CC/WD 19160-6:2019

# Addressing — Digital interchange models

THE CALENDARING AND SCHEDULING CONSORTIUM

TC VCARD

Ronald Tse **AUTHOR**

Peter Tam **AUTHOR**

**CALCONNECT STANDARD**

WORKING DRAFT

## WARNING FOR DRAFTS

This document is not a CalConnect Standard. It is distributed for review and comment, and is subject to change without notice and may not be referred to as a 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.



**The Calendaring and Scheduling Consortium, Inc.**
4390 Chaffin Lane
McKinleyville
California 95519
United States of America

copyright@calconnect.org
www.calconnect.org

# CONTENTS

# FOREWORD

The Calendaring and Scheduling Consortium ("CalConnect") is a global non-profit organization with the aim to facilitate interoperability of collaborative technologies and tools through open standards.

CalConnect works closely with international and regional partners, of which the full list is available on our website (https://www.calconnect.org/about/liaisons-and-relationships).

The procedures used to develop this document and those intended for its further maintenance are described in the CalConnect Directives.

In particular the different approval criteria needed for the different types of CalConnect documents should be noted. This document was drafted in accordance with the editorial rules of the CalConnect Directives.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. CalConnect 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 provided in the Introduction.

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

This document was prepared by Technical Committee *VCARD*.

# 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

The interchange of addresses require a common structure of address data. For instance of a software application in need of user input of address information, users need assistance from the application to enter the correct format and structure of the address for it to be machine readable and interoperable among different machines or systems which process the address data.

This document focuses on enabling software applications the digital interchange of address profiles, address instances and instructions for their input and display, together with model definitions and a registry mechanism to ensure they are publicly available.

This standard complements the other parts in the family of ISO 19160 standards:

— ISO 19160-1 describes conceptual models for addressing that allow specification of international address profiles. This International Standard further provides methods to utilize these models in a way suitable for electronic interchange.

— ISO 19160-4 defines key terms for postal addressing, postal address components and constraints on their use. It focuses on use cases for postal applications, specifying the methods to write or detect addresses on mail items. 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).

Addressability, and features provided within the object which the address points to, is not described in this document. Specifically, the `AddressableObject` from ISO 19160-1 is considered out of scope. The `AddressableObject` model is considered orthogonal to models in this document, but can easily be used together, for example, in a navigation map where an address points to an addressable object, which in turn provides a list of its extant facilities.

# 1. SCOPE

This document specifies a set of data models suitable for machine encoding of the digital storage and transmission of address information, called the “Address Interchange Object” (“AXO”) models, and describes the usage of them.

Specifically, this document provides:

– data models for the interchange of address profiles conforming to ISO 19160-1 (*Addressing — Part 1: Conceptual model*)

– data models for the interchange of address instances conforming to a specific profile of ISO 19160-1, *Addressing — Part 1: Conceptual model*

– data models for entry and display templates for entering and displaying address instances conforming to the profile and encoding rules above; and

– the management and operations of a register of address profiles conforming to ISO 19160-1, *Addressing — Part 1: Conceptual model*.

## 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, *Conceptual schema language - Geographic information*

— ISO 19106, *Profiles - Geographic information*

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

— ISO 19157, *Data quality - Geographic information*

— ISO 19135-1, *Procedures for item registration - Geographic information - Part 1: Fundamentals*

# 3. TERMS AND DEFINITIONS

For the purposes of this document, the terms and definitions given in ISO 19160-1 and the following apply.

## 3.1. lineage

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

[SOURCE: ISO 19115-1, Clause 4.9]

## 3.2. locale

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

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

## 3.3. 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, Clause 4.5]

## 3.4. provenance

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

## 3.5. data type

specification of a value domain with operations allowed on values in this domain

[SOURCE: ISO 19103, Clause 4.14]

## 3.6. primitive data type

A data type that has no super type. The primitive data type of a data type is the data type itself, if the data type has no super type, and otherwise the primitive data type of the super type of the data type.

[SOURCE: ISO/IEC 10179, Clause 4.23]

## 3.7. user defined data type

*data type* (Clause 3.5) defined by the user in an interchange address profile through the composure of other *data types* (Clause 3.5) and constraints

## 3.8. user defined data type definition

definition of a *user defined data type* (Clause 3.7)

## 3.9. address feature

marking on an address instance to indicate what it is capable of

## 3.10. address layout template

specification of layout and positioning of address components for an interchange address instance of an address class

## 3.11. address display template

address layout template (Clause 3.10) for the display of interchange address instances of an address class

## 3.12. address form template

address layout template (Clause 3.10) of an input form for the entry of address instances of an address class

## 3.13. address processor

entity that processes interchange address instances

## 3.14. address profile distributor

entity that distributes interchange address profiles

## 3.15. signature

the string of bits resulting from the signature process

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

## 3.16. 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, Clause 4.18]

## 3.17. verification key

a data item which is mathematically related to an entity's *signature key* (Clause 3.16) and which is used by the verifier in the verification process

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

## 3.18. object identifier

oid

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

[SOURCE: ISO/IEC 15961-3, Clause 3.1.16]

## 3.19. language identifier

language symbol

symbol that uniquely identifies a particular language

[SOURCE: ISO 639-3, Clause 3.3]

## 3.20. script

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

[SOURCE: ISO 15924, Clause 3.7]

## 3.21. script code

combination of characters used to represent the name of a *script* (Clause 3.20)

[SOURCE: ISO 15924, Clause 3.8]

## 3.22. URI

uniform resource identifier

[SOURCE: ISO 19103, Clause 5.3]

# 4. CONFORMANCE

## 4.1. General

This part of ISO 19160-6 defines four classes of requirements and conformance. Appendix A specifies how conformance with these classes shall be tested.

## 4.2. Address profile register

To conform to this standard, an address profile register shall satisfy all of the requirements specified for a register in ISO 19135-1. See Appendix A.3.

## 4.3. IxAddressProfile

Any machine-readable description of an ISO 19160-1 profile for which conformance is claimed shall pass all the requirements described in the abstract test suite in Appendix A.6.

## 4.4. IxAddressClass

Any machine-readable description of an ISO 19160-1 address class for which conformance is claimed shall pass all the requirements described in the abstract test suite in Appendix A.4.

## 4.5. IxAddressComponent

Any machine-readable description of an ISO 19160-1 address component for which conformance is claimed shall pass all the requirements described in the abstract test suite in Appendix A.5.

## 4.6. IxAddressInstance

Any instance of IxAddressProfile class for which IxAddressInstance conformance is claimed shall pass the requirements described in the abstract test suite in Appendix A.2.

## 4.7. FormTemplate

Any FormTemplate in Interchange Address Profile model for which FormTemplate conformance is claimed shall pass the requirements described in the abstract test suite in Appendix A.7.

## 4.8. DisplayTemplate

Any DisplayTemplate in Interchange Address Profile model for which DisplayTemplate conformance is claimed shall pass the requirements described in the abstract test suite in Appendix A.8.

# 5. ADDRESS PROFILE REGISTRY

## 5.1. General

The ISO address profile registry will be managed with version control software, publicly available with an API and/or a graphical user interface, such as a web-based interface, to satisfy both programatic and human use for the dissemination, display and management of the registry.

The ISO address profile register is a non-hierarchical register. References to principal registers and sub registers are excluded from this document.

This address profile registry follows the requirements specified in ISO 19135-1, and additional requirements specified in this standard.

Rules for managing a register of geographical information items, including the submission of information, are found in ISO 19135-1, Clause 6.

NOTE Reference to the ISO 19135-1 requirement is denoted by η (i.e. [η2] = Requirement 2).

## 5.2. Roles and responsibilities in the management of an address profile registry

The roles and responsibilities of the register owner, register manager, submitting organizations, control body, registry manager and register user are set out in ISO 19135-1.

Requirement 1 [η1]: The register owner shall set terms and conditions regarding different levels of access to the register and making the contents available to the public. In addition, the register owner shall specify the time period in which the approval process shall be completed.

Requirement 2 [η2]: The register owner shall appoint a register manager. A register owner may serve as the register manager for any register that it has established or it may appoint another organization to serve as the register manager.

Requirement 3 [η3]: The register owner shall decide whether a control body is required for the register and if so appoint the control body. The register owner may serve as the control body for any register that it has established or it may delegate that role to a subgroup within the organization or to the register manager.

Requirement 4 [η4]: A register owner shall specify the criteria that determine which organizations may act as submitting organizations.

Requirement 5 [η5]: The register owner shall clarify the process for a submitting organization to appeal decisions of the control body (if such a body is appointed). The register owner may establish a procedure for such a process. The specification of this procedure shall include appropriate time limits for completion of the process. An alternative solution may be for a submitting organization to resubmit a new proposal with changes or a better justification.

Requirement 6 [η6]: A register manager shall manage a register in conformance with ISO 19135-1, Clause 6.

Requirement 7 [η7]: Upon request, the register manager shall distribute an information package containing a description of the register and how to submit proposals for changes to the content of the register. The information package shall describe what proposed changes to the content may be considered to be substantive.

Requirement 8 [η8]: The register manager shall accept proposals from submitting organizations and manage the proposals as specified in ISO 19135-1, Clause 6.4. The register manager shall pass proposals to the control body for decisions as to acceptability and shall serve as the point of contact between the control body and the submitting organization for negotiations regarding changes to the proposal.

Requirement 9 [η9]: The register manager shall determine whether a submitting organization is qualified in accordance with the criteria established by the register owner.

Requirement 10 [η10]: If a control body is appointed, it shall accept proposals from the register manager and render a decision regarding each proposal within the time limits specified by the register owner.

Requirement 11 [η11]: A registry manager shall ensure the integrity of any register held in the registry and shall provide means for electronic access to the registry for register managers, control body members, and register users.

Requirement 12 [η12]: Register managers shall consider the requirements of different categories of users in selecting methods for publishing the content of a register.

Requirement 13: The ISO address profile register shall have a publicly available record of changes where historical content shall remain publicly available.

Submitting organizations for the ISO address profile register consist of organizations and/or persons responsible for defining and maintaining address profiles.

## 5.3. Unmodified ISO 19135-1:2015 requirements

The unmodified ISO 19135-1 requirements are as follows:

Requirement 14 [η13]: Every register shall have a technical document describing the item classes to be registered.

NOTE For the ISO address profile register that technical document is this document.

Requirement 15 [η14]: Items shall be individually managed, moving through a set of well-defined states. Information about the temporal history of each item shall be maintained.

Requirement 16 [η16]: A clarification shall not cause any substantive semantic or technical change to a registered item.

Requirement 17 [η17]: Clarification shall be accomplished by updating the existing item in the register. The clarification shall be recorded with a justification of the change and the date on which the register transaction was made.

Requirement 18 [η19]: Retirement shall be accomplished by leaving the item in the register, having its status changed to retired, and including the date on which the register transaction was made.

Requirement 19 [η21]: The register manager shall review proposals received from third parties for completeness and return proposals to the submitting organization if the proposal is incomplete or if the submitting organization is not qualified, else initiate the approval process.

Requirement 20 [η22]: The approval process shall be completed within the time period specified by the register owner.

Requirement 21 [η23]: A registry manager shall ensure that information about valid, invalidated, superseded, or retired items in the register is readily available to users.

## 5.4. Modified ISO 19135-1:2015 requirements

The ISO 19135-1 requirements modified for this document are as follows:

Requirement 22 [η15]: If an item is superseded by another item, the date the succession occurred shall be captured, along with references to and from the item that superseded it. At any given time, only one item in the

series should be “valid”.

NOTE 1 The requirement that only one item in the series is “valid” is removed.

Requirement 23 [ηi20]: If a register item is deemed to be no longer suitable for the use in the production of new data and has been superseded by a new register item, the original item shall remain in the register, shall have its status changed to superseded, have a reference to the item(s) that superseded it, including the date on which the register transaction was made.

NOTE 2 The option of removing a superseded item from the register is removed.

Requirement 24 [ηi18]: If an item in a register is found to have substantive error, it shall be left in the register, have its status changed to invalid, have a reference to the item(s) that replaced it, and have the date when the register transaction was made.

NOTE 3 The option of removing an invalidated item from the register is removed.

## 5.5. Specific requirements

Requirement 25: The ISO address profile registry shall follow ISO 19160-1, Clause 6 in the management of address profiles, including the submission of information.

## 5.6. Content requirements from ISO 19135-1:2015

The ISO address profile register shall conform to the core register schema in ISO 19135-1, Clause 7. This clause sets out specific requirements as follows:

NOTE Reference to the ISO 19135-1 requirement is denoted by η (i.e. [ηi2] = Requirement 2).

Requirement 26 [ηi24]: The core register shall conform to the register schema as specified in UML in ISO 19135-1, Clause 7.

Requirement 27 [ηi25]: The attribute identifier that designates an item class held in a register that conforms to ISO 19135-1, Clause 7, shall uniquely denote the item class within the context of the register.

Requirement 28 [ηi26]: The attribute `itemIdentifier` is represented as a `CharacterString` that is used to uniquely denote that item within an item class and is intended for information processing. Once a value has been assigned, it shall not be reused. The class/identifier union shall be unique within the register.

# 6. PROCESS OF ADDRESS PROFILE AND ADDRESS INSTANCE INTERCHANGE

Practical usage of AXO models rely on establishment of an address profile registry.

This clause sets out specific requirements in relation with ISO 19135-1.

NOTE Reference to the ISO 19135-1 requirement is denoted by η (i.e. [η2] = Requirement 2).

The roles and responsibilities of the register owner, register manager, submitting organizations, control body, registry manager and register user are set out in ISO 19135-1, Clause 5.

## 6.1. Address profiles

### 6.1.1. General

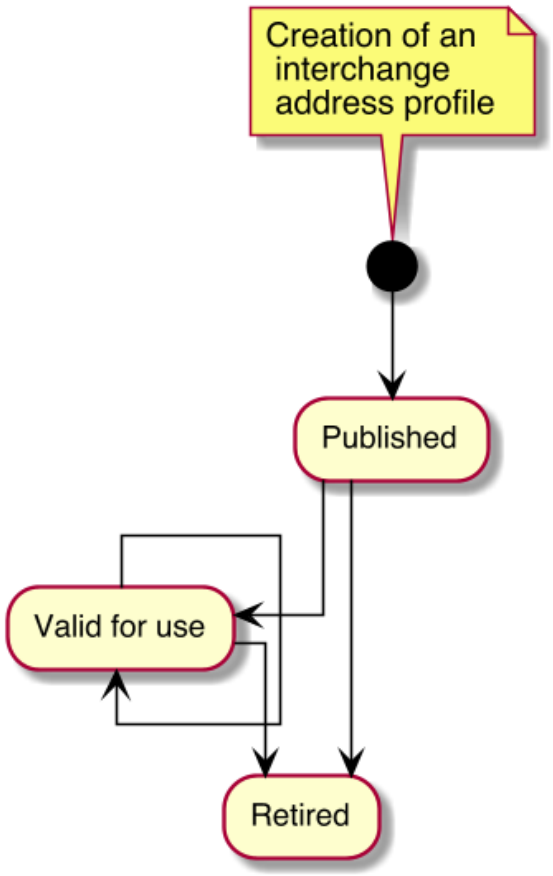


**Figure 1 — Lifecycle of an interchange address profile**

### 6.1.2. Creating interchange address profiles

Address profiles that comply with ISO 19160-1, Annex B and ISO 19106 Conformance class 1 are converted into interchange address profiles in accordance with requirements specified in Clause 9.

### 6.1.3. Publishing interchange address profiles

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

### 6.1.4. Updating interchange address profiles

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

### 6.1.5. Using interchange address profiles

Applications retrieve suitable interchange address 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 interchange address profile, and shall periodically check with the publisher (or a registry that the publisher distributes via) the latest version of the address profile.

### 6.1.6. Retiring interchange address profiles

Publisher can indicate the validity period of an interchange 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 interchange address profile, the publisher shall distribute a new version of the address profile with an expired validity period.

## 6.2. Address instances

### 6.2.1. General

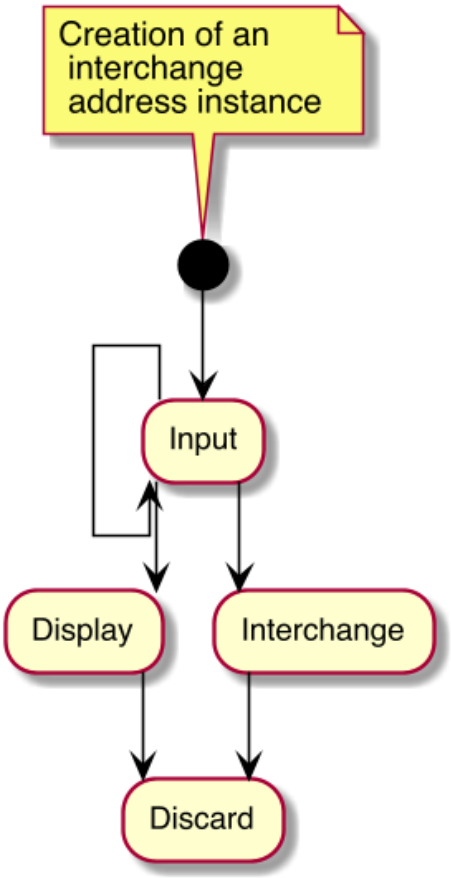


**Figure 2 — Lifecycle of an interchange address instance**

### 6.2.2. Creating an interchange address instance

This section describes how an interchange address instance is created. The desired interchange address profile shall be already retrieved for creating an interchange 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 interchange 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 features are marked on the interchange address instance.

### 6.2.3. Interchange of an interchange 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.

### 6.2.4. Displaying an interchange address instance

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

### 6.2.5. Adding address features

#### 6.2.5.1. Improving quality of an interchange 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 feature "https://standards.iso.org/19160/-6/features/specified" to the address instance.

To ensure that the resulting interchange 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 `AddressFeature` "https://standards.iso.org/19160/-6/features/confirmed" to the address instance.

#### 6.2.5.2. Verifying an interchange address instance

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

### 6.2.6. Adding associated data to an interchange address instance

A processor of an interchange 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 features could be added into the interchange address instance.

### 6.2.7. Discarding an interchange address instance

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

# 7. DATA TYPES

## 7.1. Primitive and core data types

These are the common data types used within this document.

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

- `CharacterString`
- `DateTime`, `Date`, `Time`
- `Number`, `Integer`, `Decimal`, `Real`
- `Vector`
- `Boolean`

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

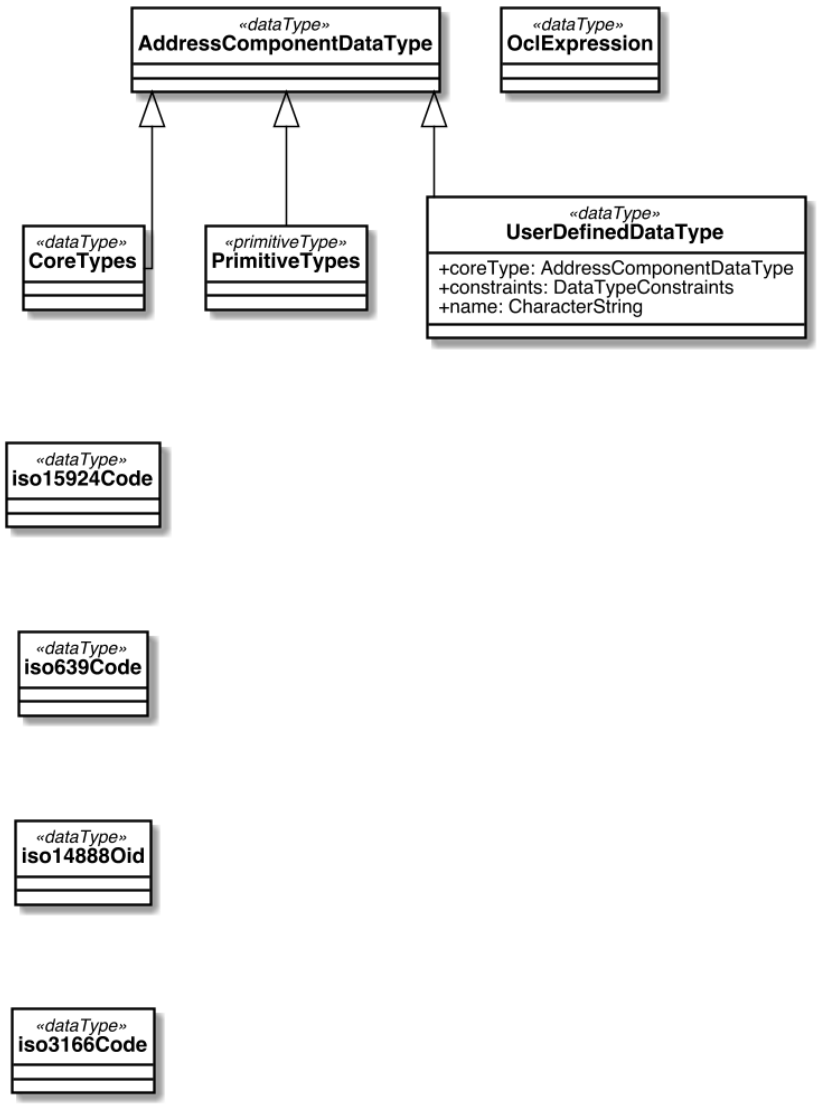


**Figure 3 – Common data types used in this document**

## 7.2. 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 |

### 7.2.1. 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”.

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.

# 8. COMMON MODELS

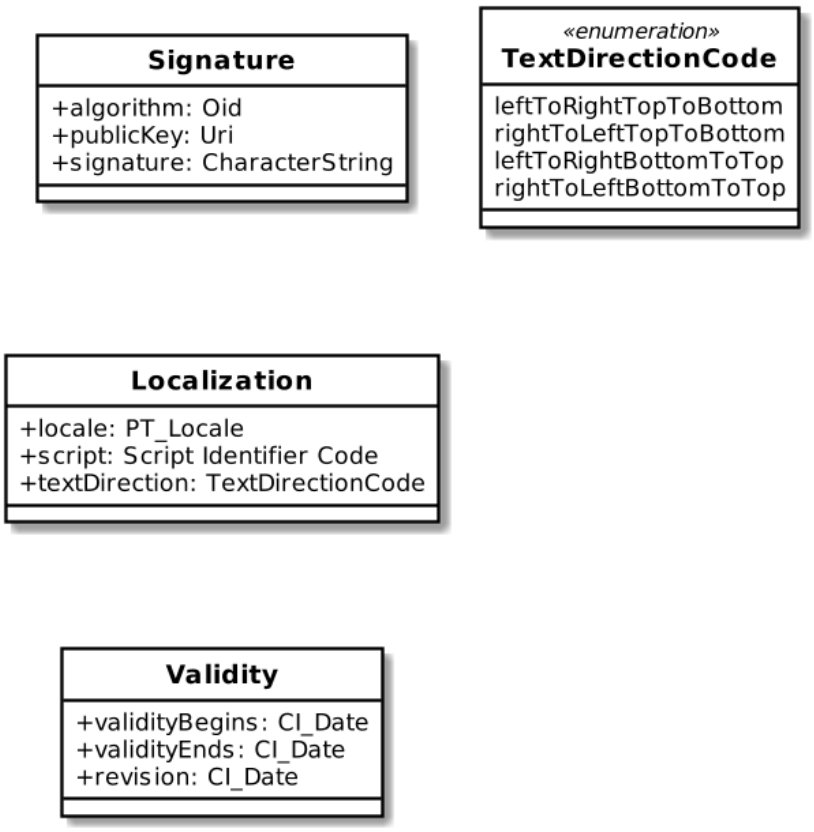


**Figure 4 — Common models data model**

## 8.1. 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.

Table 1 — Localization attributes

| Name | Definition | Mandatory/ Optional/ Conditional | Max Occur | Data Type |
|---|---|---|---|---|
| locale | The locale of the parent object. | M | 1 | ISO 19115-1 `PT_Locale` |
| script | The type of written script used in the parent object. | M | 1 | ISO 15924 `Script Identifier Code` |
| textDirection | Indicating in which direction the text of the parent should be read. | M | 1 | `TextDirectionCode` |

## 8.2. Validity

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

Table 2 — Validity attributes

| Name | Definition | Mandatory/ Optional/ Conditional | Max Occur | Data Type |
|---|---|---|---|---|
| validityBegins | The date and time when this object becomes valid. | M | 1 | ISO 19115-1 `CI_Date` |
| validityEnds | The date and time when this object becomes invalid. | M | 1 | ISO 19115-1 `CI_Date` |
| revision | Issuance date/time of this object. | M | 1 | ISO 19115-1 `CI_Date` |

## 8.3. Signature

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

Table 3 — Signature attributes

| Name | Definition | Mandatory/ Optional/ Conditional | Max Occur | Data Type |
|---|---|---|---|---|
| algorithm | The public key cryptographic algorithm used for this digital signature. | M | 1 | ??? `Oid` |
| 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. | M | 1 | `Uri` |
| signature | The actual digital signature value encoded in Base64 format. | M | 1 | `CharacterString` |

## 8.4. TextDirectionCode

An enumeration value of TextDirectionCode represents the reading direction of textual data whether it is from left to right or right to left, and from top to bottom or bottom to top.

Table 4 — TextDirectionCode values

| Name | Definition |
|---|---|

| | |
|---|---|
| leftToRightTopToBottom | Indicating that text should be read left to right, and top to bottom. |
| rightToLeftTopToBottom | Indicating that text should be read right to left, and top to bottom. |
| leftToRightBottomToTop | Indicating that text should be read left to right, and bottom to top. |
| rightToLeftBottomToTop | Indicating that text should be read right to left, and bottom to top. |

# 9. INTERCHANGE ADDRESS PROFILE

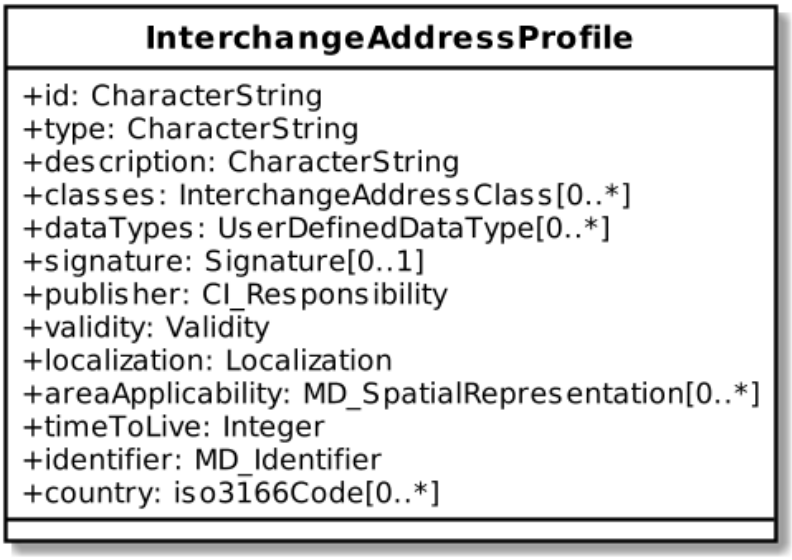


**Figure 5 — Interchange address profile data model**

## 9.1. InterchangeAddressProfile

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.

Table 5 — InterchangeAddressProfile attributes

| Name | Definition | Mandatory/ Optional/ Conditional | Max Occur | Data Type |
|---|---|---|---|---|
| id | Unique identifier of this Interchange Address Profile. | M | 1 | `CharacterString` |
| type | Intended usage of this profile. | M | 1 | `CharacterString` |
| description | Textual description of this profile. | M | 1 | `CharacterString` |
| classes | TODO: attribute classes's definition | O | N | `InterchangeAddressClass` |
| dataTypes | TODO: attribute dataTypes's definition | O | N | `UserDefinedDataType` |

| | | | | |
|---|---|---|---|---|
| signature | The digital signature to verify the integrity of this profile, and the identity of the publishing authority. | O | 1 | `Signature` |
| 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. | M | 1 | ISO 19115-1 `CI_Responsibility` |
| validity | The period of date and time that this profile should be considered as valid. | M | 1 | `Validity` |
| localization | TODO: attribute localization's definition | M | 1 | `Localization` |
| areaApplicability | The geographic representation of which this interchange address profile applies to. Overlapping geographic areas are allowed across different interchange address profiles. | O | N | `MD_SpatialRepresentation` |
| timeToLive | The maximum time interval between refreshing of this profile via an authoritative source, in seconds. | M | 1 | `Integer` |
| identifier | Used to identify this interchange address profile and provide description of it. | M | 1 | `MD_Identifier` |
| country | The country of which this interchange address profile represents. | O | N | `iso3166Code` |

# 10. INTERCHANGE ADDRESS CLASS

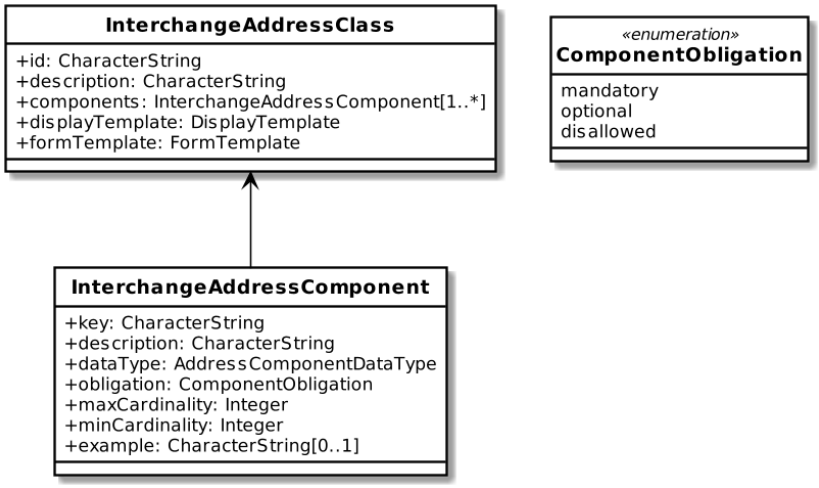


**Figure 6 — Interchange address class data model**

## 10.1. InterchangeAddressClass

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.

Table 6 — InterchangeAddressClass attributes

| | | | | |
|---|---|---|---|---|

| Name | Definition | Mandatory/ Optional/ Conditional | Max Occur | Data Type |
|---|---|---|---|---|
| id | Unique identifier of this interchange address class. | M | 1 | `CharacterString` |
| description | Textual description of this interchange address class. | M | 1 | `CharacterString` |
| components | TODO: attribute components's definition | M | N | `InterchangeAddressComponent` |
| displayTemplate | TODO: attribute displayTemplate's definition | M | 1 | `DisplayTemplate` |
| formTemplate | TODO: attribute formTemplate's definition | M | 1 | `FormTemplate` |

## 10.2. InterchangeAddressComponent

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

Table 7 — InterchangeAddressComponent attributes

| Name | Definition | Mandatory/ Optional/ Conditional | Max Occur | Data Type |
|---|---|---|---|---|
| key | An identifier of this interchange address component, shall be unique within the interchange address class. | M | 1 | `CharacterString` |
| description | Textual description of this component. | M | 1 | `CharacterString` |
| dataType | Describes the type of value accepted by this component. This takes an `AddressComponentDataType` value. | M | 1 | `AddressComponentDataType` |
| obligation | Whether this component is mandatory, optional or disallowed. | M | 1 | `ComponentObligation` |

| | | | | |
|---|---|---|---|---|
| | Values represented by the `ComponentObligation` object. | | | |
| maxCardinality | The maximum number of components within this address class. | M | 1 | `Integer` |
| minCardinality | The minimum number of components within this address class. | M | 1 | `Integer` |
| example | A textual example to demonstrate the correct use of this component. | O | 1 | `CharacterString` |

## 10.3. ComponentObligation

Table 8 – ComponentObligation values

| Name | Definition |
|---|---|
| mandatory | Indicating that a component must exist. |
| optional | Indicating that a component can exist. |
| disallowed | Indicating that a component must not exist. |

# 11. ADDRESS FEATURE

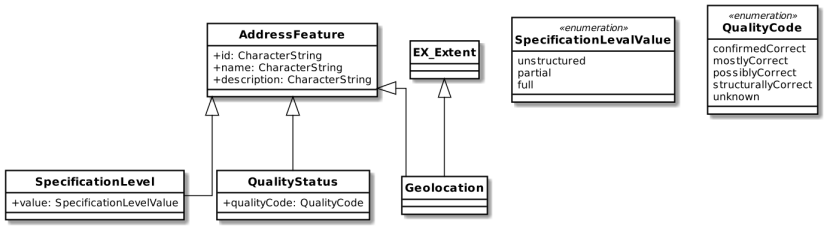


**Figure 7 — Address feature data model**

## 11.1. AddressFeature

Address feature 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 features, this class should be extended upon.

Table 9 — AddressFeature attributes

| Name | Definition | Mandatory/ Optional/ Conditional | Max Occur | Data Type |
|---|---|---|---|---|
| id | Unique identifier of the address feature. | M | 1 | `CharacterString` |
| name | Human readable name of the address feature. | M | 1 | `CharacterString` |
| description | Human readable description of the address feature. | M | 1 | `CharacterString` |

## 11.2. SpecificationLevel

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.

Table 10 — SpecificationLevel attributes

| Name | Definition | Mandatory/ Optional/ Conditional | Max Occur | Data Type |
|---|---|---|---|---|
| value | TODO: attribute value's definition | M | 1 | `SpecificationLevelValue` |

## 11.3. QualityStatus

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.

Table 11 — QualityStatus attributes

| Name | Definition | Mandatory/ Optional/ Conditional | Max Occur | Data Type |
|---|---|---|---|---|
| qualityCode | TODO: attribute qualityCode's definition | M | 1 | `QualityCode` |

## 11.4. Geolocation

An address is commonly linked to one or more geolocations. The value of a location is given by `EX_Extent` defined in ISO 19115-1, Clause 6.6.1, which supports geographic, temporal and vertical specification.

## 11.5. SpecificationLevalValue

Table 12 — SpecificationLevalValue values

| Name | Definition |
|---|---|
| unstructured | The user has entered free-form text with no regard of structuring them. |
| partial | Some components may be structurally correct, but some other components are still unstructured. |
| full | All components are accurately separated and cannot be further split. |

## 11.6. QualityCode

Table 13 — QualityCode values

| Name | Definition |
|---|---|
| confirmedCorrect | Indicating that the address is confirmed to be correct to the best of knowledge of the address processor. |
| mostlyCorrect | Indicating with high confidence that the address is correct. |
| possiblyCorrect | Indicating with confidence that the address is correct. |
| structurallyCorrect | 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. |

# 12. INTERCHANGE ADDRESS INSTANCE

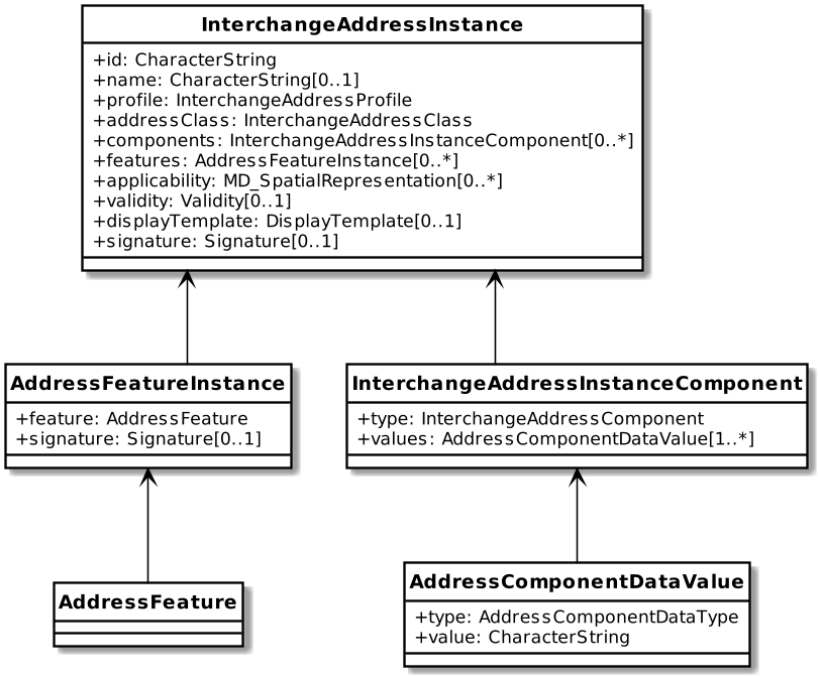


**Figure 8 — Interchange address instance data model**

## 12.1. InterchangeAddressInstance

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.

Table 14 — InterchangeAddressInstance attributes

| Name | Definition | Mandatory/ Optional/ Conditional | Max Occur | Data Type |
|---|---|---|---|---|
| id | Unique identifier for this interchange address instance. | M | 1 | `CharacterString` |
| name | Human-readable name for the interchange address instance. | O | 1 | `CharacterString` |

| | | | | |
|---|---|---|---|---|
| profile | The interchange address profile which contains the interchange address class it conforms to. | M | 1 | InterchangeAddressProfile |
| addressClass | The interchange address class it conforms to. | M | 1 | InterchangeAddressClass |
| components | TODO: attribute components's definition | O | N | InterchangeAddressInstanceComponent |
| features | TODO: attribute features's definition | O | N | AddressFeatureInstance |
| applicability | Spatial representation of the geographic area that this interchange address instance covers. | O | N | MD_SpatialRepresentation |
| validity | TODO: attribute validity's definition | O | 1 | Validity |
| displayTemplate | Display template that this interchange address instance should be shown with. This allows an address originator to provide a desired | O | 1 | DisplayTemplate |

| | | | | |
|---|---|---|---|---|
| | display view of the address. | | | |
| signature | Cryptographic signature of the interchage address instance itself togerther with its content. | O | 1 | `Signature` |

## 12.2. InterchangeAddressInstanceComponent

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.

Table 15 — InterchangeAddressInstanceComponent attributes

| Name | Definition | Mandatory/ Optional/ Conditional | Max Occur | Data Type |
|---|---|---|---|---|
| type | The interchange address component the value is for. | M | 1 | `InterchangeAddressComponent` |
| values | Values for the interchange address component specified in `type`. | M | N | `AddressComponentDataValue` |

## 12.3. AddressFeatureInstance

The address feature instance is incorporated into an interchange address instance to mark its features as determined by an address processor.

Table 16 — AddressFeatureInstance attributes

| Name | Definition | Mandatory/ Optional/ Conditional | Max Occur | Data Type |
|---|---|---|---|---|
| feature | The particular address feature that has been fulfilled. | M | 1 | `AddressFeature` |
| signature | Cryptographic signature used to ensure that the feature is marked by an address process trusted for verification of this feature. The signature | O | 1 | `Signature` |

| | | | | |
|---|---|---|---|---|
| | generated shall incorporate the `id` attribute of the interchange address instance that owns it. | | | |

## 12.4. AddressComponentDataValue

A single value for the interchange address instance component.

Table 17 — AddressComponentDataValue attributes

| Name | Definition | Mandatory/ Optional/ Conditional | Max Occur | Data Type |
|---|---|---|---|---|
| type | The data type of its value. | M | 1 | `AddressComponentDataType` |
| value | The value of the specified component. | M | 1 | `CharacterString` |

# 13. LAYOUT TEMPLATE

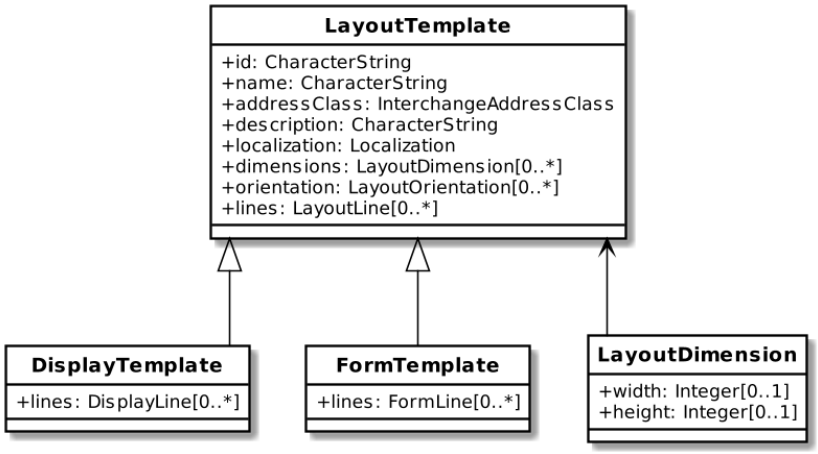


**Figure 9 — Layout template data model**

## 13.1. LayoutTemplate

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.

Table 18 — LayoutTemplate attributes

| Name | Definition | Mandatory/ Optional/ Conditional | Max Occur | Data Type |
|---|---|---|---|---|
| id | Unique identifier for the layout template. | M | 1 | `CharacterString` |
| name | Descriptive name of the layout template. | M | 1 | `CharacterString` |

| | | | | |
|---|---|---|---|---|
| addressClass | TODO: attribute addressClass's definition | M | 1 | `InterchangeAddressClass` |
| description | Textual description of the layout template. | M | 1 | `CharacterString` |
| localization | Locale and script information of the layout template. | M | 1 | `Localization` |
| dimensions | Physical dimensions of the rectangular bounding box for the rendered layout output. | O | N | `LayoutDimension` |
| orientation | Horizontal or vertical of the text orientation. | O | N | `LayoutOrientation` |
| lines | TODO: attribute lines's definition | O | N | `LayoutLine` |

## 13.2. DisplayTemplate

Table 19 — DisplayTemplate attributes

| Name | Definition | Mandatory/ Optional/ Conditional | Max Occur | Data Type |
|---|---|---|---|---|
| lines | TODO: attribute lines's definition | O | N | `DisplayLine` |

## 13.3. FormTemplate

Table 20 — FormTemplate attributes

| Name | Definition | Mandatory/ Optional/ Conditional | Max Occur | Data Type |
|---|---|---|---|---|
| lines | TODO: attribute lines's definition | O | N | `FormLine` |

## 13.4. LayoutDimension

Table 21 — LayoutDimension attributes

| Name | Definition | Mandatory/ Optional/ Conditional | Max Occur | Data Type |
|---|---|---|---|---|

| | | | | |
|---|---|---|---|---|
| width | TODO: attribute width's definition | 0 | 1 | `Integer` |
| height | TODO: attribute height's definition | 0 | 1 | `Integer` |

# 14. LAYOUT LINE

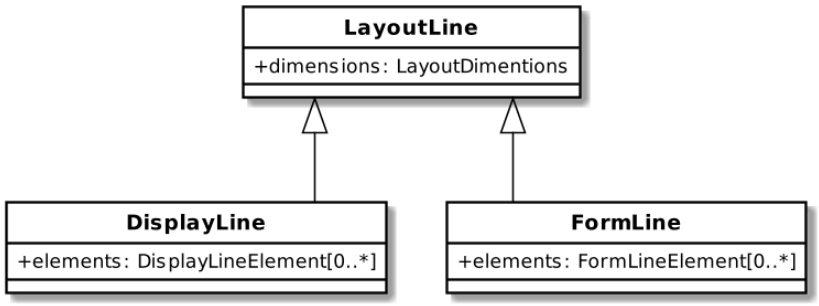


**Figure 10 – Layout line data model**

## 14.1. LayoutLine

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.

Table 22 – LayoutLine attributes

| Name | Definition | Mandatory/ Optional/ Conditional | Max Occur | Data Type |
|---|---|---|---|---|
| dimensions | Physical dimensions of the rectangular bounding box for the rendered Line. | M | 1 | `LayoutDimentions` |

## 14.2. DisplayLine

Table 23 – DisplayLine attributes

| Name | Definition | Mandatory/ Optional/ Conditional | Max Occur | Data Type |
|---|---|---|---|---|
| elements | TODO: attribute elements's definition | O | N | `DisplayLineElement` |

## 14.3. FormLine

Table 24 – FormLine attributes

| Name | Definition | Mandatory/ Optional/ Conditional | Max Occur | Data Type |
|---|---|---|---|---|

| | | | | |
|---|---|---|---|---|
| elements | TODO: attribute elements's definition | 0 | N | `FormLineElement` |

# 15. DISPLAY TEMPLATE

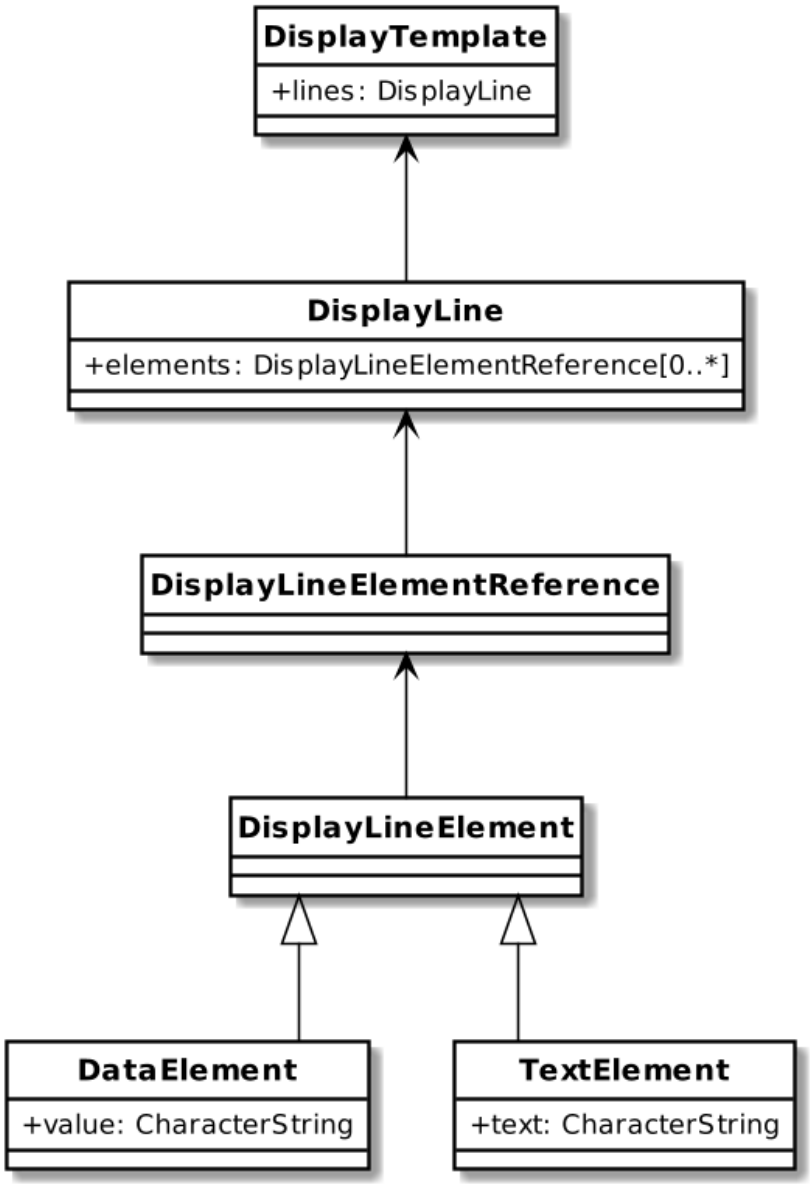


**Figure 11 — Display template data model**

## 15.1. DisplayTemplate

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.

Table 25 — DisplayTemplate attributes

| Name | Definition | Mandatory/ Optional/ Conditional | Max Occur | Data Type |
|---|---|---|---|---|
| lines | TODO: attribute lines's definition | M | 1 | `DisplayLine` |

## 15.2. DisplayLine

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`).

Table 26 — DisplayLine attributes

| Name | Definition | Mandatory/ Optional/ Conditional | Max Occur | Data Type |
|---|---|---|---|---|
| elements | TODO: attribute elements's definition | O | N | `DisplayLineElementReference` |

## 15.3. DisplayLineElement

This represents an element within a display line.

## 15.4. DataElement

Representing a variable data value that is contained in the interchange address component, such as the PO box number following the phrase "PO Box".

Table 27 — DataElement attributes

| Name | Definition | Mandatory/ Optional/ Conditional | Max Occur | Data Type |
|---|---|---|---|---|
| value | TODO: attribute value's definition | M | 1 | `CharacterString` |

## 15.5. TextElement

Representing static text, such as the phrase "PO Box" preceding the actual PO box number, for display layouts for PO boxes.

Table 28 — TextElement attributes

| Name | Definition | Mandatory/ Optional/ Conditional | Max Occur | Data Type |
|---|---|---|---|---|
| text | TODO: attribute text's definition | M | 1 | `CharacterString` |

# 16. FORM TEMPLATE

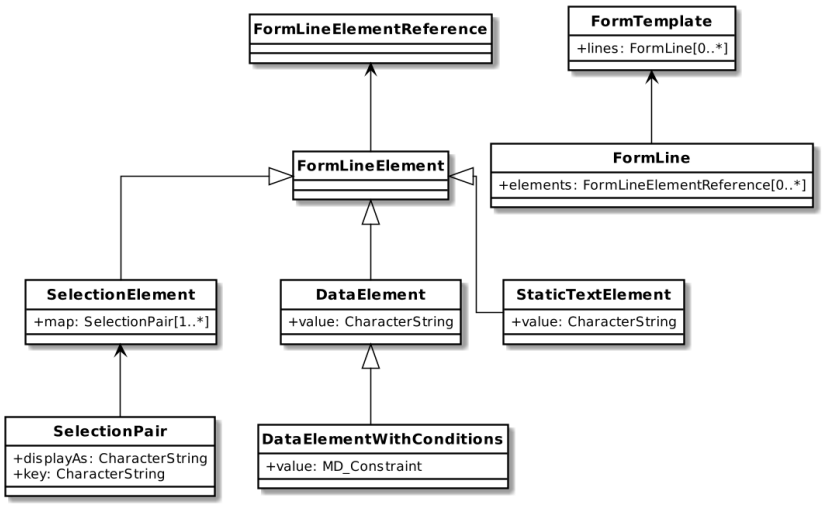


**Figure 12 — Form template data model**

## 16.1. FormTemplate

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 PATDL.

Table 29 — FormTemplate attributes

| Name | Definition | Mandatory/ Optional/ Conditional | Max Occur | Data Type |
|---|---|---|---|---|
| lines | TODO: attribute lines's definition | O | N | `FormLine` |

## 16.2. FormLine

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`).

Table 30 — FormLine attributes

| Name | Definition | Mandatory/ Optional/ | Max Occur | Data Type |
|---|---|---|---|---|

| | | | | |
|---|---|---|---|---|
| | | Conditional | | |
| elements | TODO: attribute elements's definition | O | N | `FormLineElementReference` |

## 16.3. FormLineElement

This represents an element within a form line.

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.

## 16.4. SelectionPair

Table 31 — SelectionPair attributes

| Name | Definition | Mandatory/ Optional/ Conditional | Max Occur | Data Type |
|---|---|---|---|---|
| displayAs | TODO: attribute displayAs's definition | M | 1 | `CharacterString` |
| key | TODO: attribute key's definition | M | 1 | `CharacterString` |

## 16.5. SelectionElement

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`.

Table 32 — SelectionElement attributes

| Name | Definition | Mandatory/ Optional/ Conditional | Max Occur | Data Type |
|---|---|---|---|---|
| map | TODO: attribute map's definition | M | N | `SelectionPair` |

## 16.6. DataElement

Table 33 — DataElement attributes

| Name | Definition | Mandatory/ Optional/ Conditional | Max Occur | Data Type |
|---|---|---|---|---|

| | | | | |
|---|---|---|---|---|
| value | 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`). | M | 1 | `CharacterString` |

## 16.7. StaticTextElement

Table 34 — StaticTextElement attributes

| Name | Definition | Mandatory/ Optional/ Conditional | Max Occur | Data Type |
|---|---|---|---|---|
| value | 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. | M | 1 | `CharacterString` |

## 16.8. DataElementWithConditions

Table 35 — DataElementWithConditions attributes

| Name | Definition | Mandatory/ Optional/ Conditional | Max Occur | Data Type |
|---|---|---|---|---|
| value | TODO: attribute value’s definition | M | 1 | `MD_Constraint` |

# APPENDIX A
(INFORMATIVE)
# ABSTRACT TEST SUITES

## A.1. Introduction

The abstract test suites for the conformance classes defined by this part of ISO 19160-6 are presented in Appendix A.2 to Appendix A.8.

## A.2. Conformance class: IxAddressInstance

Table A.1 — IxAddressInstance test 1: Associations

| | |
|---|---|
| Test purpose | Check that the model contains the associations as specified. |
| Test method | Inspect the model |
| Reference | Clause 12 |
| Test type | Basic |

Table A.2 — IxAddressInstance test 2: 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 12 |
| Test type | Basic |

## A.3. Conformance class: Address profile register

Refer to ISO 19135-1 for requirements.

## A.4. Conformance class: IxAddressClass

Table A.3 — IxAddressClass test 1: Associations

| | |
|---|---|
| Test purpose | Check that the model contains the associations as specified. |
| Test method | Inspect the model |
| | |

| | |
|---|---|
| Reference | Clause 10 |
| Test type | Basic |

## A.5. Conformance class: IxAddressComponent

Table A.4 — IxAddressComponent test 1: Associations

| | |
|---|---|
| Test purpose | Check that the model contains the associations as specified. |
| Test method | Inspect the model |
| Reference | ??? |
| Test type | Basic |

## A.6. Conformance class: IxAddressProfile

Table A.5 — IxAddressProfile test 1: Associations

| | |
|---|---|
| Test purpose | Check that the model contains the associations as specified. |
| Test method | Inspect the model |
| Reference | Clause 9 |
| Test type | Basic |

Table A.6 — 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 9 |
| Test type | Basic |

## A.7. Conformance class: FormTemplate

Table A.7 — 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 16 |
| Test type | Basic |

## A.8. Conformance class: DisplayTemplate

Table A.8 — 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 15 |
| Test type | Basic |

# APPENDIX B
(INFORMATIVE)
# 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 AddressFeature.

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 1109,
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.

# APPENDIX C
(INFORMATIVE)
# EXAMPLES OF OBJECTS SPECIFIED IN THIS DOCUMENT

Models specified in this document can be represented in various object structures, including in XML ISO 19139 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": { ... },
    "formTemplate": { ... }
  }
}
```

## C.3. Validity

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

## C.4. PublisherInformation

```
"publisher": {
  "publisherName": "UK Post Office",
  "publisherUri": "https://www.postoffice.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.postoffice.co.uk/profile-signature.key",
  "signature": "BOLVMNoGNM1TLglnlxgm0a9t"
}
```

### C.7. IxAddressClass

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

### 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. AddressFeature

```
"addressFeature": {
  "feature": "https://standards.iso.org/19160/-6/features/specified",
  "signature": [ ... ]
}
```

## C.14. DisplayTemplate

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

## C.15. Form template (FormTemplate)

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

# APPENDIX D
(INFORMATIVE)
# 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
    },
  },

  "addressClasses": {
    "minimalAddress": {
      "availableFields": [
        {
          "componentType": "addressLine",
          "min": 1,
          "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 639-3, *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, *Portable Operating System Interface (POSIX®) Base Specifications, Issue 7 - Information technology*

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

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

— [7] ISO/IEC 15961-3, *Data protocol for radio frequency identification (RFID) for item management - Information technology - Part 3: RFID data constructs*

— [8] ISO 19160-4, *Addressing - Part 4: International postal address components and template language*

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

— [10] ISO/IEC 10179, *Processing languages - Information technology - Document Style Semantics and Specification Language (DSSSL)*

— [11] ISO 19139, *AUTOFILL*