ISO NWIP ISO 19160-6:2018(E)

ISO TC 211/WG 7

Secretariat : SN

Addressing — Digital interchange models

Adressage — Modèles d'échange numérique

NWIP stage

Warning for WDs and CDs

This document is not an ISO International Standard. It is distributed for review and comment. It is subject to change without notice and may not be referred to as an International Standard.

Recipients of this draft are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation.

ISO copyright office
Ch. de Blandonnet 8 • CP 401
CH-1214 Vernier, Geneva, Switzerland
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
copyright@iso.org
www.iso.org

Contents

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.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular the different approval criteria needed for the different types of ISO documents should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

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. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation on the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see the following URL: www.iso.org/iso/foreword.html.

This document was prepared by Technical Committee ISO/TC 211, *Geographic information / Geomantics*.

A list of all parts in the ISO 19160 series can be found on the ISO website.

This is the first edition of this document.

Introduction

Addresses are among the most commonly exchanged information on the Internet, and the interchange of them is crucial to a number of Internet applications, such as electronic commerce, contact exchange, non-postal deliveries, as well as location scheduling.

Yet, addresses can mean much more than just geolocation information:

— As an identity, such as an office address

— As reference points (waypoints) in routing information

— As a delivery point

ISO 19160-1 describes the Addressing Conceptual models that allows specification of international address profiles, and this International Standard further provides methods to utilize them in a way suitable for electronic interchange.

Specifically, this International Standard adapts models from ISO 19160-1 as Address Interchange models, to facilitate interchange and interaction of addresses between humans and applications, and between applications, and for applications to interact and interchange international addresses, while supporting human input and human-machine interaction, without loss of fidelity.

The lifecycle of an address entry exchanged on the Internet typically starts with manual input of a human actor. This data, structured or unstructured, is then submitted to an Internet-connected application, and the application may in turn transmit this information to other applications or external parties on behalf of the user who provided the address. This transmission is usually performed to fulfil service delivery to the user. Within the process, there may be machine-human interactions that require display of the address in human-readable form, as well as machine-to-machine interactions on the address, such as for data validation.

Additional caution shall be placed on the accuracy (or lack thereof) of human input addresses. While an address specified by a human actor may unambiguously distinguish a location, there may be intention or unintentional omissions or additions to an "official" address (if there was one).

The models in this document depends on data models defined in ISO 19115-1 and ISO19157.

Addressing — Digital interchange models

1. Scope

This document specifies the "Address Interchange Object" models ("AXO"), suitable for machine encoding of the digital storage and transmission of the following components:

— description of an address profile conforming to ISO 19160-1, *Addressing — Part 1: Conceptual model*;

— address instances that conform to a specific profile of ISO 19160-1; and

— a template for entering and displaying addresses conforming to a profile of ISO 19160-1.

2. Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements 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 19160-1, *Addressing -- Part 1: Conceptual model*

ISO 19103, *Geographic information -- Conceptual schema language*

ISO 19115-1, *Geographic information -- Metadata -- Part 1: Fundamentals*

3. Terms and definitions

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

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at http://www.iso.org/obp

— IEC Electropedia: available at http://www.electropedia.org

3.1
address
structured information that allows the unambiguous determination of an object for purposes of identification and location

[SOURCE: ISO 19160-1]

3.2
addressable object
object that may be assigned an *address* (3.1)

[SOURCE: ISO 19160-1]

3.3
address class
description of a set of addresses that share the same address components, operations, methods, relationships, and semantics

[SOURCE: ISO 19160-1]

3.4
interchange address class
representation of an *address class* (3.3) suitable for interchange

3.5
address component
constituent part of an *address* (3.1)

[SOURCE: ISO 19160-1]

3.6
interchange address component
representation of an *address component* (3.5) suitable for interchange

3.7
lineage
provenance 3.14, source(s) and production process(es) used in producing a resource

[SOURCE: ISO 19115-1, 4.9]

3.8
locale
definition of the subset of a user's environment that depends on language and cultural conventions

[SOURCE: ISO/IEC/IEEE 9945: 2009, 4.211, modified — The notes given in ISO/IEC/IEEE 9945:2009 for this entry have been omitted.]

3.9
parent address
address (3.1) of a *parent addressable object* (3.10)

3.10
parent addressable object
addressable object (3.2) that fully encloses one or more other addressable objects

3.11
profile
set of one or more base standards or subsets of base standards, and, where applicable, the identification of chosen clauses, classes, options and parameters of those base standards, that are necessary for accomplishing a particular function

[SOURCE: ISO 19106: 2004, 4.5]

3.12
interchange address profile
profile (3.11) that contains data requirements and layout templates for multiple *address classes*

3.13
interchange address instance
an *address* (3.1) that is set within the context of an *interchange address class* (3.4)

3.14
provenance
organization or individual that created, accumulated, maintained and used records

3.15
data type
specification of a value domain (ISO 19103, 4.37) with operations (ISO 19103, 4.26) allowed on values in this domain

[SOURCE: ISO 19103, 4.14]

3.16
primitive data type
data type (3.15) defined as "Primitive Type" in ISO 19103, Section 7.2

3.17
user defined data type
data type (3.15) defined by the user in an *interchange address profile* (3.12) through the composure of other *data types* (3.15) and constraints

3.18
user defined data type definition
definition of a *user defined data type* (3.17)

3.19
address capability
marking on an *address instance* (3.13) to indicate its status

3.20
address layout template
specification of layout and positioning of *address components* (3.5) for an *interchange address instance* (3.13) of an *address class* (3.3)

3.21
address display template
address layout template (3.20) for the display of *interchange address instances* (3.13) of an *address class* (3.3)

3.22
address form template
address layout template (3.20) of an input form for the entry of *interchange address instances* (3.13) of an *address class* (3.3)

3.23
address processor
entity that processes *interchange address instances* (3.13)

3.24
address profile distributor
entity that distributes *interchange address profiles* (3.12)

3.25
signature
the string of bits resulting from the signature process

[SOURCE: ISO/IEC 14888-3, 4.15]

3.26
signature key
a secret data item specific to an entity and usable only by this entity in the signature process

[SOURCE: ISO/IEC 14888-3, 4.18]

3.27
verification key
a data item which is mathematically related to an entity's *signature key* (3.26) and which is used by the verifier in the verification process

[SOURCE: ISO/IEC 14888-3, 4.15]

3.28
object identifier
oid
a value (distinguishable from all other such values) which is associated with an object

[SOURCE: ISO/IEC 15961: 2004, 3.1.16]

3.29
language identifier
language symbol
symbol that uniquely identifies a particular language

[SOURCE: ISO/IEC 639-3: 2007, 3.3]

3.30
script
set of graphic characters used for the written form of one or more languages

[SOURCE: ISO 15924, 3.7]

3.31
script code
combination of characters used to represent the name of a *script* (3.30)

[SOURCE: ISO 15924, 3.8]

3.32
URI
uniform resource identifier

[SOURCE: ISO 19103, 5.3]

4. Process of address profile and address instance interchange

4.1. Address profiles

4.1.1. General

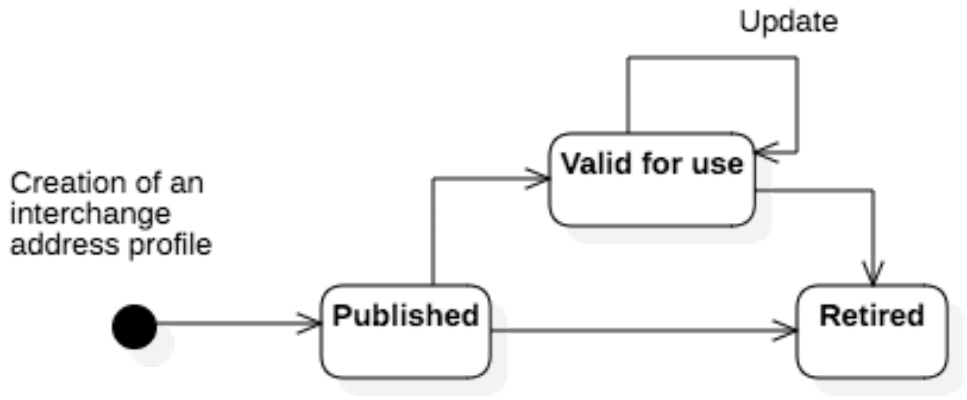

Figure 1 — Lifecycle of an interchange address profile

4.1.2. Creating interchange address profiles

Address profiles are created by publishers.

4.1.3. Publishing address profiles

Publishers distribute their address profile to others through exchange or a registry.

4.1.4. Updating address profiles

Publishers can update an address profile and re-distribute it by publishing using a new version number that supersedes the previously published one.

4.1.5. Using address profiles

Applications retrieve suitable profiles to:

— render address input forms according to the profile-specified form template; or

— display addresses according to the profile-specified display template.

Applications shall consider the validity period of an address profile, and shall periodically check with the publisher (or a registry that the publisher distributes via) the latest version of the address profile.

4.1.6. Retiring address profiles

Publisher can indicate the validity period of an address profile in the profile itself, which if the validity end date has passed, would indicate that the profile is retired.

To immediately retire an address profile, the publisher shall distribute a new version of the address profile with an expired validity period.

4.2. Address instances

4.2.1. General

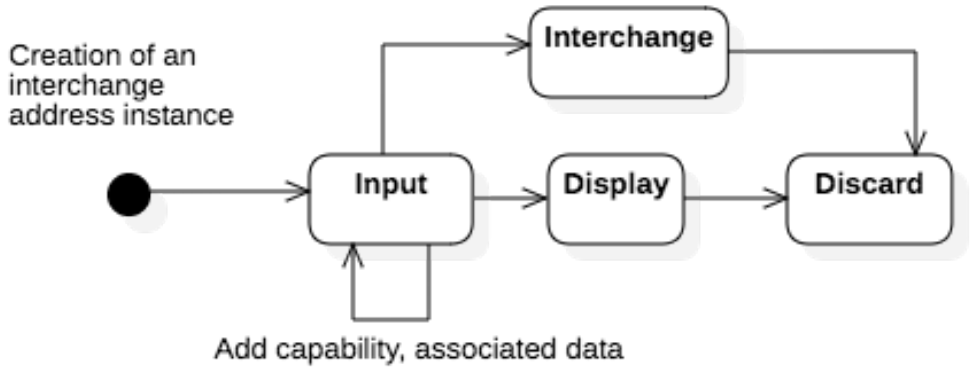

Figure 2 — Lifecycle of an interchange address instance

4.2.2. Creating an address instance

This section describes how an address instance is created. The desired address profile shall be already retrieved for creating an address instance that conforms to it.

Typically, a user enters an address through an application interface that implements an input format that conforms to the address profile's form template, such as an application that runs on an operating system or an Internet application. Such input interface may or may not provide a graphical form.

Conforming applications should not expect general users to be able to input an address with a fully-deduced structure. Immediately after input no capabilities are marked on the address instance.

4.2.3. Interchange of an address instance

After a user inputs an address into a structured address form, the user submits this address to the designated recipient, which could be an e-retailer, an electronic business card, or a calendar event.

4.2.4. Displaying an address instance

The recipient or service that receives an address instance either already has the interchange address profile or should obtain the address profile definition. With the interchange address profile, it could then display the address instance according to the address profile's display template.

4.2.5. Adding address capabilities

4.2.5.1. Improving quality of an address instance

The recipient or service could process the address, such as to further clean up the address by improving the conformance level by fully deducing the address structure, which would add an address capability "https://standards.iso.org/19160/-6/capabilities/specified" to the address instance.

To ensure that the resulting address instance with a fully-specified structure is correct, the processor may wish to confirm with the user the resulting address, such as in the case where the address is used for shipping information. This would add the AddressCapability "https://standards.iso.org/19160/-6/capabilities/confirmed" to the address instance.

4.2.5.2. Verifying an address instance

The recipient or service could further validate the address, such as with a postal or addressing authority. This would add the AddressCapability "https://verifyingauthority/verified" to the address instance written by the verifying authority.

4.2.6. Adding associated data to an address instance

A processor of an address instance could add extra information such as delivery instructions or routes as associated data.

If the user already has a verified, structured address, then the service could save the effort for verifying the address. Custom address capabilities could be added into the address instance.

4.2.7. Discarding an address instance

When a service no longer needs the address, it should be disposed of.

5. Data types

5.1. Primitive and core data types

These are the common data types used within this document.

Primitive data types (`PrimitiveTypes`) are defined in ISO 19103, 7.2, including:

— `CharacterString`

— `DateTime`, `Date`, `Time`

— `Number`, `Integer`, `Decimal`, `Real`

— `Vector`

— `Boolean`

Core data types (`CoreTypes`) are defined in ISO 19103, A.2.

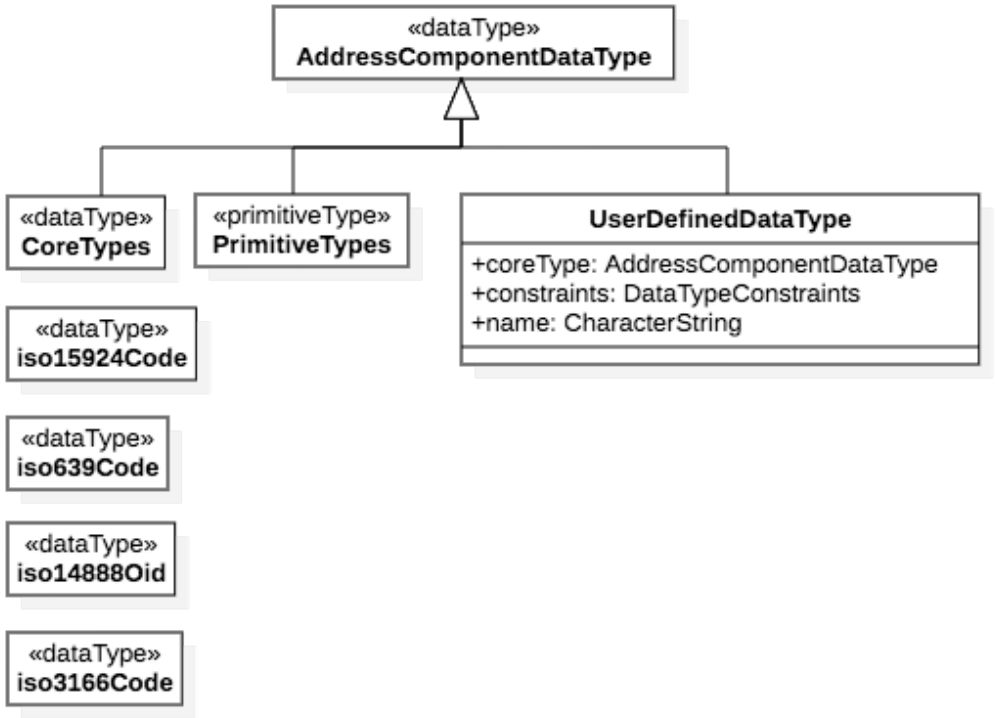

Figure 3 — Common data types used in this document

5.2. User-defined data types

User defined data types are defined within the profile that utilizes them. A user defined data type can be composed of multiple primitive data types, and constraints can be set on them.

coreType AddressComponentDataType

constraints DataTypeConstraints

5.2.1. Data Type Constraints

Constraints that can be set depend on the underlying data type. For example, the maximum number of Post Office Box numbers.

— Number types: "maxValue" and "minValue".

— Number and characters: "maxLength" and "minLength".

NOTE This section is to be further developed. (TODO)

As a constraint, it is also possible to provide a fixed list of accepted values to this type, such as the number of states in a country or districts in a city.

6. Common models

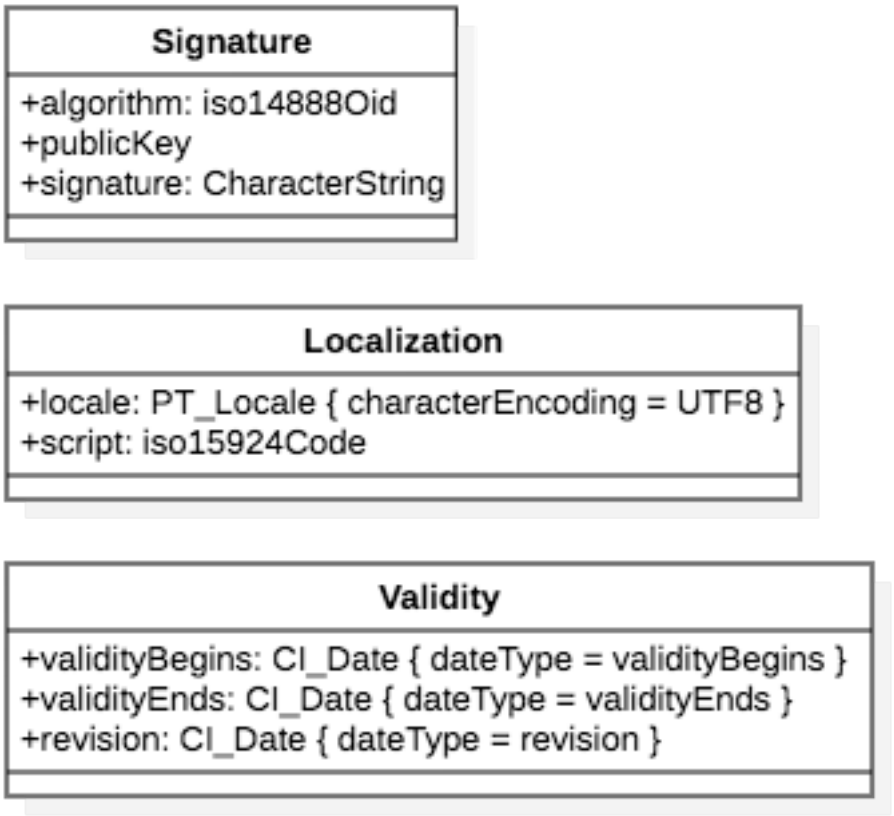

Figure 4 — Common data models used in this document

6.1. Signature

A cryptographic signature used to determine data integrity and validity of the object it belongs to.

algorithm — the public key cryptographic algorithm used for this digital signature, represented by the algorithm's object identifier (OID) defined in ISO/IEC 14888-3.

publicKey — a reference to the actual public key used to verify the digital signature, a URI where the public key of the signer used for this signature is found. Represented as a `Uri`.

NOTE Public key information should be verified on a separate communication channel from the signed object itself to ensure the integrity and validity of the public key.

signature the actual digital signature value represented as a `CharacterString`, encoded in Base64 format.

6.2. Localization

The language and script used within this interchange address profile.

Represented by the `PT_Locale` object defined in ISO 19115-1 and an ISO 15924 script code.

locale the locale of the parent object, represented by `PT_Locale` with `characterEncoding` set to `UTF-8`.

script the type of written script used in the parent object, represented by the script code from ISO 15924.

Localization information includes:

— locale: ISO 19115-1 `PT_Locale`

— script: ISO 15924 Script Identifier Code

6.3. Validity

The time interval where this interchange address profile is determined valid, and the revision number (represented as a date).

Each of such is represented as a `CI_Date` object defined in ISO 19115-1.

validityBegins the date and time when this object becomes valid. The type of `CI_Date` should be `validityBegins`.

validityEnds the date and time when this object becomes invalid. The type of `CI_Date` should be `validityEnds`.

revision issuance date/time of this object. The type of `CI_Date` should be `revision`.

7. Interchange address profile

7.1. General

The interchange address profile corresponds to a representation of the *address profile* in ISO 19160-1 suitable for digital distribution.

This model includes the necessary information for a machine-readable implementation to understand how to represent addresses, their requirements, as well as their input and display.

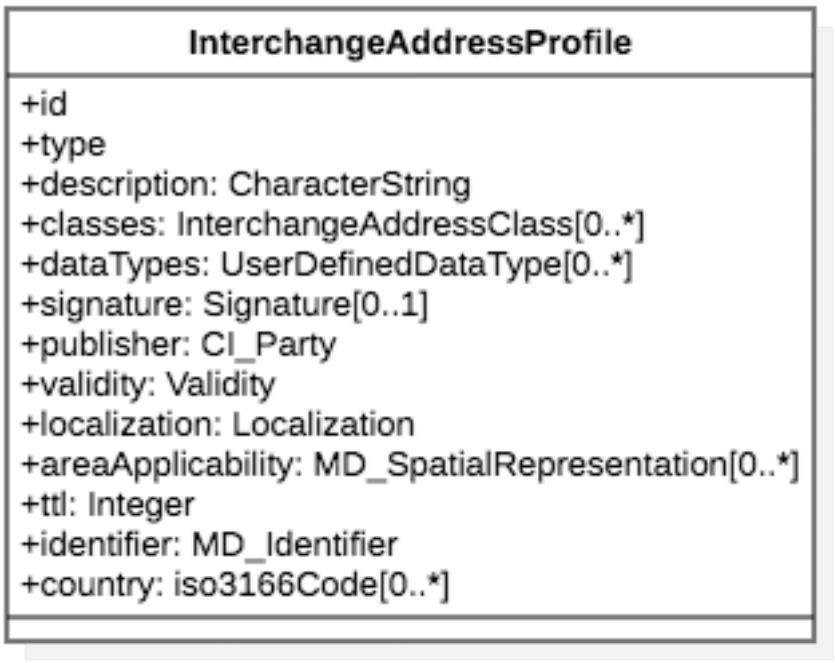

Figure 5 — Interchange address profile data model

7.2. Attributes

ID	Unique identifier of this Interchange Address Profile. `Uri`.
Type	Intended usage of this profile. `Uri`.
Description	Textual description of this definition. `CharacterString`.
TTL (time-to-live)	The maximum time interval between refreshing of this profile via an authoritative source, in seconds. `Integer`.
Country	The country of which this interchange address profile represents. Expressed as the 2-digit country code specified in ISO 639-1.

7.3. Publisher

Details about the publisher of this interchange address profile, including the name of publisher and an accessible URI that can be used to update this interchange address profile itself.

Represented by the `CI_Party` object defined in ISO 19115-1, the URI should be represented as a `CI_OnlineResource` object specified within `CI_Party.contactInfo.onlineResource`.

7.4. Identifier

Used to identify this interchange address profile and provide description of it.

Represented by the `MD_Identifier` object defined in ISO 19115-1.

7.5. Area applicability

The geographic representation of which this interchange address profile applies to. Overlapping geographic areas are allowed across different interchange address profiles.

Represented by one or more `MD_SpatialRepresentation` objects defined in ISO 19115-1.

EXAMPLE Countries that have overlapping geographic claims are allowed to represent their purported claims in interchange address profiles.

8. Interchange address class

8.1. General

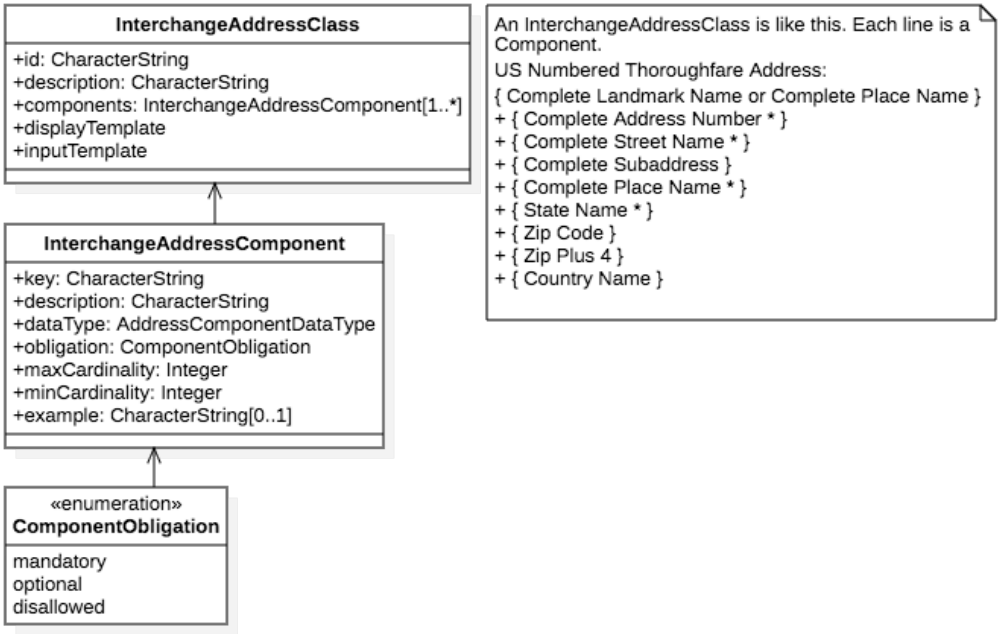

Figure 6 — Interchange address class data model

Interchange address class corresponds to the `addressClass` specified in ISO 19160-1.

A single profile can include and support multiple types of address formats, such as a numbered street address and a "PO Box" address simultaneously. Each of these address formats is represented as an interchange address class.

EXAMPLE The US Numbered Thoroughfare Address with this syntax can be represented as an interchange address class: "{ Complete Landmark Name or Complete Place Name }+ { Complete Address Number * }+ { Complete Street Name * }+ { Complete Subaddress }+ { Complete Place Name * }+ { State Name * }+ { Zip Code }+ { Zip Plus 4 }+ { Country Name }"

An interchange address class also provides a display template and a form template to allow the display and entry of an address instance of the address class.

8.2. Attributes

ID Unique identifier of this interchange address class. `Uri`.

Description Textual description of this definition. `CharacterString`.

8.3. Interchange address component

The interchange address component corresponds to the `addressComponent` defined in ISO 19160-1.

8.3.1. Attributes

Key — an identifier of this interchange address component, shall be unique within the interchange address class. `CharacterString`.

Description — Textual description of this component. `CharacterString`.

Obligation — Whether this component is mandatory, optional or disallowed. Values represented by the `ComponentObligation` object.

maxCardinality — the maximum number of components within this address class.

minCardinality — the minimum number of components within this address class.

dataType — describes the type of value accepted by this component. This takes an `AddressComponentDataType` value.

9. Address capability

9.1. General

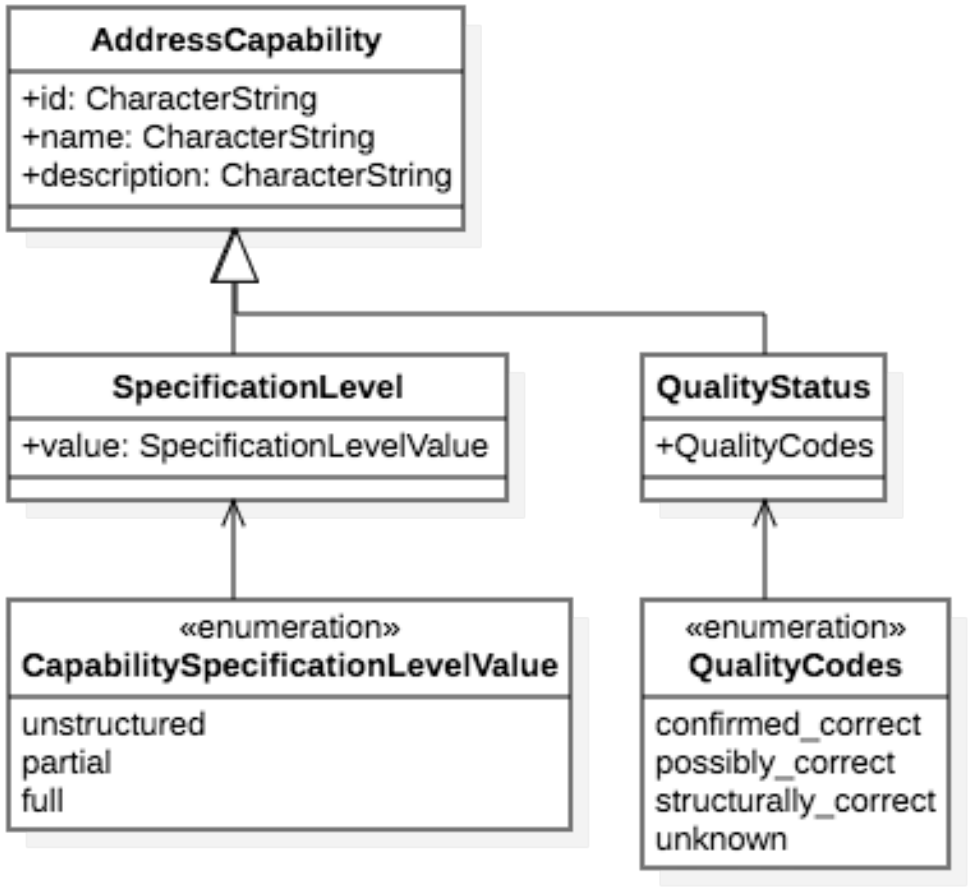

Figure 7 — Interchange address capability data model

Address capability represents the status of a particular aspect of an interchange address instance, as determined by an address processor. Each of the aspects may be differentiated by multiple statuses.

This is an abstract class. To represent specific capabilities, this class should be extended upon.

9.2. Attributes

ID unique identifier of the address capability. `CharacterString`.

name human readable name of the address capability. `CharacterString`.

description human readable description of the address capability. `CharacterString`.

9.3. Capabilities

9.3.1. Quality

An address entered may not have been validated for correctness. An address processor, such as a postal handling entity, may decide to validate the address to a certain degree in order to determine the correctness of the address.

The allowed values of validation statuses are:

— Confirmed correct, indicating that the address is confirmed to be correct to the best of knowledge of the address processor.

— Possibly correct, indicating with confidence that the address is correct.

— Structurally correct, indicating that the address components have the correct value types.

— Unknown, indicating that this address has not been validated in any way, and it cannot be assigned a quality.

9.3.2. Specification level

When an address is first manually entered by a common person, it is unrealistic to expect this person to fully adhere to the defined structure.

The allowed values of specification levels are:

— Fully structured, all components are accurately separated and cannot be further split.

— Partially structured, some components may be structurally correct, but some other components are still unstructured.

— Unstructured, the user has entered free-form text with no regard of structuring them.

EXAMPLE 1	Digital addresses entered on e-commerce sites and address books are often partially structured, with defined country, region and city, but with street addresses often unstructured.
EXAMPLE 2	Digital addresses for utility installations (e.g. electricity, water supplies) are often fully structured.

Therefore, an organization who accepts these addresses may wish to re-structure them or fill in any missing address components.

10. Interchange address instance

10.1. General

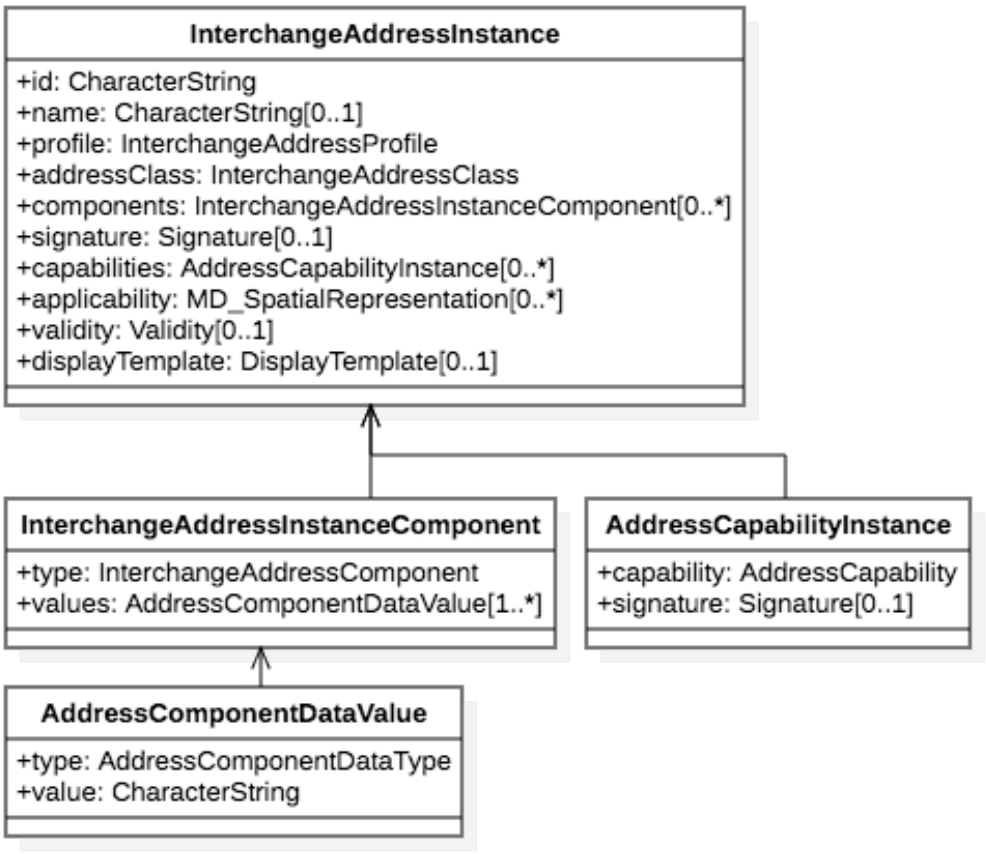

Figure 8 — Interchange address instance data model

An interchange address instance is the representation of an address that conforms to an interchange address class, which in turn belongs to an interchange address profile.

An address instance is considered to conform to an interchange address profile if it conforms to any of the profile's address classes.

10.2. Attributes

ID	unique identifier for this interchange address instance.
name	a human-readable name for this interchange address instance.
profile	the interchange address profile which contains the interchange address class it conforms to.
addressClass	the interchange address class it conforms to.
signature	a cryptographic signature of the interchange address instance itself together with its content. Optional. Represented as a `Signature`.
validity	time interval representing validity of the interchange address instance.

applicability spatial representation of the geographic area that this interchange address instance covers. Optional, multiple allowed. `MD_SpatialRepresentation`.

displayTemplate display template that this interchange address instance should be shown with. This allows an address originator to provide a desired display view of the address.

10.3. Interchange address instance component

An interchange address component instance represents a data value for a corresponding interchange address component.

Each data value only applies to a single interchange address component which belongs to an interchange address class.

10.3.1. Attributes

type the interchange address component the value is for.

values one or more values for the interchange address component specified in `type`. The cardinality of this field depends on the type of value. Represented as an `AddressComponentDataValue`.

10.3.2. Address component data value

A single value for the interchange address instance component.

type the data type of its value. A value from `AddressComponentDataType`.

value the value of the specified component. `CharacterString`.

10.4. Address capability instance

The address capability instance is incorporated into an interchange address instance to mark its capabilities as determined by an address processor:

capability the particular address capability that has been fulfilled. Represented by a `Uri` value that incorporates the address capability identifier with the capability status value.

signature cryptographic signature used to ensure that the capability is marked by an address process trusted for verification of this capability. The signature generated shall incorporate the `id` attribute of the interchange address instance that owns it. Optional. Represented by `Signature`.

11. Interchange layout template

11.1. General

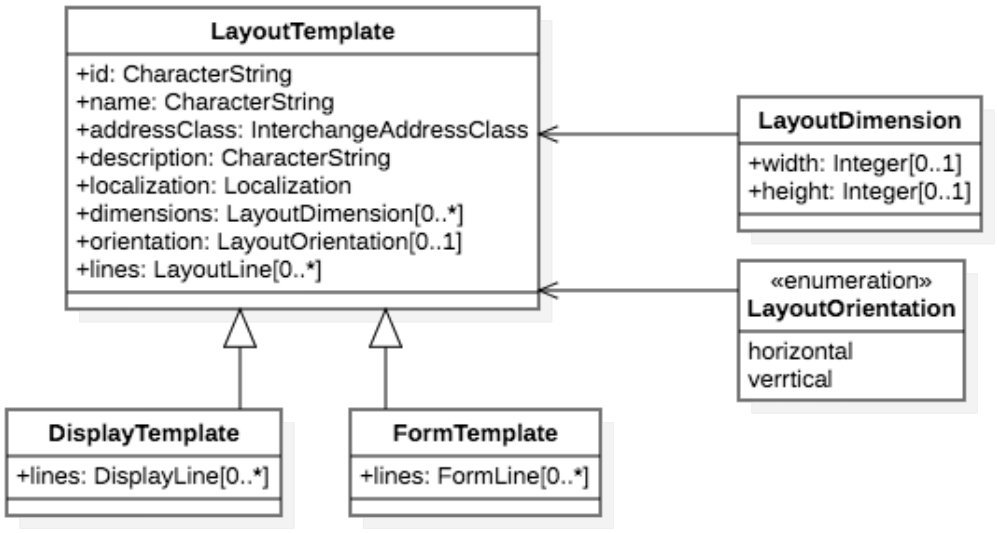

Figure 9 — Interchange layout template data model

The interchange layout template serves as the basis of the interchange display template and the interchange form template.

A layout template is designed to present an interchange address instance in the case of a display template, or present an input form for entry of an interchange address instance in the case of an interchange form template.

The interchange layout template is based on the following assumptions:

— the resulting layout is displayed in a bounded rectangular area

— the entire template is in the same script and locale.

The action of inserting components from a given interchange address instance into a layout template is called "render".

EXAMPLE A postal mail label can be considered an interchange address instance rendered according to a layout template.

11.2. Attributes

ID a unique identifier for the layout template. `CharacterString`.

Name the descriptive name of the layout template. `CharacterString`.

Description textual description of the layout template. `CharacterString`.

Localization locale and script information of the layout template. Represented by the `Localization` model.

Dimensions physical dimensions of the rectangular bounding box for the rendered layout output. Represented by `LayoutDimension`.

Orientation whether the text orientation is horizontal or vertical.

EXAMPLE Some East Asian languages allow using a vertical flow text orientation on postal mail.

This structure of a bounding box allows flexible formatting, such as the ability to hide or shrink certain lines when horizontal space is limited.

11.3. Layout line

11.3.1. General

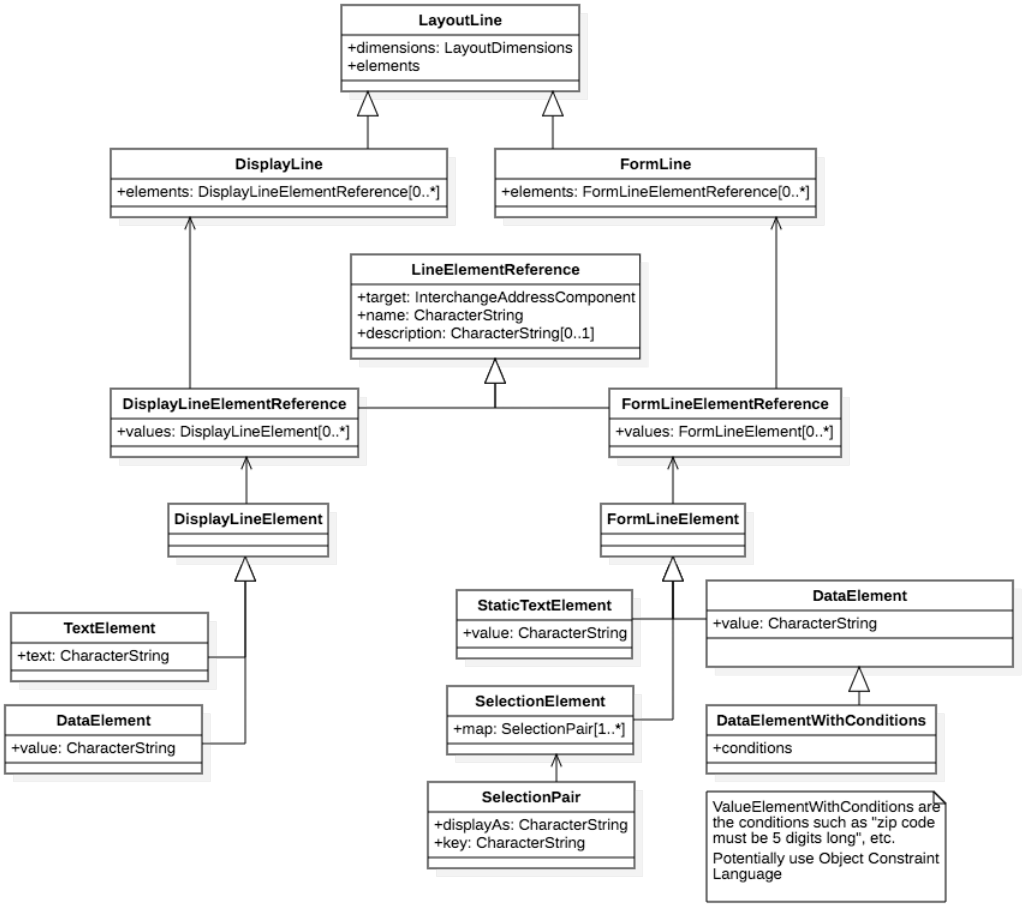

Figure 10 — Interchange layout line data model

A layout line represents a line in a rendered address display or an address input form.

It forms the basis of the address display line and address form line.

11.3.2. Attributes

Dimensions physical dimensions of the rectangular bounding box for the rendered line. Represented by `LayoutDimension`.

Elements collection of line elements that make up the line.

11.4. Line element reference

11.4.1. General

A line element reference is used to link an abstract line element to the interchange address component. The latter is used to retrieve metadata for the line element, such as the schema to determine what data values are accepted.

It is the basis of the display line element reference and the form line element reference.

11.4.2. Attributes

Target the interchange address component this reference refers to.

Name human readable name of what this line element reference refers to. `CharacterString`.

Description human readable description of what this line element reference refers to. Optional. `CharacterString`.

12. Display template

12.1. General

The display template is used for displaying an interchange address instance belonging to an interchange address class. Generally, one address class is represented by just one display template.

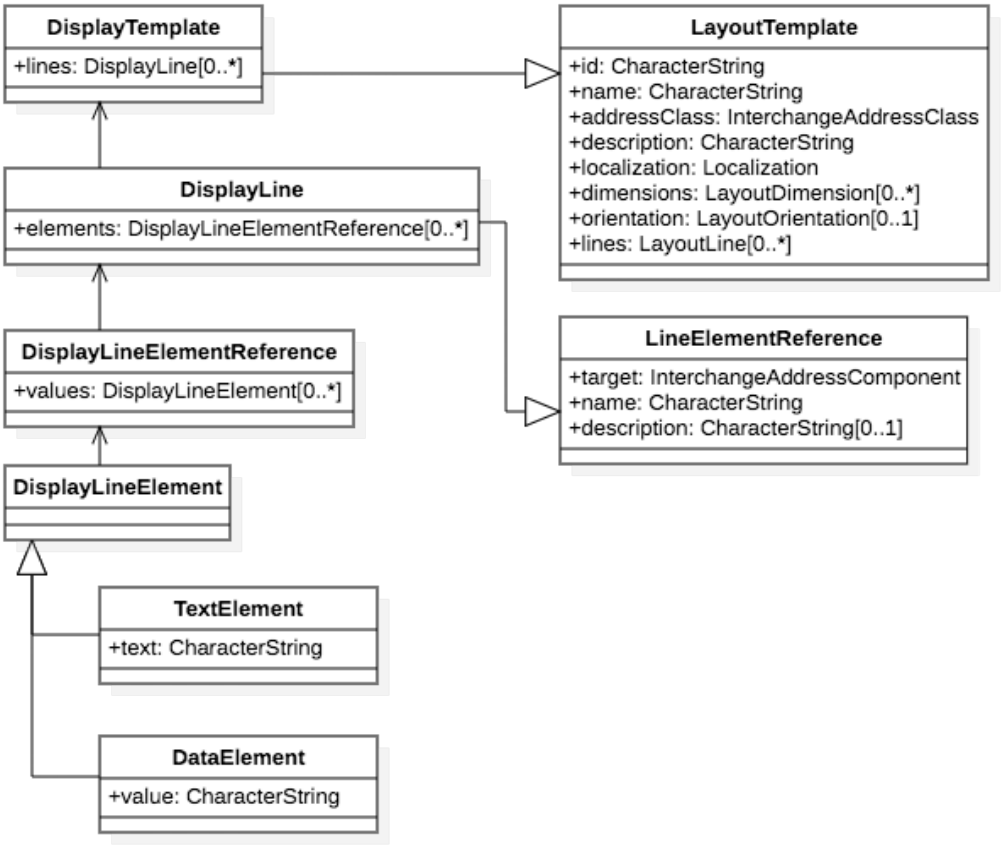

Figure 11 — Interchange display template data model

NOTE this part is intentionally separate from PADTL for a more accurate approach to representations.

12.2. Display line

A display line represents a line within a display layout.

It extends from the `LineElementReference` model, and takes a set of elements that contain display line element references (`DisplayLineElementReference`).

12.3. Display line element reference

12.4. Display line element

This represents an element within a display line.

The two types of display line elements include:

— display line text element, representing static text, such as the phrase "PO Box" preceding the actual PO box number, for display layouts for PO boxes.

— display line data element, representing a variable data value that is contained in the interchange address component, such as the PO box number following the phrase "PO Box".

13. Form templates

13.1. General

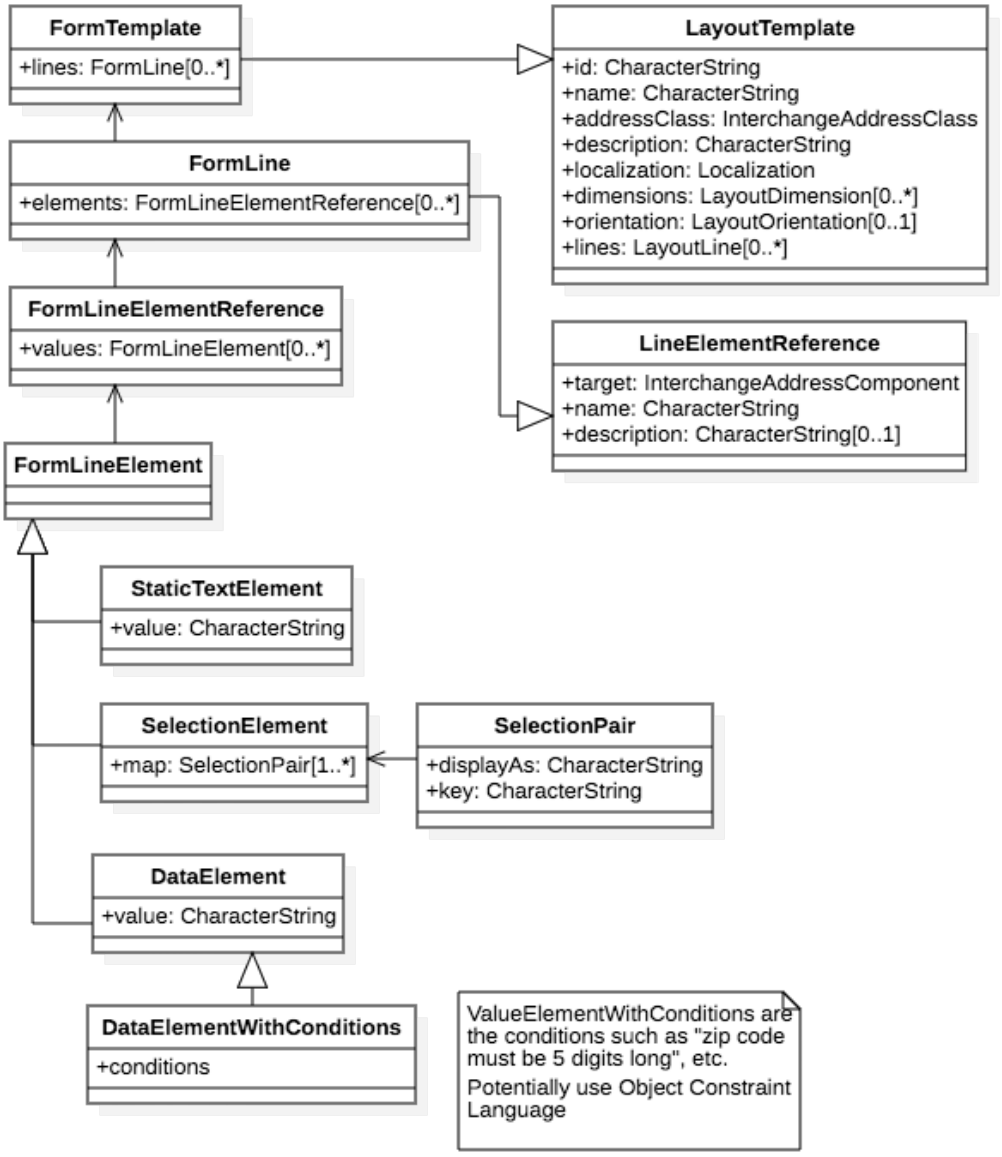

Figure 12 — Interchange form template data model

This section shows how an input form can be rendered according to the interchange address class.

The form template should use should also provide an example for display purposes in input (e.g., Help section).

The form template is used for displaying an input form for entry of address data into an interchange address instance. Generally, one address class is represented by just one input template.

NOTE this part has no equivalence in PADTL.

13.2. Form line

A form line represents a line within a form layout.

It extends from the `LineElementReference` model, and takes a set of elements that contain form line element references (`FormLineElementReference`).

13.3. Form line element reference

13.4. Form line element

This represents an element within a form line.

The three types of form line elements include:

— static text element, representing static text, such as the phrase "PO Box" preceding the actual PO box number, for form layouts for PO boxes. This element is meant to be shown in the form but not allow modification.

— data element, representing a variable data value that is contained in the interchange address component, such as the PO box number following the phrase "PO Box". This element indicates that a form field is required, and the data values accepted shall adhere to the data value type defined in its associated interchange address component (through `FormLine`).

— selection element, representing a selectable mapping between display string to code, e.g. in the case of the US States, "Rhode Island" the state name is mapped to the "RI" postal code. Represented by `SelectionElement` and within `SelectionPair`.

In an empty form with fields generated from data elements, if the `example` values of the associated interchange address component are specified, such values should be used as "placeholders" for form input.

13.5. User experience considerations

It should be assumed that users may not often be able to enter an address thoroughly in the correct, detailed structure.

Therefore, the form should be easy to understand and take into account that easy switching between interchange address classes is important.

An application that adheres to this document may find that it is easiest for the user to first select the appropriate addressClass, then allowing users to (or automatically) add necessary components as they type.

A sample input should be provided for demonstration purposes.

Annex A
(normative)

Abstract test suites

A.1. Introduction

The abstract test suites for the conformance classes defined by this part of ISO 19160 are presented in A.2 to A.5.

A.2. Conformance class: IxAddressProfile

Table A.1 — IxAddressProfile test 1: Associations

Test purpose	Check that the model contains the associations as specified.
Test method	Inspect the model
Reference	Clause 7
Test type	Basic

Table A.2 — IxAddressProfile test 2: Attributes

Test purpose	For each class and type in the model, check that the model appropriately includes the mandatory, optional and conditional attributes.
Test method	Inspect the model
Reference	Clause 7
Test type	Basic

A.3. Conformance class: IxAddressInstance

Table A.3 — IxAddressInstance test 1: Attributes

Test purpose	For each instance in the class, check that the instance appropriately includes the mandatory, optional and conditional attributes.
Test method	Inspect the model
Reference	Clause 10
Test type	Basic

A.4. Conformance class: FormTemplate

Table A.4 — FormTemplate test 1: Attributes

Test purpose	For each class and type in the model, check that the model appropriately includes the mandatory, optional and conditional attributes.
Test method	Inspect the model
Reference	Clause 13
Test type	Basic

A.5. Conformance class: DisplayTemplate

Table A.5 — DisplayTemplate test 1: Attributes

Test purpose	For each class and type in the model, check that the model appropriately includes the mandatory, optional and conditional attributes.
Test method	Inspect the model
Reference	Clause 12
Test type	Basic

Annex B
(normative)

Usage

B.1. Accuracy and Verification

An authority, such as the local post office, could "verify" a structured address that it is confirmed that this address instance is a "deliverable address" through an AddressCapability.

The owner of the address, such as the tenant of an office, could provide its signed, structured address on an electronic business card. This allows the recipient of the business card to know whether the senders address is authentic. If this address is verified to be a "deliverable address", the recipient will know that items sent to this address will very likely be deliverable.

Extra steps need to be taken here to allow this.

B.2. Address As Identity

Addresses do not only specify a location, in some cases they are part of the identity. For example, in business cards, an address can mean more than just an address, such as with vanity value.

This document shall support this functionality for it to be useful in contact exchange.

This address represents a complete textual address instance.

```
Suites 1107-1111,
Floor 11,
Central Building,
1-3 Pedder Street,
Central,
Central & Western District,
Hong Kong Island,
Hong Kong
```

And this address represents the identical address as above, with a reduction of information that is already implied, which that does not reduce its correctness. This can be achieved by supplying a `displayTemplate` in the interchange address instance model.

```
Suite 1111,
1 Pedder Street,
Central,
Hong Kong
```

B.3. Address As Destination

In certain cases, an address is expected to be reachable either by person and/or post.

An address instance should support being used in conjunction with routing information, acting as a waypoint, and/or supporting a source-defined route.

For example, written instructions on how to deliver to a place that is unambiguous but terribly difficult to locate.

NOTE E.g., some buildings have split floors — rooms may have the same floor identifier, but is actually inaccessible from the same floor.

Annex C
(normative)

Examples of objects specified in this document

Models specified in this document can be represented in various object structures, including in XML ISO 19139: 2006 and in JSON.

C.1. IxAddressProfile

```
{

  id: "http://www.iso.org/tc211/tc211-sample.adp",
  type: "iso-19160-address-profile",
  publisher: "http://www.iso.org/tc211/",
  signature: "...",

  name: "TC 211 Minimal Address Profile",
  localization: {
    language: "en",
    script: "en",
  },

area: {
  countries: ["uk"],
},

  dataTypes: {
    addressedObjectIdentifier: {
      primitiveType: Integer,
    }
  }
  addressComponents: { ... }
  addressClasses: { ... }
}
```

C.2. IxAddressClass

```
addressClasses: {
  streetAddress: {
    description: Street Address,
    availableFields: [ (addressComponent) 1..* ... ]
displayTemplate: { ... ],
inputTemplte: {}

  }
}
```

C.3. Validity

```
validity: {
validFrom: 20171129Z000000,
validTo: 20191129Z000000
}
```


C.4. PublisherInformation

```
publisher: {
publisherName: UK Post Office,
publisherUri: https://www.post.co.uk
}
```

C.5. LocalizationInformation

```
publisher: {
language: en,
script: Latn
}
```

C.6. Signature

```
signature: {
algorithm: 1.2.3.4.5.6.7.8.9,
publicKey: https://www.post.co.uk/profile-signature.key,
signature: BOLVMNoGNM1TLglnlxgm0a9t
}
```

C.7. IxAddressClass

```
addressClassDescription: {
id: streetAddress,
description: A typical street address,
addressComponents: [ ... ],
displayTemplate: { ... },
inputTemplate: { ... },
}
```

C.8. User Defined Data Types

```
dataTypes: {[
  name: addressNumberValue,
coreType: Integer,
constraints: [ ... ],
])
```

C.9. Data Type Constraints

```
constraints: [{
  maxValue: 10000,
  minValue: 1,
}]
```

C.10. IxAddressComponent

```
addressComponentDescription: {
key: addressNumber,
description: Street number,
datatype: addressNumberValue
}
```

C.11. IxAddressInstance

```
addressInstance: {
profileId: https://standards.iso.org/19160/-6/profiles/uk.adp,
components: [ ... ],
signature: { ... },
cap: [ ... ]
}
```

C.12. IxAddressInstanceComponent

```
addressComponentInstance: {
type: addressNumber,
values: [ 1001 ]
}
```

C.13. AddressCapability

```
addressCapability: {
capability: https://standards.iso.org/19160/-6/capabilities/specified,
signature: [ ... ]
}
```

C.14. DisplayTemplate

```
DisplayTemplate: {
...
}
```

C.15. Form template (FormTemplate)

```
FormTemplate: {
...
}
```


Annex D
(normative)

Examples

D.1. Example of address profiles defined in ISO 19160-1

D.1.1. ISO 19160-1 C2

```
profile = {

  id: "http://www.iso.org/tc211/tc211-minimal.adp",
  type: "iso-19160-address-profile",
  publisher: "http://www.iso.org/tc211/",
  signature: "...",

  name: "TC 211 Minimal Address Profile",
  locale: {
    language: "en",
    script: "en",
  },

  addressComponents: {
    addressLine: {
      dataType: CharacterString,
      minCardinality: 1,
      maxCardinality: n
    },
  },

  addressClasses: {
    minimalAddress: {
      availableFields: [
        {
          componentType: addressLine,
          min: 1,
          max: n,
          description: "One line of this address",
          require: true,
        }
      ],
      displayTemplates: [
        {
          /* TODO */
          orientation: horizontal,
          text: "(\n)*"
          }
        }
      ]
    }
  }
}
```

Address Instance

```
addressInstance1 = {
  profile: "http://www.iso.org/tc211/tc211-minimal.adp",
  components: [
```

```
    {
      type: addressLine,
      value: 14 Church Street,
    },
    {
      type: addressLine,
      value: Hatfield
    },
    {
      type: addressLine,
      value: South Africa
    }
  ]
}

addressInstance2 = {
  profile: "http://www.iso.org/tc211/tc211-minimal.adp",
  components: [
    {
      type: addressLine,
      value: Statue of Liberty
    },
    {
      type: addressLine,
      value: Liberty Island
    },
    {
      type: addressLine,
      value: New York
    },
    {
      type: addressLine,
      value: NY
    }
  ]
}
```

D.1.2. ISO 19160-1 C3

```
profile = {

  id: "http://www.iso.org/tc211/tc211-sample.adp",
  type: "iso-19160-address-profile",
  publisher: "http://www.iso.org/tc211/",
  signature: "...",

  name: "TC 211 Minimal Address Profile",
  locale: {
    language: "en",
    script: "en",
  },

  dataTypes: {
    addressNumberValue: {
      primitiveType: Integer,
      maxValue: 10000,
      minValue: 1,
    },
    boxNumberValue: {
```



```
    primitiveType: Integer,
    maxValue: 100000,
    minValue: 1,
  }
}

addressComponents: {
  addressNumber: {
    dataType: addressNumberValue,
  },
  boxNumber: {
    dataType: boxNumberValue,
  },

  /* Table C.4. Address component type */
  thoroughfareName: {
    dataType: thoroughfareNameValue,
  },
  localityName: {
    dataType: CharacterString,
  },
  postOfficeName: {
    dataType: CharacterString
  },
  postCode: {
    dataType: CharacterString
  },
  countryName: {
    dataType: thoroughfareName,
  }
  addressNumber: {
    dataType: addressedObjectIdentifier,
  },
},

addressClasses: {
  streetAddress: {
    description: Street Address,
    availableFields: [
      {
        componentType: addressNumber,
        minCardinality: 1,
        maxCardinality: 1,
        required: true,
      },
      {
        componentType: thoroughfareName,
        minCardinality: 1,
        maxCardinality: 1,
        required: true,
      },
      {
        componentType: placeName
        dataType: CharacterString,
        minCardinality: 1,
        maxCardinality: 1,
        required: true,
      },
      {
```

```
          componentType: postCode
          minCardinality: 1,
          maxCardinality: 1,
          required: true,
        },
        {
          componentType: countryName
          minCardinality: 1,
          maxCardinality: 1,
          required: false,
        },
      ],
      displayTemplates: [
        {
          /* TODO */
        }
      ]
    },
    boxAddress: {
      availableFields: [
        {
          componentType: boxNumber,
          minCardinality: 1,
          maxCardinality: 1,
          required: true,
        },
        {
          componentType: postOfficeName,
          dataType: CharacterString,
          minCardinality: 1,
          maxCardinality: 1,
          required: true,
        },
        {
          componentType: postCode
          minCardinality: 1,
          maxCardinality: 1,
          required: true,
        },
        {
          componentType: countryName
          minCardinality: 1,
          maxCardinality: 1,
          required: false,
        },
      ],
      displayTemplates: [
        {
          /* TODO */
        }
      ]
    }
  }
}
```

Address Instance

```
addressInstance1 = {
  profile: "http://www.iso.org/tc211/tc211-sample.adp#streetAddress",
```



```
  components: [
    {
      type: addressNumber,
      value: 99
    },
    {
      type: thoroughfareName,
      value: {
        name: Lombardy,
        type: Street
      }
    },
    {
      type: placeName,
      value: The Hills,
    },
    {
      type: postCode,
      value: 0039,
    },
    {
      type: countryName,
      value: South Africa
    }
  ]
}

boxInstance1 = {
  profile: "http://www.iso.org/tc211/tc211-sample.adp#boxAddress",
  components: [
    {
      type: boxNumber,
      value: 345
    },
    {
      type: postOfficeName,
      value: Orlando,
    },
    {
      type: postCode,
      value: 2020
    },
    {
      type: countryName,
      value: South Africa
    }
  ]
}
```


Bibliography

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

[2] ISO/IEC 639-3:2007, *Codes for the representation of names of languages — Part 3: Alpha-3 code for comprehensive coverage of languages*

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

[4] ISO/IEC/IEEE 9945:2009, *Information technology -- Portable Operating System Interface (POSIX®) Base Specifications, Issue 7*

[5] ISO/IEC 14888-3, *Information technology -- Security techniques -- Digital signatures with appendix -- Part 3: Discrete logarithm based mechanisms*

[6] ISO 15924, *Information and documentation -- Codes for the representation of names of scripts*

[7] ISO/IEC 15961:2004, *Information technology -- Radio frequency identification (RFID) for item management -- Data protocol: application interface*

[8] ISO 19106:2004, *Geographic information -- Profiles*

[9] ISO 19139:2006, *Geographic information — Metadata — XML schema implementation*

[10] ISO/IEC/IEEE 31320-2:2012, *Information technology -- Modeling Languages -- Part 2: Syntax and Semantics for IDEF1X97 (IDEFobject)*

ICS XXX

Price based on 42 pages