERTYPE shall only use ONEOF.

NOTE 2 The use of ONEOF conforms with mainstream object-oriented modelling. This rule is applied to keep the encoded exchange structure simple.

5.3 Identification of application schemas

Each application schema shall include a name and a version identification.

NOTE The inclusion of version identification ensures that a supplier and a user agree which version of the application schema describes the contents of a particular data transfer.

5.4 Transfer of sound, video and other binary data

Sound, video and binary data may be transferred as a binary attribute within the data transfer. To do this the supplier shall define a binary attribute.

NOTE 1 The encoding used to create an exchange structure can impose a limit on the number of bytes that can be used for any attribute. ENV ISO 10303-21:1994 imposes a maximum of 32767 bytes.

Alternatively, the supplier may use external files for the transfer of such data. To do this the supplier shall make use of the entity "external_file", when creating their application schema. This entity is defined in the schema "external_files" given in Annex E. An entity that has an "external_file" as an attribute may also have other attributes that are relevant to the interpretation of the external file (for example location or description of camera lens).

NOTE 2 The mechanism for transferring files is outside the scope of this European Prestandard. As a result, the data transfer of external files does not form part of the standardised geographic data transfer.

6 Data transfer content

6.1 General

A data transfer may contain an encoded application schema or application data or both. This clause defines how to map the application schema into the encoded application schema in the exchange structure, (using the schema "application_encoding": see Annex A) and how to map the data (using the schema "data_encoding": see Annex B) into the exchange structure.

6.2 Encoded application schema

Suppliers shall use the EXPRESS language when creating an application schema to describe the data to be transferred. This schema shall be sent to the user, to permit interpretation of the data. To allow this, this European Prestandard defines an EXPRESS schema, called "application_encoding", which defines how the EXPRESS application schema itself is to be transferred.

The rules for creating the encoded application schema are given in the schema "application_encoding", in Annex A. The schema "application_encoding" describes those parts of the EXPRESS language that are used to write an application schema, as defined in clause 5.2.

NOTE 1 This approach of encoding the application schema, instead of transferring it as plain text has been adopted for several reasons :

- it ensures a certain level of conformance testing on the application schema since its encoding will be carried out automatically ;
- it ensures that the restrictions on the use of the EXPRESS language that this European Prestandard imposes are followed ;
- it allows the inclusion of the encoded application schema in the same file as the encoded data, thereby making the connection between the data and their schema clear and unambiguous.

It is permitted to transfer the application schema as a plain text file in addition to its encoded form, by using the attribute "plain_text_file" of the entity "schema_details".

If the application schema as plain text conflicts with the encoded application schema, then the encoded application schema shall take precedence.

Where the encoded application schema has been previously transferred from a supplier to a user, then in subsequent transfers the encoded application schema may contain only a reference to the name and version of the application schema.

NOTE 2 The schema "application_encoding" parallels some of the work towards creating ISO 10303-22 (see Annex I), but has a different objective.

Table 1 shows the mapping of application schema elements to instances within the schema "application_encoding". The relationships between instances in the schema "application_encoding" are explained in Annex A.

NOTE 3 "object_type" in ENV 12657 maps to an EXPRESS entity in this European Prestandard and an "attribute_type" in ENV 12657 maps to an attribute of an EXPRESS entity. If "relationship_type" in ENV 12657 has attributes then it maps to an EXPRESS entity but if it does not have attributes then it maps to an attribute of an EXPRESS entity. The mapping used can be made explicit in an exchange structure by the use of the entity "declaration_mapping", as defined in the schema "data_encoding".

Table 1 - Mapping of application schema elements to schema "application_encoding" instances

Application schema	Encoded application schema instances (schema "application_encoding")
Real data type	real_data_type
Integer data type	integer_data_type
Logical data type	logical_data_type
Boolean data type	boolean_data_type
String data type	string_data_type
Binary data type	binary_data_type
Array data type	array_data_type
List data type	list_data_type
Bag data type	bag_data_type
Set data type	set_data_type
Defined data type	defined_data_type
Enumeration data type	enumeration_data_type
Select data type	select_data_type
Entity declaration	entity_declaration
Attribute	attribute_declaration
Derived attribute	derived_attribute_declaration
Inverse attribute	inverse_attribute_declaration
Uniqueness rule	unique_rule_declaration
Domain rule	domain_rule_declaration
Subtype	entity_declaration (attribute is_a_sub_of)
Supertype	entity_declaration (attribute is_a_super_of)
Schema declaration	schema_declaration
Constant declaration	constant_declaration
Function declaration	function_declaration
Procedure declaration	procedure_declaration
Rule declaration	rule_declaration
Reference interface specification	schema_declaration (attribute external_schemas)
Comment	not mapped
Application schema name	schema_details (attribute schema_name)
Application schema version	schema_details (attribute schema_version)

6.3 Geographic Data

6.3.1 Data_encoding schema

The data are encoded for transfer using the schema "data_encoding". (See Annex B)

The data consist of a collection of instances of entities. (The entities are defined in an application schema.) This collection is a "dataset" in the exchange structure. The connection between the instances of entities and their parent "dataset" is made through each instance indicating to which "dataset" it belongs. The "dataset" does not explicitly list all the instances that it contains.

The description of every entity in the exchange structure (including its sequence of attributes) is defined by the application schema in which they are described. This may be deduced from either the application schema written in the EXPRESS language, or from its representation in the encoded application schema.

Every entity to be transferred is encoded as an instance of the entity "express_entity". This refers to an "entity_identifier", and to its attributes. The "entity_identifier" connects the instance with the corresponding entry in the encoded application schema, by giving the name of the corresponding "entity_definition" in an encoded application schema, and the name and version of the application schema from which the encoded application schema has been constructed.

The attributes of an "express_entity" are defined in the encoded application schema. Each attribute shall be either a plain attribute (string, real number, integer, etc.), an "express_entity" (for association with another express_entity), a "geometric_primitive" or a "topological_primitive" (which are defined in the "spatialGO" schema), a "quality_elements" (which is defined in the "quality_schema" schema) or an "external_file" (which is defined in the "external_files" schema).

The connection between the attribute instances and their definition is not directly from the attributes to the encoded application schema, but through the "express_entity" and the order of the attributes. The order of the attributes in the data transfer shall be the same as that defined in the encoded application schema, allowing for the EXPRESS rules governing inheritance.

NOTE The order in which the attributes are to be transferred is defined by ENV ISO 10303-11:1994, and can be summarised as follows. If an entity in the application schema is defined as a SUBTYPE of one or more entities, then it inherits the attributes defined for those entities, which are called its supertypes. These inherited attributes are transferred first, and within the inherited attributes, the order of transfer follows the listed order of the supertypes.

Table 2 shows the mapping of elements of application data to instances within the "data_encoding" schema. The relationships between instances in the "data_encoding" schema are explained in Annex B.

Table 2 - Mapping of data instances to EXPRESS entity instances

Application data	Encoded data instances ("data_encoding" schema)
Real value	real_value
Integer value	integer_value
Logical value	logical_value
Boolean value	boolean_value
String value	string_value
Binary value	binary_value
Array value	aggregate_attribute
List value	aggregate_attribute
Bag value	aggregate_attribute
Set value	aggregate_attribute
Enumerated value	enumerated_value
Entity instance	express_entity
Attributes: plain aggregated other entities	attribute value reference to aggregate_attribute instance name reference to express_entity instance name
Inverse attribute	not mapped
Derived attribute	not mapped

6.3.2 Data type length limits

The EXPRESS language does not impose any limits on the length of the various data types. This European Prestandard imposes a maximum length of 32767 characters for the value of any attribute in the data transfer.

NOTE This limit arises from the use of ENV ISO 10303-21:1994 for the exchange structure.

7 Data transfer

7.1 General

The exchange structure shall be ENV ISO 10303-21:1994 for file transfer and for message handling. Encoding rules are the rules to go from a formal representation of an application dataset and the data it describes to an exchange structure. Decoding rules are the reverse of encoding rules.

File transfer is defined in 7.2 and Annex C, message handling in 7.3 and Annex D. Encoding rules for file transfer are defined in 7.2.2. Encoding rules for message handling are defined in 7.3.

7.2 File transfer

7.2.1 General

This clause describes rules controlling the appearance of data within a plain text representation of the data. This representation is based on ENV ISO 10303-21:1994. An ENV ISO 10303-21:1994 exchange structure contains one header section and one data section. This European Prestandard does not describe the encoding rules defined by ENV ISO 10303-21:1994. It only lists any additional rules that shall be applied.

7.2.2 File datasets

The content of an exchange structure for file transfer is defined by the schema "file_transfer" (see Annex C). The exchange structure contains a single "transfer_set".

Within the "transfer_set" there shall be "file_dataset"s. Each "file_dataset" is a "dataset" (as described in 6.3.1).

Each "file_dataset" is independent of the others. The option of putting more than one "file_dataset" in the same "transfer_set" is only provided for convenience of file transfer.

7.2.3 Header section

An exchange structure shall contain certain information that describes the data transfer itself, but not the data content. This is defined in ENV ISO 10303-21:1994. It is made up as follows :

- description : specifies the edition of ENV ISO 10303-21:1994 that has been used to create the exchange structure ;
- name : plain text information about the exchange structure. This includes the supplier, and time of supply, and the software used to create the exchange structure ;
- schema : an indication of the schema used to create the exchange structure. This shall be "file_transfer" (as defined in Annex C).

NOTE If the header entity "description" is used for information such as a copyright statement, a security statement or a description of each dataset in the "transfer_set", then in each case the correct and complete form of this information is contained in the metadata within the data section of the "transfer_set", and if that form disagrees with the information in the "description" then the form in the metadata takes precedence.

7.2.4 Data section

The "transfer_set" instances shall be encoded as defined in ENV ISO 10303-21:1994 according to the EXPRESS schema "file_transfer", and shall conform to the following rules which concern the order of instances in the data section as shown in figure 2.

NOTE The SCOPE construct in ENV ISO 10303-21:1994 is not sufficiently well defined for use in this European Prestandard.

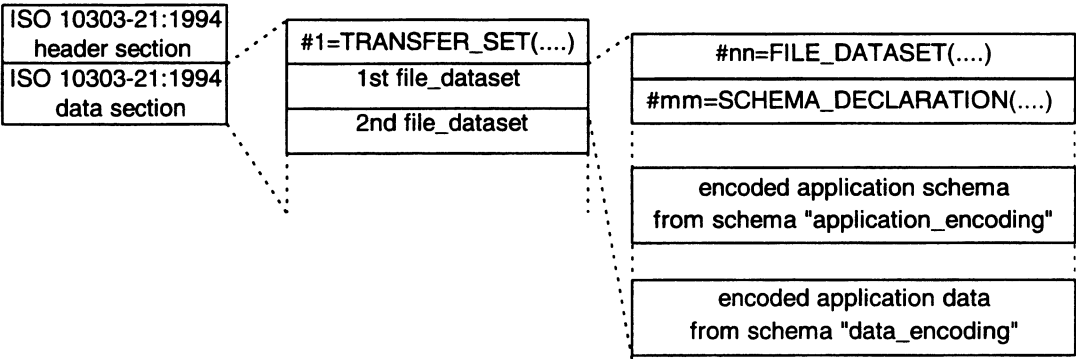


Figure 2 - Order of "transfer_set" instances for file transfer

The first instance in the data section shall be "transfer_set" and its instance name shall be #1.

If the "transfer_set" contains more than one "file_dataset", then all the instances for a single "file_dataset" shall appear together in the exchange structure.

This European Prestandard does not define any sequence of instances within the "file_dataset" contents.

This European Prestandard does not define any sequence of instance names.

7.2.5 Short names

This European Prestandard defines alternative short names for all the entities that are defined in the schemas of this European Prestandard. These are listed in Annex F. These short names may be used in an exchange structure in place of the entity names used in the schemas themselves. Short names and long names shall not be mixed within an exchange structure. An exchange structure shall only be made up of instances taken from Annex F.

7.2.6 Character Set

The exchange structure of ENV ISO 10303-21:1994 is a continuous stream of 8-bit bytes that are encodings of the graphic characters of the Latin-1 alphabet of ISO 8859-1 (decimal values 32 to 126).

For string values, the use of control directives is required to encode the portion of ISO 8859 not included in the basic alphabet, or to include ISO/IEC 10646-1 characters in a string (as described in ENV ISO 10303-21:1994).

7.3 Message handling

The content of a "message_set" is defined by the schema "messaging" (see Annex D). Within a "message_set" there shall be one "message_dataset".

NOTE The order of instances for messaging shall be the same as for file transfer, as shown in Figure 2.

This clause describes rules controlling the appearance of data within a plain text representation of the data. This representation is based on ENV ISO 10303-21:1994. An ENV ISO 10303-21:1994 exchange structure contains one header section and one data section. This European Prestandard does not describe the encoding rules defined by ENV ISO 10303-21:1994. It only lists any additional rules that shall be applied.

The additional rules for message handling are those defined in 7.2.5 and 7.2.6 of this European Prestandard.

Annex A (normative)

"application_encoding" schema

This annex provides a representation of the conceptual schema for representing an application schema, which has been written in the EXPRESS language, as defined in ENV ISO 10303-11:1994. This representation is itself written in EXPRESS. Figure A.1 shows a subset of this information in EXPRESS-G, the graphical form of the EXPRESS language.

```

SCHEMA application_encoding;

(* Description:
   application_encoding this schema defines how an application schema,
                           written in EXPRESS, shall be encoded.
*)
REFERENCE FROM external_files(external_file);

ENTITY aggregate_data_type
ABSTRACT SUPERTYPE OF (ONEOF (list_data_type, set_data_type,
                                bag_data_type, array_data_type));
    aggregation_of :      type_declaration;
END_ENTITY;

(* Descriptions:
   aggregate_data_type  collection of the available EXPRESS aggregation
                           types.
   aggregation_of       type that is being aggregated.
*)

ENTITY array_data_type
SUBTYPE of (aggregate_data_type);
    low_bound :          INTEGER;
    high_bound :         INTEGER;
    uniqueflag :         OPTIONAL BOOLEAN;
    option :             OPTIONAL BOOLEAN;
WHERE
    valid_limits :      low_bound <= high_bound;
END_ENTITY;

(* Descriptions:
   low_bound            index of first element in this array.
   high_bound           index of last element in this array.
   uniqueflag           boolean flag to indicate whether the elements
                           of the array shall be unique.
   option               boolean flag to indicate whether elements of
                           the array are optional.
   valid_limits         bounds on number of elements shall be sensible.
*)

```

```
ENTITY attribute_declaration;  
    name : valid_identifier;  
    option : BOOLEAN;  
    attribute_domain : type_declaration;  
INVERSE  
    is_a_property_of : SET OF entity_declaration  
                      FOR attributes;  
END_ENTITY;  
  
(* Descriptions:  
    name          name of the attribute being described.  
    option        boolean flag to indicate whether this attribute  
                  is optional or not.  
    attribute_domain domain from which this attribute can take  
                  values.  
    is_a_property_of each entity lists its attributes.  
*)  
  
ENTITY bag_data_type  
SUBTYPE of (aggregate_data_type);  
    low_bound : OPTIONAL INTEGER;  
    high_bound : OPTIONAL INTEGER;  
WHERE  
    valid_limits : low_bound <= high_bound;  
END_ENTITY;  
  
(* Descriptions:  
    low_bound      minimum number of elements that can be present  
                  in this bag. Default is zero.  
    high_bound     maximum number of elements that can be present  
                  in this bag. Default is an unlimited number of  
                  elements.  
    valid_limits   bounds on number of elements shall be sensible.  
*)  
  
ENTITY binary_data_type;  
    width : OPTIONAL INTEGER;  
    fixwidth : BOOLEAN;  
END_ENTITY;  
  
(* Descriptions:  
    width          maximum length of this binary.  
    fixwidth       flag to indicate if this binary is of a fixed  
                  specified width.  
*)  
  
ENTITY boolean_data_type;  
END_ENTITY;  
  
ENTITY constant_declaration;  
    name : valid_identifier;  
    base_domain : type_declaration;  
    initialisation : STRING;  
INVERSE  
    applies_within : schema_declaration  
                   FOR constants;  
UNIQUE  
    unique_own_name : name;  
END_ENTITY;
```

```
(* Description:
    name                name given to this constant.
    base_domain         the domain within which the constant value is
                        defined.
    initialisation      expression or value for this constant.
    applies_within      each schema_declaration lists its constants.
    unique_own_name      each constant shall be given a unique name.
*)
```

```
ENTITY defined_data_type;
    name :              valid_identifier;
    based_on :          type_declaration;
    where_rules :       SET OF domain_rule_declaration;
UNIQUE
    unique_own_name :   name;
WHERE
    not_own_base :      SELF <> based_on;
END_ENTITY;
```

```
(* Descriptions:
    defined_data_type    user-defined type for a domain.
    name                name of this user-defined type.
    based_on            type that this user-defined type is based on.
    where_rules         constraints placed upon the based_on type to
                        create this user-defined type.
    unique_own_name      each defined_data_type shall be given a unique
                        name.
    not_own_base         a user-defined type cannot be based on itself.
*)
```

```
ENTITY derived_attribute_declaration;
    name :              valid_identifier;
    based_on :          type_declaration;
    derived_expression : STRING;
END_ENTITY;
```

```
(* Descriptions:
    name                name given to this derived attribute.
    based_on            the domain for the value that is to be derived.
    derived_expression  expression showing how the derived value is to
                        be calculated.
*)
```

```
ENTITY domain_rule_declaration;
    label :             valid_identifier;
    where_condition :    STRING;
END_ENTITY;
```

```
(* Descriptions:
    label               name given to this WHERE rule.
    where_condition     plain text of WHERE condition, which shall
                        conform with the syntax rules of EXPRESS.
*)
```

ENTITY **entity_declaration**;

```
    name : valid_identifier;
    is_a_super_of : SET OF entity_declaration;
    is_abstract : BOOLEAN;
    is_a_sub_of : LIST OF UNIQUE entity_declaration;
    attributes : LIST OF UNIQUE attribute_declaration;
    derived_attributes : SET OF derived_attribute_declaration;
    inverse_attributes : SET OF inverse_attribute_declaration;
    uniqueness_rules : SET OF unique_rule_declaration;
    where_rules : SET OF domain_rule_declaration;
```

INVERSE

```
    is_a_part_of : schema_declaration
                  FOR entities;
```

UNIQUE

```
    unique_own_name : name;
END_ENTITY;
```

(* Descriptions:

name	name of the entity_declaration being described This is the name of one of the entities declared in the application schema.
is_a_super_of	set of those entity_declarations of which the entity described in this entity_declaration is a SUPERTYPE. This set may be empty. A SUPERTYPE shall always be a ONEOF.
is_abstract	boolean flag to indicate whether the entity described in this entity_declaration is an ABSTRACT SUPERTYPE or not. An ABSTRACT SUPERTYPE cannot be instantiated.
is_a_sub_of	ordered list of those entity_declarations of which the entity described in this entity_declaration is a SUBTYPE. A LIST is used since SUBTYPES need to be ordered to indicate the sequence for inherited attributes.
attributes	ordered list of the attributes of the entity described in this entity_declaration.
derived_attributes	set of derived attributes of this entity.
inverse_attributes	set of inverse attributes of this entity.
uniqueness_rules	UNIQUE rules that apply to the entity described in this entity_declaration. Together with WHERE rules these act as constraints.
where_rules	WHERE rules that apply to the entity described in this entity_declaration. Together with UNIQUE rules these act as constraints.
is_a_part_of	each schema_declaration lists those entity_declarations that are part of it.
unique_own_name	each entity shall be given a unique name.

*)

ENTITY **enumeration_data_type**;

```
    name : valid_identifier;
    enumerated_values : LIST [1:?] OF UNIQUE STRING;
UNIQUE
    unique_own_name : name;
WHERE
    enum_are_uppercase : upperdomain enumerated_values);
END_ENTITY;
```

```
(* Descriptions:
    name                name given to this enumeration.
    enumerated_values    possible values that this enumeration can take.
                        These shall be in upper-case only.
    unique_own_name      each enumeration shall be given a unique name.
*)
```

```
ENTITY external_type;
    name :                valid_identifier;
    renamed_as :          OPTIONAL valid_identifier;
END_ENTITY;
```

```
(* Descriptions:
    external_type        type defined in another schema, that is being
                        referenced in this schema.
    name                 name of this external type.
    renamed_as           types from another schema can be given a new
                        name for use within this schema.
*)
```

```
ENTITY function_declaration;
    name :                valid_identifier;
    parameters :          LIST [1:?] OF STRING;
    returns :             STRING;
    algorithm :           LIST [1:?] OF STRING;
INVERSE
    belongs_to :          schema_declaration
                        FOR functions;
UNIQUE
    unique_own_name :     name;
END_ENTITY;
```

```
(* Descriptions:
    name                name given to this function.
    parameters          list of parameters for this function. Each
                        parameter comprises its name and its type.
    returns             type of value returned by this function.
    algorithm           plain text of function, which shall conform
                        with the syntax rules of EXPRESS.
    belongs_to          each schema_declaration lists its functions.
    unique_own_name      each function shall be given a unique name.
*)
```

```
ENTITY integer_data_type;
END_ENTITY;
```

```
ENTITY inverse_attribute_declaration;
    name :                valid_identifier;
    low_bound :           OPTIONAL INTEGER;
    high_bound :          OPTIONAL INTEGER;
    duplicates :          BOOLEAN;
    target_entity :        entity_declaration;
    target_attribute :     attribute_declaration;
END_ENTITY;
```

```
(* Descriptions:
  name                name given to this inverse attribute.
  low_bound           minimum number of items in this
                      inverse_attribute.
  high_bound          maximum number of items in this
                      inverse_attribute.
  duplicates          boolean flag to indicate whether duplicates are
                      allowed. A BAG allows duplicates, a SET does
                      not.
  target_entity       entity that this inverse attribute refers to.
  target_attribute    attribute within the target_entity that this
                      inverse attribute refers to.
*)
```

ENTITY list_data_type

```
SUBTYPE of (aggregate_data_type);
  low_bound :          OPTIONAL INTEGER;
  high_bound :         OPTIONAL INTEGER;
  uniqueflag :         OPTIONAL BOOLEAN;
WHERE
  valid_limits :      low_bound <= high_bound;
END_ENTITY;
```

```
(* Descriptions:
  low_bound           minimum number of elements that can be present
                      in this list. Default is zero.
  high_bound          maximum number of elements that can be present
                      in this list. Default is an unlimited number of
                      elements.
  uniqueflag          boolean flag to indicate whether the elements
                      of the list shall be unique.
  valid_limits        bounds on number of elements shall be sensible.
*)
```

```
ENTITY logical_data_type;
END_ENTITY;
```

ENTITY procedure_declaration;

```
  name :              valid_identifier;
  parameters :        LIST [1:?] OF STRING;
  algorithm :          LIST [1:?] OF STRING;
INVERSE
  belongs_to :         schema_declaration
                      FOR procedures;

UNIQUE
  unique_own_name :    name;
END_ENTITY;
```

```
(* Descriptions:
  name          name given to this procedure.
  parameters    list of parameters for this procedure. Each
                parameter comprises the optional word VAR, its
                name and its type.

  algorithm     plain text of procedure, which shall conform
                with the syntax rules of EXPRESS.

  belongs_to    each schema_declaration lists its procedures.
  unique_own_name each procedure shall be given a unique name.
*)
```

```
ENTITY real_data_type;
  precision :          OPTIONAL INTEGER;
END_ENTITY;
```

```
(* Description:
  precision          a real number may have a specified precision.
*)
```

```
ENTITY rule_declaration;
  name :              valid_identifier;
  constrains :        SET [1:?] OF entity_declaration;
  algorithm :         LIST [1:?] OF STRING;
  where_conditions :  SET [1:?] OF STRING;
INVERSE
  belongs_to :        schema_declaration
                     FOR global_rules;
UNIQUE
  unique_own_name :   name;
END_ENTITY;
```

```
(* Descriptions:
  name          name given to this global rule.
  constrains    set of those entities affected by this rule.
  algorithm     plain text of rule, which shall conform with
                the syntax rules of EXPRESS.

  where_conditions constraints within this rule. These shall
                conform to EXPRESS syntax.

  belongs_to    each schema_declaration lists the global rules
                that it uses.

  unique_own_name each global rule shall be given a unique name.
*)
```

```
ENTITY schema_declaration;
  details :          schema_details;
  referenced_schemas : SET OF schema_details;
  entities :         SET OF entity_declaration;
  global_rules :     SET OF rule_declaration;
  functions :        SET OF function_declaration;
  procedures :       SET OF procedure_declaration;
  constants :       SET OF constant_declaration;
END_ENTITY;
```

```
(* Descriptions:
  details          description of this application schema.
  referenced_schemas schemas that are REFERENCED by this application
                    schema.
  entities         entities defined by this application schema.
  global_rules     global rules for this application schema.
  functions        functions for this application schema.
  procedures       procedures for this application schema.
  constants        constants that are used in this application
                    schema.
*)
```

```
ENTITY schema_details;
  schema_name :      valid_identifier;
  schema_version :  STRING;
  named_entities :  OPTIONAL SET[1:?] OF external_type;
  plain_text_file :  OPTIONAL external_file;
END_ENTITY;

(* Description:
  schema_name       name of the schema being described.
  schema_version    identifying string to distinguish this version
                    of the schema from any earlier or later
                    version.
  named_entities    a REFERENCED schema may specify a set of
                    entities that are to be interfaced from that
                    schema.
  plain_text_file   external file containing a plain text version
                    of the application schema in EXPRESS.
*)
```

```
ENTITY select_data_type;
  name :            valid_identifier;
  selected_from :   SET [2:?] OF type_declaration;
UNIQUE
  unique_own_name : name;
END_ENTITY;

(* Descriptions:
  name              name given to this select_data_type.
  selected_from     possible types for this select_data_type.
  unique_own_name   each select_data_type shall be given a unique
                    name.
*)
```

```
ENTITY set_data_type
SUBTYPE of (aggregate_data_type);
  low_bound :      OPTIONAL INTEGER;
  high_bound :     OPTIONAL INTEGER;
WHERE
  valid_limits :   low_bound <= high_bound;
END_ENTITY;
```

```
(* Descriptions:
  low_bound          minimum number of elements that can be present
                      in this set. Default is zero.
  high_bound         maximum number of elements that can be present
                      in this set. Default is an unlimited number of
                      elements.
  valid_limits       bounds on number of elements shall be sensible.
*)
```

```
ENTITY string_data_type;
  width :              OPTIONAL INTEGER;
  fixwidth :          BOOLEAN;
END_ENTITY;
```

```
(* Descriptions:
  width              maximum length of this string.
  fixwidth           flag to indicate if this string is of a fixed
                      specified width.
*)
```

```
ENTITY unique_rule_declaration;
  label :             valid_identifier;
  unique_attributes : LIST [1:?] OF UNIQUE
                      attribute_declaration;
INVERSE
  constrains :       entity_declaration
                      FOR uniqueness_rules;
END_ENTITY;
```

```
(* Descriptions:
  label              name given to this UNIQUE rule.
  unique_attributes  SET of the attributes of this entity that have
                      to be unique.
  constrains         each entity lists the UNIQUE rules that it is
                      constrained by.
*)
```

```
TYPE type_declaration = SELECT(defined_data_type, integer_data_type,
                                boolean_data_type, logical_data_type,
                                real_data_type, string_data_type,
                                binary_data_type, external_type,
                                aggregate_data_type,
                                enumeration_data_type, select_data_type,
                                entity_declaration);
END_TYPE;
```

```
(* Description:
  type_declaration   collection of all the possible sorts of domain.
*)
```

```
TYPE valid_identifier = STRING;
WHERE
  valid_whole :      valid_id(SELF);
END_TYPE;
```

```
(* Descriptions:
    valid_whole                shall start with lower-case a to z, and shall
                                only contain lower-case a to z, "_" or 0 to 9.
                                Upper-case characters that have been used in an
                                application schema shall be mapped to lower-
                                case.
*)
```

```
FUNCTION upperdomain (enum: LIST OF STRING) : BOOLEAN;
    LOCAL
        i : INTEGER;
        j : INTEGER;
        test : STRING;
    END_LOCAL;
    REPEAT i:=LOINDEX(enum) TO HIINDEX(enum);
        test := enum[i];
        IF ((test[1] < 'A') OR (test[1] > 'Z'))
            THEN RETURN (FALSE);
        END_IF;
        IF (LENGTH(test) > 1) THEN
            REPEAT j:=2 TO LENGTH(test);
                IF (((test[j] < 'A') OR (test[j] > 'Z')) AND
                    ((test[j] < '0') OR (test[j] > '9')) AND
                    (test[j] <> '_'))
                    THEN RETURN (FALSE);
                END_IF;
            END_REPEAT;
        END_IF;
    END_REPEAT;
    RETURN (TRUE);
END_FUNCTION;
```

```
FUNCTION valid_id (id: STRING) : BOOLEAN;
    LOCAL
        i : INTEGER;
    END_LOCAL;
    IF ((id[1] < 'a') OR (id[1] > 'z'))
        THEN RETURN (FALSE);
    END_IF;
    IF (LENGTH(id) > 1) THEN
        REPEAT i:=2 TO LENGTH(id);
            IF (((id[i] < 'a') OR (id[i] > 'z')) AND
                ((id[i] < '0') OR (id[i] > '9')) AND
                (id[i] <> '_'))
                THEN RETURN (FALSE);
            END_IF;
        END_REPEAT;
    END_IF;
    RETURN (TRUE);
END_FUNCTION;
```

```
END_SCHEMA; (* application_encoding *)
```