SDMX Standards: Section 3A PART I

SDMX-ML:

Schema and Documentation

Message Namespace

(Version 3.0)

September 2021

Contents

[1 Introduction 2](#_Toc82183889)

[2 Schema Documentation 2](#_Toc82183890)

[2.1 Message Namespace 2](#_Toc82183891)

[2.1.1 Summary 2](#_Toc82183892)

[2.1.2 Global Elements 3](#_Toc82183893)

[2.1.3 Complex Types 3](#_Toc82183894)

[2.1.4 Simple Types 18](#_Toc82183895)

[2.2 Message Footer Namespace 19](#_Toc82183896)

[2.2.1 Summary 19](#_Toc82183897)

[2.2.2 Global Elements 19](#_Toc82183898)

[2.2.3 Complex Types 19](#_Toc82183899)

[2.2.4 Simple Types 21](#_Toc82183900)

# Introduction

At the core of the SDMX XML messages are the message namespaces. These namespaces define the general structure of all messages and define the specific messages that are available for exchange in SDMX.

There are two namespaces associated with the messages. The main namespace schema which defines every message in SDMX is http://www.sdmx.org/resources/sdmxml/schemas/v3\_0/message. Associated with this is another sub-namespace, http://www.sdmx.org/resources/sdmxml/schemas/v3\_0/message/footer. This namespace defines footer level information that is available in messages which might require non-standard payload information to be transmitted.

In general, every message in SDMX follows common format of:

* Header
* Payload (0..n)
* Footer (0..1)

The header and payload elements exist in the message namespace, but the content of the payload is defined in the namespaces that are specific the functionality of the messages. Note that the header follows a common construct, which is then restricted according to the requirements of the message which is using it.

# Schema Documentation

## Message Namespace

**http://www.sdmx.org/resources/sdmxml/schemas/v3\_0/message**

### Summary

Referenced Namespaces:

| **Namespace** | **Prefix** |
| --- | --- |
| http://www.sdmx.org/resources/sdmxml/schemas/v3\_0/common | common |
| http://www.sdmx.org/resources/sdmxml/schemas/v3\_0/data/structurespecific | dsd |
| http://www.sdmx.org/resources/sdmxml/schemas/v3\_0/message/footer | footer |
| http://www.sdmx.org/resources/sdmxml/schemas/v3\_0/metadata/generic | metadata |
| http://www.sdmx.org/resources/sdmxml/schemas/v3\_0/registry | registry |
| http://www.sdmx.org/resources/sdmxml/schemas/v3\_0/structure | structure |
| http://www.w3.org/2001/XMLSchema | xs |

Contents:

7 Global Elements  
16 Complex Types  
1 Simple Type

### Global Elements

**Structure (StructureType):** Structure is a message that contains structural metadata. It may contain any of the following; categorisations, category schemes, code lists, concepts (concept schemes), constraints (attachment and content) data flows, hierarchical code lists, metadata flows, metadata structure definitions, organisation schemes, processes, reporting taxonomies, and structure sets.

**StructureSpecificData (StructureSpecificDataType):** StructureSpecificData is used to convey data structure specific according to data structure definition. The payload of this message (i.e. the data sets) will be based on XML schemas which are specific to the data structure definition and the orientation (i.e. the observation dimension) of the data.

**GenericMetadata (GenericMetadataType):** GenericMetadata contains reported metadata in a format which supports any metadata structure definition.

**RegistryInterface (RegistryInterfaceType):** RegistryInterface is used to conduct all interactions with the SDMX Registry Services.

**SubmitStructureRequest (SubmitStructureRequestType):** SubmitStructureRequest is used to submit structure definitions to the repository. The structure resources (key families, agencies, concepts and concept schemes, code lists, etc.) to be submitted may be communicated in-line or be supplied in a referenced SDMX-ML Structure messages external to the registry. A response will indicate status and contain any relevant error information.

**SubmitStructureResponse (SubmitStructureResponseType):** SubmitStructureResponse is returned by the registry when a structure submission request is received. It indicates the status of the submission, and carries any error messages which are generated, if relevant.

**Error (ErrorType):** Error is used to communicate that an error has occurred when responding to a request in an non-registry environment. The content will be a collection of error messages.

### Complex Types

***MessageType*:** MessageType is an abstract type which is used by all of the messages, to allow inheritance of common features. Every message consists of a mandatory header, followed by optional payload (which may occur multiple times), and finally an optional footer section for conveying error, warning, and informational messages.

Content:

Header, {any element with namespace of http://www.sdmx.org/resources/sdmxml/schemas/v3\_0/message}\*, Footer?

Element Documentation:

| **Name** | **Type** | **Documentation** |
| --- | --- | --- |
| Header | *BaseHeaderType* |  |
| Footer | FooterType | Footer is used to communicate information such as error and warnings after the payload of a message. |

**StructureType:** StructureType defines the contents of a structure message. The payload is optional since this message may be returned from a web service with only information in the footer.

Derivation:

*MessageType* (restriction)   
   StructureType

Content:

Header, Structures?, Footer?

Element Documentation:

| **Name** | **Type** | **Documentation** |
| --- | --- | --- |
| Header | StructureHeaderType |  |
| Structures | StructuresType |  |
| Footer | FooterType | Footer is used to communicate information such as error and warnings after the payload of a message. |

**StructureSpecificDataType:** StructureSpecificDataType defines the structure of the structure specific data message. Note that the data set payload type is abstract, and therefore it will have to be assigned a type in an instance. This type must be derived from the base type referenced. This base type defines a general structure which can be followed to allow for generic processing of the data even if the exact details of the data structure specific format are not known.

Derivation:

*MessageType* (restriction)   
   StructureSpecificDataType

Content:

Header, DataSet\*, Footer?

Element Documentation:

| **Name** | **Type** | **Documentation** |
| --- | --- | --- |
| Header | StructureSpecificDat aHeaderType |  |
| DataSet | *DataSetType* |  |
| Footer | FooterType | Footer is used to communicate information such as error and warnings after the payload of a message. |

**GenericMetadataType:** GenericMetadataType defines the contents of a generic metadata message.

Derivation:

*MessageType* (restriction)   
   GenericMetadataType

Content:

Header, MetadataSet\*, Footer?

Element Documentation:

| **Name** | **Type** | **Documentation** |
| --- | --- | --- |
| Header | GenericMetadataHeade rType |  |
| MetadataSet | MetadataSetType |  |
| Footer | FooterType | Footer is used to communicate information such as error and warnings after the payload of a message. |

**RegistryInterfaceType:** This is a type which describes a structure for holding all of the various dedicated registry interface message types.

Derivation:

*MessageType* (restriction)   
   RegistryInterfaceType

Content:

Header, (SubmitRegistrationsRequest | SubmitRegistrationsResponse | QueryRegistrationRequest | QueryRegistrationResponse | SubmitStructureRequest | SubmitStructureResponse | SubmitSubscriptionsRequest | SubmitSubscriptionsResponse | QuerySubscriptionRequest | QuerySubscriptionResponse | NotifyRegistryEvent)?, Footer?

Element Documentation:

| **Name** | **Type** | **Documentation** |
| --- | --- | --- |
| Header | BasicHeaderType |  |
| SubmitRegistrationsR equest | SubmitRegistrationsR equestType | SubmitRegistrationsRequest is sent to the registry by an agency or data/metadata provider to request one or more registrations for a data set or metadata set. The data source to be registered must be accessible to the registry services at an indicated URL, so that it can be processed by those services. |
| SubmitRegistrationsR esponse | SubmitRegistrationsR esponseType | SubmitRegistrationsResponse is sent to the agency or data/metadata provider in response to a submit registrations request. It indicates the success or failure of each registration request, and contains any error messages generated by the registration service. |
| QueryRegistrationReq uest | QueryRegistrationReq uestType | QueryRegistrationRequest is used to query the contents of a registry for data sets and metadata sets. It specifies whether the result set should include metadata sets, data sets, or both. The search can be characterized by providing constraints including reference periods, data regions, and data keys. |
| QueryRegistrationRes ponse | QueryRegistrationRes ponseType | QueryRegistrationResponse is sent as a response to any query of the contents of a registry. The result set contains a set of links to data and/or metadata If the result set is null, or there is some other problem with the query, then appropriate error messages and statuses will be returned. |
| SubmitStructureReque st | SubmitStructureReque stType | SubmitStructureRequest is used to submit structure definitions to the repository. The structure resources (key families, agencies, concepts and concept schemes, code lists, etc.) to be submitted may be communicated in-line or be supplied in a referenced SDMX-ML Structure messages external to the registry. A response will indicate status and contain any relevant error information. |
| SubmitStructureRespo nse | SubmitStructureRespo nseType | SubmitStructureResponse is returned by the registry when a structure submission request is received. It indicates the status of the submission, and carries any error messages which are generated, if relevant. |
| SubmitSubscriptionsR equest | SubmitSubscriptionsR equestType | SubmitSubscriptionsRequest contains one or more requests submitted to the registry to subscribe to registration and change events for specific registry resources. |
| SubmitSubscriptionsR esponse | SubmitSubscriptionsR esponseType | SubmitSubscriptionsResponse is the response to a submit subscriptions request. It contains information which describes the success or failure of each subscription request, providing any error messages in the event of failure. If successful, it returns the registry URN of the subscription, and the subscriber-assigned ID. |
| QuerySubscriptionReq uest | QuerySubscriptionReq uestType | QuerySubscriptionRequest is used to query the registry for the subscriptions of a given organisation. |
| QuerySubscriptionRes ponse | QuerySubscriptionRes ponseType | QuerySubscriptionResponse is sent as a response to a subscription query. If the query is successful, the details of all subscriptions for the requested organisation are sent. |
| NotifyRegistryEvent | NotifyRegistryEventT ype | NotifyRegistryEvent is sent by the registry services to subscribers, to notify them of specific registration and change events. Basic information about the event, such as the object that triggered it, the time of the event, the action that took place, and the subscription that triggered the notification are always sent. Optionally, the details of the changed object may also be provided. |
| Footer | FooterType | Footer is used to communicate information such as error and warnings after the payload of a message. |

**SubmitStructureRequestType:** SubmitStructureRequestType defines the structure of a registry submit structure request document.

Derivation:

*MessageType* (restriction)   
   RegistryInterfaceType (restriction)   
         SubmitStructureRequestType

Content:

Header, SubmitStructureRequest

Element Documentation:

| **Name** | **Type** | **Documentation** |
| --- | --- | --- |
| Header | BasicHeaderType |  |
| SubmitStructureReque st | SubmitStructureReque stType | SubmitStructureRequest is used to submit structure definitions to the repository. The structure resources (key families, agencies, concepts and concept schemes, code lists, etc.) to be submitted may be communicated in-line or be supplied in a referenced SDMX-ML Structure messages external to the registry. A response will indicate status and contain any relevant error information. |

**SubmitStructureResponseType:** SubmitStructureResponseType defines the structure of a registry submit registration response document.

Derivation:

*MessageType* (restriction)   
   RegistryInterfaceType (restriction)   
         SubmitStructureResponseType

Content:

Header, SubmitStructureResponse, Footer?

Element Documentation:

| **Name** | **Type** | **Documentation** |
| --- | --- | --- |
| Header | BasicHeaderType |  |
| SubmitStructureRespo nse | SubmitStructureRespo nseType | SubmitStructureResponse is returned by the registry when a structure submission request is received. It indicates the status of the submission, and carries any error messages which are generated, if relevant. |
| Footer | FooterType | Footer is used to communicate information such as error and warnings after the payload of a message. |

**ErrorType:** ErrorType describes the structure of an error response.

Content:

ErrorMessage+

Element Documentation:

| **Name** | **Type** | **Documentation** |
| --- | --- | --- |
| ErrorMessage | CodedStatusMessageTy pe | ErrorMessage contains the error message. It can occur multiple times to communicate message for multiple errors, or to communicate the error message in parallel languages. If both messages for multiple errors and parallel language messages are used, then each error message should be given a code in order to distinguish message for unique errors. |

***BaseHeaderType*:** BaseHeaderType in an abstract base type that defines the basis for all message headers. Specific message formats will refine this

Content:

ID, Test, Prepared, Sender, Receiver\*, Name\*, Structure\*, DataProvider?, MetadataProvider?, DataSetAction?, DataSetID\*, Extracted?, ReportingBegin?, ReportingEnd?, EmbargoDate?, Source\*

Element Documentation:

| **Name** | **Type** | **Documentation** |
| --- | --- | --- |
| ID | IDType | ID identifies an identification for the message, assigned by the sender. |
| Test | xs:boolean | Test indicates whether the message is for test purposes or not. |
| Prepared | HeaderTimeType | Prepared is the date the message was prepared. |
| Sender | SenderType | Sender is information about the party that is transmitting the message. |
| Receiver | PartyType | Receiver is information about the party that is the intended recipient of the message. |
| Name | TextType | Name provides a name for the transmission. Multiple instances allow for parallel language values. |
| Structure | *PayloadStructureType* | Structure provides a reference to the structure (either explicitly or through a structure usage reference) that describes the format of data or reference metadata. In addition to the structure, it is required to also supply the namespace of the structure specific schema that defines the format of the data/metadata. For cross sectional data, additional information is also required to state which dimension is being used at the observation level. This information will allow the structure specific schema to be generated. For generic format messages, this is used to simply reference the underlying structure. It is not mandatory in these cases and the generic data/metadata sets will require this reference explicitly. |
| DataProvider | DataProviderReferenc eType | DataProvider identifies the provider of the data for a data message. |
| MetadataProvider | MetadataProviderRefe renceType | MetadataProvider identifies the provider of the metadata for a metadata message. |
| DataSetAction | ActionType | DataSetAction code provides a code for determining whether the enclosed message is an Update or Delete message (not to be used with the UtilityData message). |
| DataSetID | IDType | DataSetID provides an identifier for a contained data set. |
| Extracted | xs:dateTime | Extracted is a time-stamp from the system rendering the data. |
| ReportingBegin | ObservationalTimePer iodType | ReportingBegin provides the start of the time period covered by the message (in the case of data). |
| ReportingEnd | ObservationalTimePer iodType | ReportingEnd provides the end of the time period covered by the message (in the case of data). |
| EmbargoDate | xs:dateTime | EmbargoDate holds a time period before which the data included in this message is not available. |
| Source | TextType | Source provides human-readable information about the source of the data. |

**StructureHeaderType:** StructureHeaderType defines the structure for structural metadata messages.

Derivation:

*BaseHeaderType* (restriction)   
   StructureHeaderType

Content:

ID, Test, Prepared, Sender, Receiver\*, Name\*, Source\*

Element Documentation:

| **Name** | **Type** | **Documentation** |
| --- | --- | --- |
| ID | IDType | ID identifies an identification for the message, assigned by the sender. |
| Test | xs:boolean | Test indicates whether the message is for test purposes or not. |
| Prepared | HeaderTimeType | Prepared is the date the message was prepared. |
| Sender | SenderType | Sender is information about the party that is transmitting the message. |
| Receiver | PartyType | Receiver is information about the party that is the intended recipient of the message. |
| Name | TextType | Name provides a name for the transmission. Multiple instances allow for parallel language values. |
| Source | TextType | Source provides human-readable information about the source of the data. |

**StructureSpecificDataHeaderType:** StructureSpecificDataHeaderType defines the header structure for a structure specific data message.

Derivation:

*BaseHeaderType* (restriction)   
   StructureSpecificDataHeaderType

Content:

ID, Test, Prepared, Sender, Receiver\*, Name\*, Structure+, DataProvider?, DataSetAction?, DataSetID\*, Extracted?, ReportingBegin?, ReportingEnd?, EmbargoDate?, Source\*

Element Documentation:

| **Name** | **Type** | **Documentation** |
| --- | --- | --- |
| ID | IDType | ID identifies an identification for the message, assigned by the sender. |
| Test | xs:boolean | Test indicates whether the message is for test purposes or not. |
| Prepared | HeaderTimeType | Prepared is the date the message was prepared. |
| Sender | SenderType | Sender is information about the party that is transmitting the message. |
| Receiver | PartyType | Receiver is information about the party that is the intended recipient of the message. |
| Name | TextType | Name provides a name for the transmission. Multiple instances allow for parallel language values. |
| Structure | StructureSpecificDat aStructureType | Structure provides a reference to the structure (either explicitly or through a structure usage reference) that describes the format of data or reference metadata. In addition to the structure, it is required to also supply the namespace of the structure specific schema that defines the format of the data/metadata. For cross sectional data, additional information is also required to state which dimension is being used at the observation level. This information will allow the structure specific schema to be generated. For generic format messages, this is used to simply reference the underlying structure. It is not mandatory in these cases and the generic data/metadata sets will require this reference explicitly. |
| DataProvider | DataProviderReferenc eType | DataProvider identifies the provider of the data for a data message. |
| DataSetAction | ActionType | DataSetAction code provides a code for determining whether the enclosed message is an Update or Delete message (not to be used with the UtilityData message). |
| DataSetID | IDType | DataSetID provides an identifier for a contained data set. |
| Extracted | xs:dateTime | Extracted is a time-stamp from the system rendering the data. |
| ReportingBegin | ObservationalTimePer iodType | ReportingBegin provides the start of the time period covered by the message (in the case of data). |
| ReportingEnd | ObservationalTimePer iodType | ReportingEnd provides the end of the time period covered by the message (in the case of data). |
| EmbargoDate | xs:dateTime | EmbargoDate holds a time period before which the data included in this message is not available. |
| Source | TextType | Source provides human-readable information about the source of the data. |

**GenericMetadataHeaderType:** GenericMetadataHeaderType defines the header format for generic reference metadata.

Derivation:

*BaseHeaderType* (restriction)   
   GenericMetadataHeaderType

Content:

ID, Test, Prepared, Sender, Receiver\*, Name\*, Structure+, MetadataProvider?, DataSetAction?, DataSetID\*, Extracted?, Source\*

Element Documentation:

| **Name** | **Type** | **Documentation** |
| --- | --- | --- |
| ID | IDType | ID identifies an identification for the message, assigned by the sender. |
| Test | xs:boolean | Test indicates whether the message is for test purposes or not. |
| Prepared | HeaderTimeType | Prepared is the date the message was prepared. |
| Sender | SenderType | Sender is information about the party that is transmitting the message. |
| Receiver | PartyType | Receiver is information about the party that is the intended recipient of the message. |
| Name | TextType | Name provides a name for the transmission. Multiple instances allow for parallel language values. |
| Structure | GenericMetadataStruc tureType | Structure provides a reference to the structure (either explicitly or through a structure usage reference) that describes the format of data or reference metadata. In addition to the structure, it is required to also supply the namespace of the structure specific schema that defines the format of the data/metadata. For cross sectional data, additional information is also required to state which dimension is being used at the observation level. This information will allow the structure specific schema to be generated. For generic format messages, this is used to simply reference the underlying structure. It is not mandatory in these cases and the generic data/metadata sets will require this reference explicitly. |
| MetadataProvider | MetadataProviderRefe renceType | MetadataProvider identifies the provider of the metadata for a metadata message. |
| DataSetAction | ActionType | DataSetAction code provides a code for determining whether the enclosed message is an Update or Delete message (not to be used with the UtilityData message). |
| DataSetID | IDType | DataSetID provides an identifier for a contained data set. |
| Extracted | xs:dateTime | Extracted is a time-stamp from the system rendering the data. |
| Source | TextType | Source provides human-readable information about the source of the data. |

**BasicHeaderType:** BasicHeaderType defines the most basic header information used in simple message exchanges.

Derivation:

*BaseHeaderType* (restriction)   
   BasicHeaderType

Content:

ID, Test, Prepared, Sender, Receiver

Element Documentation:

| **Name** | **Type** | **Documentation** |
| --- | --- | --- |
| ID | IDType | ID identifies an identification for the message, assigned by the sender. |
| Test | xs:boolean | Test indicates whether the message is for test purposes or not. |
| Prepared | HeaderTimeType | Prepared is the date the message was prepared. |
| Sender | SenderType | Sender is information about the party that is transmitting the message. |
| Receiver | PartyType | Receiver is information about the party that is the intended recipient of the message. |

**PartyType:** PartyType defines the information which is sent about various parties such as senders and receivers of messages.

Attributes:

id

Content:

Name\*, Contact\*

Attribute Documentation:

| **Name** | **Type** | **Documentation** |
| --- | --- | --- |
| id | IDType | The id attribute holds the identification of the party. |

Element Documentation:

| **Name** | **Type** | **Documentation** |
| --- | --- | --- |
| Name | TextType | Name is a human-readable name of the party. |
| Contact | ContactType | Contact provides contact information for the party in regard to the transmission of the message. |

**SenderType:** SenderType extends the basic party structure to add an optional time zone declaration.

Derivation:

PartyType (extension)   
   SenderType

Attributes:

id

Content:

Name\*, Contact\*, Timezone?

Attribute Documentation:

| **Name** | **Type** | **Documentation** |
| --- | --- | --- |
| id | IDType | The id attribute holds the identification of the party. |

Element Documentation:

| **Name** | **Type** | **Documentation** |
| --- | --- | --- |
| Name | TextType | Name is a human-readable name of the party. |
| Contact | ContactType | Contact provides contact information for the party in regard to the transmission of the message. |
| Timezone | TimezoneType | Timezone specifies the time zone of the sender, and if specified can be applied to all un-time zoned time values in the message. In the absence of this, any dates without time zone are implied to be in an indeterminate "local time". |

**ContactType:** ContactType provides defines the contact information about a party.

Content:

Name\*, Department\*, Role\*, (Telephone | Fax | X400 | URI | Email)\*

Element Documentation:

| **Name** | **Type** | **Documentation** |
| --- | --- | --- |
| Name | TextType | Name contains a human-readable name for the contact. |
| Department | TextType | Department is designation of the organisational structure by a linguistic expression, within which the contact person works. |
| Role | TextType | Role is the responsibility of the contact person with respect to the object for which this person is the contact. |
| Telephone | xs:string | Telephone holds the telephone number for the contact person. |
| Fax | xs:string | Fax holds the fax number for the contact person. |
| X400 | xs:string | X400 holds the X.400 address for the contact person. |
| URI | xs:anyURI | URI holds an information URL for the contact person. |
| Email | xs:string | Email holds the email address for the contact person. |

### Simple Types

**HeaderTimeType:** Provides a union type of xs:date and xs:dateTime for the header fields in the message.

Union of:

xs:dateTime, xs:date.

## Message Footer Namespace

**http://www.sdmx.org/resources/sdmxml/schemas/v3\_0/message/footer**

### Summary

Referenced Namespaces:

| **Namespace** | **Prefix** |
| --- | --- |
| http://www.sdmx.org/resources/sdmxml/schemas/v3\_0/common | common |
| http://www.w3.org/2001/XMLSchema | xs |

Contents:

1 Global Element  
2 Complex Types  
1 Simple Type

### Global Elements

**Footer (FooterType):** Footer is used to communicate information such as error and warnings after the payload of a message.

### Complex Types

**FooterType:** FooterType describes the structure of a message footer. The footer is used to convey any error, information, or warning messages. This is to be used when the message has payload, but also needs to communicate additional information. If an error occurs and no payload is generated, an Error message should be returned.

Content:

Message+

Element Documentation:

| **Name** | **Type** | **Documentation** |
| --- | --- | --- |
| Message | FooterMessageType | Message contains a single error, information, or warning message. A code is provided along with an optional severity. The text of the message can be expressed in multiple languages. |

**FooterMessageType:** FooterMessageType defines the structure of a message that is contained in the footer of a message. It is a status message that have a severity code of Error, Information, or Warning added to it.

Derivation:

StatusMessageType (restriction)   
   CodedStatusMessageType (extension)   
         FooterMessageType

Attributes:

code, severity?

Content:

Text+

Attribute Documentation:

| **Name** | **Type** | **Documentation** |
| --- | --- | --- |
| code | xs:string | The code attribute holds an optional code identifying the underlying error that generated the message. This should be used if parallel language descriptions of the error are supplied, to distinguish which of the multiple error messages are for the same underlying error. |
| severity | SeverityCodeType |  |

Element Documentation:

| **Name** | **Type** | **Documentation** |
| --- | --- | --- |
| Text | TextType | Text contains the text of the message, in parallel language values. |

### Simple Types

**SeverityCodeType:**

Derived by restriction of xs:string .

Enumerations:

|  |  |
| --- | --- |
| **Value** | **Documentation** |
| Error | Error indicates the status message coresponds to an error. |
| Warning | Warning indicates that the status message corresponds to a warning. |
| Information | Information indicates that the status message corresponds to an informational message. |