

Table of contents for: HL7_V3_Meta-Model

Diagrams for: **HL7_V3_Meta-Model**.....3
 Figure1 - **Use Case and Interaction Models**.....3
 Figure 2 - **Information Model**4
 Figure 3 - **Data type and Vocabulary Domain Models**.....5
 Figure 4 - **Message Design Model (left side)**.....6
 Figure 5 - **Message Design Model (right side)**.....7

Identifications:8
 Subject Areas for: **HL7_V3_Meta-Model**.....8

Classes in: **HL7_V3_Meta-Model**.....12
 Class: **Actor**12
 Class: **Alias_name**13
 Class: **Application_role**13
 Class: **Association**.....14
 Class: **Attribute**15
 Class: **Attribute_domain_constraint**.....16
 Class: **Attribute_type**16
 Class: **Choice_branch**17
 Class: **Choice_MET**18
 Class: **Class**.....18
 Class: **Code_subsystem**19
 Class: **Code_system**.....19
 Class: **Coded_term**.....19
 Class: **Collection_MET**20
 Class: **Common_message_element_type**.....21
 Class: **Composite_aggregation**.....21
 Class: **Composite_data_type**21
 Class: **Composite_MET**22
 Class: **Data_type**22
 Class: **Data_type_alias**.....23
 Class: **Data_type_category**.....23
 Class: **Data_type_component**.....24

Class: **Data_type_generalization**.....24
 Class: **Derivative_domain**.....25
 Class: **Domain_version**25
 Class: **Enumerated_domain**26
 Class: **Generalization_relationship**.....26
 Class: **Generic_type_parameter**26
 Class: **Hierarchical_message_description**27
 Class: **HL7_committee**28
 Class: **HMD_attribute_row**.....28
 Class: **HMD_class_row**28
 Class: **HMD_domain_constraint**.....28
 Class: **HMD_notation**.....29
 Class: **HMD_other_row**29
 Class: **HMD_relationship_row**.....29
 Class: **HMD_row**29
 Class: **Interaction**30
 Class: **Interaction_model_category**.....31
 Class: **Interaction_sequence**31
 Class: **LOINC_link**.....32
 Class: **Message_element_type**32
 Class: **Message_information_model**.....33
 Class: **Message_MET**34
 Class: **Message_row_control**.....34
 Class: **Message_type**.....35
 Class: **MET_component**36
 Class: **MIM_aggregation**.....36
 Class: **MIM_association**37
 Class: **MIM_attribute**.....37
 Class: **MIM_attribute_domain_constraint**38
 Class: **MIM_class**38
 Class: **MIM_generalization**.....38
 Class: **MIM_relationship**38
 Class: **MIM_state**39
 Class: **Model**39
 Class: **Primitive_data_type**40

Class: Primitive_MET	41
Class: Project	41
Class: Refined_message_information_model	41
Class: RMIM_attribute_domain_constraint	42
Class: RMIM_attribute_row	42
Class: RMIM_class_row	42
Class: RMIM_notation	43
Class: RMIM_note	43
Class: RMIM_other_row	43
Class: RMIM_relationship_row	44
Class: RMIM_row	44
Class: RMIM_state_row	46
Class: State	46
Class: State_transition	47
Class: Storyboard	47
Class: Storyboard_example	48
Class: Subject_area	48
Class: Subject_class	49
Class: Trigger_event	50
Class: Union_message_type	50
Class: Use_case	51
Class: Use_case_category	52
Class: Use_case_relationship	52
Class: Use_case_sequence	53

Class: V23_data_type	53
Class: V23_field_segment	53
Class: V23_fields	54
Class: V23_segments	54
Class: Version_MET	54
Class: Vocabulary_domain	55

Data type definitions in: HL7_V3_Meta-Model	56
Data type: Boolean : Boolean	56
Data type: CodedElement : CodedElement	56
Data type: CompoundHx : CompoundHx	56
Data type: Date : Date	57
Data type: DateTime : DateTime	57
Data type: DescriptiveText : DescriptiveText	57
Data type: Enumerated : Enumerated	57
Data type: IdentifierString : IdentifierString	57
Data type: Integer : Integer	58
Data type: MultiplicityString : MultiplicityString	58
Data type: NameString : NameString	58
Data type: String : String	58
Data type: VersionNumber : VersionNumber	58
Data type categories for: HL7_V3_Meta-Model	59

Figure 2 - Information Model

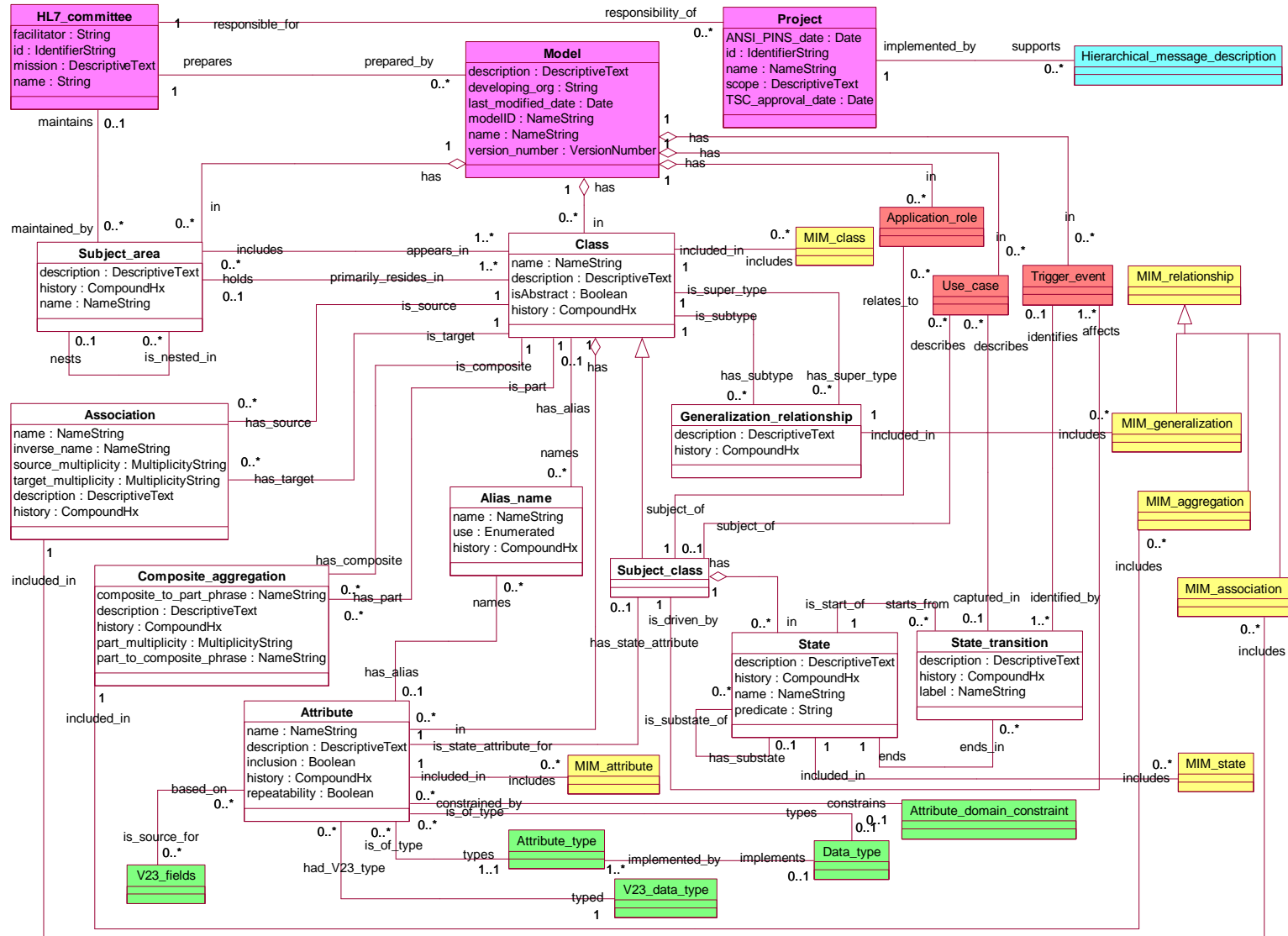


Figure 3 - Data type and Vocabulary Domain Models

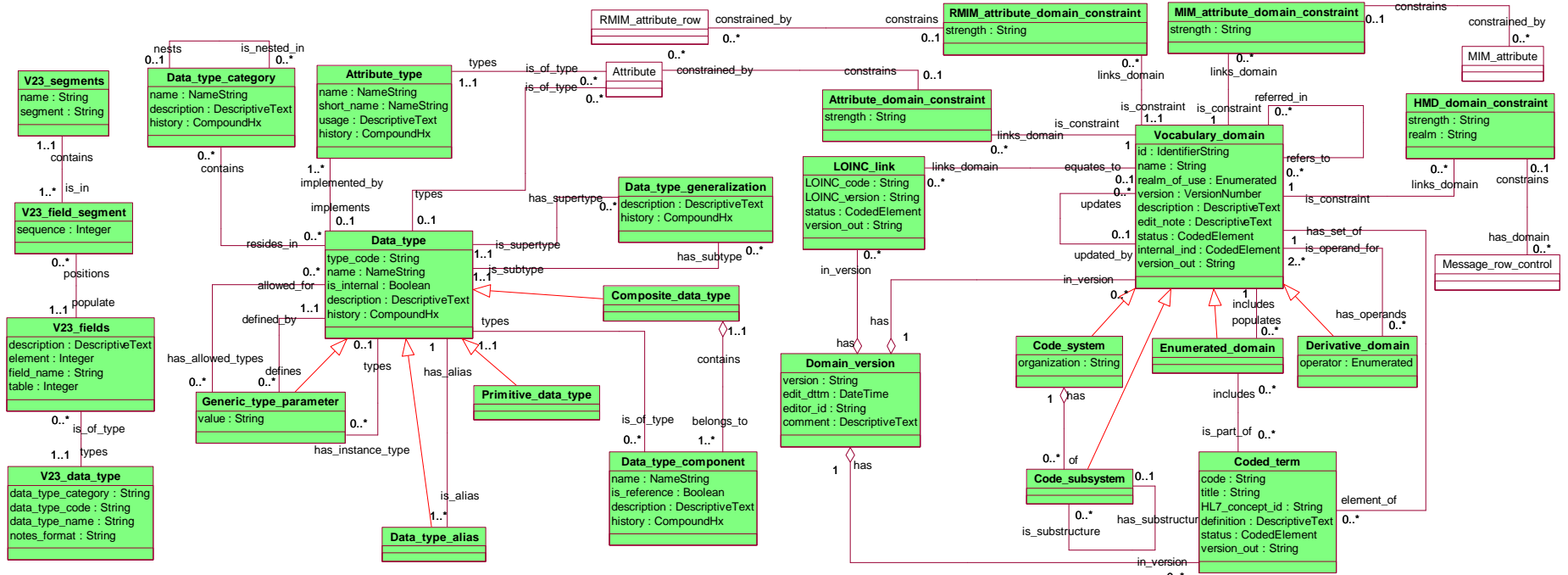


Figure 4 - Message Design Model (left side)

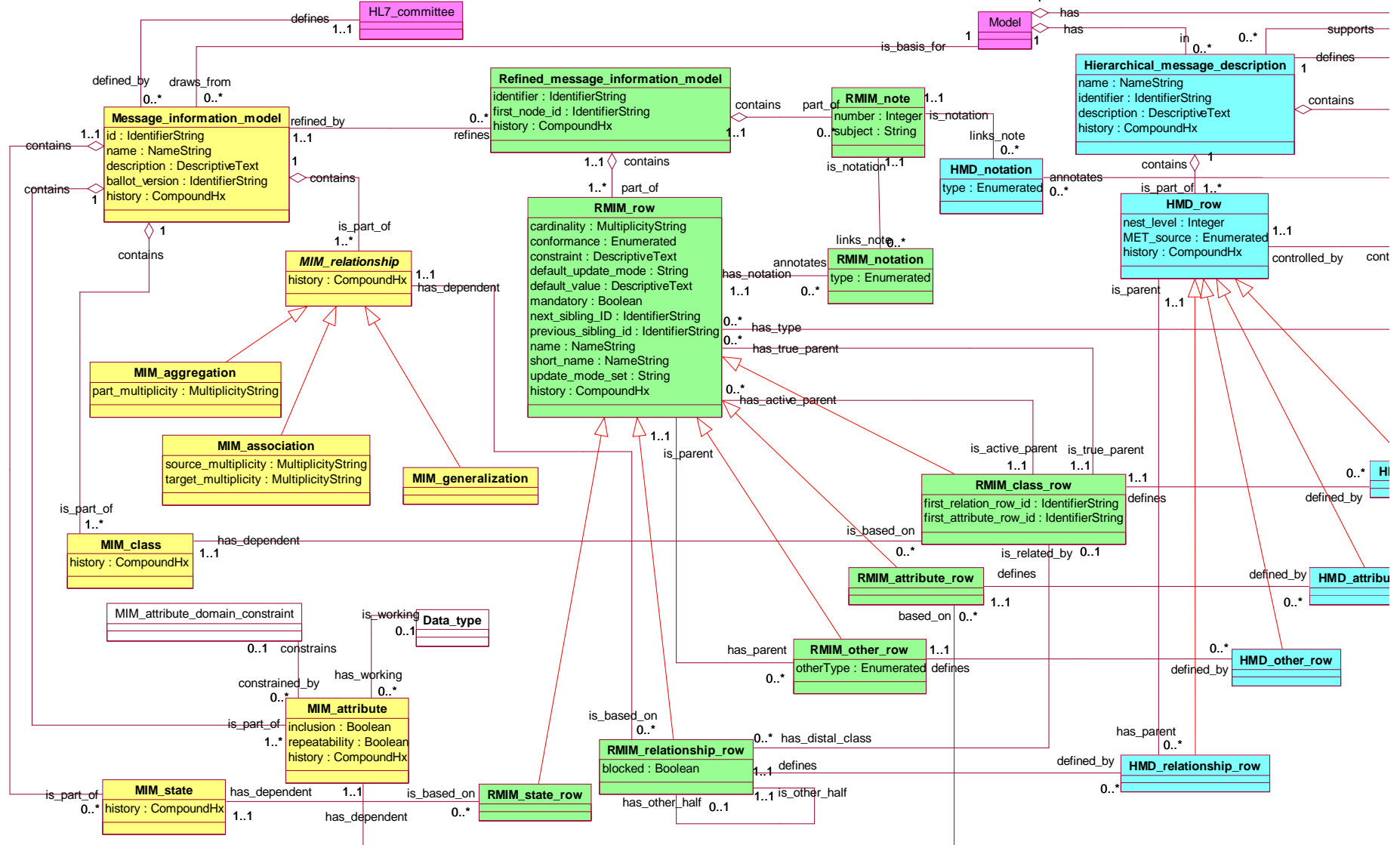
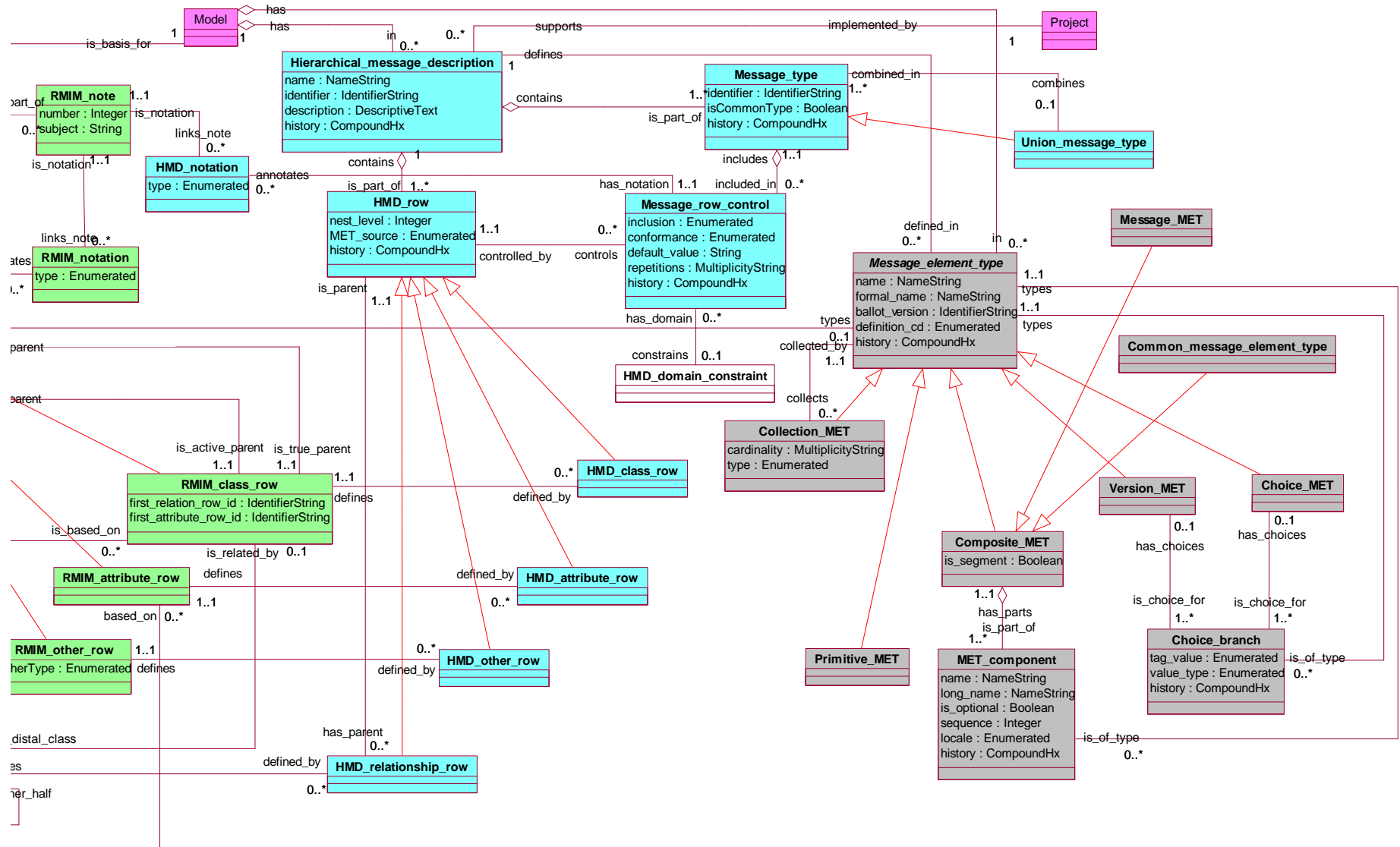


Figure 5 - Message Design Model (right side)



Model: **HL7_V3_Meta-Model**

This model is Copyright by HL7

Identifications:

Organization: HL7

Version: V 1-13 19991020

ModelID: MET_0113

Developed by: Modeling and Methodology

Description of: **HL7 V3 Meta-Model**

This model represents the meta-model of the third release of the HL7 Version 3 MDF, as updated in October 1999.

It includes the datatype and domain model, extensions to the information, interaction and use case models, and the results of two complete revisions of the Message Design Model. The latter includes the message element type model, a model of the message information model, a model of the refined message information model (R-MIM) and the relationships between these artifacts and the elements of the HMD.

This version is felt to be complete, it matches the current HL7 tooling, and is mostly correct, but probably needs some revision after the message design concepts have been tested.

Subject Areas for: **HL7 V3 Meta-Model**

Subject Area: **MET Data type model**

The data type model defines the structure of the data types that may be assigned to information model attributes when these attributes are included in messages. It expresses the hierarchical relationship between data types and their components. It defines the role for attribute types in the information model. It also includes the structure of HL7 Version 2.x fields and data types.

Contains classes:

Attribute
Attribute_type
Composite_data_type
Data_type
Data_type_alias
Data_type_category
Data_type_component
Data_type_generalization
Generic_type_parameter
HL7_committee
Model
Primitive_data_type
V23_data_type
V23_field_segment
V23_fields
V23_segments

Subject Area: **MET Hierarchical message description**

The Hierarchical message description model specifies the semantic links between elements of a MIM, the message object diagram (MOD), the abstract message definition for a set of message structures, and the message structures themselves.

Contains classes:

Hierarchical_message_description
HMD_attribute_row
HMD_class_row
HMD_domain_constraint
HMD_notation
HMD_other_row
HMD_relationship_row
HMD_row
Message_element_type
Message_row_control
Message_type
Model
RMIM_attribute_row
RMIM_class_row
RMIM_note

RMIM_other_row
RMIM_relationship_row
Union_message_type

Subject Area: MET Information model

The information model defines the content of messages exchanged with HL7. Classes, connections, attributes, and states are the primary building blocks of the information model. Classes provide abstractions of the objects represented by the model. The semantic relationships between classes are expressed using connections. The three types of connections are Generalization-specialization, Whole-part, and Instance. Attributes are the facts applicable to the objects of the class, and states capture changes that trigger events have upon the subject classes of the information model.

Contains classes:

- Alias_name**
- Association**
- Attribute**
- Attribute_domain_constraint**
- Attribute_type**
- Class**
- Composite_aggregation**
- Data_type**
- Generalization_relationship**
- HL7_committee**
- State**
- State_transition**
- Subject_area**
- Subject_class**
- V23_data_type**
- V23_fields**

Subject Area: MET Interaction model

The interaction model specifies the information flows that are needed to support the use cases defined in the use case model.

It includes the information flows or interactions, the trigger events, the application roles that send and receive the interactions and scenarios that provide an interaction trace for a series of events.

Contains classes:

- Application_role**
- HL7_committee**
- Interaction**
- Interaction_model_category**
- Interaction_sequence**
- Storyboard**
- Storyboard_example**
- Trigger_event**
- Use_case**
- Use_case_sequence**

Subject Area: MET Message element types

The message element type model expresses the semantic relationship between the meta-model elements that define the type structure used to represent HL7 information structures for communications.

Contains classes:

- Choice_branch**
- Choice_MET**
- Collection_MET**
- Common_message_element_type**
- Composite_MET**
- Hierarchical_message_description**
- Message_element_type**
- Message_MET**
- MET_component**
- Model**
- Primitive_MET**
- Version_MET**

Subject Area: MET Message information model

The message information model subject area is that sub-set of the message specification model that includes all of the elements that make up a MIM.

Contains classes:

- Association**
- Attribute**
- Class**
- Composite_aggregation**
- Data_type**

Generalization_relationship
Hierarchical_message_description
Message_information_model
MIM_aggregation
MIM_association
MIM_attribute
MIM_attribute_domain_constraint
MIM_class
MIM_generalization
MIM_relationship
MIM_state
State

Subject Area: MET Message specification model

The message specification model maps the information content of the information model into the abstract and concrete message specifications needed to communicate between computer applications.

It includes: the message information model which is the sub-set of the information model needed to support a set of messages; the hierarchical message description that maps the information content of the MIM into a set of message formats; and the message element type model which describes the type structure used to convey messages.

Contains classes:

Choice_branch
Choice_MET
Collection_MET
Common_message_element_type
Composite_MET
Data_type
Hierarchical_message_description
HL7_committee
HMD_attribute_row
HMD_class_row
HMD_domain_constraint
HMD_notation
HMD_other_row
HMD_relationship_row

HMD_row
Message_element_type
Message_information_model
Message_MET
Message_row_control
Message_type
MET_component
MIM_aggregation
MIM_association
MIM_attribute
MIM_attribute_domain_constraint
MIM_class
MIM_generalization
MIM_relationship
MIM_state
Model
Primitive_MET
Project
Refined_message_information_model
RMIM_attribute_row
RMIM_class_row
RMIM_notation
RMIM_note
RMIM_other_row
RMIM_relationship_row
RMIM_row
RMIM_state_row
Union_message_type
Version_MET

Subject Area: MET Model identification and scope

The components that define the overall model, the project and the domain information models that support the project.

Contains classes:

HL7_committee
Model
Project

Subject Area: MET Refined message information model

Contains classes:

- Message_element_type**
- Message_information_model**
- MIM_attribute**
- MIM_class**
- MIM_relationship**
- MIM_state**
- Model**
- Refined_message_information_model**
- RMIM_attribute_row**
- RMIM_class_row**
- RMIM_notation**
- RMIM_note**
- RMIM_other_row**
- RMIM_relationship_row**
- RMIM_row**
- RMIM_state_row**

Subject Area: MET Use case model

The use case model is a collection of actors, use cases and scenarios that comprise a high level functional analysis of healthcare. For HL7, the purpose of this analysis is to the requirements for messaging between computer applications . The Use Case Model documents the institutional, medical, and business practices that the message(s) being created will support.

Contains classes:

- Actor**
- HL7_committee**
- Use_case**
- Use_case_category**
- Use_case_relationship**

Subject Area: MET Vocabulary domain model

Defines the structure and relationships for vocabulary domains that are used to constrain coded information model attributes in the information and message design models.

Contains classes:

- Attribute**
- Attribute_domain_constraint**
- Code_subsystem**
- Code_system**
- Coded_term**
- Derivative_domain**
- Domain_version**
- Enumerated_domain**
- HMD_domain_constraint**
- LOINC_link**
- MIM_attribute**
- MIM_attribute_domain_constraint**
- RMIM_attribute_domain_constraint**
- RMIM_attribute_row**
- Vocabulary_domain**

Subject Area: Meta 1 Use case and Interaction models

Defined for graphing purposes only.

Contains classes:

- Actor**
- Application_role**
- Class**
- Interaction**
- Interaction_model_category**
- Interaction_sequence**
- Message_type**
- Model**
- State_transition**
- Storyboard**
- Storyboard_example**
- Subject_class**
- Trigger_event**
- Use_case**
- Use_case_category**
- Use_case_relationship**
- Use_case_sequence**

Subject Area: Meta 2 Information model

Defined for graphing purposes only.

Contains classes:

- Alias_name
- Application_role
- Association
- Attribute
- Attribute_domain_constraint
- Attribute_type
- Class
- Composite_aggregation
- Data_type
- Generalization_relationship
- Hierarchical_message_description
- HL7_committee
- MIM_aggregation
- MIM_association
- MIM_attribute
- MIM_class
- MIM_generalization
- MIM_relationship
- MIM_state
- Model
- Project
- State
- State_transition
- Subject_area
- Subject_class
- Trigger_event
- Use_case
- V23_data_type
- V23_fields

Subject Area: Meta 3 Data type and Domain models

Contains classes:

- Attribute
- Attribute_domain_constraint
- Attribute_type

- Code_subsystem
- Code_system
- Coded_term
- Composite_data_type
- Data_type
- Data_type_alias
- Data_type_category
- Data_type_component
- Data_type_generalization
- Derivative_domain
- Domain_version
- Enumerated_domain
- Generic_type_parameter
- HMD_domain_constraint
- LOINC_link
- MIM_attribute
- MIM_attribute_domain_constraint
- Primitive_data_type
- V23_data_type
- V23_field_segment
- V23_fields
- V23_segments
- Vocabulary_domain

Information model in: **HL7_V3_Meta-Model**

Classes in: **HL7_V3_Meta-Model**

Class: Actor

Is part of: **Model**

Associated with: **Use_case**
Use_case_category

Description of: Actor

An actor is a role played by someone or something that interacts directly with the elements represented by the classes in the information model. With one exception, actors cannot represent information systems. The exception is a special actor with the literal name "some information system". The name is chosen to reinforce the notion that use cases are not built on a priori knowledge of the functionality of specific healthcare information systems.

Composition for: Actor

in (1,1) :: Model :: has (0,n)

The relationship between actors and the models of which they are a part.

Associations for: Actor

participates_in (0,n) :: Use_case :: involves (1,n)

included_in (0,n) :: Use_case_category :: includes (0,n)

Attributes of: Actor

description : DescriptiveText

A short informative statement that allows people to determine, with certainty, whether a particular real world role is an instance of the actor.

history : CompoundHx

This is a compound data type that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

name : NameString

The actors in the model are each given a unique name. The actor name is a singular noun or noun phrase.

Class: Alias_name

Associated with: **Attribute**
 Class

Description of: Alias_name

Classes and attributes in the information model may have one or more alias names. These allow for special uses in HL7, such as short names, and for translations to other languages.

Associations for: Alias_name

names (0,1) :: Attribute :: has_alias (0,n)

names (0,1) :: Class :: has_alias (0,n)

Attributes of: Alias_name

history : CompoundHx

This is a compound data type that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

name : NameString

The alias name for the linked element.

use : Enumerated

A code indicating the use of this alias.

Class: Application_role

Is part of: **Model**

Associated with: **Interaction**
 Interaction
 Interaction_model_category
 Subject_class

Description of: Application_role

Application roles are abstractions that name roles that may be played by health-care information system components when sending or receiving HL7 messages. Thus they are a defined set of responsibilities with respect to interactions. The role may have responsibility to send or receive one or more interactions. The application role and its responsibilities may form the basis for establishing conformance specifications for a standard.

Application roles should be stereotyped with respect to their responsibilities for information about a subject class. The technical committee should start its thinking about application roles from the perspective that there are three fundamental purposes for message exchange, three basic "messaging modes". These are:

Declarative - This includes messages that are sent with the intent of conveying information from one party to another. For example, a healthcare provider might send a message every time a person is registered as a patient with that provider.

Imperative - This includes messages that direct or request a party to do something. For example, a healthcare provider might send a message to a laboratory every time the provider needs the laboratory to perform a test. Note, that even though the message must include information about the test and the patient the test is for, the primary purpose of the message is to request that the test be done.

Interrogative - This includes messages that ask for information, that ask a question. For example, a healthcare provider might send a message to an MPI mediator asking whether the MPI mediator has information about a specific person.

Application roles will have stereotyped names constructed as <subject class> <relationship>. These stereotypes are specific to the messaging modes.

- For the declarative mode, the typical roles are "declarer" and "recipient"
- For the imperative model, the typical roles are "placer" and "filler"
- For the interrogative mode, the typical roles are "questioner" and "answerer"

There is no requirement that a Technical Committee create all of these "<subject class><relationship>" roles. Nor is it limited to these possibilities. But it is expected that the committee will consider these stereotypes.

Composition for: **Application_role**

in (1,1) :: Model :: has (0,n)

The relationship between application roles and the models of which they are a part.

Associations for: **Application_role**

receives (0,n) :: Interaction :: received_by (1,1)

A reference to the application role that is responsible for receiving the message involved in this interaction. The receiving role must be prepared to accept the message and to fulfill the receiver responsibility.

sends (0,n) :: Interaction :: sent_by (1,1)

The sending role has responsibilities to recognize the trigger event for the interaction and to cause the appropriate message to be sent.

included_in (0,n) :: Interaction_model_category :: includes (0,n)

relates_to (1,1) :: Subject_class :: subject_of (0,n)

Links each Application role to the Subject class for which it plays a role. The nature of this relationship is stereotyped as discussed in the description for the Application_role.

Attributes of: **Application_role**

description : DescriptiveText

The text that describes the application role and summarizes the interaction responsibilities that are part of that role.

history : CompoundHx

This is a compound data type that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

identifier : IdentifierString

An identifier assigned to the application role. The identifier is unique within the scope of the model in which the application role is defined. In HL7, committees manage the unique identifiers for their application roles, and concatenate the committee identifier as "Cnn_<identifier>."

name : NameString

A name assigned to the application role. The name is unique within the scope of the model in which the application role is defined.

Class: **Association**

Associated with: **Class**
 Class
 MIM_association

Description of: **Association**

An association between classes depicts the occurrence of a reference attribute used to connect class instances (objects). The associated objects can be of the same or different classes. When the association is defined one of the two classes is designated the "source class" and the other the "target" class. These designations are necessary to unambiguously define an association, but the designations have no semantic implications about the roles of the associated classes within the

information model being defined. The selection of which class to label as "source" is arbitrary.

Associations for: **Association**

has_source (1,1) :: Class :: is_source (0,n)

A reference to the class from which the association perspective is captured.

has_target (1,1) :: Class :: is_target (0,n)

A reference to the class that is the target of the association.

included_in (0,n) :: MIM_association :: includes (1,1)

Attributes of: **Association**

description : DescriptiveText

A short informative statement that describes the relationship between the classes connected by the association.

history : CompoundHx

This is a compound data type that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

Note, the version data must fit within the range of the applicable versions for both classes to which this element is attached.

inverse_name : NameString

A short action phrase that specifies the role of the destination class in the association. Each association between the same pair of classes must have a unique inverse name.

name : NameString

A short action phrase that specifies the role of the source class in the association. Each association between the same pair of classes must have a unique name.

source_multiplicity : MultiplicityString

A set of values and value ranges indicating the number of source class instances involved in the connection. In value ranges the minimum shall be zero or more and the maximum shall be equal to or greater than the minimum. The maximum number may be expressed as unlimited.

target_multiplicity : MultiplicityString

A set of values and value ranges indicating the number of destination class instances involved in the connection. In value ranges the minimum shall be zero or more and the maximum shall be equal to or greater than the minimum. The maximum number may be expressed as unlimited.

Class: **Attribute**

Is part of: **Class**

Associated with: **Alias_name**
Attribute_domain_constraint
Attribute_type
Data_type
MIM_attribute
Subject_class
V23_data_type
V23_fields
V23_segments

Description of: **Attribute**

Attributes in the information model are the major source of the data content used in HL7 communications. Attributes are abstractions of the data captured about classes. Attributes capture separate aspects of the class and take their values independent of one another. Attribute domain specifications are captured in datatypes.

Composition for: **Attribute**

in (1,1) :: Class :: has (0,n)

The relationship between attributes and the classes of which they are a part.

Associations for: **Attribute**

has_alias (0,n) :: Alias_name :: names (0,1)

constrained_by (0,1) :: Attribute_domain_constraint :: constrains (0,n)

is_of_type (1,1) :: Attribute_type :: types (0,n)

A reference between an attribute and its attribute type. The attribute type code must also be the terminal component of the attribute name.

is_of_type (0,1) :: Data_type :: types (0,n)

A link between an attribute and the datatype that has been assigned to it. An attribute is assigned a datatype the first time that it is used in an HMD, and retains that type thereafter.

included_in (0,n) :: MIM_attribute :: includes (1,1)

is_state_attribute_for (0,1) :: Subject_class :: has_state_attribute (1,1)

The state attribute of a class contains a value indicating the current state of the class. In the event that the class has concurrent states, the attribute must be a set of state values.

had_V23_type (1,1) :: V23_data_type :: typed (0,n)

Provides an indication of the data type used in Version 2.x for a particular attribute, if such prior usage has been identified.

based_on (0,n) :: V23_fields :: is_source_for (0,n)

Provides a linkage for an information model attribute to its equivalent version 2.x field, if such linkage exists and has been identified.

stems_from (0,n) :: V23_segments :: source_of (0,n)

Many attributes are traced to equivalent content in HL7 Version 2.x. This connection is secondary to the path that traces an attribute to an HL7 field to a segment. It is provided for modelers who wish to specify particular segments for information model attributes.

Attributes of: **Attribute**

description : DescriptiveText

A short informative description of the Class characteristic captured by the Attribute.

history : CompoundHx

This is a compound data type that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

Note, the version data must fit within the range of the applicable versions for the class of which this element is a part.

inclusion : Boolean

An indication of whether the inclusion of the attribute is mandatory in all HL7 HMDs and messages. Setting the inclusion to mandatory in the information model is deprecated.

name : NameString

Singular nouns are used for attribute names. Attribute names are unique within the class they describe and within the set of attributes inherited by the class they describe. The terminal element of the name shall indicate the attribute type for the attribute.

repeatability : Boolean

Indicates whether this attribute may repeat when it is included in the message structure of a hierarchical message description. The default is false, non-repeating.

Class: **Attribute domain constraint**

Associated with: **Attribute**
Vocabulary_domain

Description of: **Attribute domain constraint**

Constrains a coded attribute to a particular vocabulary domain.

For any class, the special attribute status_cd has as its domain all of the states of the class.

Associations for: **Attribute domain constraint**

constrains (0,n) :: Attribute :: constrained_by (0,1)

links_domain (1,1) :: Vocabulary_domain :: is_constraint (0,n)

Attributes of: **Attribute domain constraint**

strength : String

The strength of the constraint is either CWE (coded with exceptions) or CNE (coded, no exceptions). If no value is given, CWE is the default.

Class: **Attribute_type**

Associated with: **Attribute**
Data_type

Description of: **Attribute_type**

An indication of the form of the attribute, and of its usage. The use of attribute type words in attribute names aids in creating uniformity in the names, helps to avoid unintentional redundancy, and adds clarity to the model.

Associations for: **Attribute_type**

types (0,n) :: Attribute :: is_of_type (1,1)

A reference between an attribute and its attribute type. The attribute type code must also be the terminal component of the attribute name.

implemented_by (0,1) :: Data_type :: implements (1,n)

Each Attribute type may be implemented by one or more data types.

Attributes of: **Attribute_type**

history : CompoundHx

This is a compound data type that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

name : NameString

The full name of the attribute type.

short_name : NameString

The representation of the attribute type that is appended to the name of the attribute to indicate the attribute type.

usage : DescriptiveText

A brief description of the intended usage of this attribute type.

Class: **Choice_branch**

Associated with: **Choice_MET**
 Message_element_type
 Version_MET

Description of: **Choice_branch**

A choice branch provides one alternative for a choice MET or a version MET. The branch includes a tag used to identify the branch when it is instantiated and a type that represents the branch.

Associations for: **Choice_branch**

is_choice_for (0,1) :: Choice_MET :: has_choices (1,n)

Each branch of a Choice_MET includes a type. One of these choices is used in a Choice MEI. A choice branch must be part of a version MET or a choice MET, but not a part of both.

is_of_type (1,1) :: Message_element_type :: types (0,n)

Each choice branch must be typed by an MET.

is_choice_for (0,1) :: Version_MET :: has_choices (1,n)

Each branch of a Version_MET includes a type. All of these choices are used in a Choice MEI. A choice branch must be part of a version MET or a choice MET, but not a part of both.

Attributes of: **Choice_branch**

history : CompoundHx

This is a compound data type that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

tag_value : Enumerated

The value that identifies a particular choice. The tag values must be unique within a single choice MET or version MET.

The branch must have a unique tag value whether or not that value governs the selection of the choice in an MEI. Whether this is the governing value for a particular HMD is determined by the value_type attribute. If that attribute has a value of "Other" then this value rules. If the value_type is "State" then a set of "states" will govern the choice. The state values are linked to the choice branch by the mandatory association to a choice node which contains the states as choice values.

value_type : Enumerated

The value type determines whether this choice branch is governed by the tag value, or whether it is governed by a set of choice values linked to the (mandatory) associated choice node, which values are determined by associations to states in the MIM.

The presently allowed values for this element are "State" or "Other" where "Other" draws the value from tag value.

Class: Choice_MET

Subtype of: **Message_element_type**

Associated with: **Choice_branch**

Description of: Choice_MET

A composite message element type for which only one of the branches will be sent in an instance.

Associations for: Choice_MET

has_choices (1,n) :: Choice_branch :: is_choice_for (0,1)

Each branch of a Choice_MET includes a type. One of these choices is used in a Choice MEI. A choice branch must be part of a version MET or a choice MET, but not a part of both.

Class: Class

Supertype of: **Subject_class**

Is part of: **Model**

Composite of: **Attribute**

Associated with: **Alias_name**
Association
Association
Composite_aggregation
Composite_aggregation
Generalization_relationship
Generalization_relationship
MIM_class
Subject_area
Subject_area

Description of: Class

An abstraction of a set of real-world things (objects) such that all of the objects have the same characteristics and all instances are subject to and conform to the same rules. Classes are the people, places, roles, things, and events about which information is kept.

Composition for: Class

has (0,n) :: Attribute :: in (1,1)

The relationship between attributes and the classes of which they are a part.

in (1,1) :: Model :: has (0,n)

The relationship between classes and the models of which they are a part.

Associations for: Class

has_alias (0,n) :: Alias_name :: names (0,1)

is_source (0,n) :: Association :: has_source (1,1)

A reference to the class from which the association perspective is captured.

is_target (0,n) :: Association :: has_target (1,1)

A reference to the class that is the target of the association.

is_composite (0,n) :: Composite_aggregation :: has_composite (1,1)

A reference to the class that is the composition the relationship.

is_part (0,n) :: Composite_aggregation :: has_part (1,1)

A reference to the class that is the part class in the relationship.

is_subtype (0,n) :: Generalization_relationship :: has_subtype (1,1)

The linkage between a generalization relationship and the subtype that participates in that connection.

is_super_type (0,n) :: Generalization_relationship :: has_super_type (1,1)

The linkage between a generalization relationship and the supertype that participates in that connection.

included_in (0,n) :: MIM_class :: includes (1,1)

appears_in (0,n) :: Subject_area :: includes (1,n)

The linkage between a Subject area and each of the Classes that are in that Subject area.

primarily_resides_in (0,1) :: Subject_area :: holds (1,n)

The linkage between a Class and the Subject area that is its primary residence. This must be established if a Class resides in more than one Subject area.

Attributes of: Class

description : DescriptiveText

A short informative statement that allows one to tell, with certainty, whether a particular real world thing is an instance of the Class as conceptualized in the information model.

history : CompoundHx

This is a compound data type that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

isAbstract : Boolean

This variable indicates whether or not this Class is an abstract class. An abstract class is a class that can not be instantiated and is customarily the generalization class in a generalization/specialization structure.

name : NameString

The Classes in the information model are given a unique name. The Class name is a singular noun or noun phrase.

Class: Code subsystem

Subtype of: **Vocabulary_domain**

Is part of: **Code_system**

Associated with: **Code_subsystem**
Code_subsystem

Description of: Code subsystem

Many code systems have subsystems. These are, e.g., axes in SNOMED, classes, and subclasses in hierarchical codes such as ICD. The structure of coding systems is idiosyncratic.

Composition for: Code subsystem

of (1,1) :: Code_system :: has (0,n)

Associations for: Code subsystem

has_substructure (0,n) :: Code_subsystem :: is_substructure (0,1)

A code sub-system may be further decomposed in a hierarchical fashion by implementing this association.

is_substructure (0,1) :: Code_subsystem :: has_substructure (0,n)

A code sub-system may be further decomposed in a hierarchical fashion by implementing this association.

Class: Code system

Subtype of: **Vocabulary_domain**

Composite of: **Code_subsystem**

Description of: Code system

A system is a published code system by an organization, that defines it, publishes it, maintains it, and thus guarantees for its usefulness and continuity.

ORGANIZATION: e.g. WHO, College of American Pathologists, ISO, IANA, and HL7 of course.

NAME: e.g. ICD, ICPM, ICPC, SNOMED, 639-1, MIME types, ...

Composition for: Code system

has (0,n) :: Code_subsystem :: of (1,1)

Attributes of: Code system

organization : String

The name of the organization that maintains the code system. Examples are WHO, College of American Pathologists, ISO, IANA, and HL7.

Class: Coded term

Is part of: **Domain_version**

Associated with: **Enumerated_domain**
Vocabulary_domain

Description of: Coded term

This class is not expected to be exhaustively instantiated. Its purpose is not to copy the lists of terms of all external code sets. Rather it will be used for those codes that must be enumerated to define a particular HL7 domain.

It can also be used to capture and manage HL7-defined code sets.

Composition for: **Coded term**

in_version (1,1) :: Domain_version :: has (0,n)

Associations for: **Coded term**

is_part_of (0,n) :: Enumerated_domain :: includes (0,n)

An enumeration is a set of terms.

element_of (1,1) :: Vocabulary_domain :: has_set_of (0,n)

This relationship indicates that a vocabulary domain is a set of terms and every term belongs to one vocabulary domain.

Through subsystems and derived vocabulary domains, any term can be member of more than one vocabulary domain.

Attributes of: **Coded term**

code : String

The text string used within the coding system to identify this concept.

definition : DescriptiveText

A textual representation of the meaning of this entry as it is represented in the coding system from which it came.

Any term may have a definition stored. For the terms that are referenced from external coding systems, the definition will not be included..

These terms may be used to maintain HL7 code tables, in which case, the definition is mandatory.:

A freshly defined HL7 code table would be an Enumeration domain (refers to itself as a "base"). All items in the enumeration would be terms with definition attached to them.

HL7_concept_id : String

the unique item identifier assigned by HL7 to this concept. This concept identifier is globally unique to a concept throughout all HL7 tables. That is, if the concept male occurred in another vocabulary domain in addition to the Gender domain, it would again have an item identifier of "1". If a universal terminology of medicine

becomes available, the universal concept identifier from that terminology could be used in place of this HL7 assigned identifier.

status : CodedElement

The status of this entry within this domain table. The values for Status come from the vocabulary domain EditStatus. Some values for status are Proposed, Rejected, Active, Obsolete, and Inactive.

title : String

A textual name for the term.

version_out : String

The version number of the table at which this entry was deleted. A blank Vout value means that the row continues to exist in the current version of the table.

Class: **Collection MET**

Subtype of: **Message_element_type**

Associated with: **Message_element_type**

Description of: **Collection MET**

Provides for the inclusion of a collection of other METs as a component of an MET.

The collection may be a List (ordered collection of instances), a Set (unordered collection of unique instances) or a Bag (unordered collection of instances which might not be unique).

Associations for: **Collection MET**

collects (1,1) :: Message_element_type :: collected_by (0,n)

Each collection MET collects elements of a single type.

Attributes of: **Collection MET**

cardinality : MultiplicityString

Provides the minimum and maximum number of occurrences in the collection.

type : Enumerated

Determines whether the collection is a List, a Set, or a Bag.

Class: Common message element type

Subtype of: **Composite_MET**

Description of: Common message element type

The common message element type is composite MET that has been declared as a public type available for assignment or substitution, as appropriate in an HMD.

OpenIssue: This class may require additional attributes.

Class: Composite aggregation

Associated with: **Class**
Class
MIM_aggregation

Description of: Composite aggregation

An association between classes that depicts the relationship between a composite aggregate class and its component parts.

Associations for: Composite aggregation

has_composite (1,1) :: Class :: is_composite (0,n)

A reference to the class that is the composition the relationship.

has_part (1,1) :: Class :: is_part (0,n)

A reference to the class that is the part class in the relationship.

included_in (0,n) :: MIM_aggregation :: includes (1,1)

Attributes of: Composite aggregation

composite_to_part_phrase : NameString

A short phrase representing the nature of the relationship from the perspective of the composite class. Example phrases are: "includes", "contains", "consists of", "has as parts". Other phrases may also be used. If no phrase is specified, the phrase "contains" will be assumed.

description : DescriptiveText

A short informative description of the composite aggregation connection.

history : CompoundHx

This is a compound data type that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

Note, the version data must fit within the range of the applicable versions for both classes to which this element is attached.

part_multiplicity : MultiplicityString

A set of values and value ranges indicating the number of part class instances involved in the aggregation. In value ranges the minimum shall be zero or more and the maximum shall be equal to or greater than the minimum. The maximum number may be expressed as unlimited.

part_to_composite_phrase : NameString

A short phrase representing the nature of the relationship from the perspective of the part class . Example phrases are: "is included in", "is contained in", "is part of", and "is a component of". Other phrases may also be used. If no phrase is specified, "is part of" will be assumed.

Class: Composite data type

Subtype of: **Data_type**

Composite of: **Data_type_component**

Associated with: **Composite_MET**

Description of: Composite data type

A composite data type consists of one or more named and typed components.

Composition for: Composite data type

contains (1,n) :: Data_type_component :: belongs_to (1,1)

Associations for: Composite data type

specifies (1,1) :: Composite_MET :: specified_by (0,1)

Each composite data type has an associated composite MET that is used to represent it in messages.

Class: Composite MET

Subtype of: **Message_element_type**
Supertype of: **Common_message_element_type**
Message_MET
Composite of: **MET_component**
Associated with: **Composite_data_type**

Description of: Composite MET

A message element type that contains other message elements (components). Each component message element has a name and a type. Each component of an element must have a different name, although many may be of the same type.

Composition for: Composite MET

has_parts (1,n) :: MET_component :: is_part_of (1,1)

Associations for: Composite MET

specified_by (0,1) :: Composite_data_type :: specifies (1,1)

Each composite data type has an associated composite MET that is used to represent it in messages.

Attributes of: Composite MET**is_segment : Boolean**

This attribute is intended to capture the definition of a "segment" MET for use in a segment-positional syntax (ER7).

OpenIssue: This attribute is redundant to the association to a MIM_class. If the association is present, then it is a segment. Otherwise, not.

Class: Data type

Supertype of: **Composite_data_type**
Data_type_alias
Generic_type_parameter
Primitive_data_type
Is part of: **Model**

Associated with: **Attribute**
Attribute_type
Data_type_alias
Data_type_category
Data_type_component
Data_type_generalization
Data_type_generalization
Generic_type_parameter
Generic_type_parameter
Generic_type_parameter
MIM_attribute

Description of: Data type

Datatypes are used to express the type of an attribute or of a data type component. A data type may be composite, primitive, an alias or a generic type parameter.

A generic type parameter contains a parameter that is part of the definition of a generic data type. Each generic type parameter is part of the definition for a single generic type.

A composite data type contains one or more components.

An alias provides an alternative name, alternate type code and additional description for another data type.

A primitive data type is a data type that is defined entirely by its specification. A primitive data type may have generic type parameters.

A generic data type is a type that has one or more generic type parameters. It provides a pattern for instantiating a specific, usually composite, data type.

Composition for: Data type

in (1,1) :: Model :: has (0,n)

The relationship between data types and the models in which they are first defined.

Associations for: Data type

types (0,n) :: Attribute :: is_of_type (0,1)

A link between an attribute and the datatype that has been assigned to it. An attribute is assigned a datatype the first time that it is used in an HMD, and retains that type thereafter.

implements (1,n) :: Attribute_type :: implemented_by (0,1)

Each Attribute type may be implemented by one or more data types.

has_alias (1,n) :: Data_type_alias :: is_alias (1,1)

Provides for one type to represent a simple alias for another.

resides_in (0,n) :: Data_type_category :: contains (0,n)

types (0,n) :: Data_type_component :: is_of_type (1,1)

Each component is linked to a single type either directly or through a generic type parameter.

is_subtype (0,n) :: Data_type_generalization :: has_subtype (1,1)

Each data type generalization includes a single sub-type.

is_supertype (0,n) :: Data_type_generalization :: has_supertype (1,1)

Each data type generalization provides sub-types for a single super-type..

allowed_for (0,n) :: Generic_type_parameter :: has_allowed_types (0,n)

Determines the set of types that a generic type parameter may implement.

defined_by (0,n) :: Generic_type_parameter :: defines (1,1)

Relationship between a Generic Type Parameter and the Generic type for which it is a parameter.

types (0,n) :: Generic_type_parameter :: has_instance_type (0,1)

This relationship defines the particular instantiation type for a generic instance.

is_working (0,n) :: MIM_attribute :: has_working (0,1)

If a MIM attribute is associated with a RIM attribute that does not have an assigned data type, the MIM attribute may be assigned an alternative, working data type. This association must not be instantiated otherwise.

Attributes of: **Data_type**

description : DescriptiveText

A detailed description or specification for the data type. All such descriptions are assumed to also reference a broader specification of data types.

history : CompoundHx

This is a compound data type that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

is_internal : Boolean

A data type may be defined as being "internal". An internal type is used only to define other composite data types. Internal types shall not be used in messages. For example, we define a type Binary that contains pure raw data bits, and that is used only by Multimedia Enabled Free Text.

name : NameString

The formal name for the data type.

type_code : String

The formal code assigned by the Control/Query Committee for this datatype. This is the representation of the data type that appears as the data type for attributes of the information model and data type components in the data type model.

Class: **Data_type_alias**

Subtype of: **Data_type**

Associated with: **Data_type**

Description of: **Data_type_alias**

An alias provides an alternative name, alternate type code and additional description for another data type. Its most common use is to provide a specific "user-friendly" name for a collection of other data types.

Associations for: **Data_type_alias**

is_alias (1,1) :: Data_type :: has_alias (1,n)

Provides for one type to represent a simple alias for another.

Class: **Data_type_category**

Is part of: **Model**

Associated with: **Data_type**
Data_type_category
Data_type_category
HL7_committee

Description of: **Data type category**

A data type category collects data types that represent similar real world concepts, or are represented in a similar fashion.

Composition for: **Data type category**

in (1,1) :: Model :: has (0,n)

The relationship between data type categories and the models of which they are a part.

Associations for: **Data type category**

contains (0,n) :: Data_type :: resides_in (0,n)

is_nested_in (0,1) :: Data_type_category :: nests (0,n)

nests (0,n) :: Data_type_category :: is_nested_in (0,1)

maintained_by (1,1) :: HL7_committee :: maintains (0,n)

Attributes of: **Data type category**

description : DescriptiveText

The description of the data type category expresses the unifying concept that causes a set of data types to be included in this category.

history : CompoundHx

This is a compound data type that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

name : NameString

The name given to the data type category.

Class: **Data type component**

Is part of: **Composite_data_type**

Associated with: **Data_type**

Description of: **Data type component**

A component is an element of a data type that may be valued when the data type is used in HL7 communications. The component takes its type from a data type that is any of - primitive, composite, or generic type parameter.

A component of a composite data type is like a variable, i.e. it has a name and a type. The type can be declared to be included by reference instead of by value. This is useful if you know such a component mentions an instance that is already mentioned elsewhere in the communication. In languages such as Java, where objects are always handled through references this does not make any difference.

Composition for: **Data type component**

belongs_to (1,1) :: Composite_data_type :: contains (1,n)

Associations for: **Data type component**

is_of_type (1,1) :: Data_type :: types (0,n)

Each component is linked to a single type either directly or through a generic type parameter.

Attributes of: **Data type component**

description : DescriptiveText

The description of a component should include its role within the composite of which it is a part and its relationships, if any, to other components of the same composite.

history : CompoundHx

This is a compound data type that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

is_reference : Boolean

The component type can be declared to be included by reference instead of by value. This is useful if you know such a component mentions an instance that is already mentioned elsewhere in the communication. In languages such as Java, where objects are always handled through references this does not make any difference.

name : NameString

The formal name of the data type component.

Class: **Data type generalization**

Associated with: **Data_type**
 Data_type

Description of: Data type generalization

Types can maintain an inheritance relationship with each other. We explicitly allow (and use) "multiple inheritance". However, we do use inheritance as a way to specialize subtypes from general super-types. Rather we go the other way. Abstract generalized types are used to categorize the concrete types in different ways. Thus one can get hold of all types that have a certain property of interest.

For instance, we define the generalized type Quantity to subsume all quantitative types. This is used to define one type Ratio as a ratio of any two quantities.

Similarly, we define a data type Interval that is a continuous subset of any type with an order relation. All types with an order relation are subsumed under OrderedType. Note that not all quantities are ordered (e.g. vectors are not) and there may be non-quantities that have an order relationship (ordinals, e.g. military ranks).

Associations for: Data type generalization

has_subtype (1,1) :: Data_type :: is_subtype (0,n)

Each data type generalization includes a single sub-type.

has_supertype (1,1) :: Data_type :: is_supertype (0,n)

Each data type generalization provides sub-types for a single super-type..

Attributes of: Data type generalization

description : DescriptiveText

A statement of the nature of the generalization relationship.

history : CompoundHx

This is a compound data type that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

Class: Derivative domain

Subtype of: **Vocabulary_domain**

Associated with: **Vocabulary_domain**

Description of: Derivative domain

A derivative is the vocabulary domain that results from applying a particular operator to two or more other vocabulary domains.

A derivative vocabulary domain is usually a subset of another set. Defining derivatives inclusion or exclusion of one domain with another. Derivatives can exist on different levels: e.g., in the realm of an institution (hospital department), a country (USA), a treaty (EU), and so on. Thus, each derivative can be defined on another derivative in a larger realm (e.g., a German hospital narrows the domain defined by the EU that in turn may be a subset of a WHO code).

While system and subsystem are defined externally, derivatives must be explicitly defined.

Associations for: Derivative domain

has_operands (2,n) :: Vocabulary_domain :: is_operand_for (0,n)

Relationship between a derivative, its operator, and the operands (other domains) to which the operator is applied.

Attributes of: Derivative domain

operator : Enumerated

The operator that defines the combinatorial rule applied to two or more operand domains to arrive at the derivative domain. Operators include union (+) and difference (-). Additional constructs may be provided.

Class: Domain version

Composite of: **Coded_term**
 LOINC_link
 Vocabulary_domain

Description of: Domain version

Captures each update of the vocabulary domain tables in a version, including the when the editing took place, who performed it, and comments as to what was done.

Composition for: Domain version

has (0,n) :: Coded_term :: in_version (1,1)

has (0,n) :: LOINC_link :: in_version (1,1)

has (0,n) :: Vocabulary_domain :: in_version (1,1)

Attributes of: **Domain_version**

comment : DescriptiveText

A summary of why the edits were made, and what was done.

edit_dttm : DateTime

The date and time that the edit session began

editor_id : String

An identifier of the person or group responsible for the edits.

version : String

The version number of the edit session. This number is incremented by 1 each time a new edit session takes place. The version number is used as the value of Vin and Vout as appropriate to track which table entries in the vocabulary definition tables were added, modified, or deleted during the session.

Class: **Enumerated_domain**

Subtype of: **Vocabulary_domain**

Associated with: **Coded_term**
Vocabulary_domain

Description of: Enumerated_domain

An enumerated domain is defined by an enumerated set of code terms. While large code systems are impractical to enumerate, and while some are not enumerable at all, enumeration is useful for small domains.

Associations for: Enumerated_domain

includes (0,n) :: Coded_term :: is_part_of (0,n)

An enumeration is a set of terms.

populates (1,1) :: Vocabulary_domain :: includes (0,n)

An enumeration includes terms from a particular domain. In the case where the enumeration is the entire domain, this relationship is self-referential.

Class: **Generalization_relationship**

Associated with: **Class**
Class

MIM_generalization

Description of: Generalization_relationship

Generalization is a relationship between a class and subtypes of the class. A supertype can be associated with more than one subtype. Each of the subtypes associated with a single supertype is mutually exclusive. A subtype may be associated with more than one supertype. The hierarchy or lattice of generalizations is called a generalization relationship. The subtype inherits the attributes, and associations, and of all of its supertypes.

Associations for: Generalization_relationship

has_subtype (1,1) :: Class :: is_subtype (0,n)

The linkage between a generalization relationship and the subtype that participates in that connection.

has_super_type (1,1) :: Class :: is_super_type (0,n)

The linkage between a generalization relationship and the supertype that participates in that connection.

included_in (0,n) :: MIM_generalization :: includes (1,1)

Attributes of: Generalization_relationship

description : DescriptiveText

A short informative description of the Generalization relationship.

history : CompoundHx

This is a compound data type that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

Note, the version data must fit within the range of the applicable versions for both classes to which this element is attached.

Class: **Generic_type_parameter**

Subtype of: **Data_type**

Associated with: **Data_type**
Data_type
Data_type

Description of: **Generic type parameter**

A generic type parameter contains a parameter that is part of the definition of a generic data type. Each generic type parameter is part of the definition for a single generic type.

The most common form of generic type parameter provides an instantiation type, drawn from two or more types.

Other generic type parameters constrain the generic type, such as the collection type and multiplicity for a Collection.

Associations for: **Generic type parameter**

defines (1,1) :: Data_type :: defined_by (0,n)

Relationship between a Generic Type Parameter and the Generic type for which it is a parameter.

has_allowed_types (0,n) :: Data_type :: allowed_for (0,n)

Determines the set of types that a generic type parameter may implement.

has_instance_type (0,1) :: Data_type :: types (0,n)

This relationship defines the particular instantiation type for a generic instance.

Attributes of: **Generic type parameter**

value : String

Establishes the content for a generic type parameter that defines a property of a generic type other than an instantiation type.

Class: **Hierarchical message description**

Is part of: **Model**

Composite of: **HMD_row**
Message_type

Associated with: **Message_element_type**
Project

Description of: **Hierarchical message description**

A structure that completely defines the structure of a set of messages, and reflects the relationship of the elements of these messages to components of the Refined

Message Information Model from which it derives and the Message Element Types that it defines or uses.

Composition for: **Hierarchical message description**

contains (1,n) :: HMD_row :: is_part_of (1,1)

A reference to the HMD that contains each HMD row.

contains (1,n) :: Message_type :: is_part_of (1,1)

Each message structure is contained in a single HMD.

in (1,1) :: Model :: has (0,n)

The relationship between hierarchical message descriptions and the models in which they are first defined.

Associations for: **Hierarchical message description**

defines (0,n) :: Message_element_type :: defined_in (1,1)

Each MET must be defined in one HMD.

supports (1,1) :: Project :: implemented_by (0,n)

A MIM should support a single project.

Attributes of: **Hierarchical message description**

description : DescriptiveText

A short textual description of the messages covered in the HMD..

history : CompoundHx

This is a compound data type that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

identifier : IdentifierString

Arbitrary identifier assigned by Technical Steering Committee. Committees may assign an interim identifier that starts with the committee's identifier, as "Cnn_<identifier>" so long as this composite identifier is unique.

name : NameString

The HMDs in the model are each given a name.

Class: HL7_committee

Associated with: **Data_type_category**
 Interaction_model_category
 Message_information_model
 Model
 Project
 Subject_area
 Use_case_category

Description of: HL7_committee

Unique identifier assigned to each of the Technical Committees and Special Interest Groups of the HL7 Working Group.

Associations for: HL7_committee

maintains (0,n) :: Data_type_category :: maintained_by (1,1)

maintains (0,n) :: Interaction_model_category :: maintained_by (0,1)

defines (0,n) :: Message_information_model :: defined_by (1,1)

prepares (0,n) :: Model :: prepared_by (1,1)

Links a model to the committee that prepared it.

responsible_for (0,n) :: Project :: responsibility_of (1,1)

Establishes the relationship between committees and the projects for which that committee is responsible.

maintains (0,n) :: Subject_area :: maintained_by (0,1)

maintains (0,n) :: Use_case_category :: maintained_by (0,1)

Attributes of: HL7_committee

facilitator : String

Name of the individual who facilitates modeling for this committee.

id : IdentifierString

Assigned committee identifier.

mission : DescriptiveText

The approved mission statement or charter for this committee.

name : String

The name of the Technical Committee or Special Interest Group.

Class: HMD_attribute_row

Subtype of: **HMD_row**

Associated with: **RMIM_attribute_row**

Description of: HMD_attribute_row

An attribute row represents a single attribute in the R-MIM.

Associations for: HMD_attribute_row

defined_by (1,1) :: RMIM_attribute_row :: defines (0,n)

Class: HMD_class_row

Subtype of: **HMD_row**

Associated with: **RMIM_class_row**

Description of: HMD_class_row

There is one class row in an HMD. This row is the root of the HMD.

Associations for: HMD_class_row

defined_by (1,1) :: RMIM_class_row :: defines (0,n)

Class: HMD_domain_constraint

Associated with: **Message_row_control**
 Vocabulary_domain

Description of: HMD_domain_constraint

Constrains a coded HMD attribute row to a particular vocabulary domain. Links each coded attribute in an HMD to the code domain that may be used to value it.

For any class, the special attribute status_cd has as its domain all of the states of the class. In an HMD domain specification, the special domain name '@state' can substitute for the domain name. If held is a valid state, <@state> and <@state - (held)> are valid domain specifications.

Associations for: **HMD_domain_constraint**

constrains (0,n) :: Message_row_control :: has_domain (0,1)

links_domain (1,1) :: Vocabulary_domain :: is_constraint (0,n)

Attributes of: **HMD_domain_constraint**

realm : String

May specify the realm of applicability for this attribute row.

strength : String

The strength of the constraint is either CWE (coded with exceptions) or CNE (coded, no exceptions). If no value is given, CWE is the default.

Class: **HMD_notation**

Associated with: **Message_row_control**
RMIM_note

Associations for: **HMD_notation**

annotates (1,1) :: Message_row_control :: has_notation (0,n)

links_note (1,1) :: RMIM_note :: is_notation (0,n)

Attributes of: **HMD_notation**

type : Enumerated

Indicates the type of the note. Types include:

RV : Required value

CP : Conditional Presence

CN : Constraint

DM : Domain

CT : Comment

Class: **HMD_other_row**

Subtype of: **HMD_row**

Associated with: **RMIM_other_row**

Description of: **HMD_other_row**

Links an 'other' R-MIM row into a message.

Associations for: **HMD_other_row**

defined_by (1,1) :: RMIM_other_row :: defines (0,n)

Class: **HMD_relationship_row**

Subtype of: **HMD_row**

Associated with: **HMD_row**
RMIM_relationship_row

Description of: **HMD_relationship_row**

An relationship row represents a single association, aggregation or inheritance relationship traversed in the R-MIM graph walk.

Associations for: **HMD_relationship_row**

has_parent (1,1) :: HMD_row :: is_parent (0,n)

defined_by (1,1) :: RMIM_relationship_row :: defines (0,n)

Class: **HMD_row**

Supertype of: **HMD_attribute_row**
HMD_class_row
HMD_other_row
HMD_relationship_row

Is part of: **Hierarchical_message_description**

Associated with: **HMD_relationship_row**
Message_row_control

Description of: **HMD_row**

The rows of an HMD.

Composition for: **HMD_row**

is_part_of (1,1) :: Hierarchical_message_description :: contains (1,n)

A reference to the HMD that contains each HMD row.

Associations for: **HMD_row**

is_parent (0,n) :: HMD_relationship_row :: has_parent (1,1)

controlled_by (0,n) :: Message_row_control :: controls (1,1)

Each message row control controls the presence of one unsubsumed HMD row in the message structure of which the message row control is a part.

Attributes of: **HMD_row**

history : CompoundHx

This is a compound data type that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

MET_source : Enumerated

Indicates the source of the MET - reuse a definition from this HMD, CMET or data type.

nest_level : Integer

Indicates the nesting level of the row in the HMD.

Class: **Interaction**

Is part of: **Model**

Associated with: **Application_role**
Application_role
Interaction
Interaction
Interaction_model_category
Interaction_sequence
Message_type
Trigger_event

Description of: **Interaction**

An association between a specific message (information transfer), a particular trigger event that initiates or triggers the interaction, and the roles that send and receive the interaction. An interaction is a single, one-way transfer of information. Within itself, an interaction may not specify a return message. An interaction may, however, establish a responsibility for the receiver of its message, and this responsibility may require that the receiver initiate a particular trigger

event/interaction subsequent to the receipt. That follow-on interaction may have the effect of continuing or completing a transaction that requires two or more linked message exchanges.

Composition for: **Interaction**

in (1,1) :: Model :: has (0,n)

The relationship between interactions and the models of which they are a part.

Associations for: **Interaction**

received_by (1,1) :: Application_role :: receives (0,n)

A reference to the application role that is responsible for receiving the message involved in this interaction. The receiving role must be prepared to accept the message and to fulfill the receiver responsibility.

sent_by (1,1) :: Application_role :: sends (0,n)

The sending role has responsibilities to recognize the trigger event for the interaction and to cause the appropriate message to be sent.

initiated_by_receiver (0,1) :: Interaction :: responsible_for (0,n)

A reference to an interaction that the receiver of the message must initiate once receipt of the message is acknowledged. This is an optional element in that there may no follow-on responsibility. Transactions can be established through a chain of receiver responsibilities for individual interactions.

responsible_for (0,n) :: Interaction :: initiated_by_receiver (0,1)

A reference to an interaction that the receiver of the message must initiate once receipt of the message is acknowledged. This is an optional element in that there may no follow-on responsibility. Transactions can be established through a chain of receiver responsibilities for individual interactions.

included_in (0,n) :: Interaction_model_category :: includes (0,n)

is_linked_by (0,n) :: Interaction_sequence :: links (1,1)

transfers (1,1) :: Message_type :: transferred_by (1,n)

Each interaction shall include a link to a single message structure that the interaction will transfer.

initiated_by (1,1) :: Trigger_event :: initiates (0,n)

A reference to the trigger event that triggers or initiates this interaction.

Attributes of: **Interaction**

description : DescriptiveText

Provides a description of the data content of the interaction, usually expressed in terms of the class instances that are expected to be part of the message sent by the interaction.

history : CompoundHx

This is a compound data type that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

identifier : IdentifierString

An identifier assigned to the interaction. The identifier should be unique within the scope of the model in which the interaction is defined. In HL7, committees manage the unique identifiers for their interactions, and concatenate the committee identifier as "Cnn_<identifier>."

Class: **Interaction_model_category**

Is part of: **Model**

Associated with: **Application_role**
HL7_committee
Interaction
Interaction_model_category
Interaction_model_category

Description of: **Interaction_model_category**

A major category of information represented in the interaction model. An aggregation of interrelated interactions and application roles. A category allows portions of a large model to be viewed as a whole thereby eliminating some complexity involved in understanding a large model.

Composition for: **Interaction_model_category**

in (1,1) :: Model :: has (0,n)

The relationship between interaction model categories and the models of which they are a part.

Associations for: **Interaction_model_category**

includes (0,n) :: Application_role :: included_in (0,n)

maintained_by (0,1) :: HL7_committee :: maintains (0,n)

includes (0,n) :: Interaction :: included_in (0,n)

nested_in (0,1) :: Interaction_model_category :: nests (0,n)

Interaction model categories may be nested.

nests (0,n) :: Interaction_model_category :: nested_in (0,1)

Interaction model categories may be nested.

Attributes of: **Interaction_model_category**

description : DescriptiveText

Short informative text describing the interaction model category so as to be clear what type of interactions and application roles it includes.

history : CompoundHx

This is a compound data type that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

name : NameString

The name given to the interaction model category. The identifier for the committee defining this category is prepended to the name as Cnn.

Class: **Interaction_sequence**

Is part of: **Storyboard**

Associated with: **Interaction**

Description of: **Interaction_sequence**

Captures the sequence in which a particular interaction is included in a storyboard.

Composition for: **Interaction_sequence**

is_part_of (1,1) :: Storyboard :: contains (0,n)

Each storyboard is made up of a sequence of interactions, a sequence of use cases, or both.

Associations for: **Interaction_sequence**

links (1,1) :: Interaction :: is_linked_by (0,n)

Attributes of: **Interaction_sequence**

history : CompoundHx

This is a compound data type that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

sequence_number : Integer

The order in which the interaction participates in the Storyboard.

Class: **LOINC_link**

Is part of: **Domain_version**

Associated with: **Vocabulary_domain**

Description of: **LOINC_link**

A linkage from a particular vocabulary domain to a particular LOINC code.

Composition for: **LOINC_link**

in_version (1,1) :: Domain_version :: has (0,n)

Associations for: **LOINC_link**

links_domain (0,1) :: Vocabulary_domain :: equates_to (0,n)

Attributes of: **LOINC_link**

LOINC_code : String

The LOINC code being equated to the vocabulary domain.

LOINC_version : String

The version of LOINC being referenced.

status : CodedElement

The status of the item. The values for Status come from vocabulary domain EditStatus. Some values for status are Proposed, Rejected, Active, Obsolete, and Inactive.

version_out : String

The version number of the table at which this entry was deleted. A blank Vout value means that the row continues to exist in the current version of the table.

Class: **Message_element_type**

Is Abstract Class

Supertype of: **Choice_MET**
Collection_MET
Composite_MET
Primitive_MET
Version_MET

Is part of: **Model**

Associated with: **Choice_branch**
Collection_MET
Hierarchical_message_description
MET_component
RMIM_row

Description of: **Message_element_type**

This is an abstract generalization for particular message element types (MET). It is also known simply as a type. A MET is a specification of the values that a message element can take on in its instances. It is the basic unit of structure for HL7 Version 3 messages and related information structures.

Composition for: **Message_element_type**

in (1,1) :: Model :: has (0,n)

The relationship between Message element types and the models in which they are first defined.

Associations for: **Message_element_type**

types (0,n) :: Choice_branch :: is_of_type (1,1)

Each choice branch must be typed by an MET.

collected_by (0,n) :: Collection_MET :: collects (1,1)

Each collection MET collects elements of a single type.

defined_in (1,1) :: Hierarchical_message_description :: defines (0,n)
Each MET must be defined in one HMD.

types (0,n) :: MET_component :: is_of_type (1,1)
Each MET component is typed by a single MET.

types (0,n) :: RMIM_row :: has_type (0,1)

Attributes of: **Message element type**

ballot_version : IdentifierString

The version of HL7 models under which this MET was first balloted, with its HMD.

definition_cd : Enumerated

Indicates whether this MET is defined within an HMD, as a CMET, a type for a data type, or is being re-used in an HMD.

formal_name : NameString

The formal name (sometimes called the "long name") is a very descriptive name that generally is the same as it appears in the source. Common sources for formal names are class, attribute, or association names from the Message Information Model, Object Views from the Message Object Diagram, Common Message Element Type definitions, and Data Type definitions.

history : CompoundHx

This is a compound data type that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

name : NameString

The name (sometimes called the "short name") is intended to be a mnemonic abbreviation of the formal name that is more tractable for programming, for reference as types in another MET and for use in Implementation Technology Specifications.

Class: **Message information model**

Composite of: **MIM_attribute**
 MIM_class
 MIM_relationship
 MIM_state

Associated with: **HL7_committee**
 Model
 Refined_message_information_model

Description of: **Message information model**

Is a subset of the HL7 Reference Information Model (RIM) that contains only those classes, attributes, states and relationships necessary to support the messages to be specified for a particular project. The acronym for message information model is MIM.

The elements of the MIM may have their optionality and/or cardinality constrained beyond the levels set in the RIM. For example, if a connection is optional in the RIM, it may be constrained to be mandatory in the MIM. Similarly, an attribute that may repeat any number of times in the RIM may be constrained in the MIM to a specific number of repeats.

In selecting classes, attributes, states and connections for inclusion in a MIM, a Technical Committee should follow the following process and rules. The TC is selecting the elements of a MIM that will contain: Classes; Attributes (perhaps with constrained optionality); States; Instance connections (perhaps with constrained cardinality); and whole part connections (perhaps with constrained cardinality).

The selection rules are:

- a) all selections must come from the RIM or from a single DIM that is a subset of the HL7 RIM;
- b) for every attribute selected, the committee must also select the class of which that attribute is a part;
- c) for every state selected, the committee must also select the class of which that state is a part;
- d) all state transitions between states selected for inclusion in the MIM shall be deemed members of the MIM, too.
- e) for every instance connection and whole part connection that is selected, the committee must also select both of the classes that are connected by that connection; and

f) any class that is selected must meet one of the following two criteria:

- i) the class contains at least one selected attribute, or
- ii) the class participates in two selected connections (instance or whole part).

Composition for: **Message information model**

contains (1,n) :: MIM_attribute :: is_part_of (1,1)

A MIM is a container holding links to individual components of the information model.

contains (1,n) :: MIM_class :: is_part_of (1,1)

A MIM is a container holding links to individual components of the information model.

contains (1,n) :: MIM_relationship :: is_part_of (1,1)

A MIM is a container holding links to individual components of the information model.

contains (0,n) :: MIM_state :: is_part_of (1,1)

A MIM is a container holding links to individual components of the information model.

Associations for: **Message information model**

defined_by (1,1) :: HL7_committee :: defines (0,n)

draws_from (1,1) :: Model :: is_basis_for (0,n)

A reference to the model (RIM version) from which all of the elements of the MIM will be drawn.

refined_by (0,n) :: Refined_message_information_model :: refines (1,1)

Each R-MIM refines a single MIM.

Attributes of: **Message information model**

ballot_version : IdentifierString

The version of HL7 models in which this MIM was first balloted. Balloting will freeze the contents of the MIM. Future changes to the MIM must be made with a new MIM, established under a new id.

description : DescriptiveText

Describes the general set of messages (HMDs) that will be built from this MIM.

history : CompoundHx

This is a compound data type that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

id : IdentifierString

Arbitrary identifier assigned by Technical Steering Committee. Committees may assign an interim identifier that starts with the committee's identifier, as "Cnn_<identifier>" so long as this composite identifier is unique.

name : NameString

A brief descriptive name for the MIM. This may be the same as the name of the supported project.

Class: **Message MET**

Subtype of: **Composite_MET**

Description of: **Message MET**

A message MET types all of the messages in an HMD.

Class: **Message row control**

Is part of: **Message_type**

Associated with: **HMD_domain_constraint**
HMD_notation
HMD_row

Description of: **Message row control**

An element of a message type that controls the use of a particular HMD row in messages defined by the parent message type. The message row control has one subtype that relates to HMD attribute rows.

Composition for: **Message row control**

included_in (1,1) :: Message_type :: includes (0,n)

A reference to the message structure of which each message row control is a part.

Associations for: **Message_row_control**

has_domain (0,1) :: HMD_domain_constraint :: constrains (0,n)

has_notation (0,n) :: HMD_notation :: annotates (1,1)

controls (1,1) :: HMD_row :: controlled_by (0,n)

Each message row control controls the presence of one unsubsumed HMD row in the message structure of which the message row control is a part.

Attributes of: **Message_row_control**

conformance : Enumerated

Describes the requirement of information systems to send, or receive and process, this message element in order to claim conformance to the HL7 messaging standard defined by this message type. Values are: Required (R) and Not Required (N).

default_value : String

A notation that captures the default value for this row in the message type. It provides a value that a sending system may insert when creating a message instance if it has no other value to use.

history : CompoundHx

This is a compound data type that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

inclusion : Enumerated

Shows whether a message element may appear and if it may be null.. A mandatory or required message element may appear within an optional message element. If the outer message element, which is optional, actually appears in a message instance, any mandatory inner element must appear Possible values are: Mandatory (M), and Optional (O)..

repetitions : MultiplicityString

Describes whether the message element may repeat. Shows the minimum and maximum number of repetitions for this row (and its subordinates) in this message structure. It is not consistent to have a minimum number of repetitions of zero unless the Inclusion value is Optional.

Class: **Message_type**

Supertype of: **Union_message_type**

Is part of: **Hierarchical_message_description**

Composite of: **Message_row_control**

Associated with: **Interaction**
Union_message_type

Description of: **Message_type**

A message type is part of an HMD. It defines the specific information transfer that occurs in an interaction to meet the requirements of use cases. It is a set of constraints applied to the message elements defined in the HMD. These are represented by a set of columns in the HMD. The content for those columns is specified by the message row control instances that are parts of the message type.

The message type is an atomic unit in that the entire information content defined by the message type will be sent in an interaction, or no part of it will be sent. No further decomposition is possible.

Composition for: **Message_type**

is_part_of (1,1) :: Hierarchical_message_description :: contains (1,n)

Each message structure is contained in a single HMD.

includes (0,n) :: Message_row_control :: included_in (1,1)

A reference to the message structure of which each message row control is a part.

Associations for: **Message_type**

transferred_by (1,n) :: Interaction :: transfers (1,1)

Each interaction shall include a link to a single message structure that the interaction will transfer.

combined_in (0,1) :: Union_message_type :: combines (1,n)

Attributes of: **Message_type**

history : CompoundHx

This is a compound data type that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

identifier : IdentifierString

Arbitrary, unique identifier assigned by Technical Steering Committee. Committees may assign an interim identifier that starts with the committee's identifier, as "Cnn_<identifier>" so long as this composite identifier is unique.

isCommonType : Boolean

Identifies that this message type structure is common to all of the message types in this HMD.

Class: MET component

Is part of: **Composite_MET**

Associated with: **Message_element_type**

Description of: MET component

Elements that make up a composite MET.

Each MET component has a name and a type. The names must be unique within the composite. Each component of an element must have a different name, although many may be of the same type.

Composition for: MET component

is_part_of (1,1) :: Composite_MET :: has_parts (1,n)

Associations for: MET component

is_of_type (1,1) :: Message_element_type :: types (0,n)

Each MET component is typed by a single MET.

Attributes of: MET component**history : CompoundHx**

This is a compound data type that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

is_optional : Boolean

Indicates whether this component is optional within the MET composite of which it is a part.

locale : Enumerated

Locale is intended to be a code. The intent is that all the information in a component flagged for a particular locale is specific to the definition for that locale. This allows for locale-specific information to be added to a message type. A scheme for generating unambiguous locale IDs is required.

In theory, the locale ID does not need to be transmitted in a message instance, because the locally-tagged code has its own name.

Because of the hierarchy of uniform METs the approach can be applied anywhere from data types on up. It also supports message instances carrying multiple locale-specific inclusions.

long_name : NameString

The full name for the component

name : NameString

This is the "short name" for the component.

sequence : Integer

Provides the sequence number within the composite for use in a segment-positional syntax such as ER7.

Class: MIM aggregation

Subtype of: **MIM_relationship**

Associated with: **Composite_aggregation**

Description of: MIM aggregation

A linking element that includes an individual aggregation relationship from the RIM into a particular MIM.

Associations for: MIM aggregation

includes (1,1) :: Composite_aggregation :: included_in (0,n)

Attributes of: MIM_aggregation

part_multiplicity : MultiplicityString

Expresses a multiplicity for the part in this aggregation that applies to all uses of the aggregation in this MIM, and that is a tighter constraint than that recorded in the RIM. This attribute must be valued.

Class: MIM_association

Subtype of: **MIM_relationship**

Associated with: **Association**

Description of: MIM_association

A linking element that includes one individual association from the RIM into a particular MIM.

Associations for: MIM_association

includes (1,1) :: Association :: included_in (0,n)

Attributes of: MIM_association

source_multiplicity : MultiplicityString

Expresses a multiplicity for the source class in this association that applies to all uses of the association in this MIM, and that is a tighter constraint than that recorded in the RIM. This attribute must be valued.

target_multiplicity : MultiplicityString

Optional attribute expresses a multiplicity for the target class in this association that applies to all uses of the association in this MIM, and that is a tighter constraint than that recorded in the RIM. This attribute must be valued.

Class: MIM_attribute

Is part of: **Message_information_model**

Associated with: **Attribute**
Data_type
MIM_attribute_domain_constraint
RMIM_attribute_row

Description of: MIM_attribute

Includes individual attributes from the RIM into a particular MIM, provided that the parent class for the attribute is also in the MIM..

Composition for: MIM_attribute

is_part_of (1,1) :: Message_information_model :: contains (1,n)

A MIM is a container holding links to individual components of the information model.

Associations for: MIM_attribute

includes (1,1) :: Attribute :: included_in (0,n)

has_working (0,1) :: Data_type :: is_working (0,n)

If a MIM attribute is associated with a RIM attribute that does not have an assigned data type, the MIM attribute may be assigned an alternative, working data type. This association must not be instantiated otherwise.

constrained_by (0,1) :: MIM_attribute_domain_constraint :: constrains (0,n)

has_dependent (0,n) :: RMIM_attribute_row :: based_on (1,1)

Each R-MIM attribute is based on one MIM attribute.

Attributes of: MIM_attribute

history : CompoundHx

This is a compound data type that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

inclusion : Boolean

This attribute expresses whether the inclusion of the attribute in the HMD and message structures derived from the HMD is mandatory. If the inclusion of the attribute is mandatory in the RIM, it must also be mandatory in the MIM. The default is the inclusion constraint for this attribute in the RIM.

repeatability : Boolean

This attribute expresses whether the attribute may repeat in a message. The repeatability applies to all uses of the attribute in this MIM, and must be a tighter constraint than that recorded in the RIM. The default for this attribute is the repeatability assigned to the attribute in the RIM.

Class: MIM_attribute_domain_constraint

Associated with: **MIM_attribute**
 Vocabulary_domain

Description of: MIM_attribute_domain_constraint

Constrains a coded MIM attribute to a particular vocabulary domain.

For any class, the special attribute status_cd has as its domain all of the states of the class.

Associations for: MIM_attribute_domain_constraint

constrains (0,n) :: MIM_attribute :: constrained_by (0,1)

links_domain (1,1) :: Vocabulary_domain :: is_constraint (0,n)

Attributes of: MIM_attribute_domain_constraint**strength : String**

The strength of the constraint is either CWE (coded with exceptions) or CNE (coded, no exceptions). If no value is given, CWE is the default.

Class: MIM_class

Is part of: **Message_information_model**

Associated with: **Class**
 RMIM_class_row

Description of: MIM_class

Includes individual classes from the RIM into a particular MIM.

Composition for: MIM_class

is_part_of (1,1) :: Message_information_model :: contains (1,n)

A MIM is a container holding links to individual components of the information model.

Associations for: MIM_class

includes (1,1) :: Class :: included_in (0,n)

has_dependent (0,n) :: RMIM_class_row :: is_based_on (1,1)

Attributes of: MIM_class**history : CompoundHx**

This is a compound data type that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

Class: MIM_generalization

Subtype of: **MIM_relationship**

Associated with: **Generalization_relationship**

Description of: MIM_generalization

A linking element that includes one individual generalizations from the RIM into a particular MIM.

Associations for: MIM_generalization

includes (1,1) :: Generalization_relationship :: included_in (0,n)

Class: MIM_relationship

Is Abstract Class

Supertype of: **MIM_aggregation**
 MIM_association
 MIM_generalization

Is part of: **Message_information_model**

Associated with: **RMIM_relationship_row**

Description of: MIM_relationship

Provides an abstract generalization for the MIM links to association, aggregation, and generalization connections in the RIM.

Composition for: MIM_relationship

is_part_of (1,1) :: Message_information_model :: contains (1,n)

A MIM is a container holding links to individual components of the information model.

Associations for: **MIM_relationship**

has_dependent (0,n) :: RMIM_relationship_row :: is_based_on (1,1)

Attributes of: **MIM_relationship**

history : CompoundHx

This is a compound data type that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

Class: **MIM_state**

Is part of: **Message_information_model**

Associated with: **RMIM_state_row**
State

Description of: **MIM_state**

Includes an individual state into the MIM, provided that the parent class for the state is also in the MIM.

Composition for: **MIM_state**

is_part_of (1,1) :: Message_information_model :: contains (0,n)

A MIM is a container holding links to individual components of the information model.

Associations for: **MIM_state**

has_dependent (0,n) :: RMIM_state_row :: is_based_on (1,1)

includes (1,1) :: State :: included_in (0,n)

Attributes of: **MIM_state**

history : CompoundHx

This is a compound data type that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

Class: **Model**

Composite of: **Actor**
Application_role
Class

Data_type

Data_type_category

Hierarchical_message_description

Interaction

Interaction_model_category

Message_element_type

Refined_message_information_model

Storyboard

Subject_area

Trigger_event

Use_case

Use_case_category

Associated with: **HL7_committee**
Message_information_model

Description of: **Model**

This class in the meta-model contains the elements necessary to uniquely define an aggregate model, to establish its provenance and scope, and to link it to each of the elements that make up that model.

A model is a collection of subject areas, scenarios, classes, attributes, use cases, actors, trigger events, interactions, etc. that depict the information needed to specify HL7 Version3 messages. This model is further divided into four specific models - a use case model, an information model, an interaction model, and a message design model..

Composition for: **Model**

has (0,n) :: Actor :: in (1,1)

The relationship between actors and the models of which they are a part.

has (0,n) :: Application_role :: in (1,1)

The relationship between application roles and the models of which they are a part.

has (0,n) :: Class :: in (1,1)

The relationship between classes and the models of which they are a part.

has (0,n) :: Data_type :: in (1,1)

The relationship between data types and the models in which they are first defined.

has (0,n) :: Data_type_category :: in (1,1)

The relationship between data type categories and the models of which they are a part.

has (0,n) :: Hierarchical_message_description :: in (1,1)

The relationship between hierarchical message descriptions and the models in which they are first defined.

has (0,n) :: Interaction :: in (1,1)

The relationship between interactions and the models of which they are a part.

has (0,n) :: Interaction_model_category :: in (1,1)

The relationship between interaction model categories and the models of which they are a part.

has (0,n) :: Message_element_type :: in (1,1)

The relationship between Message element types and the models in which they are first defined.

contains (0,n) :: Refined_message_information_model :: is_part_of (1,1)

has (0,n) :: Storyboard :: in (1,1)

The relationship between scenarios and the models of which they are a part.

has (0,n) :: Subject_area :: in (1,1)

The relationship between subject areas and the models of which they are a part.

has (0,n) :: Trigger_event :: in (1,1)

Sets relationship between model elements and the models of which they are a part.

has (0,n) :: Use_case :: in (1,1)

The relationship between use cases and the models of which they are a part.

has (0,n) :: Use_case_category :: in (1,1)

The relationship between use case model categories and the models of which they are a part.

Associations for: **Model**

prepared_by (1,1) :: HL7_committee :: prepares (0,n)

Links a model to the committee that prepared it.

is_basis_for (0,n) :: Message_information_model :: draws_from (1,1)

A reference to the model (RIM version) from which all of the elements of the MIM will be drawn.

Attributes of: **Model**

description : DescriptiveText

A short narrative describing the scope and intent of the model.

developing_org : String

A short form identifier of the organization responsible for the publication and maintenance of the model. . For HL7, this name shall be "HL7."

last_modified_date : Date

The date the model was last modified by the model developing organization.

modelID : NameString

A unique identifier assigned to the model by the developing organization. In HL7, these identifiers will be assigned by the Modeling and Methodology Committee.

name : NameString

A descriptive title for the model. The name in combination with the version number shall be unique within the set of models developed by any particular model developing organization.

version_number : VersionNumber

A number showing the release level of the model. The version number, in combination with the name, shall be unique for all public releases of the model.

Class: **Primitive data type**

Subtype of: **Data_type**

Associated with: **Primitive_MET**

Description of: **Primitive data type**

A primitive data type is a data type that is defined entirely by its specification. A primitive data type may have generic type parameters.

Associations for: **Primitive data type**

types (0,n) :: Primitive_MET :: is_of_type (1,1)

A Primitive MET gains its own type from a Primitive data type.

Class: Primitive MET

Subtype of: **Message_element_type**

Associated with: **Primitive_data_type**

Description of: Primitive MET

A message element type that does not contain other message elements. A primitive MET may only be defined as part of a data type MET (DMET).

Associations for: Primitive MET

is_of_type (1,1) :: Primitive_data_type :: types (0,n)

A Primitive MET gains its own type from a Primitive data type.

Class: Project

Associated with: **Hierarchical_message_description**
HL7_committee

Description of: Project

The specification of a particular, coherent set of messages and events based around one (or a few) Subject Classes. A project is undertaken to address a current need for standardizing information that flows between a number of parties in healthcare. A project also defines a ballot package that might be advanced independently of other HL7 ballot packages.

Associations for: Project

implemented_by (0,n) :: Hierarchical_message_description :: supports (1,1)

A MIM should support a single project.

responsibility_of (1,1) :: HL7_committee :: responsible_for (0,n)

Establishes the relationship between committees and the projects for which that committee is responsible.

Attributes of: Project

ANSI_PINS_date : Date

The date on which the project scope was published in the ANSI PINS system.

id : IdentifierString

Arbitrary identifier assigned by Technical Steering Committee.

name : NameString

Brief descriptive name for the project.

scope : DescriptiveText

The approved scope statement for the project. It defines the area of healthcare functionality that needs to be supported by HL7 messaging and is a high level use case that encompasses the entire project.

TSC_approval_date : Date

Date on which the project was approved by the TSC.

Class: Refined message information model

Is part of: **Model**

Composite of: **RMIM_note**
RMIM_row

Associated with: **Message_information_model**

Description of: Refined message information model

The Refined message information model (R-MIM) is a hybrid between a class model and an object model. Each class, attribute and association in the MIM is part of the R-MIM.

In addition, classes in the MIM may be cloned in the R-MIM. That is, they may be represented more than once. This is done to allow the messages to be tailored to the specific needs of different instances of a class. For example, both the Person class has roles for both patients and doctors. By and large, the attributes and associations of Person that are important to the patient role are different from those important to the doctor role. Cloning allows these differences to be represented explicitly in the R-MIM.

When a class is cloned, the clone must be given a unique name, and the original may be re-named, as well. Both the clone and the original may be constrained (beyond the constraints of the MIM) independently. Constraints involve removing attributes, tightening association cardinality, discarding associations and reducing vocabulary domains.

The R-MIM is displayed both as a UML diagram, and in a two-level tabular format in which each class and clone is a primary row, and the attributes and associations of those classes comprise the second-level rows. Selected attributes of the meta-model provide 'lay-out' information for the tabular format.

Composition for: **Refined message information model**

is_part_of (1,1) :: Model :: contains (0,n)

contains (0,n) :: RMIM_note :: part_of (1,1)

contains (1,n) :: RMIM_row :: part_of (1,1)

Associations for: **Refined message information model**

refines (1,1) :: Message_information_model :: refined_by (0,n)
Each R-MIM refines a single MIM.

Attributes of: **Refined message information model**

first_node_id : IdentifierString

Identifies the first class node in the tabular display of the R-MIM.

history : CompoundHx

A compound data element that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

identifier : IdentifierString

A unique identifier for the R-MIM.

Class: **RMIM attribute domain constraint**

Associated with: **RMIM_attribute_row**
Vocabulary_domain

Description of: **RMIM attribute domain constraint**

Constrains a coded RMIM attribute row to a particular vocabulary domain.

For any class, the special attribute status_cd has as its domain all of the states of the class.

Associations for: **RMIM attribute domain constraint**

constrains (0,n) :: RMIM_attribute_row :: constrained_by (0,1)

links_domain (1,1) :: Vocabulary_domain :: is_constraint (0,n)

Attributes of: **RMIM attribute domain constraint**

strength : String

The strength of the constraint is either CWE (coded with exceptions) or CNE (coded, no exceptions). If no value is given, CWE is the default.

Class: **RMIM attribute row**

Subtype of: **RMIM_row**

Associated with: **HMD_attribute_row**
MIM_attribute
RMIM_attribute_domain_constraint

Description of: **RMIM attribute row**

Expresses the presence of selected attributes in the R-MIM.

Associations for: **RMIM attribute row**

defines (0,n) :: HMD_attribute_row :: defined_by (1,1)

based_on (1,1) :: MIM_attribute :: has_dependent (0,n)
Each R-MIM attribute is based on one MIM attribute.

constrained_by (0,1) :: RMIM_attribute_domain_constraint :: constrains (0,n)

Class: **RMIM class row**

Subtype of: **RMIM_row**

Associated with: **HMD_class_row**
MIM_class
RMIM_relationship_row
RMIM_row
RMIM_row

Description of: **RMIM class row**

Expresses the presence of selected classes or clones thereof in the R-MIM.

Associations for: **RMIM class row**

defines (0,n) :: HMD_class_row :: defined_by (1,1)

is_based_on (1,1) :: MIM_class :: has_dependent (0,n)

is_related_by (0,n) :: RMIM_relationship_row :: has_distal_class (0,1)

is_active_parent (0,n) :: RMIM_row :: has_active_parent (1,1)

Shows the active parent relationship for an inherited row. Reflects the combined effects of inheritance and cloning.

is_true_parent (0,n) :: RMIM_row :: has_true_parent (1,1)

Establishes the true parent for each association and attribute. Is required because the act of cloning precludes determining this association through the information model or MIM.

Attributes of: **RMIM_class_row**

first_attribute_row_id : IdentifierString

Pointer to the first child attribute row for this class row.

first_relation_row_id : IdentifierString

Pointer to the first child relationship row for this class row.

Class: **RMIM_notation**

Associated with: **RMIM_note**
RMIM_row

Associations for: **RMIM_notation**

links_note (1,1) :: RMIM_note :: is_notation (0,n)

annotates (1,1) :: RMIM_row :: has_notation (0,n)

Attributes of: **RMIM_notation**

type : Enumerated

Indicates the type of the note. Types include:

RV : Required value

CP : Conditional Presence

CN : Constraint

DM : Domain

CT : Comment

Class: **RMIM_note**

Is part of: **Refined_message_information_model**

Associated with: **HMD_notation**
RMIM_notation

Description of: **RMIM_note**

A note may be placed on any row of an R-MIM or of a Hierarchical Message Definition. These notes clarify the semantic intent of the Technical Committee.

Composition for: **RMIM_note**

part_of (1,1) :: Refined_message_information_model :: contains (0,n)

Associations for: **RMIM_note**

is_notation (0,n) :: HMD_notation :: links_note (1,1)

is_notation (0,n) :: RMIM_notation :: links_note (1,1)

Attributes of: **RMIM_note**

number : Integer

Notes within an HMD are identified by number.

subject : String

Captures the subject, a column of the HMD, to which the note applies.

Class: **RMIM_other_row**

Subtype of: **RMIM_row**

Associated with: **HMD_other_row**
RMIM_row

Description of: **RMIM_other_row**

Expresses the presence of special rows in the R-MIM. There are two types of such additional rows:

item : Represents the individual elements of the message that make up a Set or List of elements, as required by repeating attributes or associations.

stc : Represents the presence of a sub-component of a data type in the message. These components are exposed in to allow the expression of constraints against them.

Associations for: **RMIM_other_row**

defines (0,n) :: HMD_other_row :: defined_by (1,1)

has_parent (1,1) :: RMIM_row :: is_parent (0,n)

Each 'other' node arises as a result of some other node, its parent.

Attributes of: **RMIM_other_row**

otherType : Enumerated

Coded value for the type of the other row. See definition of the RMIM_other_row class for the code values and their meaning.

Class: **RMIM_relationship_row**

Subtype of: **RMIM_row**

Associated with: **HMD_relationship_row**
MIM_relationship
RMIM_class_row
RMIM_relationship_row
RMIM_relationship_row

Description of: **RMIM_relationship_row**

Expresses the presence of selected association nodes (UML roles) in the R-MIM. Each relationship in the MIM and R-MIM produces two rows in the tabular R-MIM, one for the appearance of each end of the relationship in one of the R-MIM classes or clones.

Associations for: **RMIM_relationship_row**

defines (0,n) :: HMD_relationship_row :: defined_by (1,1)

is_based_on (1,1) :: MIM_relationship :: has_dependent (0,n)

has_distal_class (0,1) :: RMIM_class_row :: is_related_by (0,n)

has_other_half (1,1) :: RMIM_relationship_row :: is_other_half (0,1)

Each RMIM relationship row may be paired with a second such row to comprise a complete relationship.

is_other_half (0,1) :: RMIM_relationship_row :: has_other_half (1,1)

Each RMIM relationship row may be paired with a second such row to comprise a complete relationship.

Attributes of: **RMIM_relationship_row**

blocked : Boolean

Expresses whether this half-relationship is intended to be followed in building an HMD. This value is not normative. It may be over-ridden. When the R-MIM is diagrammed in UML, the presence of a blocked path is shown by making the UML role for the other end of the relationship un navigable.

Class: **RMIM_row**

Is Abstract Class

Supertype of: **RMIM_attribute_row**
RMIM_class_row
RMIM_other_row
RMIM_relationship_row
RMIM_state_row

Is part of: **Refined_message_information_model**

Associated with: **Message_element_type**
RMIM_class_row
RMIM_class_row
RMIM_notation
RMIM_other_row

Description of: **RMIM_row**

The R-MIM is modeled by the Rows that make up its tabular expression.

Composition for: **RMIM_row**

part_of (1,1) :: Refined_message_information_model :: contains (1,n)

Associations for: **RMIM_row**

has_type (0,1) :: Message_element_type :: types (0,n)

has_active_parent (1,1) :: RMIM_class_row :: is_active_parent (0,n)
Shows the active parent relationship for an inherited row. Reflects the combined effects of inheritance and cloning.

has_true_parent (1,1) :: RMIM_class_row :: is_true_parent (0,n)
Establishes the true parent for each association and attribute. Is required because the act of cloning precludes determining this association through the information model or MIM.

has_notation (0,n) :: RMIM_notation :: annotates (1,1)

is_parent (0,n) :: RMIM_other_row :: has_parent (1,1)
Each 'other' node arises as a result of some other node, its parent.

Attributes of: **RMIM_row**

cardinality : MultiplicityString

Expresses the minimum and maximum number of occurrences for an association or an attribute in the context of the R-MIM.

conformance : Enumerated

Expresses the constraints on this row for conformance testing. Possible values are:

R: required for conformance

(blank): not required for conformance

NP: not permitted to appear in his message variant (only used in message row controls, not in R-MIM).

constraint : DescriptiveText

Captures constraints and notes for a given row in the R-MIM and HMD. The type of note and its contents must both be expresses. The types provided for include:

Required value : states a value and, in the presence of repetition, how many times the value can/must appear

Conditional Presence : states a value that must or must not be present based on the value of another element or sub-component that is higher in the HMD

Constraint : a verbal expression of a constraint

Domain : a domain specification, as described in the vocabulary chapter

Comment : any general comment; this label should not be used for items that can be described with any of the other labels.

default_update_mode : String

A coded value drawn from the possible modes of updating that occur when an attribute is received by a system that already contains values for that attribute. The update modes and their codes are:

R : replace (this is the default)

D : delete

I : ignore

NA : not applicable

V : verify: confirm that it exists

K : key: when creating an element store it; when updating an element confirm that it exists.

The following codes apply when updating individual items in a set:

ESA : edit set: add item

ESC : edit set: change item

ESD : edit set: delete item

ESAC : edit set: add or, if the item exists, change item

default_value : DescriptiveText

The default value for this attribute. If this field is blank or Null, the default is 'NULL'. If the field contains 'No' then no default specified. Otherwise, the field contains the value of the default and if the value is an integer it must be enclosed in quotes.

history : CompoundHx

A compound data element that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

mandatory : Boolean

If this attribute has a value of "True" then this message element must have a non-null value in order for the receiver to process the message.

name : NameString

The name of the row in the R-MIM. Rows representing attributes should not be renamed.

next_sibling_ID : IdentifierString

Points to the next sibling node in the tabular R-MIM.

previous_sibling_id : IdentifierString

Points to the previous sibling node in the tabular R-MIM.

short_name : NameString

A shortened version of the name for the row.

update_mode_set : String

A set of utterances from the list of values for 'default_update_mode.'. The sender may change the update mode instance by instance to any of the values in this list.

Class: **RMIM_state_row**

Subtype of: **RMIM_row**

Associated with: **MIM_state**

Description of: **RMIM_state_row**

Expresses the presence of selected states in the R-MIM.

Associations for: **RMIM_state_row**

is_based_on (1,1) :: MIM_state :: has_dependent (0,n)

Class: **State**

Is part of: **Subject_class**

Associated with: **MIM_state**
State
State
State_transition
State_transition

Description of: **State**

The identification of a unique combination of attribute value(s) and connections which are of interest about a subject class.

The enumerated States for each Subject class must include an "Initial state" from which some designated state transition moves the class to one or more active states.

Composition for: **State**

in (1,1) :: Subject_class :: has (0,n)

This relationship between states and the subject classes of which they are a part.

Associations for: **State**

included_in (0,n) :: MIM_state :: includes (1,1)

has_substate (0,n) :: State :: is_substate_of (0,1)

States may be defined as sub-states of a parent, provided that all of the states for a given class have unique names.

is_substate_of (0,1) :: State :: has_substate (0,n)

States may be defined as sub-states of a parent, provided that all of the states for a given class have unique names.

ends (0,n) :: State_transition :: ends_in (1,1)

is_start_of (0,n) :: State_transition :: starts_from (1,1)

Attributes of: **State****description : DescriptiveText**

A short informative description of the State.

history : CompoundHx

This is a compound data type that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

Note, the version data must fit within the range of the applicable versions for the class of which this element is a part.

name : NameString

The name of this particular state which shall be unique within this class.

predicate : String

The condition or set of conditions that when true about the attribute(s) and connections of the parent subject class identifies that the subject class is in the declared state.

Class: State transition

Associated with: **State**
 State
 Trigger_event
 Use_case

Description of: State transition

Captures the semantics of transitions from state-to-state for the Subject classes. Note that a legal transition may return to the same state from which it started.

State transitions also are a critical link between leaf-level use cases from which the transitions stem, and the trigger events identified with the transitions.

Associations for: State transition

ends_in (1,1) :: State :: ends (0,n)

starts_from (1,1) :: State :: is_start_of (0,n)

identified_by (0,1) :: Trigger_event :: identifies (1,n)

This connection from state transition to trigger event is only optional because we do not expect all possible trigger events to be defined in HL7. Nevertheless, any state transition that is described in a use case must be linked to a trigger event.

Note that because of the above condition, and because of the condition on the instance connection from a use case to a state transition, there is a mandatory (1,1) relationship from any leaf-level use case to one trigger event.

The relationship from trigger event to state transition is (1..*) because although the event will probably drive the Subject class to a single state (e.g. "Canceled") and the transitions may start from several states. Thus one trigger event may identify many transitions.

captured_in (0,n) :: Use_case :: describes (0,1)

A leaf-level use case describes the events that result in a single state transition of the subject class.

Although the instance connection from use case to state transition is optional, this is only because not all use cases are leaf-level. At the leaf-level, each use case should describe one and only one state transition.

Attributes of: State transition

description : DescriptiveText

A short, informative description of the state transition.

history : CompoundHx

This is a compound data type that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

Note, the version data must fit within the range of the applicable versions for both of the states to which this transition links.

label : NameString

A word or phrase that reflects the event that causes the transition. The name shall be unique with respect to the state from which the transition starts. This label may be the same as the name of the trigger event identified with the Transition.

Class: Storyboard

Is part of: **Model**

Composite of: **Interaction_sequence**
 Storyboard_example
 Use_case_sequence

Description of: Storyboard

A storyboard is a statement of health care relevant events defined as a sequence of leaf-level use cases or interactions. The storyboard provides one set of use case instances that the modeling committee expects will typically occur in the domain.

A storyboard may also be expressed as a subset of the interaction model in which case the representation includes all interactions that are implied by the trigger events associated with the sequence of use cases or are implied by the sender and

receiver responsibilities of those interactions. Usually, an interaction diagram is constructed to show a group of interactions for a single storyboard.

The collection of storyboards that HL7 may publish does not limit the ways in which HL7 can be applied; other combinations of trigger events, interactions, and application roles that are consistent with the interaction model may also be used.

Composition for: **Storyboard**

contains (0,n) :: Interaction_sequence :: is_part_of (1,1)

Each storyboard is made up of a sequence of interactions, a sequence of use cases, or both.

in (1,1) :: Model :: has (0,n)

The relationship between scenarios and the models of which they are a part.

exemplifies (0,n) :: Storyboard_example :: is_part_of (1,1)

Each storyboard example provides a real world example for a single storyboard..

contains (0,n) :: Use_case_sequence :: contains (1,1)

Each storyboard is made up of a sequence of interactions, a sequence of use cases, or both.

Attributes of: **Storyboard**

history : CompoundHx

This is a compound data type that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

identifier : IdentifierString

An identifier assigned to the storyboard to simplify references to it. The identifier should be unique within the scope of the model in which it is defined. In HL7, committees manage the unique identifiers for their storyboards, and concatenate the committee identifier as "Cnn_<identifier>."

name : NameString

A short phrase that provides a descriptive title for the storyboard.

purpose : DescriptiveText

The purpose for which the storyboard was created. Frequently it describes the generic set of actions that the storyboard represents.

Class: **Storyboard_example**

Is part of: **Storyboard**

Description of: **Storyboard_example**

Provides a real-world example of the sequence of events captured in a Storyboard.

Composition for: **Storyboard_example**

is_part_of (1,1) :: Storyboard :: exemplifies (0,n)

Each storyboard example provides a real world example for a single storyboard..

Attributes of: **Storyboard_example**

description : DescriptiveText

A narrative example from the real world that describes a set of events represented by the sequence of use cases that make up the Storyboard which this example exemplifies.

history : CompoundHx

This is a compound data type that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

identifier : IdentifierString

A unique identifier assigned to the storyboard example.

Class: **Subject_area**

Is part of: **Model**

Associated with: **Class**
Class
HL7_committee
Subject_area
Subject_area

Description of: **Subject_area**

A major category of information represented in the information model. An aggregation of interrelated classes. A subject area allows portions of a large model to be viewed as a whole thereby eliminating some complexity involved in understanding a large model.

Subject areas will also be used by the Methodology and Modeling Committee of HL7 to designate Domain Information Models (DIM), class stewardship responsibilities, and classes of interest to a particular committee.

Composition for: **Subject_area**

in (1,1) :: Model :: has (0,n)

The relationship between subject areas and the models of which they are a part.

Associations for: **Subject_area**

holds (1,n) :: Class :: primarily_resides_in (0,1)

The linkage between a Class and the Subject area that is its primary residence. This must be established if a Class resides in more than one Subject area.

includes (1,n) :: Class :: appears_in (0,n)

The linkage between a Subject area and each of the Classes that are in that Subject area.

maintained_by (0,1) :: HL7_committee :: maintains (0,n)

is_nested_in (0,1) :: Subject_area :: nests (0,n)

The linkage between two subject areas where one of the two is nested within the other.

nests (0,n) :: Subject_area :: is_nested_in (0,1)

The linkage between two subject areas where one of the two is nested within the other.

Attributes of: **Subject_area**

description : DescriptiveText

Short informative text describing the subject area so as to be clear what type of Classes it includes.

history : CompoundHx

This is a compound data type that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

name : NameString

The name given to the subject area. A subject area name is often the plural form of the name of the central or dominant class within the subject area. Subject area names shall be unique within a given model.

Class: **Subject_class**

Subtype of: **Class**

Composite of: **State**

Associated with: **Application_role**

Attribute

Trigger_event

Use_case

Description of: **Subject_class**

A specialization of Class that is used in HL7 to identify those classes that are the focus for a set of Use Cases, Trigger events and/or Application Roles.

Composition for: **Subject_class**

has (0,n) :: State :: in (1,1)

This relationship between states and the subject classes of which they are a part.

Associations for: **Subject_class**

subject_of (0,n) :: Application_role :: relates_to (1,1)

Links each Application role to the Subject class for which it plays a role. The nature of this relationship is stereotyped as discussed in the description for the Application_role.

has_state_attribute (1,1) :: Attribute :: is_state_attribute_for (0,1)

The state attribute of a class contains a value indicating the current state of the class. In the event that the class has concurrent states, the attribute must be a set of state values.

is_driven_by (1,n) :: Trigger_event :: affects (1,1)

Reflects the fact that although a trigger event may cause multiple state transitions, all of these transitions will be within the states of a single subject class.

subject_of (0,n) :: Use_case :: describes (0,1)
Links a leaf-level use case to its Subject Class.

Class: Trigger event

Is part of: **Model**
Associated with: **Interaction**
State_transition
Subject_class

Description of: Trigger event

An occurrence in the health care domain, or within the systems that support this domain, that causes information to be exchanged in the domain or between systems. Trigger events are initiators of Interactions.

Each Trigger event is tied to one State transition for one of the Subject classes in the model. In turn, the State transition traces to a leaf-level Use case from which the transition stems, and the leaf-level use case defines the context for the trigger event.

Composition for: Trigger event

in (1,1) :: Model :: has (0,n)
Sets relationship between model elements and the models of which they are a part.

Associations for: Trigger event

initiates (0,n) :: Interaction :: initiated_by (1,1)
A reference to the trigger event that triggers or initiates this interaction.

identifies (1,n) :: State_transition :: identified_by (0,1)
This connection from state transition to trigger event is only optional because we do not expect all possible trigger events to be defined in HL7. Nevertheless, any state transition that is described in a use case must be linked to a trigger event.

Note that because of the above condition, and because of the condition on the instance connection from a use case to a state transition, there is a mandatory (1,1) relationship from any leaf-level use case to one trigger event.

The relationship from trigger event to state transition is (1..*) because although the event will probably drive the Subject class to a single state (e.g. "Canceled")

and the transitions may start from several states. Thus one trigger event may identify many transitions.

affects (1,1) :: Subject_class :: is_driven_by (1,n)
Reflects the fact that although a trigger event may cause multiple state transitions, all of these transitions will be within the states of a single subject class.

Attributes of: Trigger event

dependency : String

If the occurrence of the trigger event is dependent upon the state of one or more objects in the domain or upon the prior occurrence of a different trigger event, this dependency will be expressed in this textual component.

description : DescriptiveText

The text that describes the trigger event. When viewed along with the description of the State transitions which the event identifies, this description must have sufficient detail that the event can be reliably recognized when it occurs.

history : CompoundHx

This is a compound data type that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

identifier : IdentifierString

An identifier assigned to the trigger event. The identifier is unique within the scope of the model in which the trigger event is defined. In HL7, committees manage the unique identifiers for their trigger events, and concatenate the committee identifier as "Cnn_<identifier>."

name : NameString

A name assigned to the trigger event. The name is unique within the scope of the model in which the trigger event is defined.

Class: Union message type

Subtype of: **Message_type**
Associated with: **Message_type**

Associations for: Union message type

combines (1,n) :: Message_type :: combined_in (0,1)

Class: Use_case

Is part of: **Model**
Associated with: **Actor**
State_transition
Subject_class
Use_case_category
Use_case_relationship
Use_case_relationship
Use_case_sequence

Description of: Use_case

A use case is a summary of health care relevant events and related information system events that reflect the usage of the information in the information model and related business models. Use cases describe the interactions and information interchanges that occur in the healthcare domain and the events that cause these interchanges.

Use cases may be expressed at various levels. A use case may be a parent to several child use cases. In this circumstance, the interactions ascribed to all of the children constitute the complete interaction of the parent. The decomposition to child use cases should stop when the resulting use case involves a single actor and a single interaction in response to a single stimulus - an atomic or leaf level use case. Note that a use case may be a child of more than one parent, but must not be defined such that a trace up the parental tree from a child will run into the same child (recursion).

At the lowest level of decomposition, each leaf level use case should link to only a single state transition which, in turn, links to a trigger event that initiates interactions. If the linkage is to multiple such transitions or events, further decomposition should be considered.

Composition for: Use_case

in (1,1) :: Model :: has (0,n)

The relationship between use cases and the models of which they are a part.

Associations for: Use_case

involves (1,n) :: Actor :: participates_in (0,n)

describes (0,1) :: State_transition :: captured_in (0,n)

A leaf-level use case describes the events that result in a single state transition of the subject class.

Although the instance connection from use case to state transition is optional, this is only because not all use cases are leaf-level. At the leaf-level, each use case should describe one and only one state transition.

describes (0,1) :: Subject_class :: subject_of (0,n)

Links a leaf-level use case to its Subject Class.

included_in (0,n) :: Use_case_category :: includes (0,n)

is_source_for (0,n) :: Use_case_relationship :: links_source (1,1)

Links a use case relationship to the source of the relationship.

is_target_in (0,n) :: Use_case_relationship :: links_target (1,1)

Links the use case relationship to its target or destination.

is_linked (0,n) :: Use_case_sequence :: links (1,1)

Attributes of: Use_case

description : DescriptiveText

The text that describes the use case and provides the details necessary to understand the events that are involved in the use case.

history : CompoundHx

This is a compound data type that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

identifier : IdentifierString

An identifier assigned to the use case. The identifier is unique within the scope of the model. In HL7, committees manage the unique identifiers for their use cases, and concatenate the committee identifier as "Cnn_<identifier>."

name : NameString

A short phrase that provides a descriptive name for the use case. The name should be unique within the scope of use cases defined by a particular committee.

Class: Use case category

Is part of: **Model**
Associated with: **Actor**
HL7_committee
Use_case
Use_case_category
Use_case_category

Description of: Use case category

A major category of information represented in the use case model. An aggregation of interrelated actors and use cases. A category allows portions of a large model to be viewed as a whole thereby eliminating some complexity involved in understanding a large model.

Composition for: Use case category

in (1,1) :: Model :: has (0,n)

The relationship between use case model categories and the models of which they are a part.

Associations for: Use case category

includes (0,n) :: Actor :: included_in (0,n)

maintained_by (0,1) :: HL7_committee :: maintains (0,n)

includes (0,n) :: Use_case :: included_in (0,n)

nested_in (1,1) :: Use_case_category :: nests (0,n)

Use case categories may be nested in a hierarchy.

nests (0,n) :: Use_case_category :: nested_in (1,1)

Use case categories may be nested in a hierarchy.

Attributes of: Use case category

description : DescriptiveText

Short informative text describing the use case category so as to be clear what type of use cases it includes.

history : CompoundHx

This is a compound data type that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

name : NameString

The name given to the use case category. The identifier for the committee defining this category is prepended to the name as Cnn.

Class: Use case relationship

Associated with: **Use_case**
Use_case

Description of: Use case relationship

Use cases maintain a variety of relationships. This class captures all such flavors. See the use case model chapter of the MDF for details of the stereotypical relationships.

Associations for: Use case relationship

links_source (1,1) :: Use_case :: is_source_for (0,n)

Links a use case relationship to the source of the relationship.

links_target (1,1) :: Use_case :: is_target_in (0,n)

Links the use case relationship to its target or destination.

Attributes of: Use case relationship

history : CompoundHx

This is a compound data type that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

stereotype : String

Identifies the stereotype for the use case relationship. A blank or null value is a simple generalization relationship, meaning that the target use case Use Case 1 adds additional behavior to the source use case. A value of "extends" means that the target use case adds additional behavior to the source use case at a specified Variation Point. A value of "includes" means that the target uses the source as part of its execution.

Class: Use case sequence

Is part of: **Storyboard**

Associated with: **Use_case**

Description of: Use case sequence

Captures the sequence in which a particular use case is enacted in a storyboard.

Composition for: Use case sequence

contains (1,1) :: Storyboard :: contains (0,n)

Each storyboard is made up of a sequence of interactions, a sequence of use cases, or both.

Associations for: Use case sequence

links (1,1) :: Use_case :: is_linked (0,n)

Attributes of: Use case sequence

history : CompoundHx

This is a compound data type that holds the id and previous_ID for history, and the first_ver and last_ver for versioning.

sequence : Integer

The order in which the use case participates in the Storyboard.

Class: V23 data type

Associated with: **Attribute**
V23_fields

Description of: V23 data type

The definition of the data type as specified in Figure 2.2, Chapter 2 of HL7 Version 2.3.

Associations for: V23 data type

typed (0,n) :: Attribute :: had_V23_type (1,1)

Provides an indication of the data type used in Version 2.x for a particular attribute, if such prior usage has been identified.

types (0,n) :: V23_fields :: is_of_type (1,1)

Expresses that each field in V2.3 is typed with a data type.

Attributes of: V23 data type

data_type_category : String

The type of data type, such as alphanumeric, or chapter-specific.

data_type_code : String

The two- or three-character code for the data type. e.g. CM

data_type_name : String

The name for the data type as specified in the HL7 Chapter 2 table.

notes_format : String

The format of the data type, particularly for compound data types.

Class: V23 field segment

Associated with: **V23_fields**
V23_segments

Description of: V23 field segment

Links the fields in HL7 V2.3 to the segments in which those fields appear. Source is the HL7 data dictionary.

Associations for: V23 field segment

positions (1,1) :: V23_fields :: populate (0,n)

Links each field to the segment(s) in which it is used and the sequential position in which it appears.

is_in (1,1) :: V23_segments :: contains (1,n)

Aggregates the fields that make up each segment.

Attributes of: V23 field segment

sequence : Integer

The sequence number at which the element or field appears in the segment.

Class: V23_fields

Associated with: **Attribute**
 V23_data_type
 V23_field_segment

Description of: V23_fields

Contains all of the fields (data elements) specified in HL7 Version 2.x, as captured in the data dictionary published by HL7.

Associations for: V23_fields

is_source_for (0,n) :: Attribute :: based_on (0,n)

Provides a linkage for an information model attribute to its equivalent version 2.x field, if such linkage exists and has been identified.

is_of_type (1,1) :: V23_data_type :: types (0,n)

Expresses that each field in V2.3 is typed with a data type.

populate (0,n) :: V23_field_segment :: positions (1,1)

Links each field to the segment(s) in which it is used and the sequential position in which it appears.

Attributes of: V23_fields

description : DescriptiveText

The description of this field.

element : Integer

The unique numeric identifier assigned by HL7 for this field.

field_name : String

The name of the field in the HL7 Data Dictionary.

table : Integer

The number of the HL7 table that provides values for this field, if the field is table-based.

Class: V23_segments

Associated with: **Attribute**
 V23_field_segment

Description of: V23_segments

Contains all of the segments specified in HL7 Version 2.3, as captured in the data dictionary published by HL7.

Associations for: V23_segments

source_of (0,n) :: Attribute :: stems_from (0,n)

Many attributes are traced to equivalent content in HL7 Version 2.x. This connection is secondary to the path that traces an attribute to an HL7 field to a segment. It is provided for modelers who wish to specify particular segments for information model attributes.

contains (1,n) :: V23_field_segment :: is_in (1,1)

Aggregates the fields that make up each segment.

Attributes of: V23_segments

name : String

The name of the segment.

segment : String

The three-character segment identifier.

Class: Version_MET

Subtype of: **Message_element_type**

Associated with: **Choice_branch**

Description of: Version_MET

Provides for version-specific METs for inter-version compatibility. When it is used, all of its branches will be included in the message.

Associations for: Version_MET

has_choices (1,n) :: Choice_branch :: is_choice_for (0,1)

Each branch of a Version_MET includes a type. All of these choices are used in a Choice MEI. A choice branch must be part of a version MET or a choice MET, but not a part of both.

Class: Vocabulary domain

Supertype of: **Code_subsystem**
 Code_system
 Derivative_domain
 Enumerated_domain

Is part of: **Domain_version**

Associated with: **Attribute_domain_constraint**
 Coded_term
 Derivative_domain
 Enumerated_domain
 HMD_domain_constraint
 LOINC_link
 MIM_attribute_domain_constraint
 RMIM_attribute_domain_constraint
 Vocabulary_domain
 Vocabulary_domain
 Vocabulary_domain
 Vocabulary_domain

Description of: Vocabulary domain

A vocabulary domain is a specification of the allowable values for an attribute or component of a composite data type that has a coded datatype assigned to it. The vocabulary domain may be as simple as a reference to a table of enumerated values or it can be a collection of predicate statements.

Every collection of terms is a vocabulary domain, including the one element set, the empty set, an enumeration of a few terms, and all huge vocabulary systems, their subsystems and derivatives.

Composition for: Vocabulary domain

in_version (1,1) :: Domain_version :: has (0,n)

Associations for: Vocabulary domain

is_constraint (0,n) :: Attribute_domain_constraint :: links_domain (1,1)

has_set_of (0,n) :: Coded_term :: element_of (1,1)

This relationship indicates that a vocabulary domain is a set of terms and every term belongs to one vocabulary domain.

Through subsystems and derived vocabulary domains, any term can be member of more than one vocabulary domain.

is_operand_for (0,n) :: Derivative_domain :: has_operands (2,n)

Relationship between a derivative, its operator, and the operands (other domains) to which the operator is applied.

includes (0,n) :: Enumerated_domain :: populates (1,1)

An enumeration includes terms from a particular domain. In the case where the enumeration is the entire domain, this relationship is self-referential.

is_constraint (0,n) :: HMD_domain_constraint :: links_domain (1,1)

equates_to (0,n) :: LOINC_link :: links_domain (0,1)

is_constraint (0,n) :: MIM_attribute_domain_constraint :: links_domain (1,1)

is_constraint (0,n) :: RMIM_attribute_domain_constraint :: links_domain (1,1)

referred_in (0,n) :: Vocabulary_domain :: refers_to (0,n)

This "see also" link is used in conjunction with the comments field to direct the interested reader to other codes..

refers_to (0,n) :: Vocabulary_domain :: referred_in (0,n)

This "see also" link is used in conjunction with the comments field to direct the interested reader to other codes..

updated_by (0,n) :: Vocabulary_domain :: updates (0,1)

This is used for versioning. Actually the version numbering is less important than the maintenance of these "updates" links between versions.

updates (0,1) :: Vocabulary_domain :: updated_by (0,n)

This is used for versioning. Actually the version numbering is less important than the maintenance of these "updates" links between versions.

Attributes of: Vocabulary domain

description : DescriptiveText

A textual description of the vocabulary domain and its purpose

edit_note : DescriptiveText

Editor's notes for the domain.

id : IdentifierString

A unique, sequentially assigned number that identifies a vocabulary domain.

internal_ind : CodedElement

Indicates whether the domain is one that is maintained internally by HL7 in the primitive vocabulary domain enumeration tables, or whether the domain refers to a set maintained by an external organization.

name : String

A unique textual name for the vocabulary domain. The name is created using mixed case object oriented style names, without the use of white space or special punctuation. The name is generally singular. This name is used when the vocabulary domain is referenced by other vocabulary domain definitions. Examples of acceptable names are: Gender, OrderType, PatientType, AbnormalFlag, etc.

The name may be determined by the organization defining the system which includes this domain.

realm_of_use : Enumerated

A coded element that indicates the country/affiliate/jurisdiction for which this domain specification applies. This term allows multiple jurisdiction-specific domains to be related under a single over-arching domain.

The values for the realms of use column come from the RealmOfUse vocabulary domain.

status : CodedElement

The status of the item. The values for Status come from vocabulary domain EditStatus. Some values for status are Proposed, Rejected, Active, Obsolete, and Inactive.

version : VersionNumber

The version of the domain as assigned by HL7 or by the organization defining the system of which this domain is a part.

version_out : String

The version number of the table at which this entry was deleted. A blank Vout value means that the row continues to exist in the current version of the table.

Data type definitions in: **HL7_V3_Meta-Model**

Data type: **Boolean : Boolean**

Is a Primitive Data Type

Description of: **Boolean**

Boolean data

Data type: **CodedElement : CodedElement**

Is a Primitive Data Type

Description of: **CodedElement**

Coded data

Data type: **CompoundHx : CompoundHx**

Is a Composite Data Type

Description of: **CompoundHx**

This set of components is assigned to one attribute of most meta-model elements. These components serve to track the history of each element and designate the models in which the element is valid.

Components of: **CompoundHx**

firstVer : String

This component contains the model unique identifier (modelID) of the first model version in which this element was defined.

hxID : Integer

This component is used to track the version history of each element of the model. It contains the unique element identifier assigned to each model element. The values are assigned in the repository. Modelers should never change these values or assign new ones, but they may copy them to indicate element history.

lastVer : String

This component contains the model unique identifier (modelID) of the model for which this element ceased to be valid. A blank lastVer value means that the element is valid in the most recent HL7 models. If this value is valued, the element is no longer a member of the current RIM. Since model identifiers are monotonically increasing, a given element is valid from the model identified by firstVer up to but not including the model identified by lastVer.

prevHxID : Integer

If an element of the meta-model derives from a previously defined element, this component will be valued. It contains the unique identifier of the element's predecessor,

Data type: **Date : Date**

Is a Primitive Data Type

Description of: **Date**

Date data.

Data type: **DateTime : DateTime**

Is a Primitive Data Type

Description of: **DateTime**

DateTime data.

Data type: **DescriptiveText : DescriptiveText**

Is a Primitive Data Type

Description of: **DescriptiveText**

In most instances the information to be kept about model components includes provision for a textual description of the component. Experience in documenting such models has shown the value of structuring these descriptions in order to provide for reference to external documents, identification of open issues, explanation of modeling rationale, etc. Therefore, descriptions of model components shall be of type DescriptiveText, as follows.

A paragraph that is part of the regular description shall not begin with one of the reserved phrases. Paragraphs that begin with a reserved phrase are used to capture

comments about the rationale for modeling, to capture open issues and to express external references. The reserved phrases and their usage are shown below:

Reserved phrase "Rationale:" allows the modeler to document the rationale or justification for the specification of a particular element. It may occupy one or more paragraphs, but only one modeling rationale component should appear for any given model element. The first paragraph of the rationale must begin "Rationale:" The rationale will continue to the end of the description or until another reserved phrase is encountered.

Reserved phrase "OpenIssue:" allows the modeler to identify and discuss any open issues that remain to be resolved with respect to the model element. It may occupy one or more paragraphs, and there may be multiple open issues for a model element. The first paragraph of each open issue must begin with "OpenIssue:" The open issue will continue to the end of the description or until another reserved phrase is encountered.

Reserved phrase "ExtRef:" provides the specification of a reference to an external document, either by name or by a URL reference. Multiple external references may be contained in a given description. The external reference must be a single paragraph that starts with "ExtRef:" and must either be the final paragraph of the description or it must be followed by another reserved phrase paragraph.

Data type: **Enumerated : Enumerated**

Is a Primitive Data Type

Description of: **Enumerated**

Enumerated data

Data type: **IdentifierString : IdentifierString**

Is a Primitive Data Type

Description of: **IdentifierString**

Various data model elements in the use case model and interaction model are required to have identifiers that are unique throughout the model. These elements will be specified as being represented by an IdentifierString.

Elements that use these identifiers are: application role, interaction, message, scenario, scenario example, trigger event, and use case.

An IdentifierString is a string that contains no embedded spaces and that is built from a limited character set. The IdentifierString may include any number of the following characters: upper or lower case alphabetic characters (A-Z and a-z); the digits (0-9); the dot character (.); the hyphen character (-); and the underscore character (_). These characters may be in any order, except that the first character of the IdentifierString shall be either a digit or an upper case alphabetic character.

Note: Because the dot character (.) is an allowed member of an IdentifierString, the IdentifierString cannot be used in defining fully qualified names for elements.

Data type: Integer : Integer

Is a Primitive Data Type

Description of: Integer
Integer data.

Data type: MultiplicityString : MultiplicityString

Is a Primitive Data Type

Description of: MultiplicityString

A set of values and value ranges including the minimum and maximum occurrence are required for associations and aggregations in the meta-model. A MultiplicityString shall be used to represent this set in both the literary and graphical expressions of the model. The MultiplicityString is a constrained string. It is built according to the following rules:

1. A MultiplicityString shall have at least a minimum and a maximum value.
2. A MultiplicityString may also include an open ended range at the upper end.
3. A MultiplicityString shall be expressed either as the single element "1" (the numeral one) or as a pair of elements separated by an ellipsis (..).
4. The elements making up a MultiplicityString shall be either zero, a positive integer, or the character "*".
5. If the character "*" appears in a MultiplicityString, there must be only a single occurrence, and that occurrence shall represent the set of all positive integers that are greater than the largest of the other integers in the same MultiplicityString.

6. The minimum value for the multiplicity shall be the smallest integer in the MultiplicityString, and may not be the character "*".

7. The maximum value for the multiplicity shall be the largest integer in the MultiplicityString, and must be greater than zero.

8. The elements making up a MultiplicityString should be ordered in ascending order, but are not required to be.

Data type: NameString : NameString

Is a Primitive Data Type

Description of: NameString

The NameString is a string that contains no embedded spaces and that is built from a limited character set. The NameString may include any number of the following characters: upper or lower case alphabetic characters (A-Z and a-z); the digits (0-9); the hyphen (-), and the underscore character (_). These characters may be in any order, except that the first character of the NameString shall be an alphabetic character.

The appropriate use of upper and lower case characters, and the inclusion of special characters allow data modelers to create easily readable strings for the noun- and verb-phrases required for many of these elements. (Examples might include "is_ordered_by" or "HealthCarePractitioner" or NameString.)

For clarity of reading, the initial character of a NameString should be lower case when used for names of attributes, labels of relationships, and labels for state transitions, and should be upper case in all other uses.. No matter what conventions are used with respect to capitalization, when NameStrings are compared for uniqueness, all alphabetic characters shall be treated as though they are lower case.

Data type: String : String

Is a Primitive Data Type

Description of: String
String data.

Data type: VersionNumber : VersionNumber

Is a Primitive Data Type

Description of: **VersionNumber**

A version number is a string comprised solely of the digit characters and the dot (.) character.

Stewardship & DIMs in: **HL7_V3_Meta-Model**

Data type categories for: **HL7_V3_Meta-Model**

Data type category: **MET_Metamodel_data_types**

Specifies particular data types used in the meta-model. Other data types such as String, Boolean, Integer and Enumerated are not listed here.

Contains data types: **Boolean**
CodedElement
CompoundHx
Date
DateTime
DescriptiveText
Enumerated
IdentifierString
Integer
MultiplicityString
NameString
String
VersionNumber