Date: July 2013

UML Profile for MIF Static Models

Version 1.0

Tooling Challenge Repository: http://gforge.hl7.org/gf/project/toolchlge_018/

Related OMG Specifications:

Unified Modeling Language http://www.omg.org/spec/UML
Object Constraint Language http://www.omg.org/spec/OCL
Meta Object Facility Core http://www.omg.org/spec/MOF
MOF 2 XMI Mapping http://www.omg.org/spec/XMI

Related W3C Specifications:

Extensible Markup Language (XML) http://www.w3.org/XML/

Related HL7 Standards

HL7 Model Interchange Format
http://www.hl7.org/implement/standards/product_brief.cfm?product_id=101
HL7 Reference Information Model http://www.hl7.org/implement/standards/rim.cfm
HL7 Version 3 Normative Edition, 2012
http://www.hl7.org/implement/standards/product_brief.cfm?product_id=282
HL7 Version 3 Clinical Document Architecture
http://www.hl7.org/implement/standards/product_brief.cfm?product_id=7
Data Types - Abstract Specification
http://www.hl7.org/implement/standards/product_brief.cfm?product_id=264

The following contributors have played different roles in driving the development of this specification

Submitters
Antonio Villegas, PhD (UPC BarcelonaTech)
Antoni Olivé, Professor (UPC BarcelonaTech)
Advisors
Josep Vilalta (HL7 Spain)
Alberto Sáez (HL7 Spain)
Diego Kaminker (HL7 Argentina)
HL7 Spain Technical Committee
Sponsors
vico open modeling - http://vico.org

THIS PROFILE IS PROVIDED TO YOU "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EXPRESS, IMPLIED, OR OTHERWISE, INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT SHALL HL7 INTERNATIONAL BE LIABLE TO YOU OR ANYONE ELSE FOR ANY DIRECT, SPECIAL, INCIDENTAL, INDIRECT, OR CONSEQUENTIAL DAMAGES OF ANY KIND, OR ANY DAMAGES WHATSOEVER, INCLUDING WITHOUT LIMITATION, LOSS OF PROFIT, LOSS OF USE, SAVINGS OR REVENUE, OR THE CLAIMS OF THIRD PARTIES, WHETHER OR NOT DOL OR THE U.S. GOVERNMENT HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH LOSS, HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, ARISING OUT OF OR IN CONNECTION WITH THE POSSESSION, USE, OR PERFORMANCE OF THIS SOFTWARE.

USE OF SPECIFICATION - TERMS, CONDITIONS & NOTICES

The material in this document details an specification in accordance with the terms, conditions and notices set forth below. This document does not represent a commitment to implement any portion of this specification in any company's products. The information contained in this document is subject to change without notice.

LICENSES

GENERAL USE RESTRICTIONS

Table of Contents

1 Scope 1
1.1 MIF-UML Background 1
1.2 Intended Users of MIF-UML 1
1.3 MIF-UML External Libraries 2
1.4 MIF-UML Transformations 2

2 Conformance 4
2.1 Unified Modeling Language Conformance 4
2.2 Reference Information Model Conformance 4
2.3 Data Types Conformance 5
2.4 Tool Conformance 5

3 Normative References 6

4 Terms and Definitions 7
4.1 Definitions 7
4.2 Acronyms 10

5 Notational Conventions 11
5.1 Keywords for Requirement Statements 11
5.2 Annotations on Example Diagrams 11

6 Additional Information 12
6.1 MIF-UML Introduction and Concepts 12
6.1.1 MIF Static Models 12
6.1.2 UML Extension Mechanism 13
6.1.3 MIF-UML Goals 13
6.2 Proof of Concept 14

7 MIF-UML Modeling Guide 15
7.1 Overview 15
7.2 Modeling a static model 15
7.2.1 Setting up the environment 16
7.2.2 Load HL7 Data Types 18
7.2.3 Import UML Profile for MIF Static Models 20
7.2.4 Static Model 21
7.2.5 Entry point 23
7.2.6 Classes and relationships 24
7.2.7 Adding attributes 28
7.2.8 CMETs 31
7.2.9 Choices 32
7.2.10 Export to XML Metadata Interchange (XMI) 33
7.3 Improvements 35
7.3.1 Unified Modeling Language (UML) .... **¡Error! Marcador no definido.**
7.3.2 Model Interchange Format (MIF) 35
7.3.3 Enterprise Architect (EA) 35

8 MIF-UML Profile Reference 38
8.1 Overview 38
8.2 Foundational Area 38
8.2.1 Overview 38
8.2.2 «stereotype» mifAccess 40
8.2.3 «stereotype» mifAccount 40
8.2.4 «stereotype» mifAct 40
8.2.5 «stereotype» mifActHeir 40
8.2.6 «stereotype» mifActRelationship 40
8.2.7 «stereotype» mifContainer 40
8.2.8 «stereotype» mifContextStructure 41

8.2.9 «stereotype» mifControlAct ... 41
8.2.10 «stereotype» mifDevice ... 41
8.2.11 «stereotype» mifDeviceTask ... 41
8.2.12 «stereotype» mifDiagnosticImage ... 41
8.2.13 «stereotype» mifDiet ... 41
8.2.14 «stereotype» mifDocument ... 41
8.2.15 «stereotype» mifEmployee ... 41
8.2.16 «stereotype» mifEntity ... 42
8.2.17 «stereotype» mifEntityHeir ... 42
8.2.18 «stereotype» mifExposure ... 42
8.2.19 «stereotype» mifFinancialContract ... 42
8.2.20 «stereotype» mifFinancialTransaction ... 42
8.2.21 «stereotype» mifInfrastructureRoot ... 42
8.2.22 «stereotype» mifInvoiceElement ... 43
8.2.23 «stereotype» mifLanguageCommunication ... 43
8.2.24 «stereotype» mifLicensedEntity ... 43
8.2.25 «stereotype» mifLivingSubject ... 44
8.2.26 «stereotype» mifManagedParticipation ... 44
8.2.27 «stereotype» mifManufacturedMaterial ... 44
8.2.28 «stereotype» mifMaterial ... 44
8.2.29 «stereotype» mifNonPersonLivingSubject ... 44
8.2.30 «stereotype» mifObservation ... 44
8.2.31 «stereotype» mifOrganization ... 44
8.2.32 «stereotype» mifParticipation ... 45
8.2.33 «stereotype» mifPatient ... 45
8.2.34 «stereotype» mifPatientEncounter ... 45
8.2.35 «stereotype» mifPerson ... 45
8.2.36 «stereotype» mifPlace ... 45
8.2.37 «stereotype» mifProcedure ... 45
8.2.38 «stereotype» mifPublicHealthCase ... 45
8.2.39 «stereotype» mifQualifiedEntity ... 46
8.2.40 «stereotype» mifRole ... 46
8.2.41 «stereotype» mifRoleHeir ... 46
8.2.42 «stereotype» mifRoleLink ... 46
8.2.43 «stereotype» mifSubstanceAdministration ... 46
8.2.44 «stereotype» mifSupply ... 46
8.2.45 «stereotype» mifWorkingList ... 47
8.3 Message Communications Control Area ... 47
8.3.1 Overview ... 47
8.3.2 «stereotype» mifAcknowledgement ... 47
8.3.3 «stereotype» mifAcknowledgementDetail ... 47
8.3.4 «stereotype» mifAttachment ... 48
8.3.5 «stereotype» mifAttentionLine ... 48
8.3.6 «stereotype» mifBatch ... 48
8.3.7 «stereotype» mifCommunicationForm ... 48
8.3.8 «stereotype» mifMessage ... 48
8.3.9 «stereotype» mifParameter ... 48
8.3.10 «stereotype» mifParameterItem ... 48
8.3.11 «stereotype» mifParameterList ... 49
8.3.12 «stereotype» mifQueryAck ... 49
8.3.13 «stereotype» mifQueryByParameter ... 49
8.3.14 «stereotype» mifQueryContinuation ... 49
8.3.15 «stereotype» mifQueryEvent ... 49
8.3.16 «stereotype» mifQuerySpec ... 49
8.3.17 «stereotype» mifSortControl ... 49
8.3.18 «stereotype» mifTransmission ... 49
8.3.19 «stereotype» mifTransmissionRelationship ... 50
8.4 Special Constructs Area ... 50
8.4.1 Overview ... 50
8.4.2 «stereotype» mifAttribute ... 51

8.4.3 «stereotype» mifChoice 52
8.4.4 «stereotype» mifCMET 52
8.4.5 «stereotype» mifEntryPoint 52
8.4.6 «stereotype» mifPlayer 52
8.4.7 «stereotype» mifProperty 53
8.4.8 «stereotype» mifScoper 53
8.4.9 «stereotype» mifStaticModel 53
8.5 Enumerations 54
8.5.1 Overview 54
8.5.2 «enumeration» AffiliateKind 55
8.5.3 «enumeration» ArtifactKind 56
8.5.4 «enumeration» CodingStrengthKind 56
8.5.5 «enumeration» ConformanceKind 57
8.5.6 «enumeration» PackageKind 57
8.5.7 «enumeration» PackageRootKind 57
8.5.8 «enumeration» PublishingDomainKind 57
8.5.9 «enumeration» PublishingSectionKind 60
8.5.10 «enumeration» PublishingSubSectionKind 60
8.5.11 «enumeration» StaticModelRepresentationKind 61
8.6 Metaclasses 61
8.6.1 Overview 61
8.6.2 «metaclass» Class 61
8.6.3 «metaclass» Association 63
8.6.4 «metaclass» Package 66
8.6.5 «metaclass» Property 66
8.6.6 «metaclass» Attribute 67

9 MIF-UML Transformation Reference 68
9.1 Overview 68
9.2 Static Model 68
9.3 Entry Point 69
9.4 Class 71
9.5 Attribute 73
9.6 Association 76
9.7 Choice 80
9.8 CMET 82

Preface

HL7

Founded in 1987, Health Level Seven International (HL7) is a not-for-profit, ANSI-accredited standards developing organization dedicated to providing a comprehensive framework and related standards for the exchange, integration, sharing, and retrieval of electronic health information that supports clinical practice and the management, delivery and evaluation of health services. HL7's 2,300+ members include approximately 500 corporate members who represent more than 90% of the information systems vendors serving healthcare.

HL7 Mission

HL7 provides standards for interoperability that improve care delivery, optimize workflow, reduce ambiguity and enhance knowledge transfer among all of our stakeholders, including healthcare providers, government agencies, the vendor community, fellow SDOs and patients. In all of our processes we exhibit timeliness, scientific rigor and technical expertise without compromising transparency, accountability, practicality, or our willingness to put the needs of our stakeholders first.

HL7 Tooling Challenge

HL7 is holding a contest to encourage the development of HL7 tools. This document describes the specification of a UML Profile for MIF Static Models as a particular submission to the HL7 2012-2013 Tooling Challenge.

Typographical Conventions

The type styles shown below are used in this document to distinguish programming statements from ordinary English. However, these conventions are not used in tables or section headings where no distinction is necessary.

Times/Times New Roman - 10 pt.: Standard body text

Helvetica/Arial - 10 pt. Bold: OMG Interface Definition Language (OMG IDL) and syntax elements.

`Courier - 10 pt. Bold:` Programming language elements.

`Courier - 10 pt.:` Constraints and operations.

Helvetica/Arial - 10 pt: Exceptions

NOTE: Terms that appear in italics are defined in the glossary (section 4). Italic text also represents the name of a document, specification, or other publication.

Issues

The reader is encouraged to report any technical or editing issues/problems with this specification to http://www.omg.org/report_issue.htm.

1 Scope

1.1 MIF-UML Background

The Health Level Seven International (HL7) is a not-for-profit, ANSI-accredited standards developing organization dedicated to providing standards for the exchange, integration, sharing, and retrieval of electronic health information that supports clinical practice and the management, delivery, and evaluation of health services. HL7 standards enable semantic interoperability between almost all institutions, domains, and fields of healthcare. With HL7, all important communication tasks of a healthcare provider (hospitals, clinics, primary care centers, and other service delivery points) can be handled and the efficiency of the communication process can be improved.

HL7 Version 3 is an specific healthcare interoperability standard within HL7 designed to handle most if not all healthcare communications, using a relatively small set of constructs. The standard specifies storyboards, trigger events, and interactions between clinical applications to maintain a common representation of healthcare messaging tasks such as the update of a patient billing account, the registration of a new observation in a diagnosis, or the submission of a blood test request to a laboratory.

HL7 V3 and its model-based development framework enable semantic interoperability between different applications that adopt the same healthcare information models. The standard indicates the information each kind of clinical message or document must contain and how it is structured through the definition of information models based on the RIM. An information model is a structured specification of the information within a specific domain of interest. It expresses the concepts of information required and the properties of these concepts, including attributes, relationships, constraints, and states. The standard defines different types of information models to represent documents and messages to exchange in several healthcare domains.

The formal specification and documentation of all HL7 V3 artifacts is stored in a model repository. The Model Interchange Format (MIF) is a set of XML formats used to support the storage and exchange of HL7 version 3 artifacts as part of the HL7 Development Framework. It also encodes the validation rules around what constitutes valid HL7 V3 artifacts and provides documentation of what the semantics of the different parts of an HL7 V3 artifact are. The MIF can be transformed into derived forms such as UML/XMI or OWL.

The Unified Modeling Language (UML) is a standard general-purpose graphical modeling language in the field of software engineering that provides a visual notation that is widely known by the software industry and engineering community. UML is used to specify, visualize, modify, construct, and document software artifacts of a system under development.

1.2 Intended Users of MIF-UML

The management of the healthcare information models that define the messages and documents of the standard and the complexity of the development framework to implement them in an interoperability scenario are central aspects for the success of HL7 V3. In the software engineering literature, most of the aspects related to information models have been extensively studied over the years. Consequently, several existing tools and methodologies from this field may be applied and adapted to obtain, refine, understand, and use information models in order to fulfill the requirements of the healthcare domain.

One of the key goals for MIF-UML is to allow modelers and developers to apply MIF-UML with minimal effort in order to create new models or change existing models and ultimately to produce MIF artifacts directly in UML. The aim of this specification is to increase the number of common UML modeling tools that are compatible with the MIF format.

An information model is useless unless its purpose is properly understood by the specialists that have to work with it. This specification describes a UML profile that correctly describes MIF (Model Interchange Format) static models to the extent allowed by the UML language. The main benefits of providing a standard UML version of the healthcare information models within HL7 V3 are related to understandability, tool-support, and analyzability of the standard. The usage of UML as the modeling language of the HL7 V3 models defined in MIF format allows all members of the software engineering community to understand the standard without requiring further training. Furthermore, the healthcare community can benefit from existing modeling tools and methodologies from that community.

Nowadays, there are methods and applications to generate a significant part of the final code of a software system from its UML information model, or even interpret this model and make it executable. It is possible to check the consistency of UML models in order to solve many issues in the software development process. A UML model can be translated into a narrative description that enables non-experts or business people to understand its semantics. Also, there is a large amount of tools to visualize UML models that integrate other functionalities and help users to deal with them. In addition, there is a wide community of UML experts that may collaborate and contribute to the improvement of the HL7 V3 information models.

1.3 MIF-UML External Libraries

A central tenet of MIF-UML is reuse. To this end this specification makes use of the HL7 Version 3 Data Types Abstract Specification, Release 2 (http://www.hl7.org/implement/standards/product_brief.cfm?product_id=264). It provides the semantic definitions for the data types used in the creation of HL7 V3 specifications. These "abstract" semantic definitions are also able to be used as constraints in the creation of implementation guides, and implementation technology specifications that enable actual exchange of data are based on these definitions.

Additionally, this specification makes use of the HL7 vocabularies defined in the HL7 V3 standard (http://www.hl7.org/implement/standards/product_brief.cfm?product_id=186) when instantiating common static models. Modern health care communications and data storage makes heavy use of encoded information. In HL7, this is referred to as vocabulary. The HL7 standards define several different types of objects that implement various characteristics of vocabulary. Whereas other elements of the HL7 standards are primarily concerned with structure, vocabulary deals with content.

The HL7-defined vocabulary tables that have been developed for coded class attributes are stored in the HL7 repository, from which a number of views have been extracted to produce the HL7 Vocabulary Listings for the HL7 Reference Information Model (RIM). The views are presented in table format and include the HL7 Concept Domains, Code Systems (including, HL7-maintained systems and external systems referenced by HL7), HL7-defined Value Sets, and a Cross Reference between Concept Domains and Coded Attributes.

1.4 MIF-UML Transformations

The UML Profile for MIF static models provides an easy way of constructing MIF models directly in UML. The profile consists of three areas, as shown in Figure 1. Each area is a subset of UML 2.4 constructs that are extended by UML stereotypes. The UML extensions define the MIF concepts without an analogous representation in UML. All MIF-UML models use the standard XMI exchange format specified for UML 2.4 and may exchange MIF models between conforming UML tools.

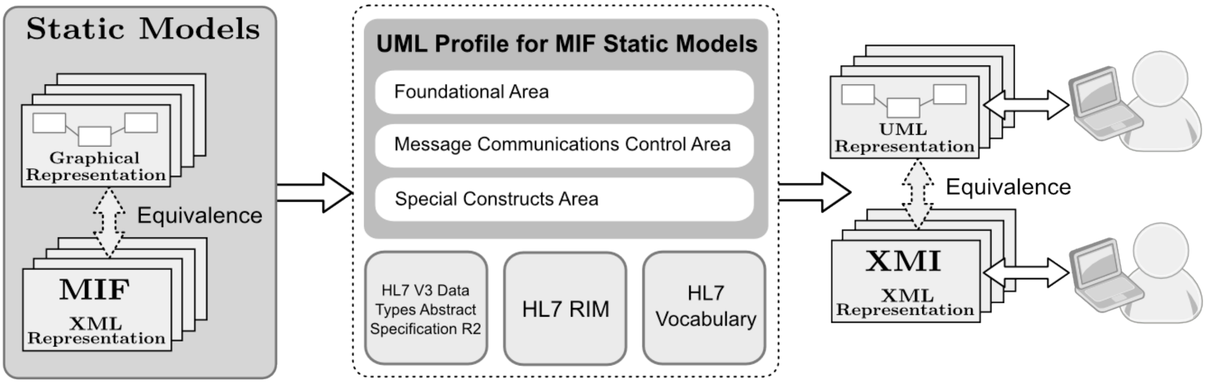


Figure 1. Components of the MIF-UML Specification.

The components of Figure 1 have distinct purposes and behaviors:

- Static Models, which are the initial models that can be represented graphically using the specific modeling language provided by HL7 V3. An equivalent representation in MIF format is also provided in the HL7 standard.
- UML Profile for MIF Static Models
    - The Foundational Area, which contains the stereotypes that represent the concepts in the HL7 Reference Information Model (RIM), including acts, roles, entities, and participations.
    - The Message Communications Control Area, which contains the stereotypes that represent the technical infrastructure of HL7, including messaging and other components.
    - The Special Constructs Area, which contains the stereotypes that provide ways to represent specific constructs from HL7 such as entry points, CMETs, and choices.
- HL7 V3 Data Types Abstract Specification R2, which contains the data types referenced within the MIF static models.
- HL7 Reference Information Model (RIM), which contains the foundational classes of the HL7 standards.
- HL7 Vocabulary, which contains the HL7 Concept Domains, Code Systems, and HL7-defined Value Sets referenced within the MIF static models.
- UML Models, which are the UML representation of the input Static Models in MIF format. It is possible to work with these models directly with their graphical representation in UML through a CASE tool like Enterprise Architect, or with their XMI representation as XML files.

As indicated in Figure 1, this structure for the MIF-UML profile provides direct “entry points” for both UML modelers who are primarily domain oriented and XMI users/processors that are primarily technically oriented. However, it also defines a clear relationship between these abstraction levels, allowing modelers to also move flexibly between them using UML modeling tools like Enterprise Architect or XML editors.

2 Conformance

This specification defines the following conformance points (also referred to as conformance targets):

- Unified Modeling Language Conformance
- Reference Information Model Conformance
- Data Types Conformance
- Tool Conformance

2.1 Unified Modeling Language Conformance

The OMG UML specification (see UML v.2.4.1, Chapter 2) defines four compliance levels:

- Level 0 (L0). This compliance level is formally defined in the UML Infrastructure. It contains a single language unit that provides for modeling the kinds of class-based structures encountered in most popular object-oriented programming languages. As such, it provides an entry-level modeling capability. More importantly, it represents a low cost common denominator that can serve as a basis for interoperability between different categories of modeling tools.

- Level 1 (L1). This level adds new language units and extends the capabilities provided by Level 0. Specifically, it adds language units for use cases, interactions, structures, actions, and activities.

- Level 2 (L2). This level extends the language units already provided in Level 1and adds language units for deployment, state machine modeling, and profiles.

- Level 3 (L3). This level represents the complete UML. It extends the language units provided by Level 2 and adds new language units for modeling information flows, templates, and model packaging.

Each UML model using the UML Profile for MIF Static Models satisfies the L3 conformance level, including all the constraints defined in the UML Superstructure specification.

2.2 Reference Information Model Conformance

The Health Level Seven (HL7) Reference Information Model (RIM) is a static model of health and health care information as viewed within the scope of HL7 standards development activities. It is the combined consensus view of information from the perspective of the HL7 working group and the HL7 international affiliates.

The HL7 RIM describes the core classes for the healthcare subject areas, the attributes for each of these classes as well as their associations and specializations. The HL7 RIM identifies six major "high-level" classes that are fundamental to understand the world of healthcare information. At the highest level of abstraction, healthcare consists of a series of clinical Acts performed to, by, on behalf of, utilizing, or in some way involving, one or more instances of a Participating Entity-in-a-Role ("the entity John Smith in the role of Patient participates as the Subject of an act of Observation").

Section 8 of this specification describes the stereotypes of the UML profile for MIF static models. The profile defines, among others, a stereotype for any class described in the RIM. The attributes of a class with one of these RIM-based stereotypes applied should fulfill the following properties:

- The name and type of the attribute must be the same as the name and type defined for one of the attributes of the class in the RIM that the stereotype represents. As an example, the RIM class Patient owns the attribute veryImportantPersonCode:CD[0..1], and inherits all the attributes of its ascendant RIM class Role (14 attributes owned by Role, in this case). A class in a static model with the stereotype mifPatient from the MIF profile should only contain the attributes from the set of attributes of Patient or Role in the RIM. No additional attributes should be declared.

- Attributes may be constrained by more specific minimum and maximum cardinality value that its definition in the RIM. For example, a class with the stereotype mifPatient may own the attribute veryImportantPersonCode::CD[1..1] which is defined in the RIM with cardinality [0..1]. The cardinality [1..1] is more specific than [0..1].

The constraints expressed in OCL in section 8 of this specification deal with the Reference Information Model conformance.

2.3 Data Types Conformance

The RIM is composed of classes with attributes which are data elements. A data element is a unit of data for which the definition, identification, representation, and permissible values are specified by means of a set of attributes. Every data element has a data type. The data type defines the set of valid values that can be assigned to a data element and their meaning (semantics). Meaningful exchange of data requires that we know the definition of values so exchanged. This is true for complex "values" such as timing specifications as well as for simpler values such as character strings or integer numbers. An instance of a data type is a data value.

Each UML model using the UML Profile for MIF Static Models must use the data types defined in the HL7 V3 Data Types Abstract Specification R2. This specification uses Unified Modeling Language (UML) class diagrams to graphically represent how data types relate to each other. Data types are shown as UML classes using the short name for the type. Properties of types are shown as UML operations. Generic types are shown as UML parameterized classes.

2.4 Tool Conformance

This specification defines tool conformance in terms of conformance points. A tool developer may assert that a given tool supports one or more of the conformance points defined in this specification as follows:

- The tool allows the application of the UML Profile for MIF Static Models to a UML model. It includes the application of the stereotypes defined in the profile.
- The tool conforms to UML 2.4.1 and provides ways to create valid UML constructors and models.
- The tool conforms to the constraints and annotations defined in the HL7 Reference Information Model (RIM).
- The tool uses the data types from the HL7 V3 Data Types Abstract Specification R2 to indicate the type of the attributes in UML models.
- The tool includes the HL7 Vocabularies and allows the association of concept domains to properties with conformance to the RIM.
- The tool evaluates the correctness of UML models with respect to the constraints defined in the UML Profile for MIF Static Models.
- The tool processes a graphical UML model and produces its equivalent XMI representation.
- The tool consumes the XMI representation of a model and produces its equivalent graphical UML representation.

At some time in the future tools may be developed that can verify these assertions with some degree of confidence. This specification uses Enterprise Architect as modeling tool.

3 Normative References

The following normative documents contain provisions which, through reference in this text, constitute provisions of this specification. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply.

[CDA] HL7 Version 3 Clinical Document Architecture, CDA® Release 2, (http://www.hl7.org/implement/standards/product_brief.cfm?product_id=7)

[HL7DT] HL7 Version 3 Standard: Data Types - Abstract Specification, Release 2 (http://www.hl7.org/implement/standards/product_brief.cfm?product_id=264)

[HL7V3] Health Level Seven Version 3 (V3) Normative Edition, HL7 V3, (http://www.hl7.org/implement/standards/product_brief.cfm?product_id=186)

[MIF] HL7 Version 3 Standard: Model Interchange Format, Release 1, Version 2.2.0 (http://www.hl7.org/implement/standards/product_brief.cfm?product_id=101)

[MOF] OMG Meta Object Facility (MOF) Core Specification, Version 2.4.1, formal/2011-08-07 (http://www.omg.org/spec/MOF/2.4.1/PDF/)

[OCL] OMG Object Constraint Language (OCL), Version 2.3.1, formal/2012-01-01 (http://www.omg.org/spec/OCL/2.3.1/PDF)

[RFC2119] Key words for use in RFCs to Indicate Requirement Levels, IETF RFC 2119, March 1997 (http://www.ietf.org/rfc/rfc2119.txt)

[RIM] HL7 Reference Information Model (RIM), Version 2.4.1 (http://www.hl7.org/implement/standards/rim.cfm)

[UML] OMG Unified Modeling Language (OMG UML), Superstructure, Version 2.4.1, formal/2011-08-06 (http://www.omg.org/spec/UML/2.4.1/Superstructure/PDF)

[XMI] OMG MOF 2 XMI Mapping Specification, Version 2.4.1. formal/2011-08-09, (http://www.omg.org/spec/XMI/2.4.1/PDF)

[XML] Extensible Markup Language (XML) (http://www.w3.org/XML/)

4 Terms and Definitions

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

4.1 Definitions

Association (UML)

An association describes a set of tuples whose values refer to typed instances. An instance of an association is called a link. A link is a tuple with one value for each end of the association, where each value is an instance of the type of the end.

Attribute (UML)

An attribute defines values that can be attached to instances of a class.

Choice (MIF HL7)

A Choice is a special construction used in MIF static models. Choices are represented as boxes bordered by a dashed line and enclose two or more classes that are part of an inheritance hierarchy (e.g. two or more Roles, two or more Entities, etc.). It is important to note that any annotations or associations connected to the Choice box apply to all classes within it. Associations connected to a specific class within the choice box apply only to that class.

Clinical Document Architecture (CDA)

The HL7 Clinical Document Architecture (CDA) is a document markup standard that specifies the structure and semantics of "clinical documents" for the purpose of exchange. A CDA document is a defined and complete information object that can include text, images, sounds, and other multimedia content. A clinical document is a documentation of clinical observations and services, with the following characteristics:

- Persistence: A clinical document continues to exist in an unaltered state, for a time period defined by local and regulatory requirements (NOTE: There is a distinct scope of persistence for a clinical document, independent of the persistence of any XML-encoded CDA document instance).
- Stewardship: A clinical document is maintained by an organization entrusted with its care.
- Potential for authentication: A clinical document is an assemblage of information that is intended to be legally authenticated.
- Context: A clinical document establishes the default context for its contents.
- Wholeness: Authentication of a clinical document applies to the whole and does not apply to portions of the document without the full context of the document.
- Human readability: A clinical document is human readable.

Class (UML)

A class describes a set of objects that share the same specifications of features, constraints, and semantics.

CMET (MIF HL7)

CMETs (Common Message Element Types) are a work product produced by a particular committee for expressing a common, useful and reusable concept. They are generally "consumed", or used by both the producing committee and other committees. A CMET can be seen as a message type fragment that is reusable by other message types. Any message type, including other CMETs, can reference a CMET. As an example, several committees may require the use of a common concept, that of a person in the role of a patient. A CMET can be defined to express this concept as a message type that clones a role played by a person, with all appropriate attributes. The CMET is then used to uniformly represent the concept for all interested committees.

Data Type (UML)

A data type is a type whose instances are identified only by their value. A data type may contain attributes to support the modeling of structured data types. A typical use of data types would be to represent programming language primitive types or CORBA basic types. For example, integer and string types are often treated as data types.

Entry Point (MIF HL7)

Entry Points designate the class(es) from which the messages begin for a particular model. Entry points are represented as clear boxes with thick black borders. A thick black arrow originates from the Entry Point box and points to a class which is considered the root, or focal class for one or more models. Each Entry Point is named, carries the artifact ID of the R-MIM which originates from it, and contains a brief description.

Enumeration (UML)

An enumeration is a data type whose values are enumerated in the model as enumeration literals.

EnumerationLiteral (UML)

An enumeration literal is a user-defined data value for an enumeration.

eXtended Markup Language (XML)

Extensible Markup Language (XML) is a simple, very flexible text format derived from SGML (ISO 8879). Originally designed to meet the challenges of large-scale electronic publishing, XML is also playing an increasingly important role in the exchange of a wide variety of data on the Web and elsewhere.

Extension (UML)

An extension is used to indicate that the properties of a metaclass are extended through a stereotype, and gives the ability to flexibly add (and later remove) stereotypes to classes.

HL7

Founded in 1987, Health Level Seven International (HL7) is a not-for-profit, ANSI-accredited standards developing organization dedicated to providing a comprehensive framework and related standards for the exchange, integration, sharing, and retrieval of electronic health information that supports clinical practice and the management, delivery and evaluation of health services. HL7's 2,300+ members include approximately 500 corporate members who represent more than 90% of the information systems vendors serving healthcare.

Health Level 7 Version 3 (HL7 V3)

HL7 V3 is a standard for exchanging messages among information systems that implement healthcare applications.

Model Interchange Format (MIF)

The Model Interchange Format (MIF) is a set of XML formats used to support the storage and exchange of artifacts as part of the HL7 version 3 development process. It also encodes the validation rules around what constitutes valid HL7 V3 artifacts and provides documentation of what the semantics of the different parts of an HL7 V3 artifact are.

Model (UML)

A formal specification of the function, structure and/or behavior of an application or system.

Object Constraint Language (OCL)

An adopted OMG standard and formal language used to describe expressions on MOF models. These expressions typically specify invariant conditions that must hold for the system being modeled or queries over objects described in a model. Note that when the OCL expressions are evaluated, they do not have side effects; i.e., their evaluation cannot alter the state of the corresponding executing system. For the purpose of this specification, references to OCL should be considered references to the Object Constraint Language Specification, cited in Normative References, above.

Package (UML)

A package is used to group elements, and provides a namespace for the grouped elements.

Profile (UML)

A profile defines limited extensions to a reference metamodel with the purpose of adapting the metamodel to a specific platform or domain.

Property (UML)

A property is a structural feature. A property owned by a classifier represents an attribute, and it may also represent an association end. It relates an instance of the class to a value or collection of values of the type of the attribute. A property related to an association or its specializations represents an end of the association. The type of property is the type of the end of the association.

Reference Information Model (RIM)

The Health Level Seven (HL7) Reference Information Model (RIM) is a static model of health and health care information as viewed within the scope of HL7 standards development activities. It is the combined consensus view of information from the perspective of the HL7 working group and the HL7 international affiliates. The RIM is the ultimate source from which all HL7 version 3.0 protocol specification standards draw their information-related content.

The RIM provides a static view of the information needs of HL7 V3 standards. It includes class and state-machine diagrams and is accompanied by use case models, interaction models, data type models, terminology models, and other types of models to provide a complete view of the requirements and design of HL7 standards. The classes, attributes, state-machines, and relationships in the RIM are used to derive domain-specific information models that are then transformed through a series of constraining refinement processes to eventually yield a static model of the information content of an HL7 standard.

The RIM is comprised of six "back-bone" classes:

- Act, which represents the actions that are executed and must be documented as health care is managed and provided.
- Participation which expresses the context for an act in terms such as who performed it, for whom it was done, where it was done, etc.
- Entity, which represents the physical things and beings that are of interest to, and take part in health care.
- Role, which establishes the roles that entities play as they participate in health care acts.
- ActRelationship, which represents the binding of one act to another, such as the relationship between an order for an observation and the observation event as it occurs.
- RoleLink, which represents relationships between individual roles.

Stereotype (UML)

A stereotype defines how an existing metaclass may be extended, and enables the use of platform or domain specific terminology or notation in place of, or in addition to, the ones used for the extended metaclass.

XML Metadata Interchange (XMI)

XMI is a widely used interchange format for sharing objects using XML. Sharing objects in XML is a comprehensive solution that builds on sharing data with XML. XMI is applicable to a wide variety of objects: analysis (UML), software (Java, C++), components (EJB, IDL, CORBA Component Model), and databases (CWM). For the purpose of this specification, references to XMI should be considered references to the XML Metadata Interchange (XMI) 2.0 Specification, cited in Normative References, above.

4.2 Acronyms

CDA	Clinical Document Architecture
HL7	Health Level Seven, International
MDA	Model Driven Architecture
MIF	Model Interchange Format
MOF	Meta Object Facility
OCL	Object Constraint Language
RIM	Reference Information Model
UML	Unified Modeling Language
XMI	XML Metadata Interchange
XML	eXtended Markup Language
XSD	XML Schema Definition

5 Notational Conventions

5.1 Keywords for Requirement Statements

The keywords “must,” “must not,” “shall,” “shall not,” “should,” “should not,” and “may” in this specification are to be interpreted as described in RFC 2119.

5.2 Annotations on Example Diagrams

Some of the diagram examples in this specification include explanatory annotations, which should not be confused as being part of the formal UML graphical notation.

In these cases, the explanatory text originates outside the UML diagram boundary, and has an arrow pointing at the feature of the diagram, which is being explained by the annotation. The color rendition of this spec shows these annotations in red.

6 Additional Information

6.1 MIF-UML Introduction and Concepts

6.1.1 MIF Static Models

The MIF models consist of a number of XML files that together define all HL7 V3 artifacts produced by HL7. These schemas define public elements for all artifacts that might reasonably be maintained or published independently. They also define complex types, simple types, groups and other schema structures used to fully define the set of content allowed (and required) in HL7 artifacts.

One of the major purposes of the MIF is documentation. Every complex type, element, attribute and other component definition includes descriptive documentation explaining what function that element provides and what type of data it is intended to represent. The documentation also frequently includes the correspondence of the element to the corresponding concept (if any) in the UML meta-model.

MIF static models contain elements like Choice, CMET or EntryPoint that are specific constructs which do not have equivalent in UML. Choice boxes boxes are bordered by a dashed line and enclose two or more classes that are part of an inheritance hierarchy. Entry points are represented on a MIF model as clear boxes with thick black borders. A thick black arrow originates from the entry point box and points to a class which is considered the root, or focal class in the model. CMETs (Common Message Element Type) are a predefined container of common elements that are re-used for several MIF models in order to avoid repetitions and simplify the model. CMETs can be seen as hyperlinks to other models and are graphically denoted as boxes with a dashed border.

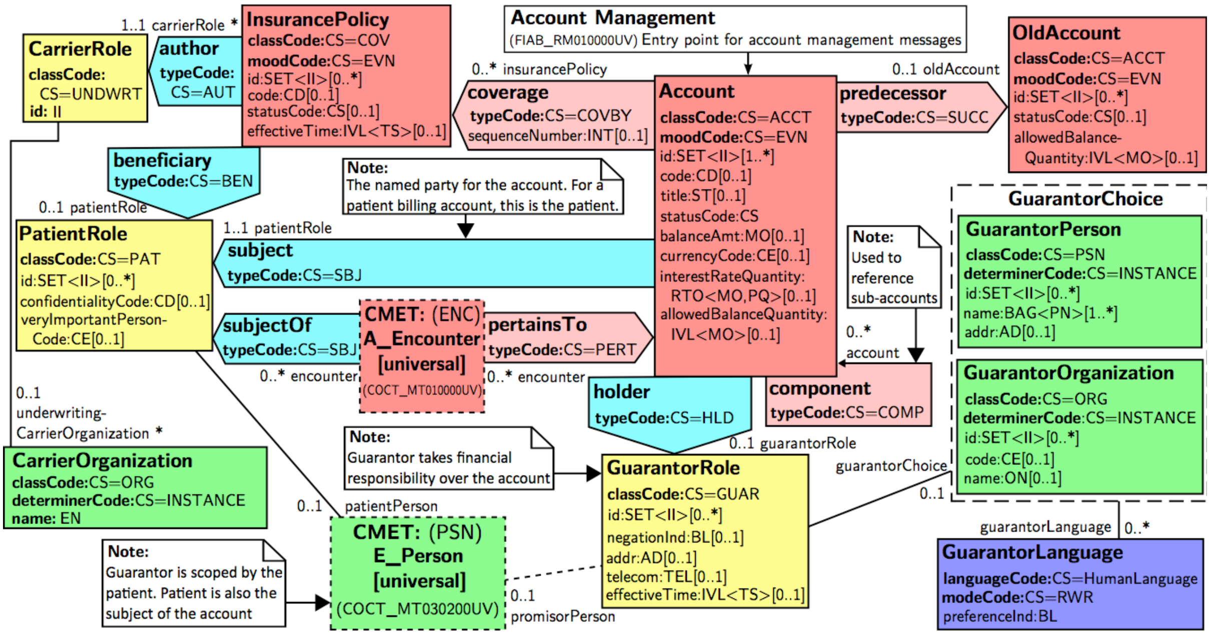


Figure 2. Patient Billing Account Event model (FIAB_RM010000UV02).

Figure 2 presents the Patient Billing Account Event model as an example of the graphical modeling language used in HL7. According to the standard, this model includes the required concepts for the creation and management of patient billing accounts specified in a non UML-compliant modeling language. Central to the model is the concept of an account, with attributes such as identifier, type (code), balance and the currency (e.g. US $) of the account. An account is an accumulator of financial and administrative information for the main purpose of supporting claims and reimbursement. Associations to the account include patient identification (subject of the account), insurance policy, encounter, and the optional specification of a guarantor for any outstanding balance in the account.

This MIF model contains the entry point Account Management connected to the central class Account, the CMETs A_EncounterUniversal and E_PersonUniversal that are links to other MIF models, a choice construction named GuarantorChoice that includes the entities GuarantorPerson and GuarantorOrganization.

6.1.2 UML Extension Mechanism

UML is defined using a metamodel, that is, a model of the modeling language itself. In principle, the metamodel cannot be changed for particular purposes or domains. Fortunately, however, the UML metamodel includes a profile mechanism that permits limited changes to UML without modifying the underlying metamodel. Profiles and constraints permit UML to be tailored to specific domains or platforms while maintaining interoperability across tools.

UML includes three main extensibility constructs: constraints, stereotypes, and tagged values. A constraint is a textual statement of a semantic relationship expressed in some formal language or in natural language. A stereotype is a new kind of model element devised by the modeler and based on an existing kind of model element. A tagged value is a named piece of information attached to any model element.

These constructs permit many kinds of extensions to UML without requiring changes to the basic UML metamodel itself. They may be used to create tailored versions of the UML for an application area. A coherent set of stereotypes with their definition of properties and constraints is modeled as a profile.

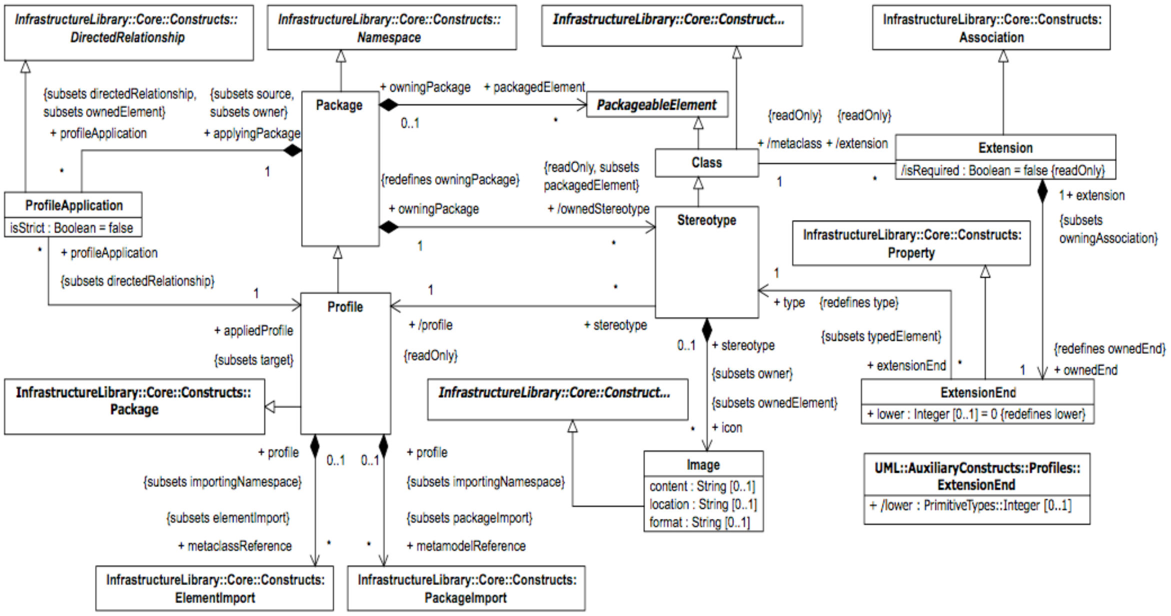


Figure 3. Elements of the UML extension mechanism defined in the UML superstructure specification.

The Profile mechanism has been defined specifically for providing a lightweight extension mechanism to the UML standard. In UML 1.1, stereotypes and tagged values were used as string-based extensions that could be attached to UML model elements in a flexible way. In subsequent revisions of UML, the notion of a Profile was defined in order to provide more structure and precision to the definition of stereotypes and tagged values. Figure 3 shows the fragment of the UML metamodel that defines the elements of the UML extension mechanism defined in UML2. It is possible to observe that Stereotype is a descendant of Class. Thus, all the tagged values are defined as properties (attributes) of the stereotype. Section 8 of this specification is based on the extension mechanism defined in UML 2.4.1.

6.1.3 MIF-UML Goals

This MIF-UML is a specification (under the OMG directives) designed to enable general use of UML and MDA tools to support the development and use of MIF static models. The primary intention of this specification is to enable the UML representation of MIF static models, so that the large base of existing UML modeling tools can be used for MIF models. The following key considerations have been implemented in the specification:

- Standards Based. Enable leverage HL7 and UML standards and standards based tools.
- Simplicity. Reduce complexity and lower the barrier for entry for HL7 MIF business and technical modelers.
- Reuse. Facilitate reuse of MIF models and, as a result, their equivalent UML models.
- Agility. Enable the UML profile to be used with other standards, technologies and layers from the healthcare domain, if required.
- Audience. Allow audiences with different levels of knowledge of the HL7 MIF technical concepts to create and use MIF specifications in UML.
- Interoperability. Ensure that a UML representation of a MIF model produced by one developer can be interpreted as expected by another.
- Completeness. Ensure that a developer can produce a UML representation of any MIF concept, including semantics, XML Schema structure, and metadata.
- Practicality. With minimal effort, an architect or developer can employ the profile in current UML development tools to develop a MIF model.

6.2 Proof of Concept

Proofs of Concept have been completed during the development of the UML profile for MIF static models such that UML models can be used to represent the content and semantics of existing MIF static models defined in MIF XML format. The next MIF and HL7 static models from the HL7 V3 2012 Normative Edition have been used to proof the correctness and validity of the profile:

- Clinical Document Architecture
  - POCD_HD000040 - CDA Hierarchical Description
- Clinical Statement
  - COCT_MT530004UV - A_SupportingClinicalStatement minimal
  - COCT_MT530000UV - A_SupportingClinicalStatement universal
- Patient Administration
  - PRPA_MT201301UV02 - Patient Activate
  - PRPA_MT201302UV02 - Patient Revise
  - PRPA_MT201303UV02 - Patient Demographics
  - PRPA_MT201304UV02 - Patient Identifiers
  - PRPA_MT201305UV02 - Patient Nullify
  - PRPA_MT201306UV02 - Patient Registry Query By Demographics
  - PRPA_MT201307UV02 - Patient Registry Query By Identifier
  - PRPA_MT201310UV02 - Patient Registry Find Candidates Response
- Pharmacy
  - PORX_MT010120UV01 - Medication Order
- Scheduling
  - PRSC_MT010101UV01 - New Appointment Notification
  - PRSC_MT010201UV01 - Revise Appointment Notification
  - PRSC_MT020101UV01 - Cancel Appointment Notification
  - PRSC_MT020201UV01 - Reschedule Appointment Notification

These MIF and HL7 static models and their equivalent UML specification are included in this submission. In addition to it, the profile specification has been developed with Enterprise Architect although it can be easily imported and used with other modeling tools that support XMI.

7 MIF-UML Modeling Guide

7.1 Overview

The main goal of this section is to describe the modeling steps required to construct valid MIF static models in UML. The stereotypes of the UML profile for MIF static models described in section 8 and the mappings between MIF and UML constructs included in the MIF-UML transformation reference of section 9 are the key components that support this modeling guide.

This specification uses Enterprise Architect as a modeling tool. The indications and steps presented here are for modelers using Enterprise Architect 10. However, this modeling guide can be easily adapted to other modeling tools and the general steps may be similar.

7.2 Modeling a static model

A static model is an artifact that represents the knowledge of a particular domain of interest. HL7 standards define a big amount of static models for several healthcare domains. The following subsections present the required modeling steps to construct a static model in UML like the model in Figure 4.

This particular model represents the arguments of a Scheduling service. In other words, if someone interested in this domain wants to realize scheduling services, this model describes the data needed to perform scheduling operations.

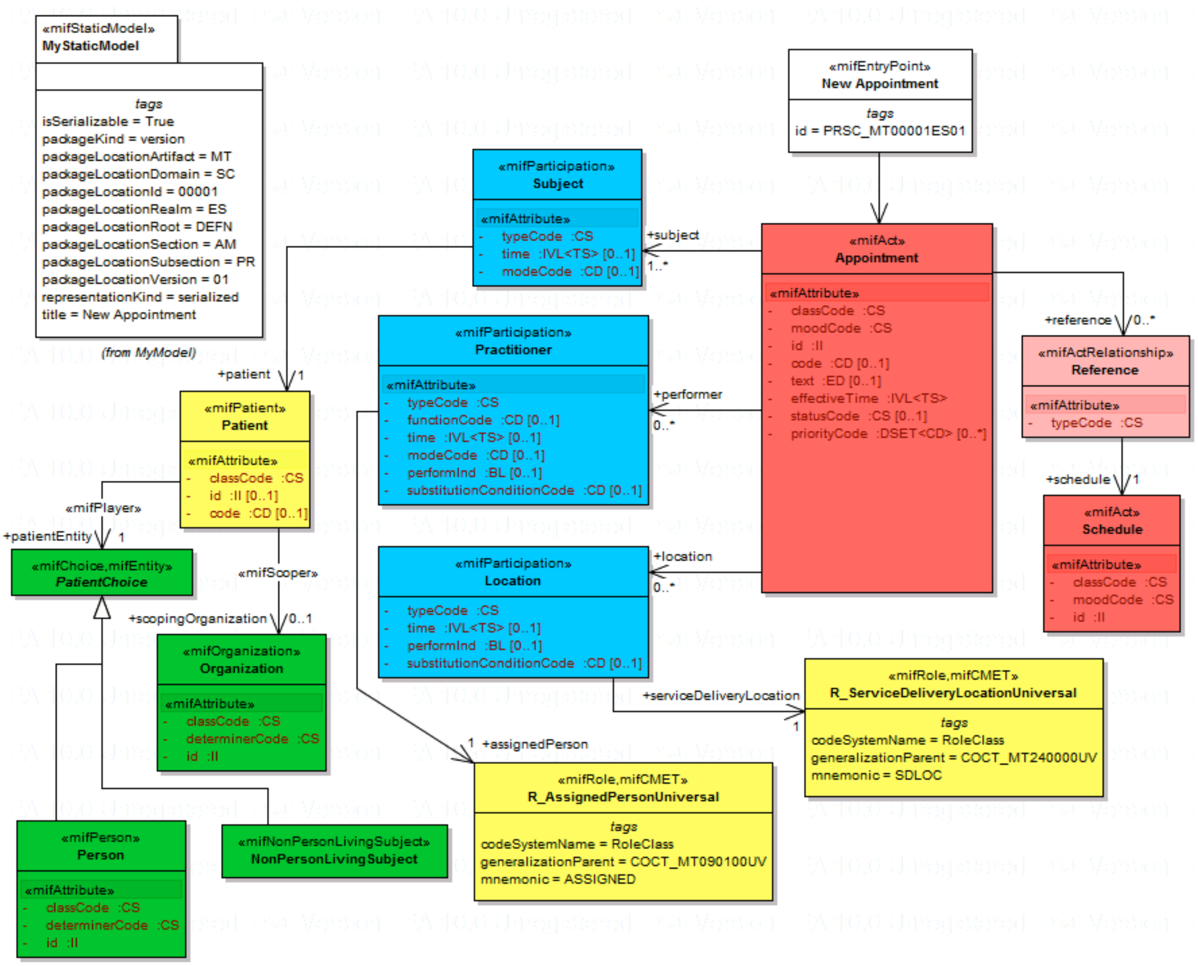


Figure 4. Example of MIF Static Model in UML using the UML Profile for MIF Static Models.

The central focus of the model is the Appointment class. This class represents the activity being scheduled.

The subject (usually the patient) participates directly or indirectly in the appointment. The modeCode attribute of subject is included to allow for appointments where the patient is not physically present. The subject time attribute allows the patient's appointment start time and duration to take place within the start time and duration for the whole appointment. Both the subject and the practitioner participations have a modeCode attribute to indicate a physical or remote presence.

The practitioner and location participations associate the resources being reserved in this appointment. These are practitioners and high-use items that are typically controlled by a schedule. Each resource can have its own effective time interval or it can default to the time interval of the appointment. The performInd attribute is a boolean indicating that this participation is controlled by a schedule and must be reserved before use. A participation that does not have this indicator set is not actively scheduled, i.e., for information purposes only.

A Schedule is a set of slots associated with a resource and may be used here to designate which schedule(s) the appointment is occupying (for example, a physician's schedule and his/her office schedule). A Schedule may also be used for any time block which is further divided into slots.

The following subsections deal with the particular steps to create the model shown in Figure 4.

7.2.1 Setting up the environment

Before starting with the construction of the model itself, it is necessary to present the steps for preparing the modeling environment. Enterprise Architect projects can be created as .eap files via the **File | New Project** menu option, which displays the Save New Enterprise Architect Project dialog. Once the project has been saved, the Model Wizard dialog displays. The modeler may select the Model Packages to include and click on the **OK** button. Enterprise Architect adds a model containing the selected Model Packages to the Project Browser. Figure 5 shows the Model Wizard dialog. It is not necessary to select any model package at this point.

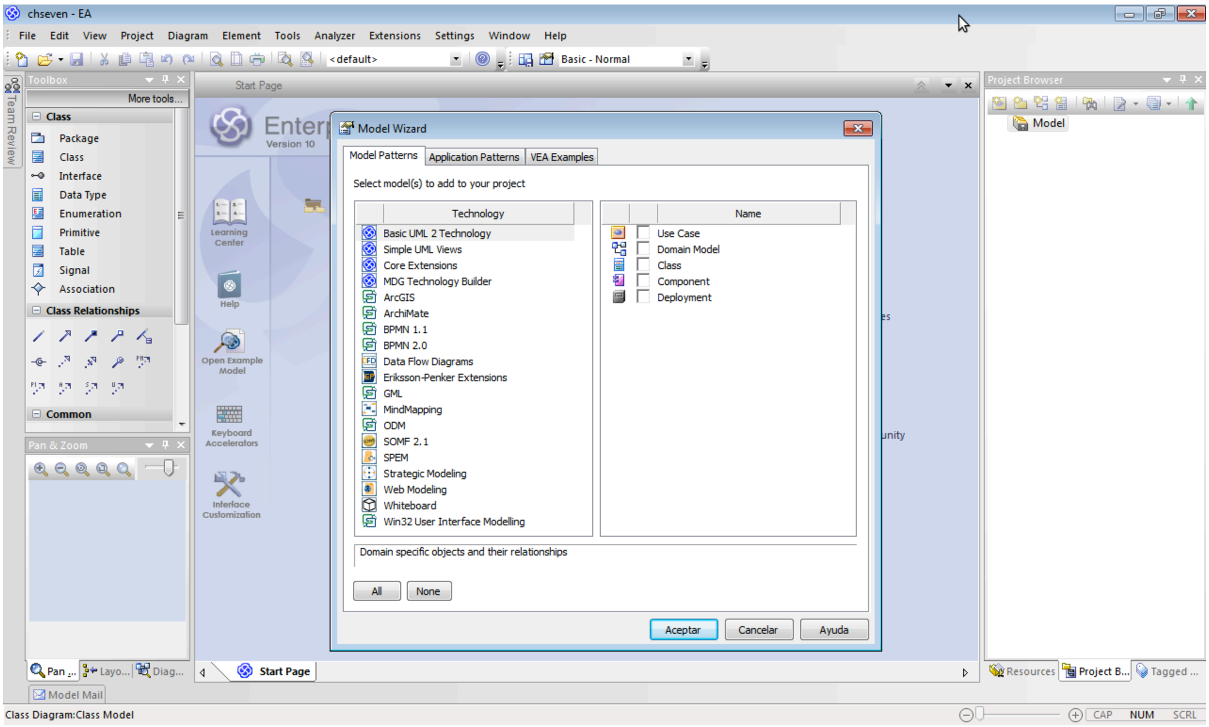


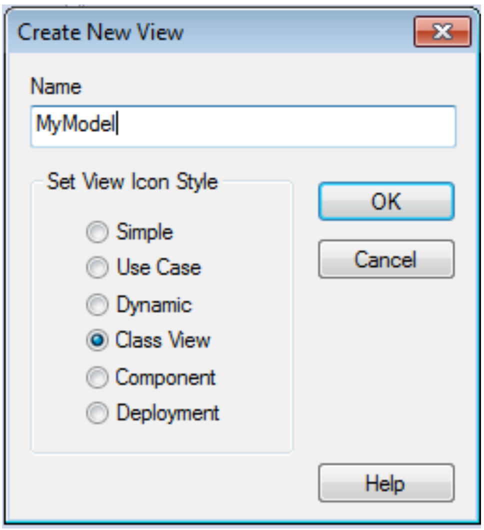
Figure 5. Model Wizard dialog of Enterprise Architect.

Now that the modeler has created a project containing at least one model, it is necessary to add a view to the model, and then add a package with diagram, elements and connectors (associations). A view is the highest-level container, or package, within a model.

There are six types of View, five of which represent conventional ways of categorizing the structures or purposes of a model, and one (*Simple View*) for developing your own categorization.

To create a View, follow the steps below:

1. Right-click on your model name in the Project Browser. The context menu displays.
2. Select the **New View** menu option. The Create New View dialog displays (see Figure 6).
3. In the **Name** field, type the name of the View.
4. In the Set View Icon Style panel, click on the radio button for the type of View to create (Class View, in this case).
5. Click on the **OK** button.



Create New View

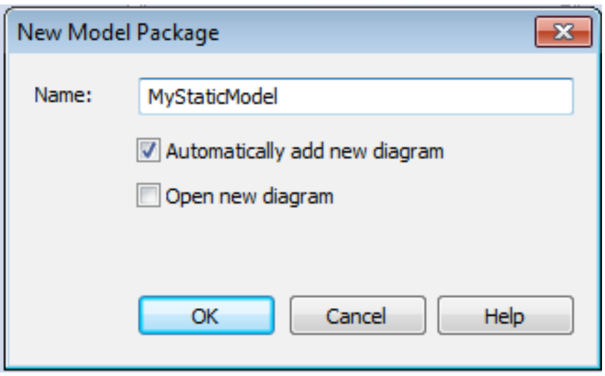
Name: MyModel

Set View Icon Style

○ Simple
○ Use Case
○ Dynamic
● Class View
○ Component
○ Deployment

Figure 6. Create New View dialog of Enterprise Architect.

Now that you have created a View in the model, you can add a package and diagram to that View or any other in the model. A *Package* is a container of model elements, and is displayed in the Project Browser as the 'folder' icon familiar to Windows users. Package contents are arranged alphabetically. In the Project Browser click on a package and, in the Project Browser toolbar, click on the **New Package** icon. Enterprise Architect displays a prompt for the package name (see Figure 7).



New Model Package

Name: MyStaticModel

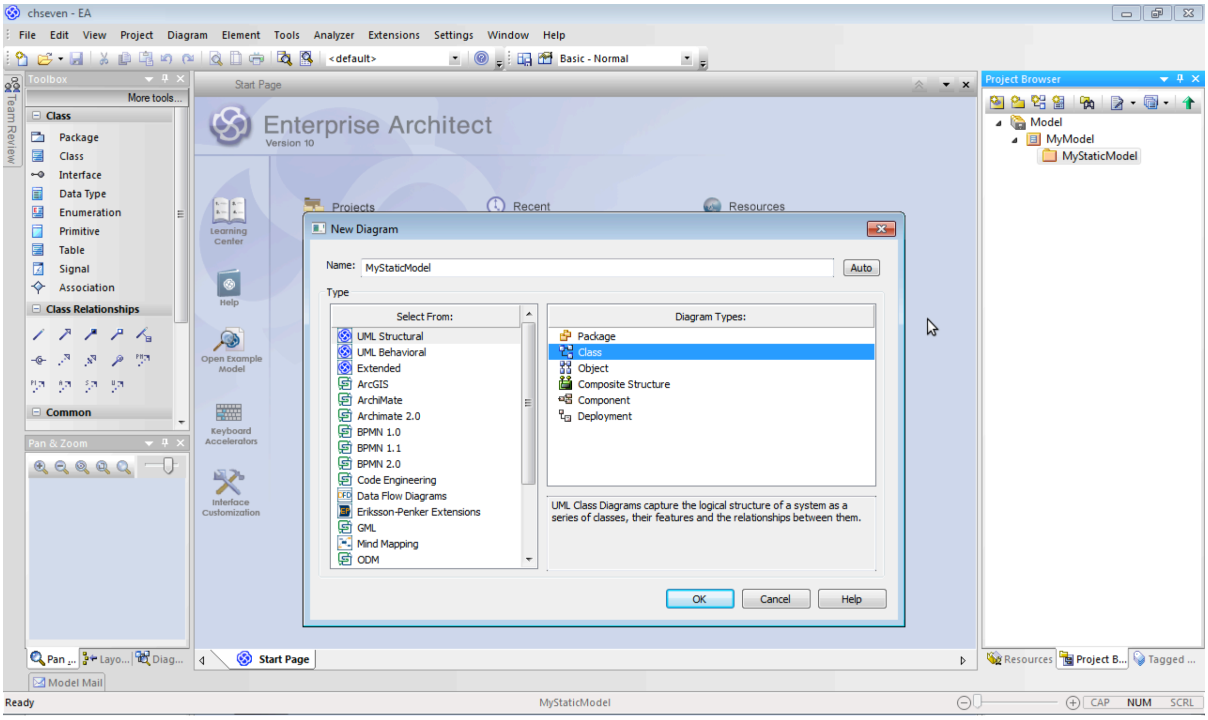
☑ Automatically add new diagram

□ Open new diagram

Figure 7. New Model Package dialog of Enterprise Architect.

This prompt also contains the **Automatically add new diagram** option for automatically creating a diagram for the package, which defaults to selected. Type in a name and click on the **OK** button. Enterprise Architect adds the new package subordinate to the package or view you selected.

Then the **New Diagram** dialog (see Figure 8) displays since the option to automatically add a new diagram was selected in the previous step. A diagram is a representation of the components or elements of your model and, depending on the type of diagram, the way those elements are connected or how they interact is different. Click on the UML Structural diagram category in the Select From panel, and select the Class diagram type in the Diagram Types panel, then click on the **OK** button. Enterprise Architect adds a class diagram object to the package, with the same name as the package. It also opens the Diagram View for your diagram, in the center of the screen.



Name: MyStaticModel

Select From: UML Structural

Diagram Types: Class

UML Class Diagrams capture the logical structure of a system as a series of classes, their features and the relationships between them.

Figure 8. New Diagram dialog of Enterprise Architect.

7.2.2 Load HL7 Data Types

MIF static models use the data types from the HL7 Abstract Data Types specification. In order to be consistent with the standard requirements, this specification provides these data types in an XMI file named datatypes.xml that can be imported in Enterprise architect.

To import a package from XMI, follow the steps below:

1. In the Project Browser window, select the view into which to import the file (MyModel, in the running example).

2. Either:

   - Right-click and select the **Import/Export | Import Package from XMI** context menu option, or

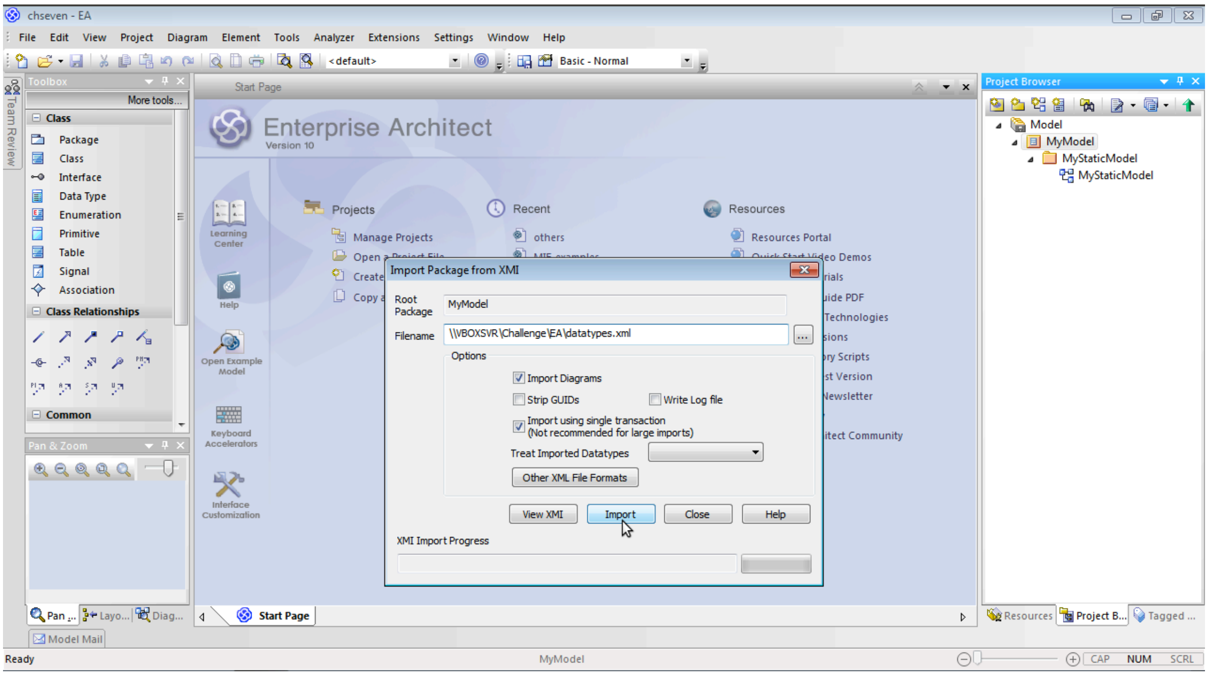
   - Select the **Project | Import/Export | Import Package from XMI** menu option.

   The Import Package from XMI dialog displays (see Figure 9).

3. In the **Filename** field, type the directory path and filename from which to import the datatypes.xml XMI file.

4. Select the **Import diagrams** checkbox to import diagrams.

5. Click on the **Import** button.



Import Package from XMI

Root Package: MyModel

Filename: \\VBOXSVR\Challenge\EA\datatypes.xml

Options

☑ Import Diagrams

☐ Strip GUIDs ☐ Write Log file

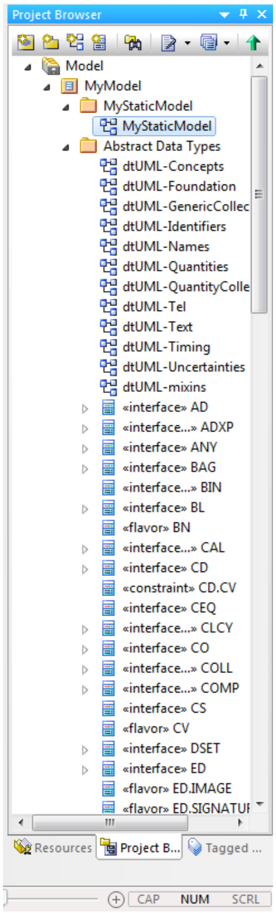
☑ Import using single transaction (Not recommended for large imports)

Treat Imported Datatypes:

XMI Import Progress

Figure 9. Import Package from XMI dialog.

Then the list of datatypes from the HL7 Abstract Data Types Specification is available to be used in the types of the attributes of clases (see Figure 10). The imported package also includes class diagrams of the data types for the sake of understandability.



Model
MyModel
MyStaticModel
MyStaticModel
Abstract Data Types
dtUML-Concepts
dtUML-Foundation
dtUML-GenericCollec
dtUML-Identifiers
dtUML-Names
dtUML-Quantities
dtUML-QuantityColle
dtUML-Tel
dtUML-Text
dtUML-Timing
dtUML-Uncertainties
dtUML-mixins
«interface» AD
«interface...» ADXP
«interface» ANY
«interface» BAG
«interface...» BIN
«interface» BL
«flavor» BN
«interface...» CAL
«interface» CD
«constraint» CD.CV
«interface» CEQ
«interface...» CLCY
«interface» CO
«interface...» COLL
«interface...» COMP
«interface» CS
«flavor» CV
«interface» DSET
«interface» ED
«flavor» ED.IMAGE
«flavor» ED.SIGNATUI

Figure 10. Package Browser including the HL7 Abstract Data Types.

7.2.3 Import UML Profile for MIF Static Models

The stereotypes described in section 8 are required to construct MIF models in UML. This specification provides a XMI file called MIF-Profile.xml with all the contents of the UML Profile for MIF Static Models. To import the UML Profile, follow the steps below:

1. Open the Resources window (**Project | Resources**).

2. Right-click on the *UML Profiles* tree node and select the **Import Profile** context menu option. The Import UML Profile dialog displays (see Figure 11).

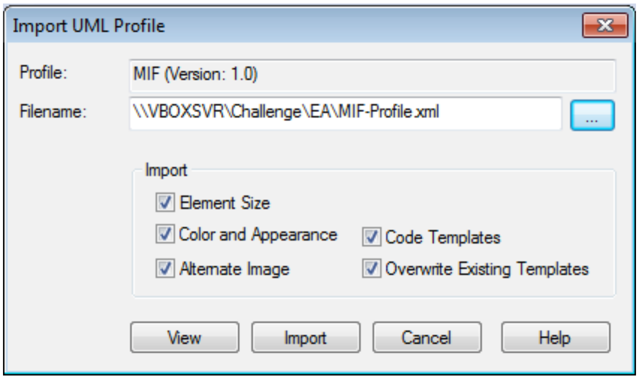
3. Locate the XML Profile file to import using the **[...]** (Browse) button.

4. Set the required import option checkboxes for all stereotypes defined in the Profile; Ensure that the next options are selected:

   - **Element Size** - to import the element size attributes

   - **Color and Appearance** - to import the color (background, border and font) and appearance (border thickness) attributes

5. Click on the **Import** button.



Profile: MIF (Version: 1.0)

Filename: \\VBOXSVR\Challenge\EA\MIF-Profile.xml

Import

☑ Element Size

☑ Color and Appearance ☑ Code Templates

☑ Alternate Image ☑ Overwrite Existing Templates

Figure 11. Import UML Profile dialog of Enterprise Architect.

The Profile is added to the *UML Profiles* folder (see Figure 12). If the Profile already exists, Enterprise Architect prompts you to overwrite the existing version and import the new one (or cancel). Once the import is complete, the Profile is ready to use.

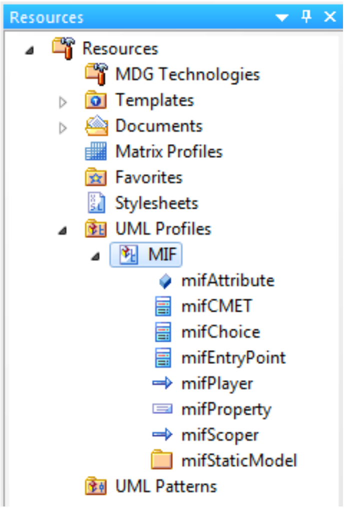


Figure 12. Resources View in Enterprise Architect showing the MIF-UML Profile.

7.2.4 Static Model

The Model Package named MyStaticModel created before is the container of all the elements included in the MIF static model. First step is to open the Class diagram within the MyStaticModel package and drag and drop this package from the Project Browser view to the class diagram, as shown in Figure 13. To indicate that this package represents the MIF static model in UML, it is necessary to apply the «mifStaticModel» stereotype, as described in section 9.2. To do that, open the Properties dialog of the package by double-clicking on its name.

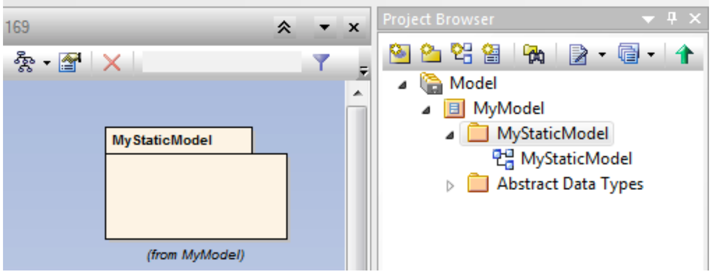


Figure 13. Package "MyStaticModel".

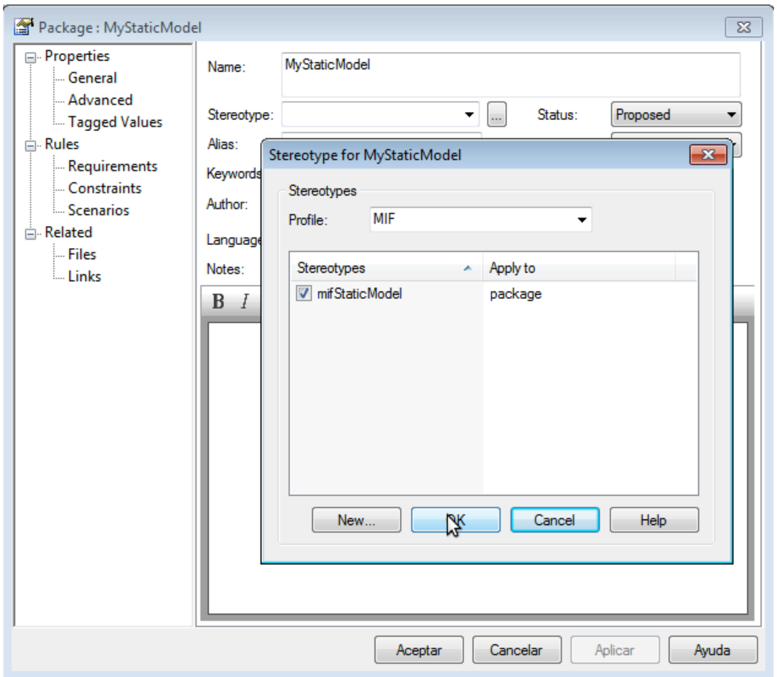
To apply a stereotype to any UML construct, using the Properties dialog, select any one of the following steps:

1. In the **Stereotype** field, type the stereotype(s) to apply as a comma-separated list.
2. Click on the drop-down arrow and select the required stereotype from the list.
3. Click on the **[...]** button to use the Stereotype Selector dialog.

The Stereotype Selector dialog (see Figure 14) enables you to apply one or more stereotypes to a UML construct, from multiple Profiles:

1. Click on the **Profile** drop-down arrow and choose the MIF profile.
2. In the Stereotypes list, enable or disable the required stereotype by selecting or deselecting the checkbox against it. In this case, select the «mifStaticModel» stereotype.
3. Click on the **OK** button to apply the selection.

Then, the stereotype «mifStaticModel» is applied to the Package MyStaticModel.


Package : MyStaticModel

Name: MyStaticModel

Stereotype:

Status: Proposed

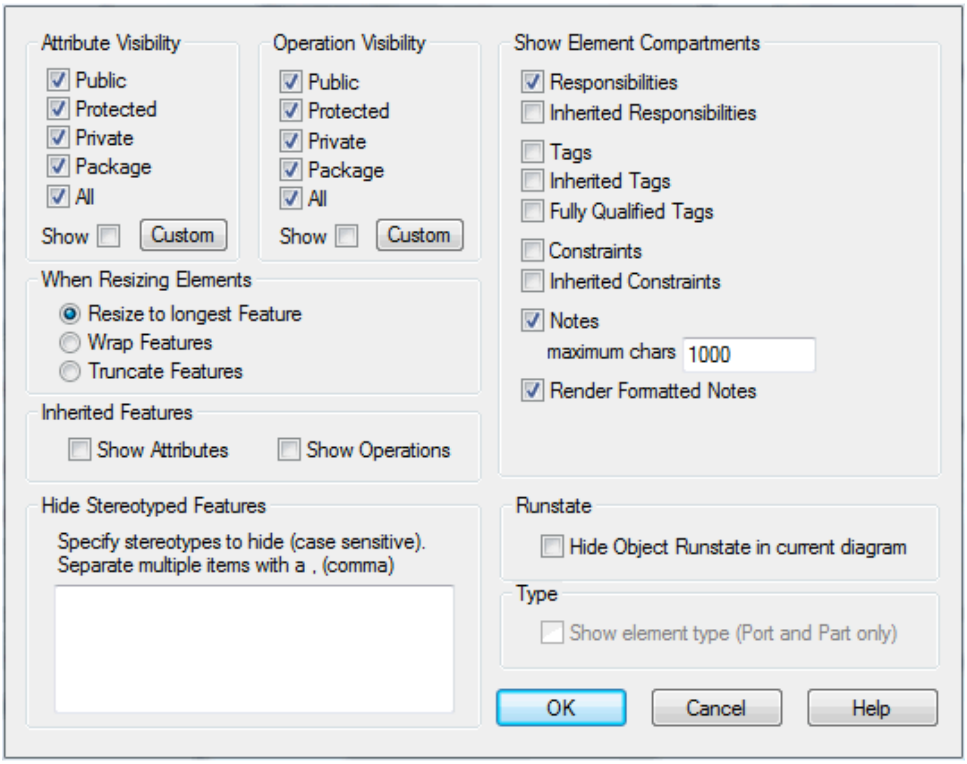
Stereotype for MyStaticModel

Profile: MIF

Stereotypes	Apply to
☑ mifStaticModel	package

Figure 14. Applying stereotype «mifStaticModel» to package MyStaticModel.

The stereotype «mifStaticModel» defines a set of properties (see section 8.4.9) to express additional knowledge present in MIF files. The value of these properties may be set when applying the stereotype to the package. To visualize this tagged values right click in MyStaticModel package and select Feature and Compartment Visibility. Then the Feature Visibility dialog opens (see Figure 15). Deselect the All checkboxes in Attribute Visibility and Operation Visibility and select the Tags checkbox in Show Element Compartments.



Attribute Visibility
☑ Public
☑ Protected
☑ Private
☑ Package
☑ All
Show ☐ Custom

Operation Visibility
☑ Public
☑ Protected
☑ Private
☑ Package
☑ All
Show ☐ Custom

Show Element Compartments
☑ Responsibilities
☐ Inherited Responsibilities
☐ Tags
☐ Inherited Tags
☐ Fully Qualified Tags
☐ Constraints
☐ Inherited Constraints
☑ Notes
maximum chars 1000
☑ Render Formatted Notes

When Resizing Elements
● Resize to longest Feature
○ Wrap Features
○ Truncate Features

Inherited Features
☐ Show Attributes ☐ Show Operations

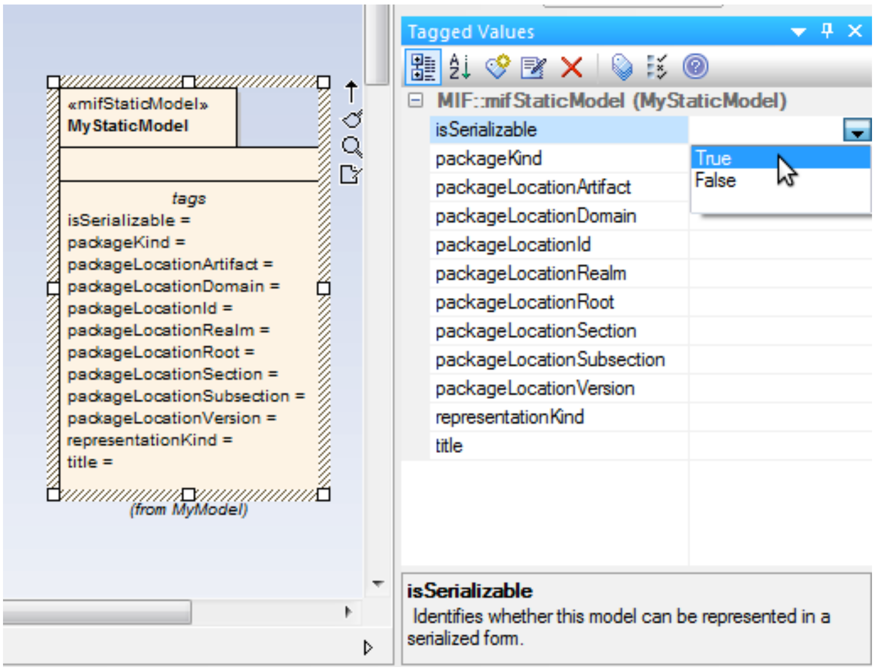
Hide Stereotyped Features
Specify stereotypes to hide (case sensitive).
Separate multiple items with a , (comma)

Runstate
☐ Hide Object Runstate in current diagram

Type
☐ Show element type (Port and Part only)

Figure 15. Feature Visibility dialog.

Then the tagged values are visible in the class diagram for the package, and their values can be set in the Tagged Values view (see Figure 16).



«mifStaticModel»
MyStaticModel

tags
isSerializable =
packageKind =
packageLocationArtifact =
packageLocationDomain =
packageLocationId =
packageLocationRealm =
packageLocationRoot =
packageLocationSection =
packageLocationSubsection =
packageLocationVersion =
representationKind =
title =

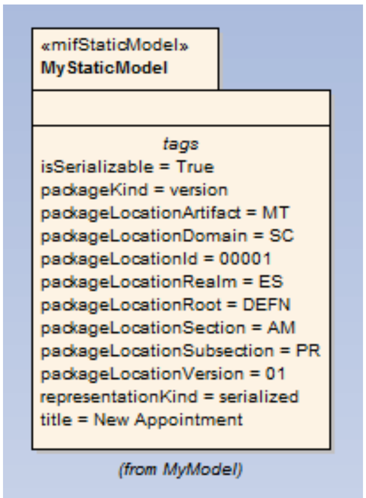
(from MyModel)

MIF::mifStaticModel (MyStaticModel)
isSerializable
packageKind
packageLocationArtifact
packageLocationDomain
packageLocationId
packageLocationRealm
packageLocationRoot
packageLocationSection
packageLocationSubsection
packageLocationVersion
representationKind
title

isSerializable
Identifies whether this model can be represented in a serialized form.

Figure 16. Setting values to tagged values of «mifStaticModel» stereotype.

As a result, Figure 17 shows MyStaticModel package with the stereotype «mifStaticModel» and all the values for the properties of the stereotype. The values indicate that MyStaticModel is the definition of the version 01 of a message type (MT) model artifact with title New Appointment and identified by the code 00001, created for the HL7 Spain affiliate (ES) to be published in the scheduling (SC) domain, within the administrative management (AM) in the practice (PR) subsection.



«mifStaticModel»
MyStaticModel

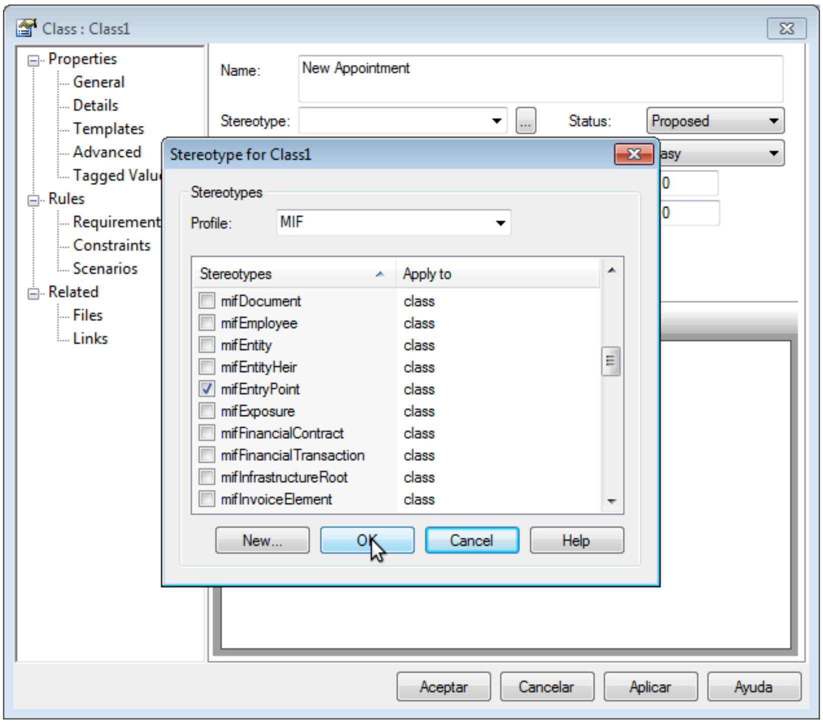
tags
isSerializable = True
packageKind = version
packageLocationArtifact = MT
packageLocationDomain = SC
packageLocationId = 00001
packageLocationRealm = ES
packageLocationRoot = DEFN
packageLocationSection = AM
packageLocationSubsection = PR
packageLocationVersion = 01
representationKind = serialized
title = New Appointment

(from MyModel)

Figure 17. Final view of MyStaticModel package.

7.2.5 Entry point

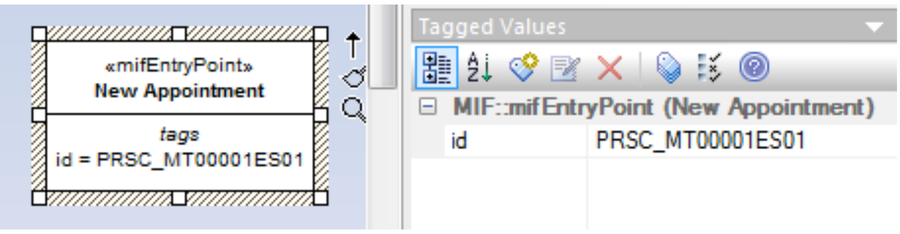
MIF models contain an entry point element that indicates which is the central element of focus in the model. As indicated in section 9.3, entry points are represented as common UML classes with the stereotype «mifEntryPoint». In the running example, create a new class named NewAppointment and apply the stereotype «mifEntryPoint» (see Figure 18). The application of this profile also requires the specification of the id attribute value, as depicted in Figure 19. In this case, the value for the id is PRSC_MT00001ES01, which is formed by joining some of the values indicated in the package with the stereotype «mifStaticModel».



Name: New Appointment
Stereotype:
Status: Proposed
Profile: MIF

Stereotypes	Apply to
□ mifDocument	class
□ mifEmployee	class
□ mifEntity	class
□ mifEntityHeir	class
☑ mifEntryPoint	class
□ mifExposure	class
□ mifFinancialContract	class
□ mifFinancialTransaction	class
□ mifInfrastructureRoot	class
□ mifInvoiceElement	class

Figure 18. Creating entry point for the static model.



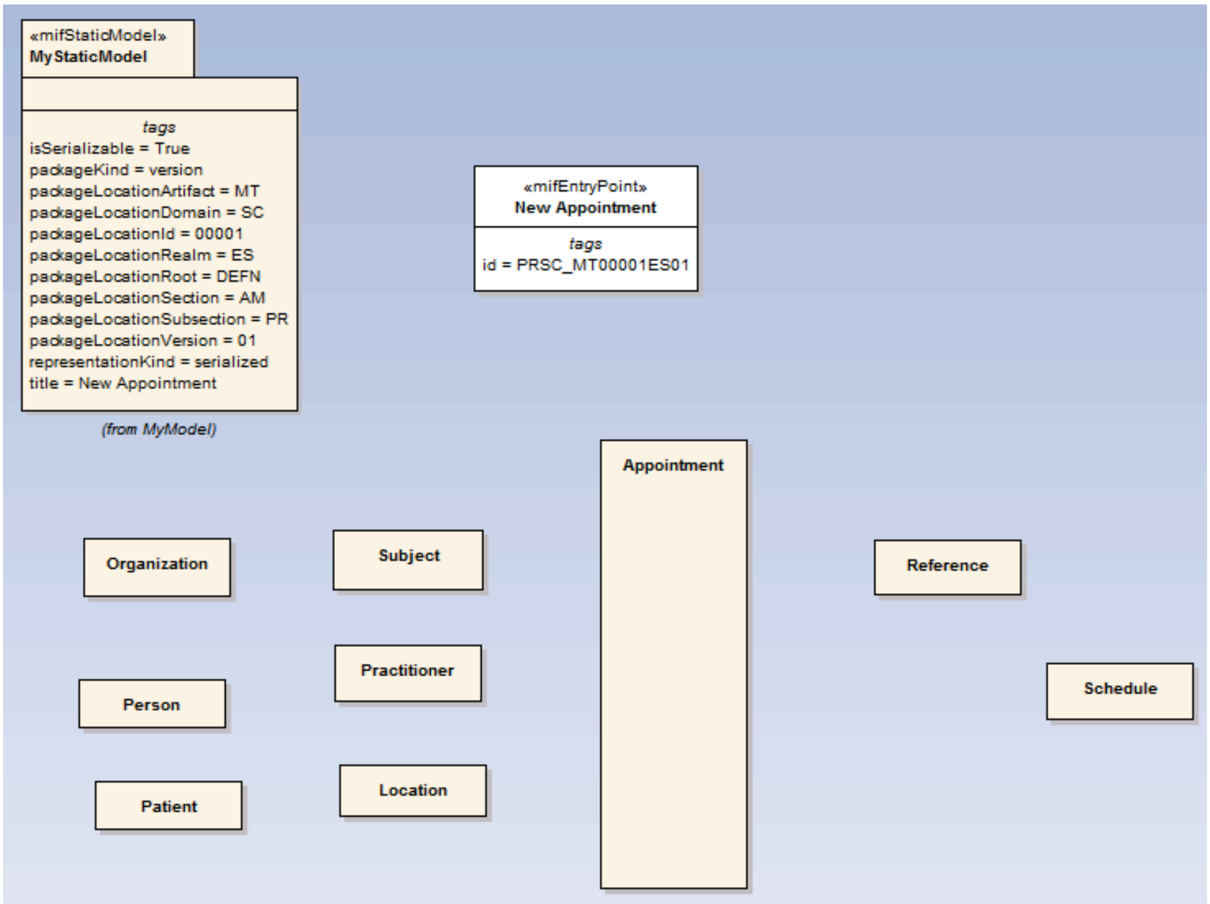
id	PRSC_MT00001ES01

Figure 19. Setting up tagged values of «mifEntryPoint» stereotype.

7.2.6 Classes and associations

Classes and their associations between them are the main elements of knowledge in a class diagram or static model. The modeler has several options for adding elements (the model building units) to a package and/or diagram. The simplest method is to use the Toolbox to the left of the diagram, which automatically lists the elements applicable to the type of diagram created. Just click on the required element and drag it onto your diagram.

Before the element appears on the diagram, the Properties dialog of that element displays. You can use the Properties dialog to define the characteristics of the element, such as its name. Type a name in the Name field, and click on the OK button. Look at the Project Browser, underneath the package in which you created the diagram. The element is listed. This is the process to create the classes presented in Figure 20.


«mifStaticModel»
MyStaticModel

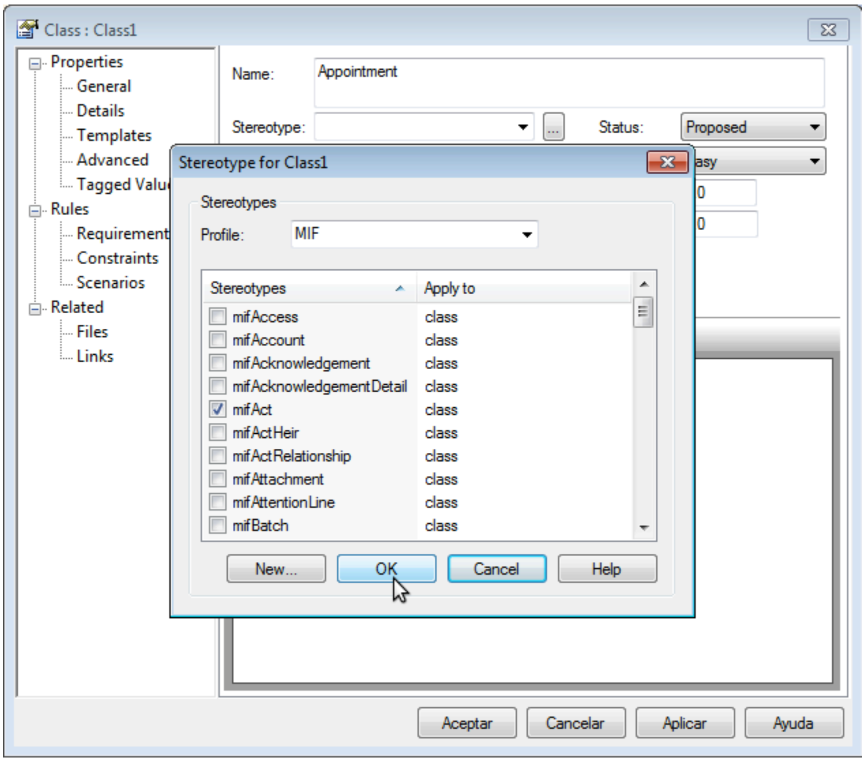
tags
isSerializable = True
packageKind = version
packageLocationArtifact = MT
packageLocationDomain = SC
packageLocationId = 00001
packageLocationRealm = ES
packageLocationRoot = DEFN
packageLocationSection = AM
packageLocationSubsection = PR
packageLocationVersion = 01
representationKind = serialized
title = New Appointment

(from MyModel)

«mifEntryPoint»
New Appointment
tags
id = PRSC_MT00001ES01

Figure 20. Classes in the static model.

Each class in a MIF static model is derived from one of the classes defined in the HL7 Reference Information Model (RIM), as indicated in section 9.4. To indicate this knowledge in UML, each class must contain the stereotype that matches with the RIM class from which it is derived. In the example, the «mifAct» stereotype is applied to Appointment because it is derived from the Act class of the RIM (see Figure 21).



Class : Class1

- Properties
  - General
  - Details
  - Templates
  - Advanced
  - Tagged Valu
- Rules
  - Requirement
  - Constraints
  - Scenarios
- Related
  - Files
  - Links

Name: Appointment

Stereotype:

Status: Proposed

Stereotype for Class1

Stereotypes

Profile: MIF

Stereotypes	Apply to
□ mifAccess	class
□ mifAccount	class
□ mifAcknowledgement	class
□ mifAcknowledgementDetail	class
☑ mifAct	class
□ mifActHeir	class
□ mifActRelationship	class
□ mifAttachment	class
□ mifAttentionLine	class
□ mifBatch	class

Figure 21. Applying stereotypes to classes in the static model.

When applying the stereotype from the Foundational Area (see 8.2) of the UML Profile for MIF static models, the color of the UML classes automatically changes in the class diagram (in version 10 of Enterprise Architect) to match the right color of the RIM (red for acts, blue for participations, yellow for roles, and green for entities).

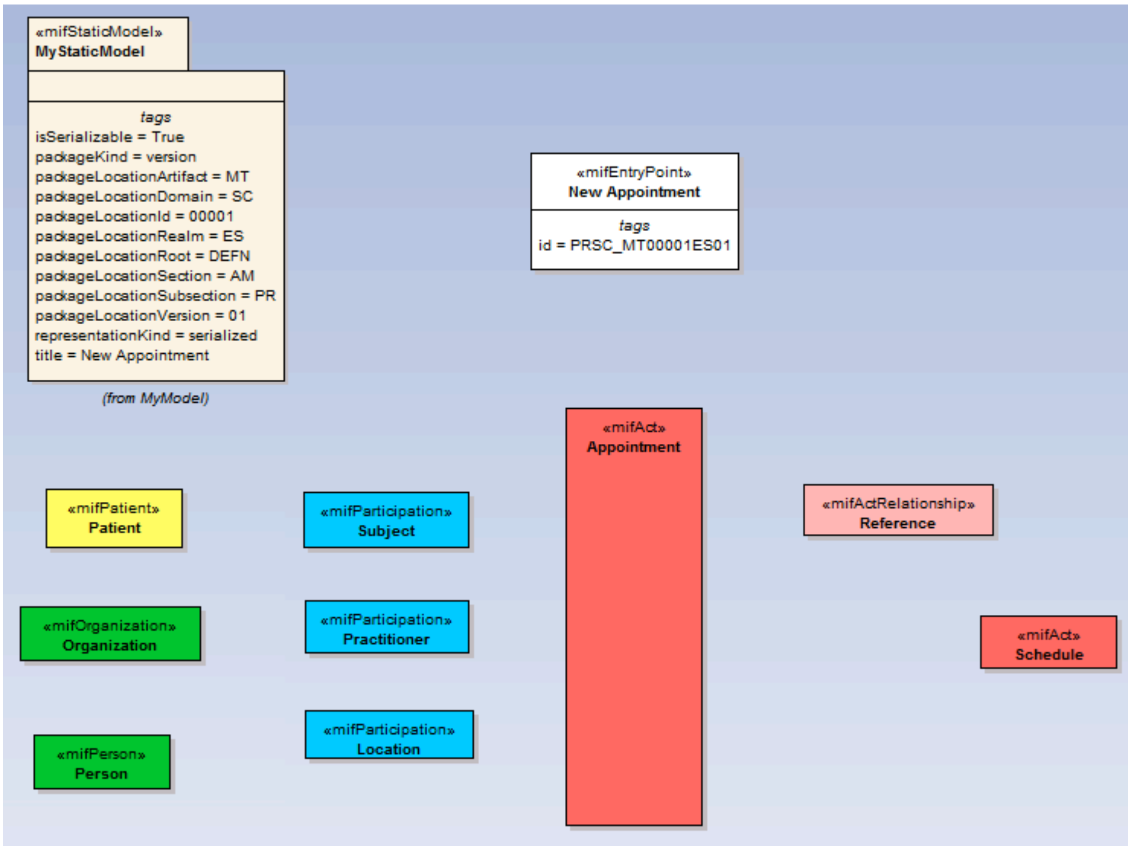


Figure 22. Static model after applying the stereotypes.

Once the classes are created and their stereotypes applied, the modeler needs to connect them with associations. Connectors define specific associations between specific elements, so the modelr usually create them directly on the diagram by dragging the required association from the Toolbox.

First of all create two elements on the diagram. Click on a connector in the Toolbox, click on the source element in the association, then drag across to the target element. This creates the selected connection between the two elements. Following this process, the modeler obtains the model shown in Figure 23. Note that the entry point must also contain an association that connects it with the main class in the model, which is Appointment in the example.

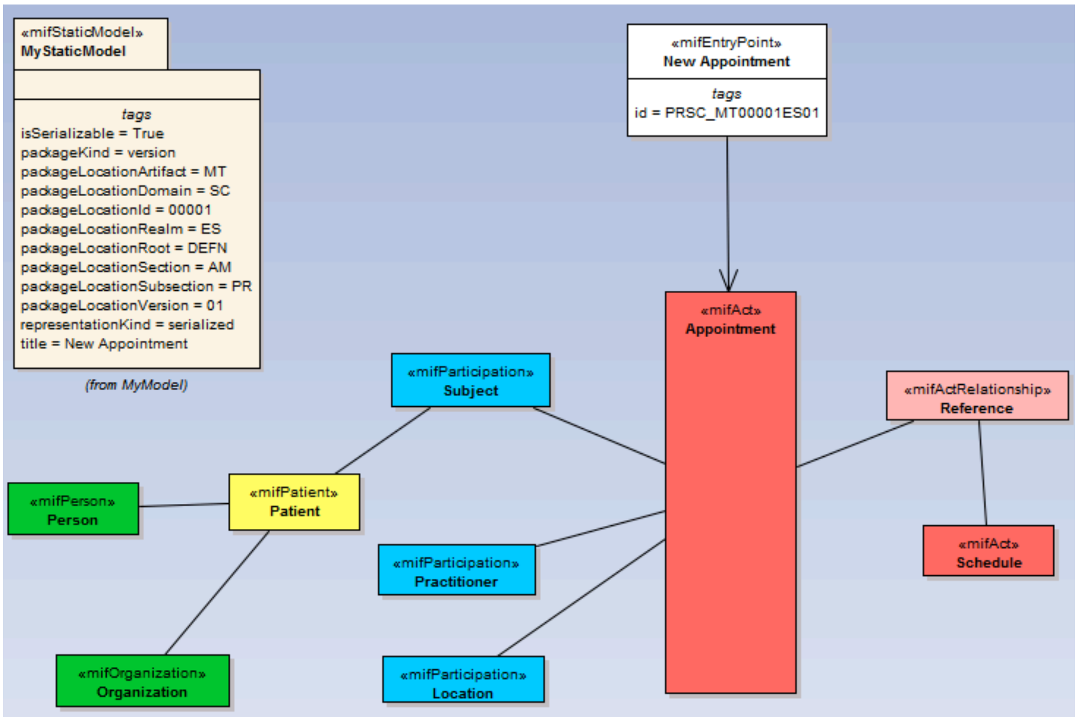
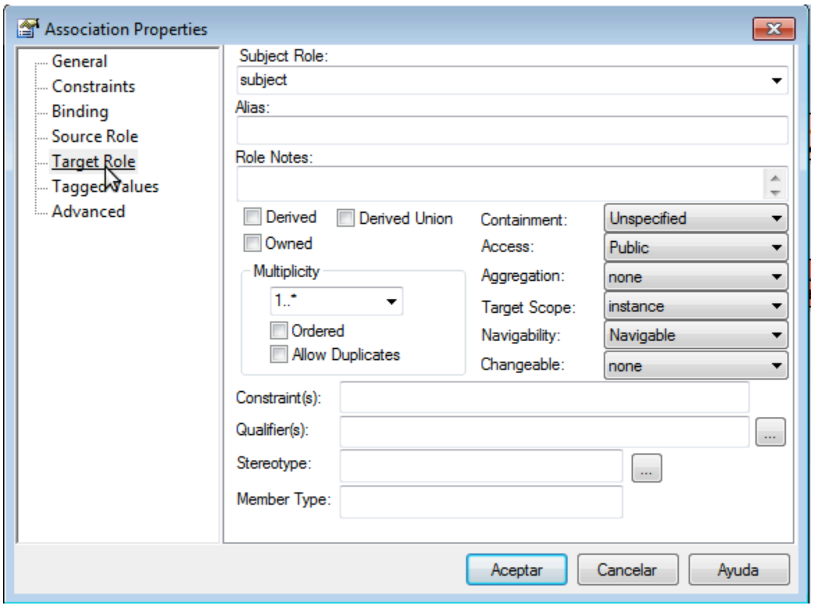


Figure 23. Associations in the static model.

If the modeler double-clicks on the connector, the connector Properties dialog displays (see Figure 24), and then it is possible to define the characteristics of the association, like the role names of the association ends and their multiplicity.



Subject Role: subject

Alias:

Role Notes:

□ Derived □ Derived Union

□ Owned

Multiplicity: 1..*

□ Ordered

□ Allow Duplicates

Containment: Unspecified

Access: Public

Aggregation: none

Target Scope: instance

Navigability: Navigable

Changeable: none

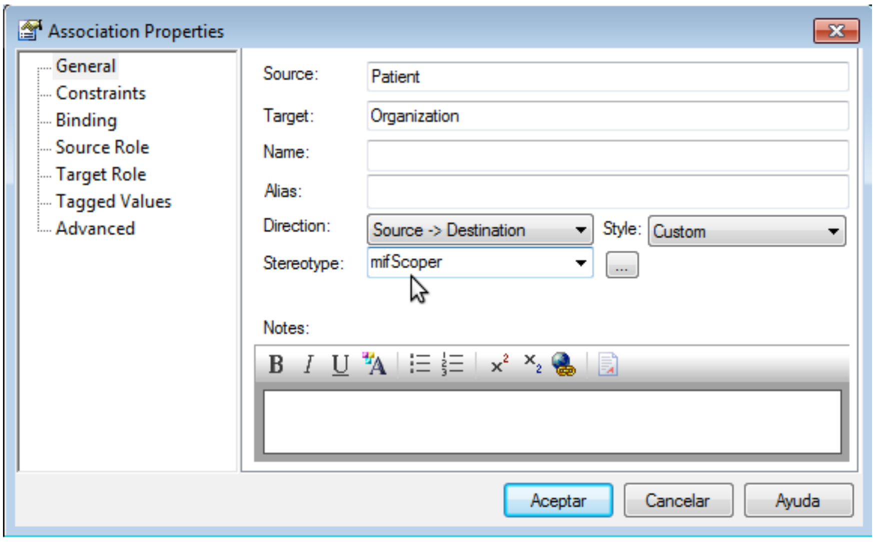
Constraint(s):

Qualifier(s):

Stereotype:

Member Type:

Figure 24. Setting up the characteristics of an association end.



Source: Patient

Target: Organization

Name:

Alias:

Direction: Source -> Destination

Style: Custom

Stereotype: mifScoper

Notes:

Figure 25. Clarifying the semantics of a association between a role class and an entity class.

Associations between roles and entities in MIF static models contain additional semantics indicating whether the entity is the player or the scoper of the role in the association (see section 9.6). To represent this knowledge, the modeler should apply the stereotypes «mifPlayer» or «mifScoper» in the associations between a class with a stereotype referencing a RIM role and a class with a stereotype representing a RIM entity (see Figure 25 and Figure 26).

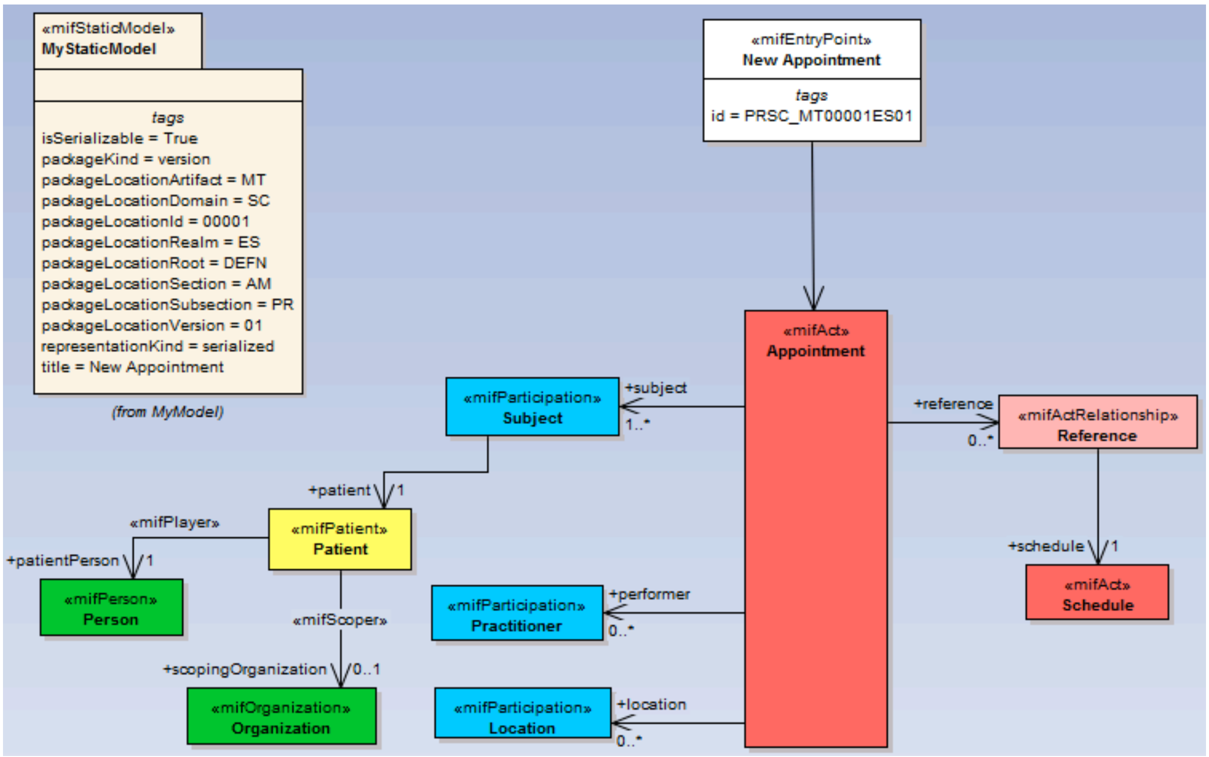


Figure 26. Classes and their associations in the static model.

MIF static models have several constraints on associations. All the associations in the model must be binary, i.e., only connect two elements, and should only connect compatible classes (see constraints [1]-[8] in section 8.6.3). As an example, if a member end of a association is a class with a stereotype representing an Act, the opposite member end must me a class with na stereotype representing a Participation or an ActRelationship. In the same way, entities connect to roles, roles connect to entities, role links, and participations, and participations connect to roles and acts.

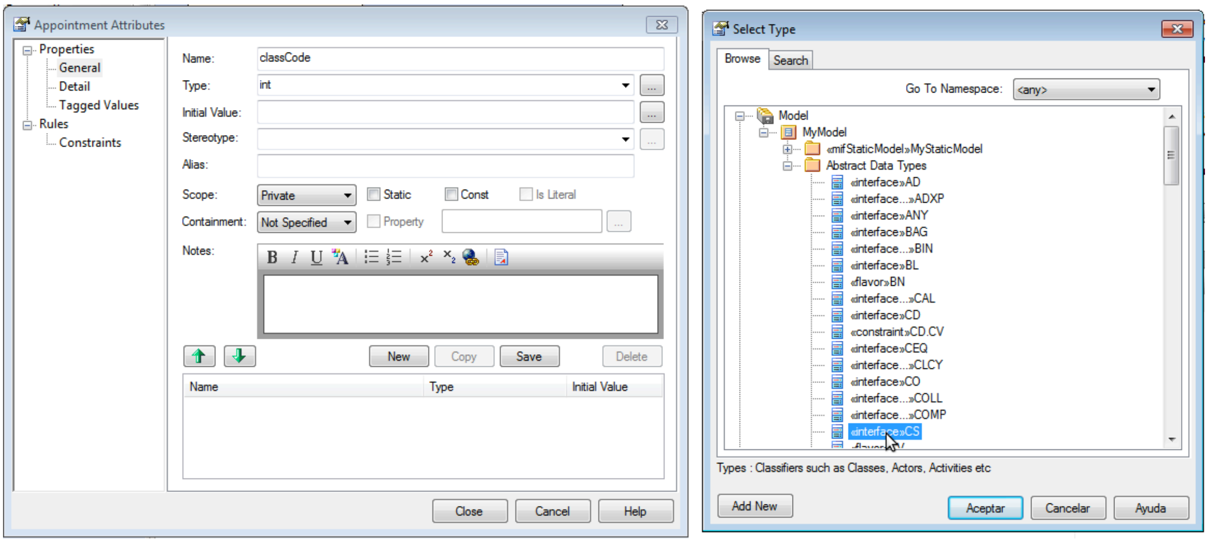
All of these constraints are formally defined in the UML Profile for MIF static models as OCL constraints in the stereotypes and metaclasses of section 8. Therefore, any issue in the construction of associations can be automatically detected taking into account these constraints.

7.2.7 Adding attributes

Attributes are features of an element that represent the properties or characteristics of that element. Elements with attributes (typically classes) display their features in a special compartment of the class box located below the name of the element. To create attributes in a class, either:

1. Right-click on the class to be edited, and from the context menu select the **Attributes** menu option
2. Click on the element and press [F9], or
3. Double-click on the class to open the Properties dialog. Then, in the Detail section select Attributes.

Then, the <Class name> Attributes dialog displays (see Figure 27).



Appointment Attributes

Name: classCode

Type: int

Initial Value:

Stereotype:

Alias:

Scope: Private

Containment: Not Specified

Notes:

Select Type

Go To Namespace: <any>

Model
- MyModel
  - «mifStaticModel»MyStaticModel
  - Abstract Data Types
    - «interface»AD
    - «interface...»ADXP
    - «interface»ANY
    - «interface»BAG
    - «interface...»BIN
    - «interface»BL
    - «flavor»BN
    - «interface...»CAL
    - «interface»CD
    - «constraint»CD.CV
    - «interface»CEQ
    - «interface...»CLCY
    - «interface»CO
    - «interface...»COLL
    - «interface...»COMP
    - «interface»CS

Types : Classifiers such as Classes, Actors, Activities etc

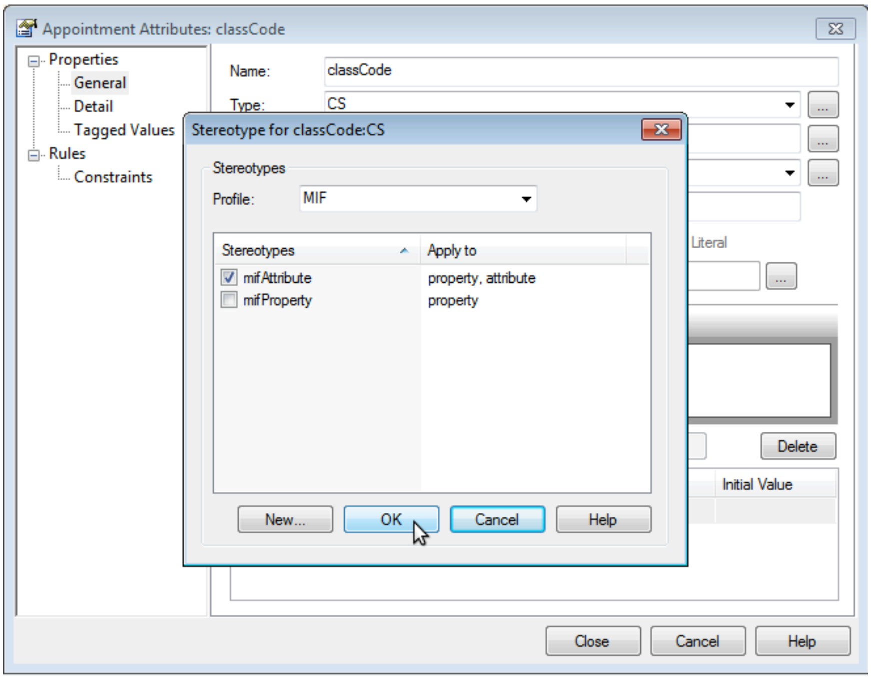
Figure 27. Attributes dialog and selection of a data type from the package of HL7 Abstract Data Types.

The General tab of the Attributes dialog is shown in left side of Figure 27. To review an existing attribute, click on the attribute name in the Attributes panel. To delete an existing attribute, click on the attribute name in the Attributes panel and click on the **Delete** button. To create a new attribute, click on the **New** button, or click on an existing attribute name in the Attributes panel, and click on the **Copy** button.

The major fields of the attribute that can be edited are:

- Name: Display the name of the attribute. For a new attribute, type the name (with no spaces).
- Type: Display the attribute's type. If necessary, click on the drop-down arrow and select a different type. The type can be defined by a data type or by a classifier element. Alternatively, click the **[…]** select button to open a dialog that can be used to select or define a different attribute classifier type that might not be in the Type drop-down list. Figure 27 (right) shows the selection of the type CS from the Abstract Data Types package. If the attribute is complex, type the name of the type directly in the text box, as in the case of IVL<TS> in Figure 31. See more details about this in section 9.5.
- Stereotype: Define the stereotype of the attribute. If necessary, either type a different stereotype name or click on the drop-down arrow and select a stereotype. It is mandatory to apply the «mifAttribute» stereotype in order to include additional semantics not present in common UML attributes that are specified in the attributes present in MIF models (seeFigure 28).
- Notes: Enter any free text notes associated with the attribute.

If you have changed the attribute details, click on the **Save** button to save the changes.



Name: classCode

Type: CS

Stereotypes

Profile: MIF

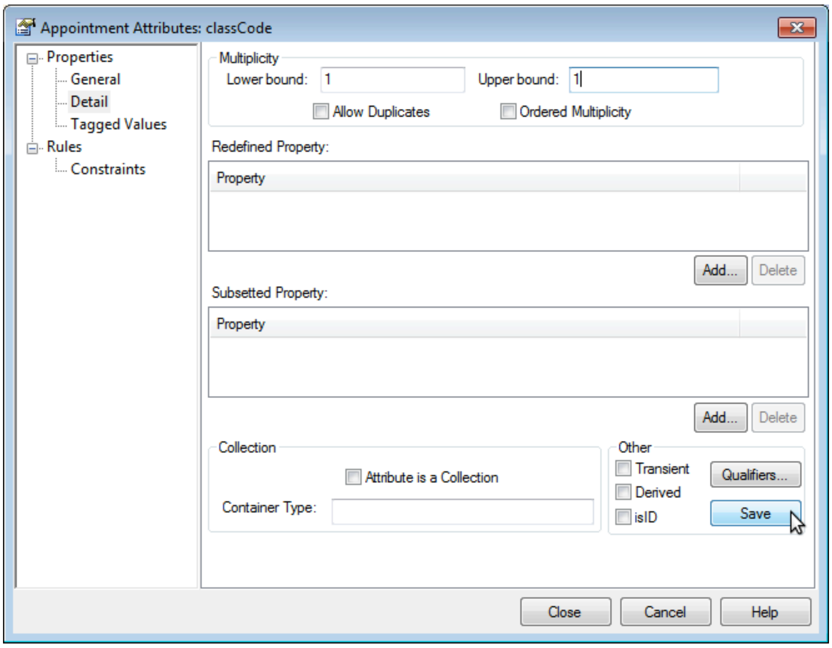
Stereotypes	Apply to
☑ mifAttribute	property, attribute
☐ mifProperty	property

Figure 28. Applying the stereotype «mifAttribute» to UML attributes in a model.

To define additional details relating to attributes, click on the **Detail** tab of the Attributes dialog (see Figure 29). The major fields of the attribute that can be edited in the Detail tab are:

- Multiplicity
  - Lower bound: Define a lower limit to the multiplicity interval of the attribute.
  - Upper bound: Define an upper limit to the multiplicity interval of the attribute.

When you have completed these fields, click on the **Save** button.



Multiplicity

Lower bound: 1 Upper bound: 1

☐ Allow Duplicates ☐ Ordered Multiplicity

Redefined Property:

Property

Subsetted Property:

Property

Collection

☐ Attribute is a Collection

Container Type:

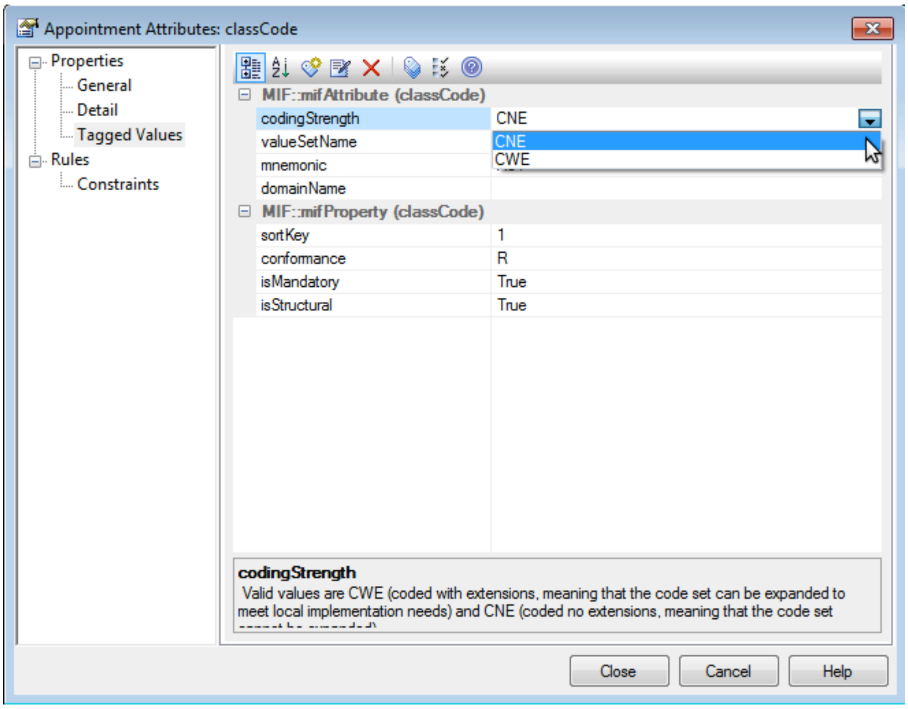
Other

☐ Transient

☐ Derived

☐ isID

Figure 29. Details tab of the Attributes dialog.



MIF::mifAttribute (classCode)

codingStrength	CNE
valueSetName	
mnemonic	
domainName	

CNE
CWE

MIF::mifProperty (classCode)

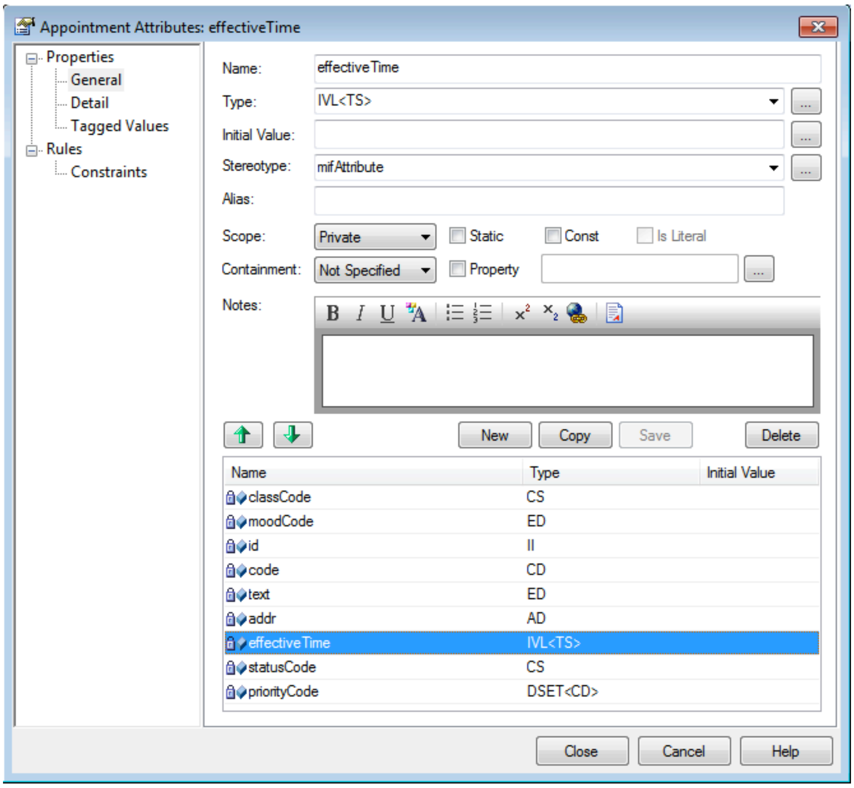
sortKey	1
conformance	R
isMandatory	True
isStructural	True

codingStrength
Valid values are CWE (coded with extensions, meaning that the code set can be expanded to meet local implementation needs) and CNE (coded no extensions, meaning that the code set cannot be expanded)

Figure 30. Setting values to the tagged values of the «mifAttribute» stereotype applied to a UML attribute.

The application of the «mifAttribute» stereotype to the attributes of a class involves setting the properties defined in this stereotype. These properties (see sections 8.6.5 and 8.6.6) specify additional characteristics not present in common UML attributes that are required in MIF attributes. To define the values for these special properties relating to MIF attributes, click on the **Tagged Values** tab of the Attributes dialog (see Figure 30).

Some of the tagged values are defined in the UML profile for MIF static models as enumerations. In these cases, there is a dropdown list showing the possible values, as shown for the codingStrength tagged value in Figure 30.



Name: effectiveTime
Type: IVL<TS>
Initial Value:
Stereotype: mifAttribute
Alias:
Scope: Private □ Static □ Const □ Is Literal
Containment: Not Specified □ Property
Notes:

Name	Type	Initial Value
classCode	CS	
moodCode	ED	
id	II	
code	CD	
text	ED	
addr	AD	
effectiveTime	IVL<TS>	
statusCode	CS	
priorityCode	DSET<CD>	

Figure 31. Selecting a complex type as the type of an attribute.

MIF static models have several constraints on attributes. All the attributes in the model must be conformant with the attributes of the RIM class referenced by the stereotype of the class owning the attribute (see RIM conformance in section 2.2). As an example, Figure 32 (left) shows the invalid definition of the attributes moodCode and addr of the Appointment class. The attribute addr is not an attribute owned by the Act class in the RIM, and the type of the moodCode attribute of the Act class in the RIM is not ED. The RIM Act class defines moodCode with the attribute CS. Figure 32 (right) shows a correct definition of attributes in the Appointment class

All of these constraints are formally defined in the UML Profile for MIF static models as OCL constraints in the stereotypes and metaclasses of section 8. Therefore, any issue in the construction of attributes can be automatically detected taking into account these constraints.

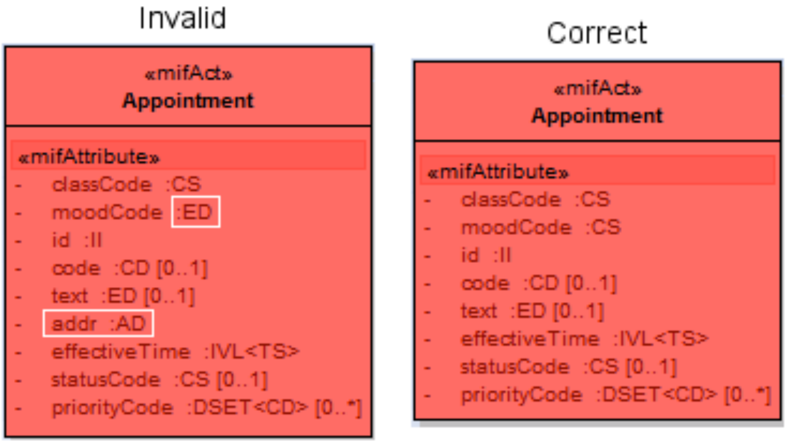


Figure 32. Conformance rules of attributes with respect to the HL7 Reference Information Model.

7.2.8 CMETs

CMETs are a special construct in MIF static models that allows modelers to express a reference to a common, reusable concept that is defined in another static model.

CMETs in UML are represented as common classes with the stereotype «mifCMET» and another stereotype from the Foundation area referencing the kind of CMET. As an example, Figure 33 shows the CMETs R_AssignedPersonUniversal and R_ServiceDeliveryLocationUniversal. Both CMETs are classes with the «mifCMET» stereotype and the «mifRole» stereotype. Thus, the central class defined in the model where the CMET is defined in detail is a class representing a role. For more details about how to represent CMETs in UML see section 9.8.

The stereotype «mifCMET» defines a set of properties (see section 8.4.4) to express additional knowledge present in MIF files. The value of these properties may be set when applying the stereotype to the CMET class. To visualize this tagged values right click in the CMET class and select Feature and Compartment Visibility. Then the Feature Visibility dialog opens (see Figure 15). Deselect the All checkboxes in Attribute Visibility and Operation Visibility and select the Tags checkbox in Show Element Compartments.

Then the tagged values are visible in the class diagram for the package, and their values can be set in the Tagged Values view (see Figure 33). The «mifCMET» stereotype defines the properties codeSystem and mnemonic, which represent the name for the code system in which the supplier structural domain for the CMET is defined and a short mnemonic used to represent the code system within the HL7 vocabulary. Also, the «mifCMET» stereotype includes the generalizationParent property which holds the reference to the static model intended to be resolved, imported and re-used.

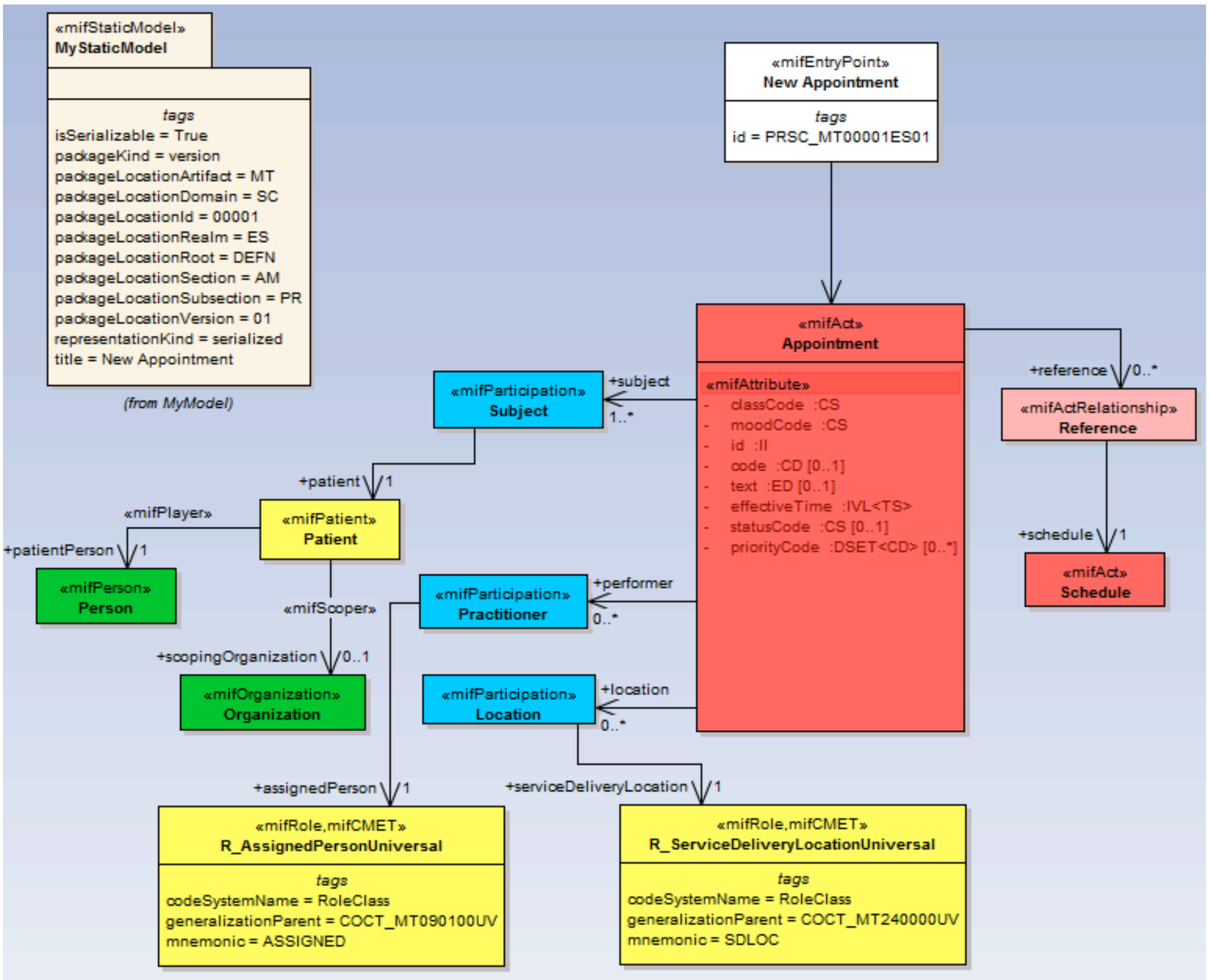
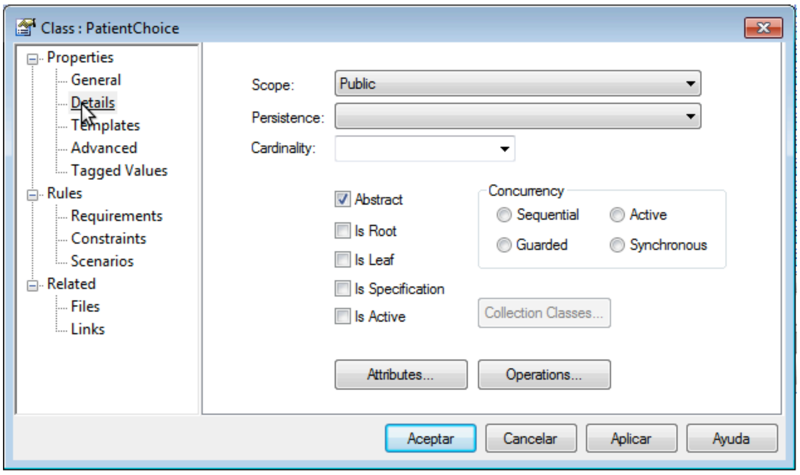


Figure 33. Final view of CMETS R_AssignedPersonUniversal and R_ServiceDeliveryLocationUniversal.

7.2.9 Choices

Choices are special constructs that appear in MIF models to define a hierarchy of elements. In UML, Choices are represented as abstract classes with the stereotype «mifChoice». Choices do not contain attributes; however both the Choice and any class contained in the Choice may participate as the source or target of associations.

To indicate that the Choice class is abstract, click on the **Detail** tab of the Properties dialog (seeFigure 34).



Scope: Public

Persistence:

Cardinality:

☑ Abstract

□ Is Root

□ Is Leaf

□ Is Specification

□ Is Active

Concurrency

○ Sequential ○ Active

○ Guarded ○ Synchronous

Figure 34. Details dialog of Class properties to indicate that the Choice class is abstract.

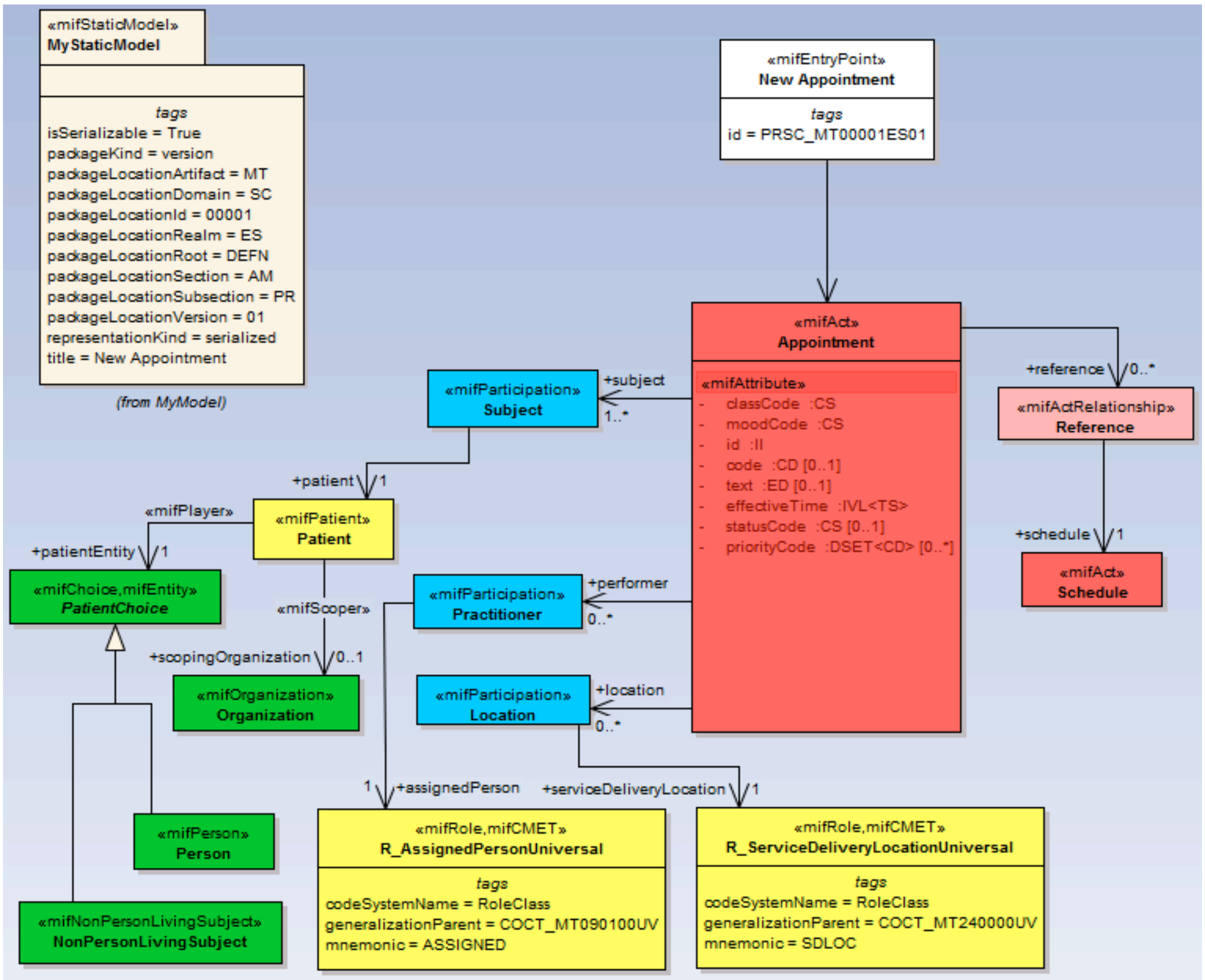


Figure 35. Final View of PatientChoice.

In addition, the Choice class must have a second stereotype from the Foundational area indicating the reference to the RIM class from which the Choice is derived. Figure 24 shows that the class Person plays the role of Patient. Alternatively, Figure 25 introduces the PatientChoice, which also contains the «mifEntity» stereotype to indicate that the choice is between entities. In this case, the player of the role Patient can be a Person or a NonPersonLivingSubject.

All the classes included in the choice are connected to the choice class with a generalization relationship indicating the hierarchy of classes. The choice class is the top-level class in the hierarchy while the classes contained in the choice are represented as subclasses in UML (see more details in section 9.7).

Note that the classes that descend from the choice in the same hierarchy must have stereotypes that are compatible with the stereotype of the choice class. In other words, a class with the stereotype «mifAct» cannot be a descendant of PatientChoice, which is a choice of entities (see constraint [4] of section 8.6.2). Any issue in the construction of choice hierarchies can be automatically detected taking into account the constraints in this specification.

7.2.10 Export to XML Metadata Interchange (XMI)

Modelers can export a package to an XMI (XML-based) file. This enables modelers to move Enterprise Architect model elements between models, for distributed development, manual version control and other benefits. It also enables limited export of Enterprise Architect model elements to Rational Rose and other tools that implement the UML 2.1 XMI 2.1 standard, UML1.4 XMI 1.2 standard, or UML 1.3 XMI 1.1 / XMI 1.0 standard.

To export a package to XMI, follow the steps below:

1. In the Project Browser window, select the package to export. In the example of Figure 36, the package selected is MyStaticModel.

2. Either:

   a. Right-click and select the **Import/Export | Export Package to XMI** context menu option, or

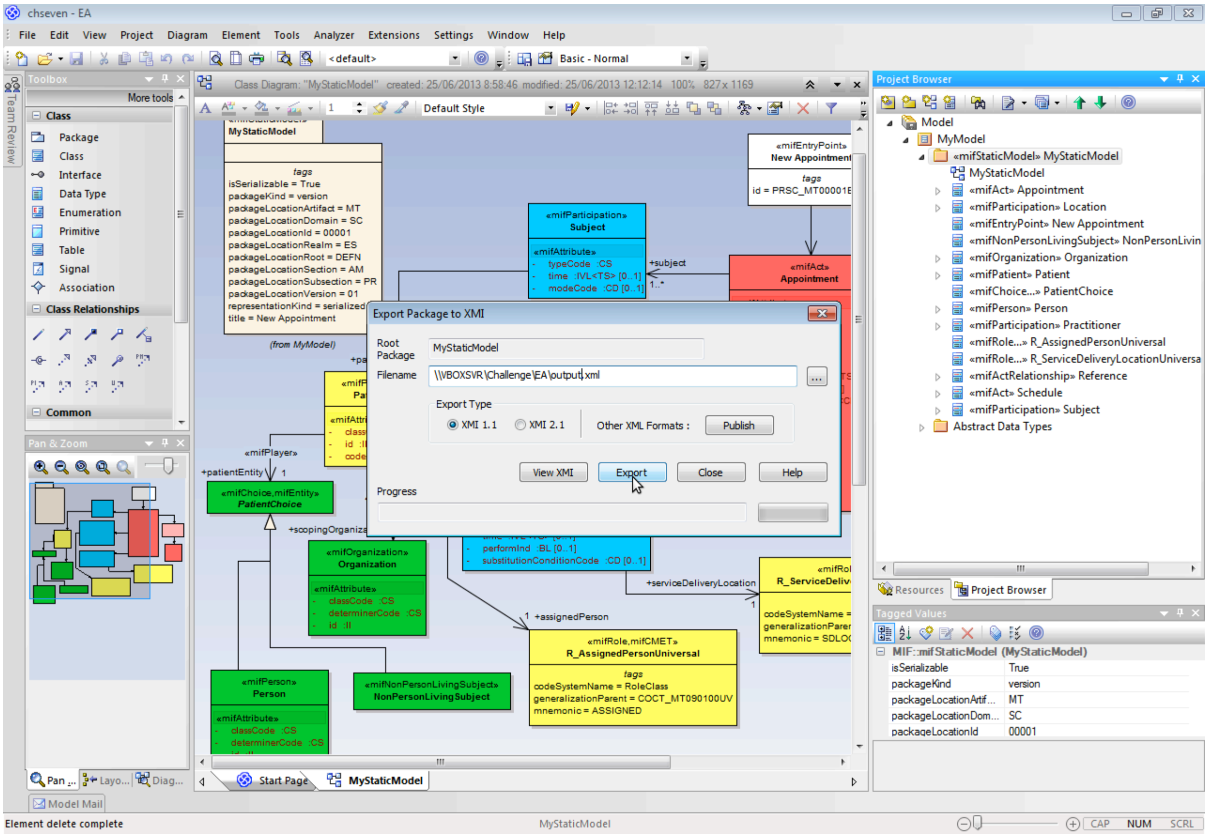
   b. Select the **Project | Import/Export | Export Package to XMI** menu option.

The Export Package to XMI dialog displays, as shown in Figure 36. In the Filename field, type the directory path and filename into which to output the XMI file. In the XMI Type: field, click on the drop-down arrow and select the appropriate XMI format:

- XMI 1.1, to generate output in XMI 1.1 format (necessary if you intend to use this file in a comparison with the model at a later time)

- XMI 2.1, to generate output in XMI 2.1 format.

Finally, click on the Export button.



Export Package to XMI

Root Package: MyStaticModel

Filename: \\VBOXSVR\Challenge\EA\output.xml

Export Type: ● XMI 1.1 ○ XMI 2.1

Progress

Element delete complete

chseven - EA
File Edit View Project Diagram Element Tools Analyzer Extensions Settings Window Help
Toolbox
Class
Package
Class
Interface
Data Type
Enumeration
Primitive
Table
Signal
Association
Class Relationships
Common
Pan & Zoom
Class Diagram: "MyStaticModel" created: 25/06/2013 8:58:46 modified: 25/06/2013 12:12:14 100% 827 x 1169
«mifParticipation» Subject
«mifAct» Appointment
«mifEntryPoint» New Appointment
«mifChoice,mifEntity» PatientChoice
«mifOrganization» Organization
«mifPerson» Person
«mifNonPersonLivingSubject» NonPersonLivingSubject
«mifRole,mifCMET» R_AssignedPersonUniversal
+subject
+assignedPerson
+serviceDeliveryLocation
Other XML Formats :
Publish
View XMI
Export
Close
Help
Project Browser
Model
MyModel
«mifStaticModel» MyStaticModel
Abstract Data Types
Tagged Values
Start Page
MyStaticModel

Figure 36. Export Package to XMI dialog.

7.3 Improvements

This subsection presents some issues found during the development of this specification that may be addressed to improve the different standards and tools involved.

7.3.1 Model Interchange Format (MIF)

MIF fragmentation

Normative editions of HL7 V3 standard contain several versions and distributions of MIF files. Concretely, the 'processable' directory of Disc 2 in the HL7 V3 2012 Normative Edition includes:

- combined package mif – The combined package mif used to generate the html representations of the normative edition contents.
- mif – The complete set of designs represented in HL7 Model Interchange Format (MIF) Release 1.
- mif-lite – The complete set of designs represented in HL7 Model Interchange Format (MIF) Release 1, but stripped of their descriptive documentation. This form of MIF files will be distributed with a license that permits them to be used as configuration files for tools.
- mif2 – The complete set of designs represented in HL7 Model Interchange Format (MIF) Release 2.1.5. (Schemas for this format are distributed as part of the V3 Generator tool.)
- mif2-lite – The complete set of designs represented in HL7 Model Interchange Format (MIF) Release 2.1.5, but stripped of their descriptive documentation. This form of MIF files will be distributed with a license that permits them to be used as configuration files for tools.

This amount of different distributions makes it difficult for non-experienced users and developers to select the correct set of MIF files. A simplification in this area might increase the usability of MIF standard.

Coded attributes

Attributes with a coded type (CD, CS, etc.) presented in MIF files usually indicate the HL7 Vocabulary from which the values of the attribute must be taken. The problem is that depending on the specific type and attribute the name of the XML attribute containing this information can be “valueSetName”, “domainName”, or “codeSystemName”. A consensus in the naming of the property holding the vocabulary or code set for MIF attributes may be convenient.

7.3.2 Enterprise Architect (EA)

Specific support for MIF static models

When creating a new project, Enterprise Architect opens the Model Wizard dialog in order to help users on selecting the kind of models to include in the project. Enterprise Architect may take into account the specific considerations of MIF static models and provide specific options to create MIF based projects directly from the Model Wizard. These actions will simplify the steps to set the modeling environment described in section 7.2.1 of this specification.

Simplification of Properties dialog

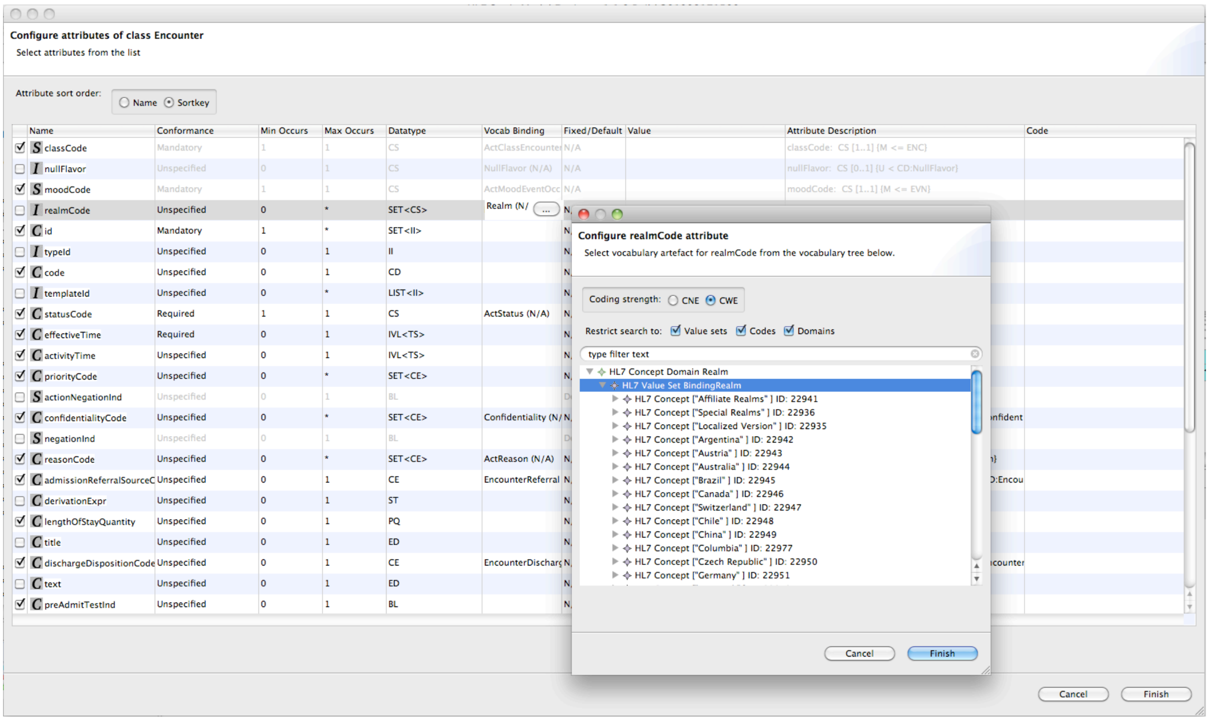
The Properties dialog in Enterprise Architect contains a lot of options and settings that can be changed in order to indicate the characteristics of a specific element. There are several tabs in the Properties dialog to cover a full range of options that usually are not used by common modelers. As an example, a modeler that wants to create an attribute owned by a class, set its multiplicity, apply a stereotype, and set the values of the specific characteristics included in the stereotype must traverse three different tabs through the Properties dialog of UML attributes (see section 7.2.7). A simplification of this dialog showing the most used options in a single, simpler tab, to adapt to the particular options of MIF models will simplify these common modeling activities.

RIM and HL7 Conformance

Section 2.2 introduced that attributes owned by classes in MIF static models must be consistent with the attributes in the Reference Information Model. Enterprise Architect may include the knowledge defined in the

RIM and when creating the attributes of a class with a stereotype referencing a MIF class, the Attributes dialog must show a checkbox list with the attributes of that RIM class. Thus, the modeler can select those attributes of interest to his/her purpose from the list of RIM attributes in order to be consistent with the RIM. Then, the modeler may have the possibility of redefine their multiplicity according to the consistency rules presented in this specification.

An example of tool that follows this process when dealing with attributes is the HL7 Static Model Designer (SMD), as shown in Figure 37. In addition to it, Enterprise Architect should include the list of HL7 Vocabularies and Data Types in order to simplify the configuration of attributes in the same way as HL7 SMD does.



Configure attributes of class Encounter

Select attributes from the list

Attribute sort order: ○ Name ● Sortkey

	Name	Conformance	Min Occurs	Max Occurs	Datatype	Vocab Binding	Fixed/Default	Value	Attribute Description	Code
☑	S classCode	Mandatory	1	1	CS	ActClassEncounter	N/A		classCode: CS [1..1] {M <= ENC}	
□	I nullFlavor	Unspecified	0	1	CS	NullFlavor (N/A)	N/A		nullFlavor: CS [0..1] {U < CD:NullFlavor}	
☑	S moodCode	Mandatory	1	1	CS	ActMoodEventOcc	N/A		moodCode: CS [1..1] {M <= EVN}	
□	I realmCode	Unspecified	0	*	SET<CS>	Realm (N/ ...	N			
☑	C id	Mandatory	1	*	SET<II>		N			
□	I typeId	Unspecified	0	1	II		N			
☑	C code	Unspecified	0	1	CD		N			
□	I templateId	Unspecified	0	*	LIST<II>		N			
☑	C statusCode	Required	1	1	CS	ActStatus (N/A)	N			
☑	C effectiveTime	Required	0	1	IVL<TS>		N			
☑	C activityTime	Unspecified	0	1	IVL<TS>		N			
☑	C priorityCode	Unspecified	0	*	SET<CE>		N			
□	S actionNegationInd	Unspecified	0	1	BL		D			
☑	C confidentialityCode	Unspecified	0	*	SET<CE>	Confidentiality (N/	N		nfident	
□	S negationInd	Unspecified	0	1	BL		D			
☑	C reasonCode	Unspecified	0	*	SET<CE>	ActReason (N/A)	N		n}	
☑	C admissionReferralSourceC	Unspecified	0	1	CE	EncounterReferral	N		D:Encou	
□	C derivationExpr	Unspecified	0	1	ST		N			
☑	C lengthOfStayQuantity	Unspecified	0	1	PQ		N			
□	C title	Unspecified	0	1	ED		N			
☑	C dischargeDispositionCode	Unspecified	0	1	CE	EncounterDischarg	N		counter	
□	C text	Unspecified	0	1	ED		N			
☑	C preAdmitTestInd	Unspecified	0	1	BL		N			

Configure realmCode attribute

Select vocabulary artefact for realmCode from the vocabulary tree below.

Coding strength: ○ CNE ● CWE

Restrict search to: ☑ Value sets ☑ Codes ☑ Domains

type filter text

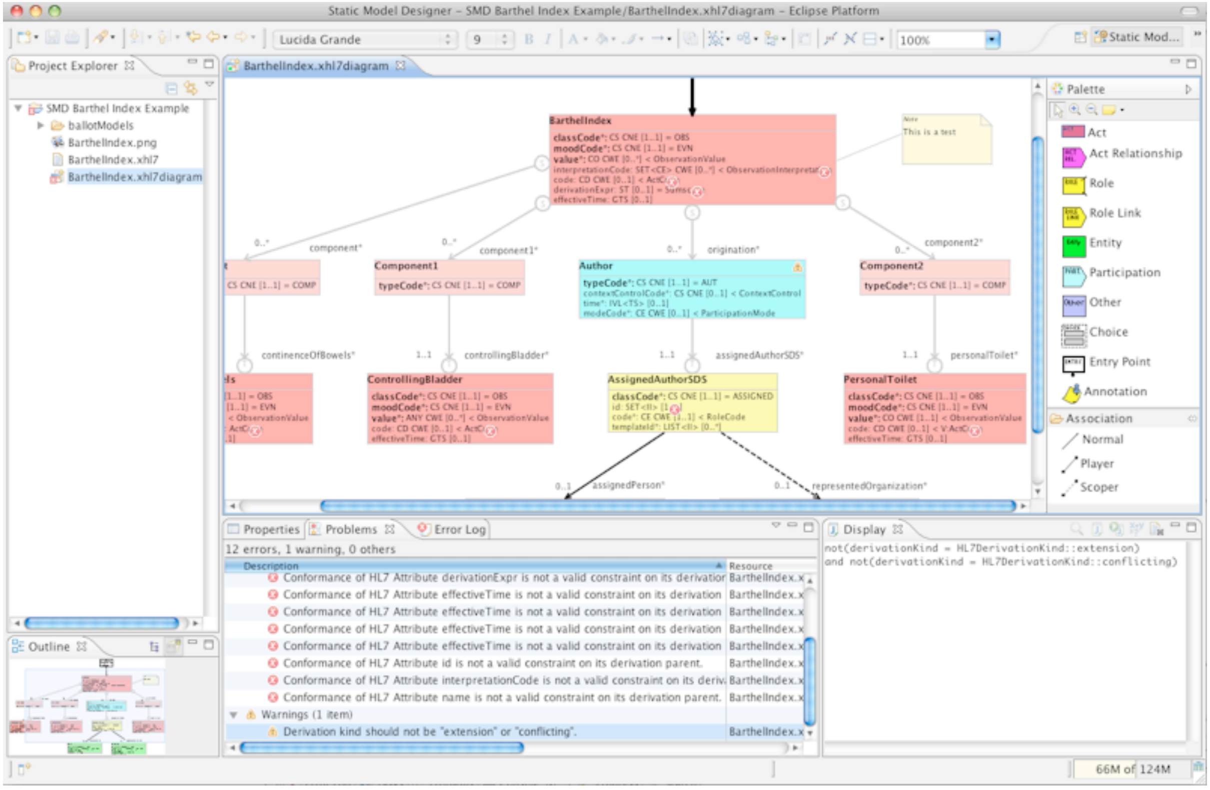
- HL7 Concept Domain Realm
  - HL7 Value Set BindingRealm
    - HL7 Concept ["Affiliate Realms"] ID: 22941
    - HL7 Concept ["Special Realms"] ID: 22936
    - HL7 Concept ["Localized Version"] ID: 22935
    - HL7 Concept ["Argentina"] ID: 22942
    - HL7 Concept ["Austria"] ID: 22943
    - HL7 Concept ["Australia"] ID: 22944
    - HL7 Concept ["Brazil"] ID: 22945
    - HL7 Concept ["Canada"] ID: 22946
    - HL7 Concept ["Switzerland"] ID: 22947
    - HL7 Concept ["Chile"] ID: 22948
    - HL7 Concept ["China"] ID: 22949
    - HL7 Concept ["Columbia"] ID: 22977
    - HL7 Concept ["Czech Republic"] ID: 22950
    - HL7 Concept ["Germany"] ID: 22951

Figure 37. Screenshot of attributes configurator dialog of HL7 Static Model Designer.

Internal usage of Stereotypes

Enterprise Architect could hide the use of stereotypes from the UML Profile for MIF static models by creating specific actions in the GUI of the tool. As an example, a modeler that wants to create an entry point needs to create a common class, and then set the stereotype «mifEntryPoint». A more user-friendly and MIF-adapted graphical user interface should include a toolbox similar to the one of HL7 Static Model Designer depicted in Figure 38 (right side), which directly includes action icons to create entry points, choices, and more.

By encapsultaing the use of stereotypes, inexperienced users can easily construct static models in a similar way as experienced modelers can do. In addition to it, Enterprise Architect could support the OCL constraints in this specification related to stereotypes in order to check the correctness of MIF models expressed in UML.



12 errors, 1 warning, 0 others

Description	Resource
Conformance of HL7 Attribute derivationExpr is not a valid constraint on its derivatior	BarthelIndex.x
Conformance of HL7 Attribute effectiveTime is not a valid constraint on its derivation	BarthelIndex.x
Conformance of HL7 Attribute effectiveTime is not a valid constraint on its derivation	BarthelIndex.x
Conformance of HL7 Attribute effectiveTime is not a valid constraint on its derivation	BarthelIndex.x
Conformance of HL7 Attribute effectiveTime is not a valid constraint on its derivation	BarthelIndex.x
Conformance of HL7 Attribute id is not a valid constraint on its derivation parent.	BarthelIndex.x
Conformance of HL7 Attribute interpretationCode is not a valid constraint on its deriv.	BarthelIndex.x
Conformance of HL7 Attribute name is not a valid constraint on its derivation parent.	BarthelIndex.x
Warnings (1 item)	
Derivation kind should not be "extension" or "conflicting".	BarthelIndex.x

```
not(derivationKind = HL7DerivationKind::extension)
and not(derivationKind = HL7DerivationKind::conflicting)
```

Figure 38. Screenshot of HL7 Static Model Designer.

8 MIF-UML Profile Reference

8.1 Overview

UML contains an extensibility capability that should accommodate most of the day-to-day needs for extensions, without requiring a change to the basic language. A stereotype is a new kind of model element with the same structure as an existing element, but with additional constraints, a different interpretation and icon, and different treatment by code generators and other back-end tools.

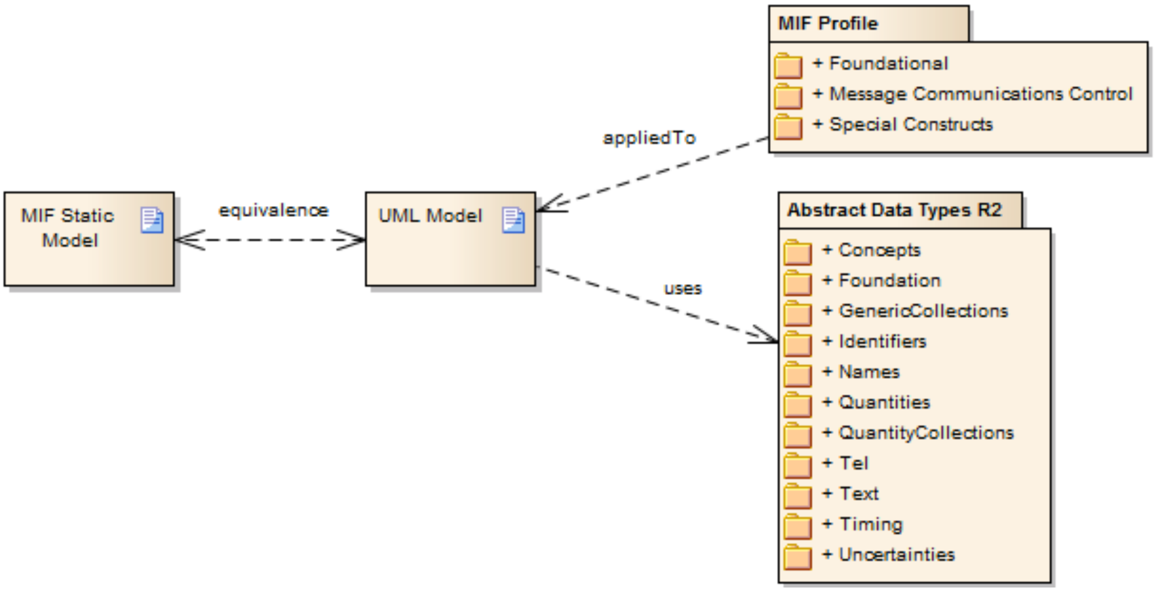


Figure 39. Overview of the MIF-UML Profile.

Stereotypes and their properties are defined in profiles. A profile is a package to group extensions intended to tailor the model toward a particular domain, technology, or implementation. Profiles can be applied to packages to extend the elements in them.

Figure 39 presents the main structure that result from the application of the MIF profile to a UML model in order to extend its elements to be able to represent the same information as in a MIF static model. Note that the UML model uses the Abstract Data Types R2 specification to represent the data types of the attributes owned by classes. Following this structure it is possible to construct equivalent MIF static models directly in UML.

The next subsections define the stereotypes contained within the three main areas of the MIF profile, including their definition, the generalization and specialization relationships, the extended metaclasses, the enumerations required, and the constraints that ensure the correctness of the profile application. Note that each stereotype name contains the prefix “mif” to avoid name collision with other stereotypes from existing profiles.

8.2 Foundational Area

8.2.1 Overview

The Foundational Area of the UML Profile for MIF Static Models comprises stereotypes that represent the "normative" content of the HL7 RIM. The UML metamodel subset covered by the Foundational Area only includes the metaclass Class, which is specifically extended by the InfrastructureRoot stereotype. It represents an abstract super-type for all RIM classes, either directly or through inheritance. Figure 40 shows the RIM-based stereotypes from the foundational area extending the metaclass Class. The color of the stereotype box is significant as it identifies the RIM class referenced by the stereotype. The Act related stereotypes are red, the Entity related stereotypes are green and the Role related stereotypes are yellow. These color guidelines are defined in the HL7 RIM.

An extension is a relationship from a stereotype to a UML metaclass (a class in the definition of the UML model itself, such as Class or Association). It indicates that the properties defined in the stereotype may be applied to instances of the metaclass that bear the stereotype. Note that an instance of a metaclass is an ordinary class as defined in a user model, so defining an extension to a metaclass means that a user-model element of the given metaclass has additional modeling-time properties whose values the modeler can specify.

A solid arrow shows an extension from the stereotype rectangle to the metaclass rectangle with a filled black arrowhead on the class end of the arrow. The keyword «metaclass» may be placed in the metaclass rectangle.

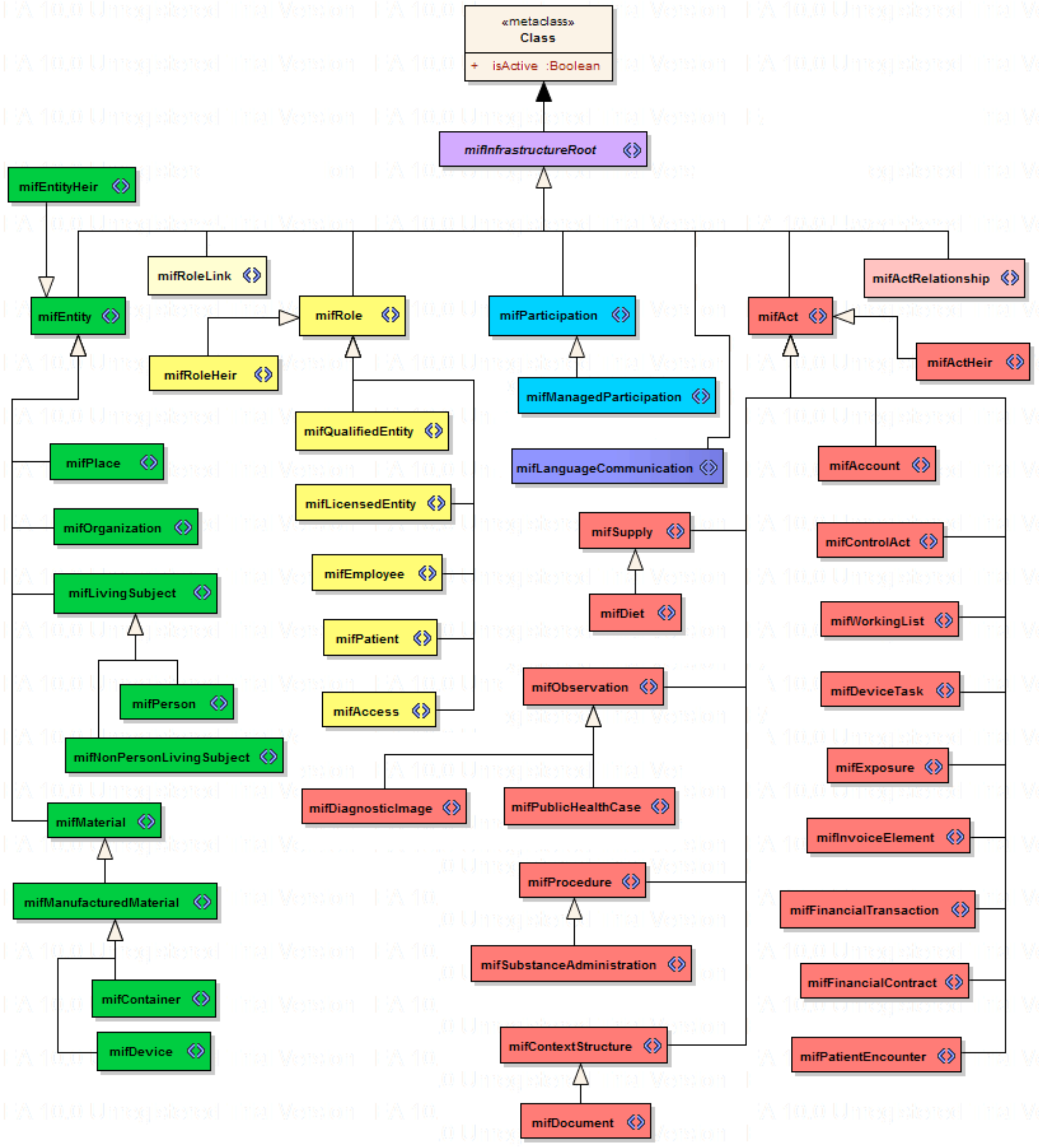


Figure 40. Stereotypes of the Foundational Area.

8.2.2 «stereotype» mifAccess

Generalization «stereotype» mifRole

Description
A role played by a device when the device is used to administer therapeutic agents (medication and vital elements) into the body, or to drain material (e.g., exudates, pus, urine, air, blood) out of the body.

8.2.3 «stereotype» mifAccount

Generalization «stereotype» mifAct

Description
A set of financial transactions that are tracked and reported together with a single balance.

8.2.4 «stereotype» mifAct

Generalization «stereotype» mifInfrastructureRoot

Specializations

- «stereotype» mifActHeir
- «stereotype» mifAccount
- «stereotype» mifSupply
- «stereotype» mifObservation
- «stereotype» mifDeviceTask
- «stereotype» mifControlAct
- «stereotype» mifWorkingList
- «stereotype» mifProcedure
- «stereotype» mifExposure
- «stereotype» mifInvoiceElement
- «stereotype» mifFinancialTransaction
- «stereotype» mifFinancialContract
- «stereotype» mifPatientEncounter
- «stereotype» mifContextStructure

Description
A record of something that is being done, has been done, can be done, or is intended or requested to be done.

8.2.5 «stereotype» mifActHeir

Generalization «stereotype» mifAct

Description
A subtype of Act defined solely as a work-around for the lack of support of the reflexive closure of generalization relationships (i.e. "Act is-a Act") by the current set of tools.

8.2.6 «stereotype» mifActRelationship

Generalization «stereotype» mifInfrastructureRoot

Description
A directed association between a source Act and a target Act.

8.2.7 «stereotype» mifContainer

Generalization «stereotype» mifManufacturedMaterial

Description
An Entity that holds other Entities.

8.2.8 «stereotype» mifContextStructure

Generalization «stereotype» mifAct

Specialization «stereotype» mifDocument

Description
A container within a document.

8.2.9 «stereotype» mifControlAct

Generalization «stereotype» mifAct

Description
An act representing a change to the state of another class, a user event (e.g. query), or a system event (e.g. time-based occurrences).

8.2.10 «stereotype» mifDevice

Generalization «stereotype» mifManufacturedMaterial

Description
A ManufacturedMaterial used in an activity without being substantially changed through that activity.

8.2.11 «stereotype» mifDeviceTask

Generalization «stereotype» mifAct

Description
An activity of an automated system.

8.2.12 «stereotype» mifDiagnosticImage

Generalization «stereotype» mifObservation

Description
An observation in the form of a spatial representational of a physical subject suitable for visual presentation.

8.2.13 «stereotype» mifDiet

Generalization «stereotype» mifSupply

Description
A supply Act dealing specifically with the feeding or nourishment of a subject.

8.2.14 «stereotype» mifDocument

Generalization «stereotype» mifContextStructure

Description
A specialization of Act that supports the characteristics unique to document management services.

8.2.15 «stereotype» mifEmployee

Generalization «stereotype» mifRole

Description
A role played by a person who is associated with an organization to receive wages or salary.

8.2.16 «stereotype» mifEntity

Generalization «stereotype» mifInfrastructureRoot

Specializations

- «stereotype» mifLivingSubject
- «stereotype» mifPlace
- «stereotype» mifMaterial
- «stereotype» mifOrganization
- «stereotype» mifEntityHeir

Description
A physical thing, group of physical things or an organization capable of participating in Acts while in a role.

8.2.17 «stereotype» mifEntityHeir

Generalization «stereotype» mifEntity

Description
A subtype of Entity defined solely as a work-around for the lack of support of the reflexive closure of generalization relationships (i.e. "Entity is-an Entity") by the current set of tools.

8.2.18 «stereotype» mifExposure

Generalization «stereotype» mifAct

Description
An interaction between entities that provides opportunity for transmission of a physical, chemical, or biological agent from an exposure source entity to an exposure target entity.

8.2.19 «stereotype» mifFinancialContract

Generalization «stereotype» mifAct

Description
A contract whose value is measured in monetary terms.

8.2.20 «stereotype» mifFinancialTransaction

Generalization «stereotype» mifAct

Description
An Act representing the movement of a monetary amount between two accounts.

8.2.21 «stereotype» mifInfrastructureRoot

Extends «metaclass**»** Class

Specializations

- «stereotype» mifEntity
- «stereotype» mifRoleLink
- «stereotype» mifRole
- «stereotype» mifParticipation
- «stereotype» mifActRelationship
- «stereotype» mifAct
- «stereotype» mifTransmissionRelationship
- «stereotype» mifAcknowledgement
- «stereotype» mifEntryPoint
- «stereotype» mifAcknowledgementDetail

- «stereotype» mifAttachment
- «stereotype» mifAttentionLine
- «stereotype» mifCommunicationForm
- «stereotype» mifTransmission
- «stereotype» mifSortControl
- «stereotype» mifParameter
- «stereotype» mifQueryEvent
- «stereotype» mifChoice
- «stereotype» mifCMET

Description

An abstract super-type for all RIM classes, either directly or through inheritance.

Constraints

[1] The attribute properties of the base class where the stereotype is applied must be consistent with the attribute properties of the RIM class referenced by the stereotype.

```
context mifInfrastructureRoot inv:
         let rimClass:Class = self.getReferencedRIMClass()
  in
  self.base_Class.ownedAttribute->forAll(a |
      if rimClass.ownedAttribute->one(p|p.name = a.name)
         then a.isConsistentWith(rimClass.ownedAttribute->any(p|p.name
= self.name))
      else true endif
    )
```

Additional Operations

[1] The query getReferencedRIMClass gives the RIM class referenced by the stereotype.

```
context mifInfrastructureRoot
def: getReferencedRIMClass():Class =
  let nameRIM:String = self.name.substring(4, self.name.size())
  in Package.allInstances()->any(p|p.name='RIM')
      ->ownedMember->select(m|m.oclIsTypeOf(Class))
          ->any(c|c.oclAsType(Class).name=nameRIM)
```

8.2.22 «stereotype» mifInvoiceElement

Generalization «stereotype» mifAct

Description

A statement and explanation of an amount owed.

8.2.23 «stereotype» mifLanguageCommunication

Generalization «stereotype» mifInfrastructureRoot

Description

The language communication capabilities of an Entity.

8.2.24 «stereotype» mifLicensedEntity

Generalization «stereotype» mifRole

Description

An Entity that is accredited with a license or qualification certifying the capability to perform specific functions.

8.2.25 «stereotype» mifLivingSubject

Generalization «stereotype» mifEntity

Specializations

- «stereotype» mifPerson
- «stereotype» mifNonPersonLivingSubject

Description
An organism, alive or not.

8.2.26 «stereotype» mifManagedParticipation

Generalization «stereotype» mifParticipation

Description
A participation that will be operated on over time and thus whose state and identity must be managed.

8.2.27 «stereotype» mifManufacturedMaterial

Generalization «stereotype» mifMaterial

Specializations

- «stereotype» mifDevice
- «stereotype» mifContainer

Description
An Entity or combination of Entities transformed for a particular purpose by a manufacturing process.

8.2.28 «stereotype» mifMaterial

Generalization «stereotype» mifEntity

Specialization «stereotype» mifManufacturedMaterial

Description
A subtype of Entity that is inanimate and locationally independent.

8.2.29 «stereotype» mifNonPersonLivingSubject

Generalization «stereotype» mifLivingSubject

Description
A subtype of LivingSubject that includes all living things except the species homo sapiens.

8.2.30 «stereotype» mifObservation

Generalization «stereotype» mifAct

Specializations

- «stereotype» mifPublicHealthCase
- «stereotype» mifDiagnosticImage

Description
An act that is intended to result in new information about a subject.

8.2.31 «stereotype» mifOrganization

Generalization «stereotype» mifEntity

Description
An Entity representing a formalized group of persons or other organizations with a common purpose and the infrastructure to carry out that purpose.

8.2.32 «stereotype» mifParticipation

Generalization «stereotype» mifInfrastructureRoot

Specialization «stereotype» mifManagedParticipation

Description
An association between an Act and a Role. The Entity playing the Role is the actor.

8.2.33 «stereotype» mifPatient

Generalization «stereotype» mifRole

Description
A LivingSubject as a recipient of health care services from a healthcare provider.

8.2.34 «stereotype» mifPatientEncounter

Generalization «stereotype» mifAct

Description
An interaction between a patient and care provider(s) for the purpose of providing healthcare-related service(s).

8.2.35 «stereotype» mifPerson

Generalization «stereotype» mifLivingSubject

Description
A human being.

8.2.36 «stereotype» mifPlace

Generalization «stereotype» mifEntity

Description
A bounded physical place or site, including any contained structures.

8.2.37 «stereotype» mifProcedure

Generalization «stereotype» mifAct

Specializations «stereotype» mifSubstanceAdministration

Description
An Act whose immediate and primary outcome (post-condition) is the alteration of the physical condition of the subject.

8.2.38 «stereotype» mifPublicHealthCase

Generalization «stereotype» mifObservation

Description
An Observation representing a condition or event, or a set of conditions or events, that has a specific significance for a population or community of persons.

8.2.39 «stereotype» mifQualifiedEntity

Generalization «stereotype» mifRole

Description
An entity that has been recognized as having certain training, experience or other characteristics that would make the entity an appropriate performer for a certain activity.

8.2.40 «stereotype» mifRole

Generalization «stereotype» mifInfrastructureRoot

Specializations

- «stereotype» mifEmployee
- «stereotype» mifPatient
- «stereotype» mifLicensedEntity
- «stereotype» mifAccess
- «stereotype» mifQualifiedEntity
- «stereotype» mifRoleHeir

Description
A competency of the Entity that plays the Role as identified, defined, guaranteed, or acknowledged by the Entity that scopes the Role.

8.2.41 «stereotype» mifRoleHeir

Generalization «stereotype» mifRole

Description
A subtype of Role defined solely as a work-around for the lack of support of the reflexive closure of generalization relationships (i.e. "Role is-a Role") by the current set of tools.

8.2.42 «stereotype» mifRoleLink

Generalization «stereotype» mifInfrastructureRoot

Description
A connection between two roles expressing a dependency between those roles and permitting the authorization or nullification of a dependent role based on status changes in its causal or directing role. The RoleLink may be operated over time and thus whose state and identity must be managed.

8.2.43 «stereotype» mifSubstanceAdministration

Generalization «stereotype» mifProcedure

Description
A type of procedure that involves a performer introducing or otherwise applying a material into or to the subject.

8.2.44 «stereotype» mifSupply

Generalization «stereotype» mifAct

Specialization «stereotype» mifDiet

Description
Provision of a material by one entity to another.

8.2.45 «stereotype» mifWorkingList

Generalization «stereotype» mifAct

Description
A dynamic list of individual instances of Act which reflect the need to view groups of Acts for clinical or administrative reasons.

8.3 Message Communications Control Area

8.3.1 Overview

The Message Communications Control Area of the UML Profile for MIF Static Models comprises a collection of stereotypes to represent the RIM classes related to the technical definition and control of message-based communication in HL7. It includes those RIM elements involved in the control, communication and acknowledgement of messages and those classes necessary to formulate, communicate and respond to query messages. Figure 41 shows the RIM-based stereotypes from the message communications control area extending the metaclass Class.

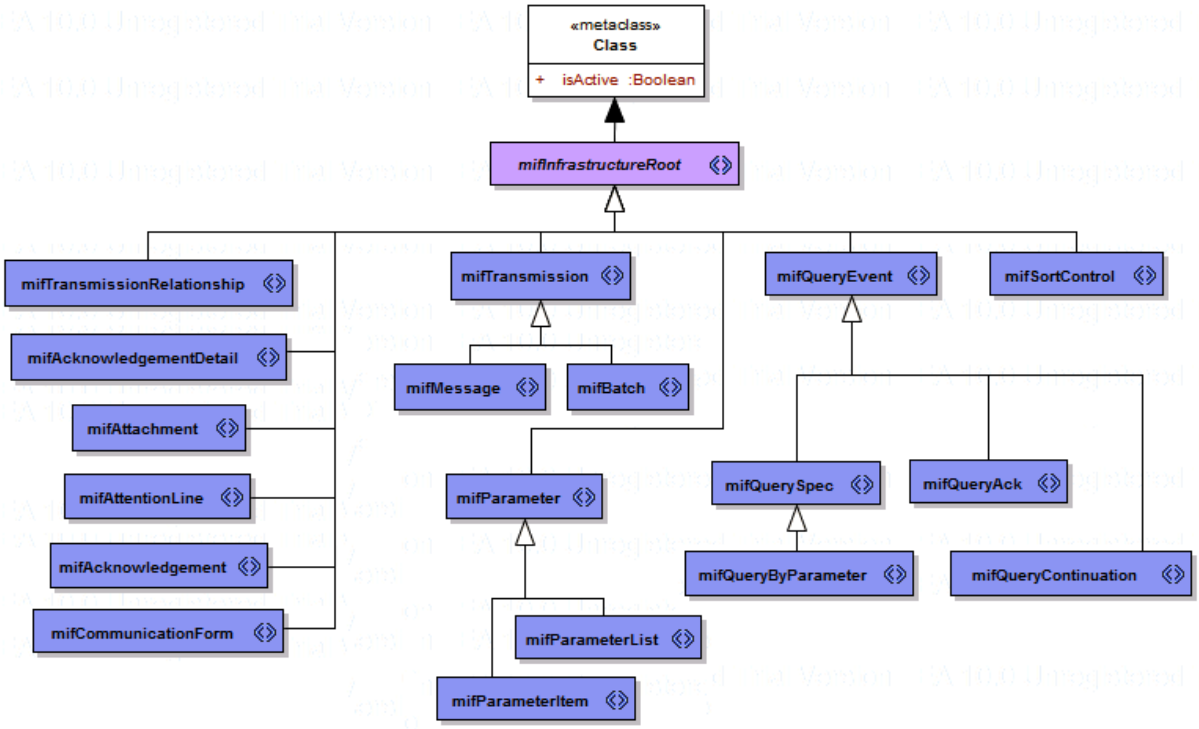


Figure 41. Stereotypes of the Message Communications Control Area.

8.3.2 «stereotype» mifAcknowledgement

Generalization «stereotype» mifInfrastructureRoot

Description
Metadata necessary when acknowledging a message.

8.3.3 «stereotype» mifAcknowledgementDetail

Generalization «stereotype» mifInfrastructureRoot

Description

A message that provides information about the communication, parsing or formal (non-business-rule) validation of the message being acknowledged.

8.3.4 «stereotype» mifAttachment

Generalization «stereotype» mifInfrastructureRoot

Description
An addressable data block which can be referred to from the interior of the message.

8.3.5 «stereotype» mifAttentionLine

Generalization «stereotype» mifInfrastructureRoot

Description
A collection of parameters related to a transmission that may need to be accessible from the transmission wrapper.

8.3.6 «stereotype» mifBatch

Generalization «stereotype» mifTransmission

Description
A message which is a collection of HL7 V3 messages.

8.3.7 «stereotype» mifCommunicationForm

Generalization «stereotype» mifInfrastructureRoot

Description
A relationship class that binds the various entities participating in the transmission (sender, receiver, respond-to) to be linked to the transmission.

8.3.8 «stereotype» mifMessage

Generalization «stereotype» mifTransmission

Description
The parent class of all HL7 Version 3 messages.

8.3.9 «stereotype» mifParameter

Generalization «stereotype» mifInfrastructureRoot

Specializations

- «stereotype» mifParameterItem
- «stereotype» mifParameterList

Description
A uniquely identifiable value or set of values to be used as a query criterion.

8.3.10 «stereotype» mifParameterItem

Generalization «stereotype» mifParameter

Description
A valued element structure (name-value pair) for an element specified in a query.

8.3.11 «stereotype» mifParameterList

Generalization «stereotype» mifParameter

Description
A named list of parameters.

8.3.12 «stereotype» mifQueryAck

Generalization «stereotype» mifQueryEvent

Description
A response to a query.

8.3.13 «stereotype» mifQueryByParameter

Generalization «stereotype» mifQuerySpec

Description
An HL7 query format proposed to replace the QRD/QRF query format.

8.3.14 «stereotype» mifQueryContinuation

Generalization «stereotype» mifQueryEvent

Description
A request for further data.

8.3.15 «stereotype» mifQueryEvent

Generalization «stereotype» mifInfrastructureRoot

Specializations

- «stereotype» mifQueryAck
- «stereotype» mifQueryContinuation
- «stereotype» mifQuerySpec

Description
An abstract class that generalizes all query message interactions.

8.3.16 «stereotype» mifQuerySpec

Generalization «stereotype» mifQueryEvent

Specialization «stereotype» mifQueryByParameter

Description
The criteria and response expectations to be applied in the query.

8.3.17 «stereotype» mifSortControl

Generalization «stereotype» mifInfrastructureRoot

Description
The sort order for instance matches to a query.

8.3.18 «stereotype» mifTransmission

Generalization «stereotype» mifInfrastructureRoot

Specializations

- «stereotype» mifBatch
- «stereotype» mifMessage

Description

Information about a specific transmission of information from one application to another.

8.3.19 «stereotype» mifTransmissionRelationship

Generalization «stereotype» mifInfrastructureRoot

Description

A directed association between a source Transmission and a target Transmission.

8.4 Special Constructs Area

8.4.1 Overview

The Special Constructs Are of the UML Profile for MIF Static Models comprises a collection of stereotypes to represent the specific constructions included in the MIF representation of a static model that do not exist in the UML modeling language. It includes, among others, Common Message Element Types (CMETs) that are references to model fragments used in several models, Choices that represent hierarchies of compatible elements, or entry points that indicate which is the central element in a model. Figure 42 shows the stereotypes from the special constructs area extending the metaclasses Class, Association, Package, and Property. Additionally, we include the metaclass Attribute to conform to Enterprise Architect requirements (although this metaclass does not belong to UML 2.4.1 metamodel).

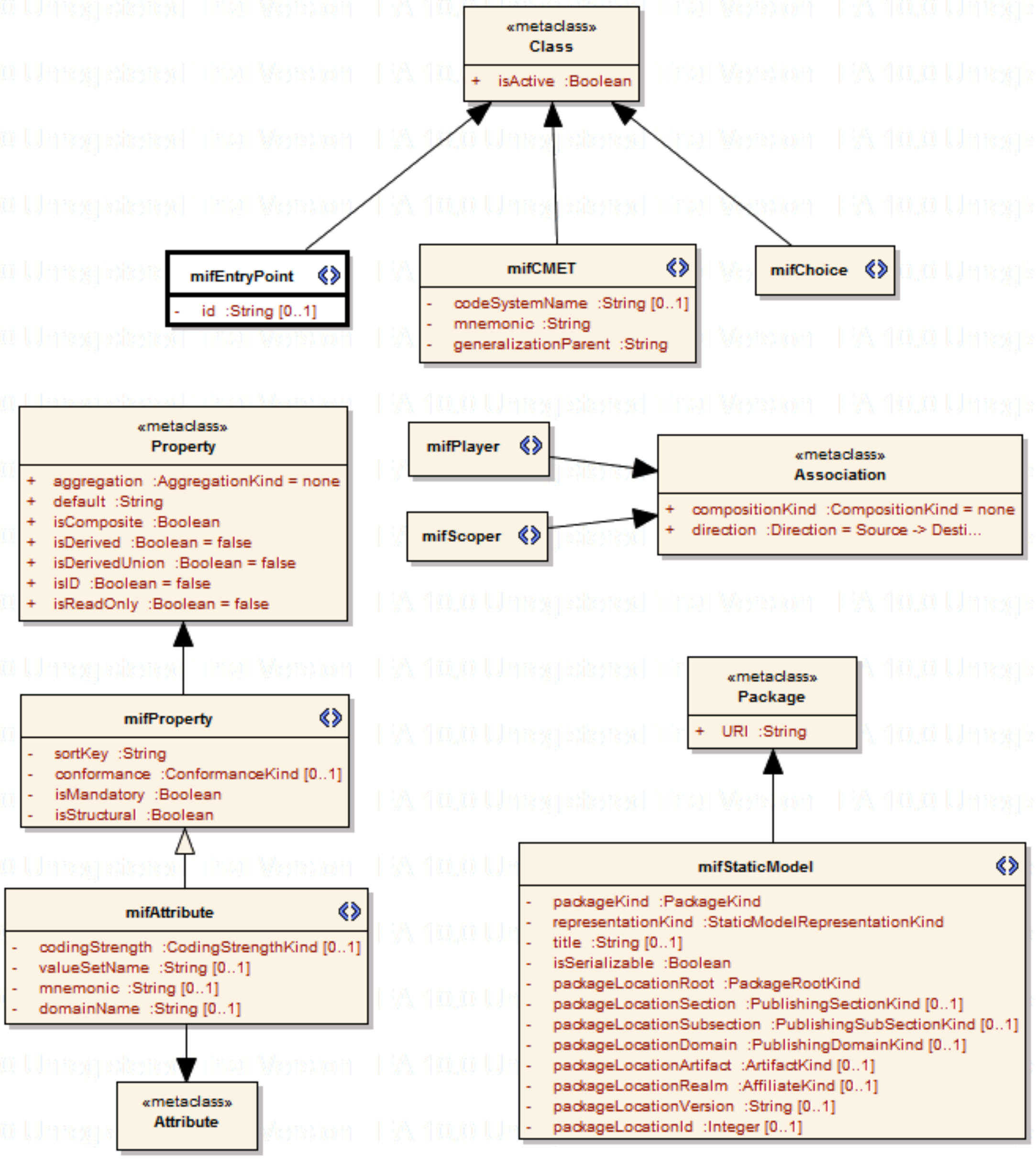

«metaclass»
Class
+ isActive :Boolean
mifEntryPoint
- id :String [0..1]
mifCMET
- codeSystemName :String [0..1]
- mnemonic :String
- generalizationParent :String
mifChoice
«metaclass»
Property
+ aggregation :AggregationKind = none
+ default :String
+ isComposite :Boolean
+ isDerived :Boolean = false
+ isDerivedUnion :Boolean = false
+ isID :Boolean = false
+ isReadOnly :Boolean = false
mifPlayer
mifScoper
«metaclass»
Association
+ compositionKind :CompositionKind = none
+ direction :Direction = Source -> Desti...
«metaclass»
Package
+ URI :String
mifProperty
- sortKey :String
- conformance :ConformanceKind [0..1]
- isMandatory :Boolean
- isStructural :Boolean
mifAttribute
- codingStrength :CodingStrengthKind [0..1]
- valueSetName :String [0..1]
- mnemonic :String [0..1]
- domainName :String [0..1]
«metaclass»
Attribute
mifStaticModel
- packageKind :PackageKind
- representationKind :StaticModelRepresentationKind
- title :String [0..1]
- isSerializable :Boolean
- packageLocationRoot :PackageRootKind
- packageLocationSection :PublishingSectionKind [0..1]
- packageLocationSubsection :PublishingSubSectionKind [0..1]
- packageLocationDomain :PublishingDomainKind [0..1]
- packageLocationArtifact :ArtifactKind [0..1]
- packageLocationRealm :AffiliateKind [0..1]
- packageLocationVersion :String [0..1]
- packageLocationId :Integer [0..1]

Figure 42. Stereotypes of the Special Constructs Area.

8.4.2 «stereotype» mifAttribute

Extends «metaclass**»** Attribute

Generalization «stereotype» mifProperty

Description

An attribute represents a data definition for an instance of a classifier. An attribute describes a range of values for that data definition.

Attributes

- codingStrength: CodingStrengthKind [0..1]

  Valid values are CWE (coded with extensions, meaning that the code set can be expanded to meet local implementation needs) and CNE (coded no extensions, meaning that the code set cannot be expanded).

- valueSetName: String [0..1]

  The descriptive name for the code system in which the supplier domain specification for the attribute is defined.

- mnemonic: String [0..1]
  A short mnemonic used to represent the code system within the HL7 vocabulary.
- domainName: String [0..1]
  The descriptive name for the code system in which the supplier domain specification for the attribute is defined.

8.4.3 «stereotype» mifChoice

Generalization «stereotype» mifInfrastructureRoot

Description
Construction that encloses two or more elements that are part of an inheritance hierarchy (e.g. two or more Roles, two or more Entities, etc.). It is important to note that any annotations or associations connected to the Choice apply to all elements within it. Associations connected to a specific element within the choice box apply only to that element.

8.4.4 «stereotype» mifCMET

Generalization «stereotype» mifInfrastructureRoot

Description
CMETs (Common Message Element Types) are a reference to an external static model intended to be resolved, imported and re-used. They are not intended to be used as a mechanism to simplify a D-MIM or R-MIM diagram, however. This means that a CMET should be expressed diagrammatically as a fully expanded instance diagram, in the D-MIM, where it is defined.

Attributes

- codeSystemName: String [0..1]
  The descriptive name for the code system in which the supplier structural domain for the CMET is defined.
- mnemonic: String [1]
  A short mnemonic used to represent the code system within the HL7 vocabulary.
- generalizationParent: String[1]
  Reference to the static model intended to be resolved, imported and re-used.

8.4.5 «stereotype» mifEntryPoint

Generalization «stereotype» mifInfrastructureRoot

Description
Entry Points designate the class(es) from which the messages begin for the particular domain. Identifies a class within the model that may be used as the initial class in a serializable representation of the model.

Attributes

- id: String [0..1]
  The identifier associated with the model that is tied to the entry-point.

8.4.6 «stereotype» mifPlayer

Extends «metaclass**»** Association

Description
Identifies the semantics of an association between two elements marked with the «stereotype» mifRole and the «stereotype» mifEntity. Each role is "played" by an Entity, called the "player".

8.4.7 «stereotype» mifProperty

Extends «metaclass**»** Property

Specialization «stereotype» mifAttribute

Description
An independently modifiable or static characteristic of an element.

Attributes

- sortKey: String [1]
  A name used in determining the sort order of the model element within its siblings.
- conformance: ConformanceKind [0..1]
  Identifies whether the element must be supported by implementors or not. If not present, indicates that support is optional.
- isMandatory: Boolean
  If true, null values may not be sent for this element.
- isStructural: Boolean
  If true, the value for the attribute is fixed.

8.4.8 «stereotype» mifScoper

Extends «metaclass**»** Association

Description
Identifies the semantics of an association between two elements marked with the «stereotype» mifRole and the «stereotype» mifEntity. Each role is "scoped" by an Entity, called the "scoper".

8.4.9 «stereotype» mifStaticModel

Extends «metaclass**»** Package

Description
A static Model in standard MIF notation. Used for RIM, CIM, LIM, etc.

Attributes

- packageKind: PackageKind [1]
  It should be set to "version" if there is one (and the name of the StaticModel set to the version value) or to "id" if there isn't a version (and the name of the StaticModel set to the id value).
- representationKind: StaticModelRepresentationKind [1]
  Identifies whether the model is represented in its flat or serializable form.
- title: String [0..1]
  The descriptive name for the package in circumstances where the 'name' is more of an identifier.
- isSerializable: Boolean
  Identifies whether this model can be represented in a serialized form.
- packageLocationRoot: PackageRootKind [1]
  Indicates the fundamental categorization of the package
- packageLocationSection: PublishingSectionKind [0..1]
  Added by Eclipse for backwards compatibility (Grahame 9-Oct 2007). To be removed later.
- packageLocationSubsection: PublishingSubSectionKind [0..1]
  Added by Eclipse for backwards compatibility (Grahame 9-Oct 2007). To be removed later.
- packageLocationDomain: PublishingDomainKind [0..1]
  Indicates the name of the domain package.
- packageLocationArtifact: ArtifactKind [0..1]
  Indicates the category of the artifacts in the package.

- packageLocationRealm: AffiliateKind [0..1]
  Indicates the geopolitical area covered by the package. Also used to distinguish "local" artifacts.
- packageLocationVersion: String [0..1]
  Indicates the name of the version package.
- packageLocationId: Integer [0..1]
  Indicates the identifier number of the package.

8.5 Enumerations

8.5.1 Overview

Enumeration types are user-definable finite sets of named elements that have a defined ordering among themselves but no other computational properties. An enumeration type has a name and a list of enumeration constants. An enumeration is shown as a rectangle with the keyword «enumeration» above the name of the enumeration type in the upper compartment. The second compartment contains a list of enumeration-literal names.

An enumeration literal has a name that can be used to identify it within its enumeration data type. The enumeration literal name is scoped within and must be unique within its enumeration. Enumeration literal names are not global and must be qualified for general use.

In a profile, it is possible to specify an enumeration as the type for an attribute of a stereotype. For example, if there is an enumeration called DaysOfWeek and an enumeration literal for each day of the week, then it is possible to create an attribute called myFavoriteDay of type DaysOfWeek in a stereotype and specify the enumeration literal called Saturday as the default value for the attribute.

The enumeration types and enumeration literals in this specification are extracted from the MIF XSD schema documentation (http://gforge.hl7.org/gf/download/frsrelease/649/7965/hl7_mifschemas-2.2.0.0.zip). These enumerations provide support to represent specific characteristics of MIF static models into UML models by using the attributes owned by the previously defined stereotypes. Each enumeration type includes a description and the list of enumeration literals it contains.

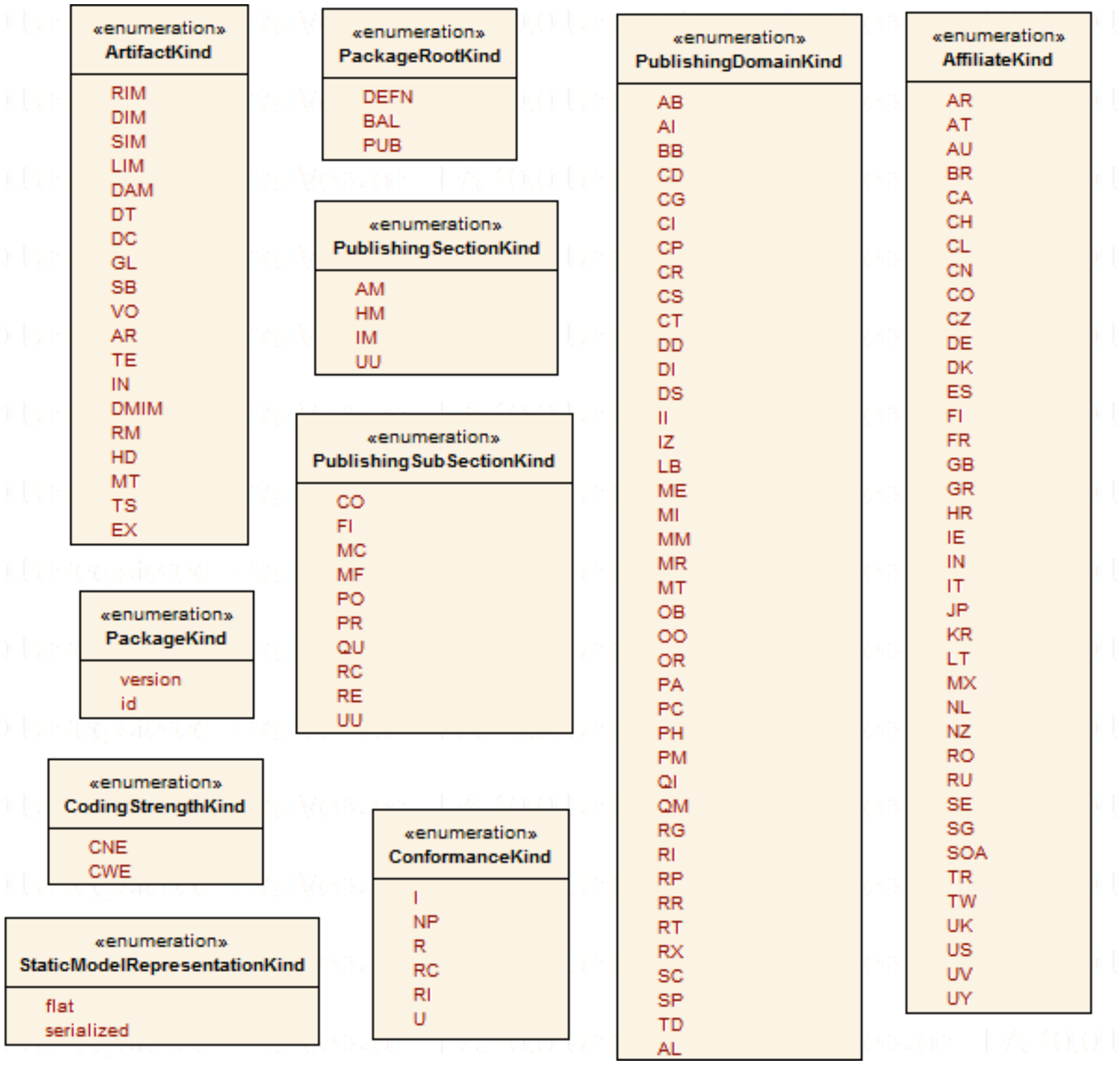


Figure 43. Enumerations referenced and used by the stereotypes of the UML profile for MIF static models.

8.5.2 «enumeration» AffiliateKind

Description

The list of HL7 affiliates.

Enumeration Literals

- AR: Argentina - Realm code for use of Argentina.
- AT: Austria - Realm code for use of Austria.
- AU: Australia - Realm code for use of Australia.
- BR: Brazil - Realm code for use of Brazil.
- CA: Canada - Realm code for use of Canada.
- CH: Switzerland - Realm code for use of Switzerland.
- CL: Chile - Realm code for use of Chile.
- CN: China - Realm code for use of China.
- CO: Columbia - Realm code for use of Localized Version.
- CZ: Czech Re- Realm code for use of Czech Republic.
- DE: Germany - Realm code for use of Germany.
- DK: Denmark - Realm code for use of Denmark.

- ES: Spain - Realm code for use of Spain.
- FI: Finland - Realm code for use of Finland.
- FR: France - Realm code for use of France.
- GB: Great Britain - Realm code for use of Great Britain.
- GR: Greece - Realm code for use of Greece.
- HR: Croatia - Realm code for use of Croatia.
- IE: Ireland - Realm code for use of Ireland.
- IN: India - Realm code for use of India.
- IT: Italy - Realm code for use of Italy.
- JP: Japan - Realm code for use of Japan.
- KR: Korea - Realm code for use of Korea.
- LT: Lithuania - Realm code for use of Lithuania.
- MX: Mexico - Realm code for use of Mexico.
- NL: The Netherlands - Realm code for use of The Netherlands.
- NZ: New Zealand - Realm code for use of New Zealand.
- RO: Romania - Realm code for use of Romania.
- RU: Russian Federation - Realm code for use of Russian Federation.
- SE: Sweden - Realm code for use of Sweden.
- SG: Singapore - Realm code for use of Localized Version.
- SOA: Southern Africa - Realm code for use of Southern Africa.
- TR: Turkey - Realm code for use of Turkey.
- TW: Taiwan - Realm code for use of Taiwan.
- UK: United Kingdom - Realm code for use of United Kingdom.
- US: United States of America - Realm code for use of United States of America.
- UV: Universal - Realm code for use of Universal realm or context, used in every instance.
- UY: Uruguay - Realm code for use of Uruguay.

8.5.3 «enumeration» ArtifactKind

Description

Identifies the kind of artifacts that can be packaged.

Enumeration Literals

- RIM: Reference Information Model.
- DIM: Domain Information Model (supercedes D-MIM).
- SIM: Serializable Information Model (supercedes R-MIM, HMD and Message Type).
- LIM: Localized Information Model (a.k.a. message profile).
- DAM: Document Analysis Model.
- DT: Datatype Model.
- DC: Document.
- GL: Glossary.
- SB: Story Board.
- VO: Vocabulary Model.
- AR: Application Role.
- TE: Trigger Event.
- IN: Interaction.
- DMIM: Domain Message Information Model.
- RM: Refined Message Information Model.
- HD: Hierarchical Message Descriptor.
- MT: Message Type.
- TS: Test Scenario.
- EX: Instance Example.

8.5.4 «enumeration» CodingStrengthKind

Description

Indicates whether or not codes are restricted to those expressed in the value set.

Enumeration Literals

- CNE: coded, non-extensible.
- CWE: coded, with extensions.

8.5.5 «enumeration» ConformanceKind

Description

Identifies allowed codes for HL7aTMs v3 conformance property.

Enumeration Literals

- I: ignored - Implementers receiving this property must not raise an error if the data is received, but will not perform any useful function with the data. This conformance level is not used in profiles or other artifacts that are specific to the "sender" or "initiator" of a communication.
- NP: not permitted - All implementers are prohibited from transmitting this content, and may raise an error if they receive it.
- R: required - All implementers must support this property. I.e. they must be able to transmit, or to receive and usefully handle the concept.
- RC: required for consumer - The element is considered "required" (i.e. must be supported) from the perspective of systems that consume instances, but is "undetermined" for systems that generate instances. Used only as part of specifications that define both initiator and consumer expectations.
- RI: required for initiator - The element is considered "required" (i.e. must be supported) from the perspective of systems that generate instances, but is "undetermined" for systems that consume instances. Used only as part of specifications that define both initiator and consumer expectations.
- U: undetermined - The conformance expectations for this element have not yet been determined.

8.5.6 «enumeration» PackageKind

Description

Identifies the kind of package used.

Enumeration Literals

- version: indicates that the Package is a version Package.
- id: indicates that the Package is not a version Package.

8.5.7 «enumeration» PackageRootKind

Description

Identifies the primary 'purpose' of the packaged content.

Enumeration Literals

- DEFN: The package represents the artifacts being defined.
- BAL: The package groups a set of content defined elsewhere intended to be submitted for ballot.
- PUB : The package groups a set of content defined elsewhere as a formal publication.

8.5.8 «enumeration» PublishingDomainKind

Description

Codes for HL7 publishing 'domain's (specific content area).

Enumeration Literals

- AB: accounting & billing - Represents the HL7 content "domain" that supports accounting and billing functions and provides support for the creation and management of patient billing accounts and the post of financial transactions against patient billing accounts for the purpose of aggregating financial transactions that will be submitted as claims or invoices for reimbursement.

- AI: trigger event control act infrastructure - Represents the HL7 content "domain" that supports trigger event control act infrastructure and covers the alternate structures of the message Trigger Event Control Acts in the HL7 Composite Message.
- BB: blood tissue and organ - Represents the HL7 content "domain" that supports blood tissue and organ domain and comprises the models, messages, and other artIfacts that are needed to support messaging related to the process of blood, tissue, and organ banking operations such as donations, eligibility, storage, dispense, administration/transfusion, explantation, and implantation.
- CD: clinical document architecture - Represents the HL7 content "domain" that supports the clinical document architecture.
- CG: clinical genomics - Represents the HL7 content "domain" that supports clinical genomics and includes standards to enable the exchange of interrelated clinical and personalized genomic data between interested parties.
- CI: transmission infrastructure - Represents the HL7 content "domain" that supports transmission infrastructure and is primarily concerned with the data content of exchanges between healthcare applications, the sequence or interrelationships in the flow of messages and the communication of significant application level exceptions or error conditions.
- CP: common product model - Represents the HL7 content "domain" that supports the common product model, which is used to improve the alignment between the different representations of products used within the body of HL7 Version 3 models.
- CR: claims and reimbursement - Represents the HL7 content "domain" that supports Claims and Reimbursement and provides support for Generic, Pharmacy, Preferred Accommodation, Physician, Oral Health Vision Care and Hospital claims for eligibility, authorization, coverage extension, pre-determination, invoice adjudication, payment advice and Statement of Financial Activity (SOFA) Release 3 of this document adds claims messaging support for Physician, Oral Health Vision Care and Hospital claims.
- CS: clinical statement - Represents the HL7 content "domain" that supports a common clinical statement pattern and is a 'pattern' designed to be used within multiple HL7 Version 3 domain models. This pattern is intended to facilitate the consistent design of communications that convey clinical information to meet specific use cases.
- CT: common types - Represents the HL7 content "domain" that supports common model types and are a work product produced by a particular committee for expressing a common, useful and reusable concept.
- DD: dummy domain - Represents the HL7 content "domain" that was created to support testing and initial set-up functions.
- DI: diagnostic imaging - This domain has been retired in favor of "imaging integration" (II).
- DS: decision support - Represents the HL7 content "domain" that provides decision support.
- II: imaging integration - Represents the HL7 content "domain" that supports imaging integration and comprises the models, implementation guides, sample documents and images that are needed to illustrate the transformation of DICOM structured reports to CDA Release 2 as well as the creation of CDA diagnostic imaging reports.
- IZ: immunization - Represents the HL7 content "domain" that supports immunization -and describes communication of information about immunization: the administration of vaccines (and/or antisera) to individuals to prevent infectious disease.
- LB: laboratory - Represents the HL7 content "domain" that supports clinical laboratory functions and comprises the models, messages, and other artifacts that are needed to support messaging related to laboratory tests or observations.
- ME: medication - Represents the HL7 content "domain" that supports medication and deals with the description of a medicine for the purposes of messaging information about medicines and the applications of these descriptions.

- MI: masterfile infrastructure - Represents the HL7 content "domain" that supports master file infrastructure and comprises the classes and attributes needed to support Master Files and Registries.
- MM: materials management - Represents the HL7 content "domain" that supports Materials Management and supports the simple scenario of a Materials Management application sending requests, notifications and queries to an auxiliary application. The intent is to establish a standard for the minimum functionality that is useful and comprehensive enough to explore the important concepts relative to inventory management.
- MR: medical records - Represents the HL7 content "domain" that supports medical records and supports clinical document management, and document querying.
- MT: shared messages - Represents the HL7 content "domain" that supports shared messages and are a work product produced for expressing common, useful and reusable message types.
- OB: observations - Represents the HL7 content "domain" that supports observations and comprises the models, messages, and other artifacts that are needed to support messaging related to resulting basic healthcare diagnostic services.
- OO: orders & observations - Represents the HL7 content "domain" that supports orders and observations and will provide over-arching support information for the "Orders" (OR) and "Observations" (OB) domains.
- OR: orders - Represents the HL7 content "domain" that supports orders and comprises the models, messages, and other artifacts that are needed to support messaging related to ordering basic healthcare services.
- PA: patient administration - Represents the HL7 content "domain" that supports Patient Administration and defines person and patient demographics and visit information about patients.
- PC: care provision - Represents the HL7 content "domain" that supports Care Provision and addresses the information that is needed for the ongoing care of individuals, populations, and other targets of care.
- PH: public health - Represents the HL7 content "domain" that supports public health and is the source of a number of Common Model Element Types (CMET) designed to meet the needs of public health data exchange.
- PM: personnel management - Represents the HL7 content "domain" that supports Personnel Management and spans a variety of clinical-administrative information functions associated with the organizations, individuals, animals and devices involved in the delivery and support of healthcare services.
- QI: query infrastructure - Represents the HL7 content "domain" that supports query infrastructure and specifies the formation of information queries and the responses to these queries to meet the needs of healthcare applications using the HL7 version 3 messaging standard.
- QM: quality measures - Represents the HL7 content "domain" that supports Quality Measures and is a standard for representing a health quality measure as an electronic document.
- RG: registries - Represents the HL7 content "domain" that supports Registries and collects HL7 artifacts for administrative registries.
- RI: informative public health - Represents the HL7 content "domain" that supports Informative Public Health.
- RP: regulated products - Represents the HL7 content "domain" that supports Regulated Products and includes standards developed as part of the family of messages targeted for the exchange of information about regulated products and the exchange of the data needed to provide approval for such products.
- RR: public health reporting - Represents the HL7 content "domain" that supports Public Health Reporting and includes messages and documents that are specifically designed to support management, reporting and investigation in the public health context.
- RT: regulated studies - Represents the HL7 content "domain" that supports Regulated Studies and includes standards developed as part of the family of messages targeted for the exchange of

information about the conduct of regulated studies, and the exchange of the data collected during those studies.

- RX: pharmacy - Represents the HL7 content "domain" that supports pharmacy and is a model used to derive message patterns to describe and communicate processes related to medication.
- SC: scheduling - Represents the HL7 content "domain" that supports Scheduling and offers a generic set of messages and behavior to implement any number of Scheduling scenarios.
- SP: specimen - Represents the HL7 content "domain" that supports Specimen and comprises the models and artifacts that are needed to support the creation of messaging related to specimen.
- TD: therapeutic devices - Represents the HL7 content "domain" that supports Therapeutic Devices and comprises the models, messages, and other artifacts that are needed to support messaging related to therapy delivery and observations made by a medical device.
- AL: artificial listing for test purposes; faux Domain for testing - Represents the HL7 content "domain" that was defined as an "artificial listing" domain to support publication testing.

8.5.9 «enumeration» PublishingSectionKind

Description

Codes for HL7 publishing 'section's (major business categories).

Enumeration Literals

- AM: administrative management - Represents the HL7 V3 publishing section that deals with the administration and management of health care activities and organizations.
- HM: health and clinical management - Represents the HL7 V3 publishing section that deals with the health care provision and clinical management.
- IM: infrastructure management - Represents the HL7 V3 publishing section that deals with the definition and management of the computing and communication infrastructure necessary to support health care.
- UU: unknown - Represents the HL7 V3 publishing section that holds specifications that are unassigned i.e., that have not yet been assigned to one of the formal publishing sections.

8.5.10 «enumeration» PublishingSubSectionKind

Description

Codes for HL7 publishing 'sub-section's (business sub-categories).

Enumeration Literals

- CO: common - Represents the HL7 V3 publishing sub-section that holds common or shared specifications within the Infrastructure Management (IM) section.
- FI: financial information - Represents the HL7 V3 publishing sub-section that holds specifications related to the management of financial information within the Administrative Management (AM) section.
- MC: message control - Represents the HL7 V3 publishing sub-section that holds specifications related to the definition and control of interoperability messages within the Infrastructure Management (IM) section.
- MF: master file - Represents the HL7 V3 publishing sub-section that holds specifications related to master file and registry management activities within the Infrastructure Management (IM) section.
- PO: operations - Represents the HL7 V3 publishing sub-section that holds specifications related to managing clinical operations within the Health and Clinical Management (HM) section.
- PR: practice - Represents the HL7 V3 publishing sub-section that holds specifications related to the management of practice settings within the Administrative Management (AM) section.

- QU: query - Represents the HL7 V3 publishing sub-section that holds specifications related to query/response activities within the Infrastructure Management (IM) section.
- RC: records - Represents the HL7 V3 publishing sub-section that holds specifications related to the definition and communication of records of clinical care within the Health and Clinical Management (HM) section.
- RE: reasoning - Represents the HL7 V3 publishing sub-section that holds specifications related to the definition and communication of reasoning (knowledge) within the Health and Clinical Management (HM) section.
- UU: unknown - Represents the HL7 V3 publishing sub-section that holds specifications that are unassigned - that have not yet been assigned to one of the formal publishing sections.

8.5.11 «enumeration» StaticModelRepresentationKind

Description

The allowed mechanisms by which a static can be serialized.

Enumeration Literals

- flat: The model is represented as a list of class definitions, not necessarily in the order they will appear when communicated, and not organized in a hierarchy for exchange.
- serialized: The model is represented as a hierarchical list of classes and associations ordered in the way they would appear in an instance for exchange.

8.6 Metaclasses

8.6.1 Overview

This section of the specification lists the UML metaclasses from the UML Superstructure Specification extended by the UML Profile for MIF static models. In addition to it, the specification includes the constraints that the instances of these metaclasses must fulfill in order to accomplish to the well-formedness rules of a correct HL7 MIF static model. The set of constraints and additional operations are defined by means of formal OCL expressions for the sake of consistency and unambiguity.

8.6.2 «metaclass» Class

Description

A class describes a set of objects that share the same features, constraints, and semantics. Class is a kind of classifier whose features are attributes and operations. Attributes of a class are represented by instances of Property that are owned by the class. Some of these attributes may represent the navigable ends of associations.

Constraints

[1] A class with the mifEntryPoint must only participate in a binary association.

```
context Class inv:
self.getNamesOfAppliedStereotypes()->includes('mifEntryPoint') implies
Association.allInstances()->select(a|a.memberEnd->one(m|m.type =
self))->size = 1
```

[2] A class with the mifChoice stereotype must be abstract and with no attributes.

```
context Class inv:
self.getNamesOfAppliedStereotypes()->includes('mifChoice') implies
self.isAbstract = true and self.ownedAttribute->isEmpty()
```

[3] A class with the mifChoice stereotype must have one and only one additional stereotype indicating the type of the choice hierarchy.

```
context Class inv:
let stereotypes = self.getNamesOfAppliedStereotypes() in
stereotypes->includes(‘mifChoice’) implies ( stereotypes->size() = 2
and
  let additionalStereotype =  stereotypes->excluding(‘mifChoice’) in
  ( self.actStereotypes()->includesAll(additionalStereotype) or
    self.roleStereotypes()->includesAll(additionalStereotype) or
    entity.actStereotypes()->includesAll(additionalStereotype) ) )
```

[4] The descendants of a class with the mifChoice stereotype must have a stereotype compatible with the stereotype that indicates the type of the choice hierarchy.

```
context Class inv:
let stereotypes = self.getNamesOfAppliedStereotypes() in
stereotypes->includes(‘mifChoice’) implies
  let additionalStereotype =  stereotypes->excluding(‘mifChoice’) in
  if self.actStereotypes()->includesAll(additionalStereotype) then
  self.allDescendants()-
>forAll(d|self.isCompatible(d.getNamesOfAppliedStereotypes(), self.
actStereotypes()))
else if self.entityStereotypes()->includesAll(additionalStereotype)
then
  self.allDescendants()-
>forAll(d|self.isCompatible(d.getNamesOfAppliedStereotypes(),
self.entityStereotypes()))
else if self.roleStereotypes()->includesAll(additionalStereotype) then
  self.allDescendants()-
>forAll(d|self.isCompatible(d.getNamesOfAppliedStereotypes(),
self.roleStereotypes()))
else true
endif endif endif
```

[5] A class with the mifCMET stereotype must have one and only one additional stereotype indicating the type of the main element of the CMET.

```
context Class inv:
let stereotypes = self.getNamesOfAppliedStereotypes() in
stereotypes->includes(‘mifCMET’) implies ( stereotypes->size() = 2 and
  let additionalStereotype =  stereotypes->excluding(‘mifCMET’) in
  ( self.actStereotypes()->includesAll(additionalStereotype) or
    self.roleStereotypes()->includesAll(additionalStereotype) or
    entity.actStereotypes()->includesAll(additionalStereotype) ) )
```

[6] A class with the mifCMET stereotype must have no attributes.

```
context Class inv:
self.getNamesOfAppliedStereotypes()->includes(‘mifCMET) implies
self.ownedAttribute->isEmpty()
```

Additional Operations

[1] The query getAppliedStereotypes() gives all the stereotypes applied to a class.

```
context Class
def: getAppliedStereotypes():Set(Stereotype) =
  Stereotype.allInstances()->select(s|s.base_Class = self)->asSet()
```

[2] The query getNamesOfAppliedStereotypes() gives all the names of the stereotypes applied to a class.

```
context Class
def: getNamesOfAppliedStereotypes():Set(String) =
self.getAppliedStereotypes().name->asSet()
```

[3] The query actStereotypes() gives all the names of the stereotypes that represent Acts.

```
context Class
def: actStereotypes():Set(String) = Set{‘mifAct’, ‘mifActHeir’,
‘mifAccount’, ‘mifControlAct’, ‘mifWorkingList’, ‘mifDeviceTask’,
‘mifExposure’, ‘mifInvoiceElement’, ‘mifFinancialTransaction’,
‘mifFinancialContract’, ‘mifPatientEncounter’, ‘mifSupply’, ‘mifDiet’,
‘mifObservation’, ‘mifDiagnosticImage’, ‘mifPublicHealthCase’,
‘mifProcedure’, ‘mifSubstanceAdministration’, ‘mifContextStructure’,
‘mifDocument’}
```

[4] The query roleStereotypes() gives all the names of the stereotypes that represent Roles.

```
context Class
def: roleStereotypes():Set(String) = Set{‘mifRole’, ‘mifRoleHeir’,
‘mifQualifiedEntity’, ‘mifLicensedEntity’, ‘mifEmployee’, ‘mifPatient’,
‘mifAccess’}
```

[5] The query entityStereotypes() gives all the names of the stereotypes that represent Entities.

```
context Class
def: entityStereotypes():Set(String) = Set{‘mifEntity’,
‘mifEntityHeir’, ‘mifPlace’, ‘mifOrganization’, ‘mifLivingSubject’,
‘mifPerson’, ‘mifNonPersonLivingSubject’, ‘mifMaterial’,
‘mifManufacturedMaterial’, ‘mifContainer’, ‘mifDevice’}
```

[6] The query participationStereotypes() gives all the names of the stereotypes that represent Participations.

```
context Class
def: participationStereotypes():Set(String) = Set{‘mifParticipation’,
‘mifManagedParticipation’}
```

[7] The query messageStereotypes() gives all the names of the stereotypes from the Message Communications Control Area.

```
context Class
def: messageStereotypes():Set(String) =
Set{‘mifTransmissionRelationship’, ‘mifAcknowledgementDetail’,
‘mifAttachment’, ‘mifAttentionLine’, ‘mifAcknowledgement’,
‘mifCommunicationForm’, ‘mifTransmission’, ‘mifMessage’, ‘mifBatch’,
‘mifParameter’, ‘mifParameterItem’, ‘mifParameterList’,
‘mifQueryEvent’, ‘mifQuerySpec’, ‘mifQueryByParameter’, ‘mifQueryAck’,
‘mifQueryContinuation’, ‘mifSortControl’}
```

[8] The query allDescendants() gives all the descendant classes in the hierarchy of specializations of a class.

```
context Class
def: allDescendants():Set(Class) = self.generalization.specific-
>union(self.generalization.specific->collect(c|c.allDescendants())-
>asSet()
```

[9] The query isCompatible() checks the compatibility of a set of stereotype names with another set of stereotype names. The sets are compatible if one of the members of the first set is included in the second set.

```
context Class
def: isCompatible(first:Set(String), second:Set(String)):Boolean =
first->one(e|second->includes(e))
```

8.6.3 «metaclass» Association

Description

An association describes a set of tuples whose values refer to typed instances. An instance of an association is called a link. A link is a tuple with one value for each end of the association, where each value is an instance of the type of the end. An association specifies a semantic relationship that can occur between typed instances. It has at least two ends represented by properties, each of which is connected to the type of the end. More than one end of the association may have the same type.

An end property of an association that is owned by an end class or that is a navigable owned end of the association indicates that the association is navigable from the opposite ends; otherwise, the association is not navigable from the opposite ends.

Constraints

[1] An association must be binary, i.e., it only contains two member ends.

```
context Association inv:
self.memberEnd->size = 2
```

[2] If a member end of an association is a class with a stereotype representing an Entity, the opposite member end must have a stereotype representing a Role or a stereotype from the Message Communications Control Area.

```
context Association inv:
let source = self.memberEnd->at(1).type in
let target = self.memberEnd->at(2).type in
if source.isCompatible(source.getNamesOfAppliedStereotypes(),
source.entityStereotypes()) then
target.isCompatible(target.getNamesOfAppliedStereotypes(),
target.roleStereotypes()) or
     target.isCompatible(target.getNamesOfAppliedStereotypes(),
target.messageStereotypes())
else if target.isCompatible(source.getNamesOfAppliedStereotypes(),
target.entityStereotypes()) then
source.isCompatible(source.getNamesOfAppliedStereotypes(),
source.roleStereotypes()) or
    source.isCompatible(source.getNamesOfAppliedStereotypes(),
source.messageStereotypes())
else true
endif endif
```

[3] If a member end of an association is a class with the stereotype mifRoleLink, the opposite member end must have a stereotype representing a Role.

```
context Association inv:
let source = self.memberEnd->at(1).type in
let target = self.memberEnd->at(2).type in
if
source.isCompatible(source.getNamesOfAppliedStereotypes(),Set{‘mifRoleL
ink’}) then
  target.isCompatible(target.getNamesOfAppliedStereotypes(),
target.roleStereotypes())
else if
target.isCompatible(target.getNamesOfAppliedStereotypes(),Set{‘mifRoleL
ink’}) then
  source.isCompatible(source.getNamesOfAppliedStereotypes(),
source.roleStereotypes())
else true
endif endif
```

[4] If a member end of an association is a class with a stereotype representing a Participation, the opposite member end must have a stereotype representing a Role or an Act.

```
context Association inv:
let source = self.memberEnd->at(1).type in
let target = self.memberEnd->at(2).type in
if
source.isCompatible(source.getNamesOfAppliedStereotypes(),source.partic
ipationStereotypes()) then
target.isCompatible(target.getNamesOfAppliedStereotypes(),
target.roleStereotypes()) or
     target.isCompatible(target.getNamesOfAppliedStereotypes(),
target.actStereotypes())
else if
target.isCompatible(target.getNamesOfAppliedStereotypes(),target.partic
ipationStereotypes()) then
source.isCompatible(source.getNamesOfAppliedStereotypes(),
source.roleStereotypes()) or
     source.isCompatible(source.getNamesOfAppliedStereotypes(),
source.actStereotypes())
```

```
else true
endif endif
```

[5] If a member end of an association is a class with the stereotype ‘mifActRelationship’, the opposite member end must have a stereotype representing an Act.

```
context Association inv:
let source = self.memberEnd->at(1).type in
let target = self.memberEnd->at(2).type in
if source.isCompatible(source.getNamesOfAppliedStereotypes(),
Set{‘mifActRelationship’}) then
  target.isCompatible(target.getNamesOfAppliedStereotypes(),
target.actStereotypes())
else if target.isCompatible(target.getNamesOfAppliedStereotypes(),
Set{‘mifActRelationship’}) then
source.isCompatible(source.getNamesOfAppliedStereotypes(),source.actSte
reotypes())
else true
endif endif
```

[6] If a member end of an association is a class with a stereotype representing an Act, the opposite member end must have a stereotype representing a Participation or the stereotype ‘mifActRelationship’.

```
context Association inv:
let source = self.memberEnd->at(1).type in
let target = self.memberEnd->at(2).type in
if source.isCompatible(source.getNamesOfAppliedStereotypes(),
source.actStereotypes())
then
target.isCompatible(target.getNamesOfAppliedStereotypes(),
target.participationStereotypes()) or
     target.isCompatible(target.getNamesOfAppliedStereotypes(),
Set{‘mifActRelationship’})
else if target.isCompatible(source.getNamesOfAppliedStereotypes(),
target.actStereotypes()) then
source.isCompatible(source.getNamesOfAppliedStereotypes(),source.partic
ipationStereotypes()) or
source.isCompatible(source.getNamesOfAppliedStereotypes(),
Set{‘mifActRelationship’})
else true
endif endif
```

[7] If a member end of an association is a class with a stereotype representing a Role, the opposite member end must have a stereotype representing an Entity or Participation or the stereotype ‘mifRoleLink’.

```
context Association inv:
let source = self.memberEnd->at(1).type in
let target = self.memberEnd->at(2).type in
if source.isCompatible(source.getNamesOfAppliedStereotypes(),
source.roleStereotypes())
then target.isCompatible(target.getNamesOfAppliedStereotypes(),
target.entityStereotypes()) or
target.isCompatible(target.getNamesOfAppliedStereotypes(),
target.participationStereotypes()) or
     target.isCompatible(target.getNamesOfAppliedStereotypes(),
Set{‘mifRoleLink’})
else if target.isCompatible(source.getNamesOfAppliedStereotypes(),
target.roleStereotypes())
then
source.isCompatible(source.getNamesOfAppliedStereotypes(),source.entity
Stereotypes()) or
source.isCompatible(source.getNamesOfAppliedStereotypes(),source.partic
ipationStereotypes()) or
source.isCompatible(source.getNamesOfAppliedStereotypes(),
Set{‘mifRoleLink’})
```

```
else true
endif endif
```

[8] If an association has the stereotype ‘mifPlayer’ or ‘mifScoper’, a member end of the association must be a class with a stereotype representing a Role and the opposite member end must be a class with a stereotype representing an Entity.

```
context Association inv:
let source = self.memberEnd->at(1).type in
let target = self.memberEnd->at(2).type in
self.isCompatible(self.getNamesOfAppliedStereotypes, Set{‘mifPlayer,
‘mifScoper’}) implies
if source.isCompatible(source.getNamesOfAppliedStereotypes(),
source.roleStereotypes())
then target.isCompatible(target.getNamesOfAppliedStereotypes(),
target.entityStereotypes
else if target.isCompatible(source.getNamesOfAppliedStereotypes(),
target.roleStereotypes())
then
source.isCompatible(source.getNamesOfAppliedStereotypes(),source.entity
Stereotypes
else true
endif endif
```

Additional Operations

[1] The query getAppliedStereotypes() gives all the stereotypes applied to an association.

```
context Association
def: getAppliedStereotypes():Set(Stereotype) =
Stereotype.allInstances()->select(s|s.base_Association = self)->asSet()
```

[2] The query getNamesOfAppliedStereotypes() gives all the names of the stereotypes applied to an association.

```
context Association
def: getNamesOfAppliedStereotypes():Set(String) =
self.getAppliedStereotypes().name->asSet()
```

8.6.4 «metaclass» Package

Description

A package is used to group elements, and provides a namespace for the grouped elements. A package is a namespace for its members, and may contain other packages. Only packageable elements can be owned members of a package. By virtue of being a namespace, a package can import either individual members of other packages, or all the members of other packages. In addition a package can be merged with other packages.

8.6.5 «metaclass» Property

Description

A property is a structural feature. A property related to a classifier by ownedAttribute represents an attribute, and it may also represent an association end. It relates an instance of the class to a value or collection of values of the type of the attribute. A property related to an Association by memberEnd or its specializations represents an end of the association. The type of property is the type of the end of the association.

Additional Operations

[1] The query isConsistentWith() specifies, for any two Properties if they are logically consistent. A property is consistent with another property if the type of the property conforms to the type of the other property, and the multiplicity of the property (if specified) is contained in the multiplicity of the other property.

```
context Property
def: isConsistentWith(other:Property):Boolean =
other.type.conformsTo(self.type) and
((other.lowerBound()->notEmpty() and self.lowerBound()->notEmpty())
     implies other.lowerBound() >= self.lowerBound()) and
```

```
    ((other.upperBound()->notEmpty() and self.upperBound()->notEmpty())
         implies other.lowerBound() <= self.lowerBound())
```

8.6.6 «metaclass» Attribute

Description

An attribute is a feature owned by a class. The metaclass Attribute is no longer included into the UML 2.4.1 Superstructure specification --it is substituted by Property. The reason to be included in the specification of the MIF profile is to conform to the specification of the profile in Enterprise Architect, which requires the explicit inclusion of the metaclass Attribute to indicate the extension of attributes of classes.

9 MIF-UML Transformation Reference

9.1 Overview

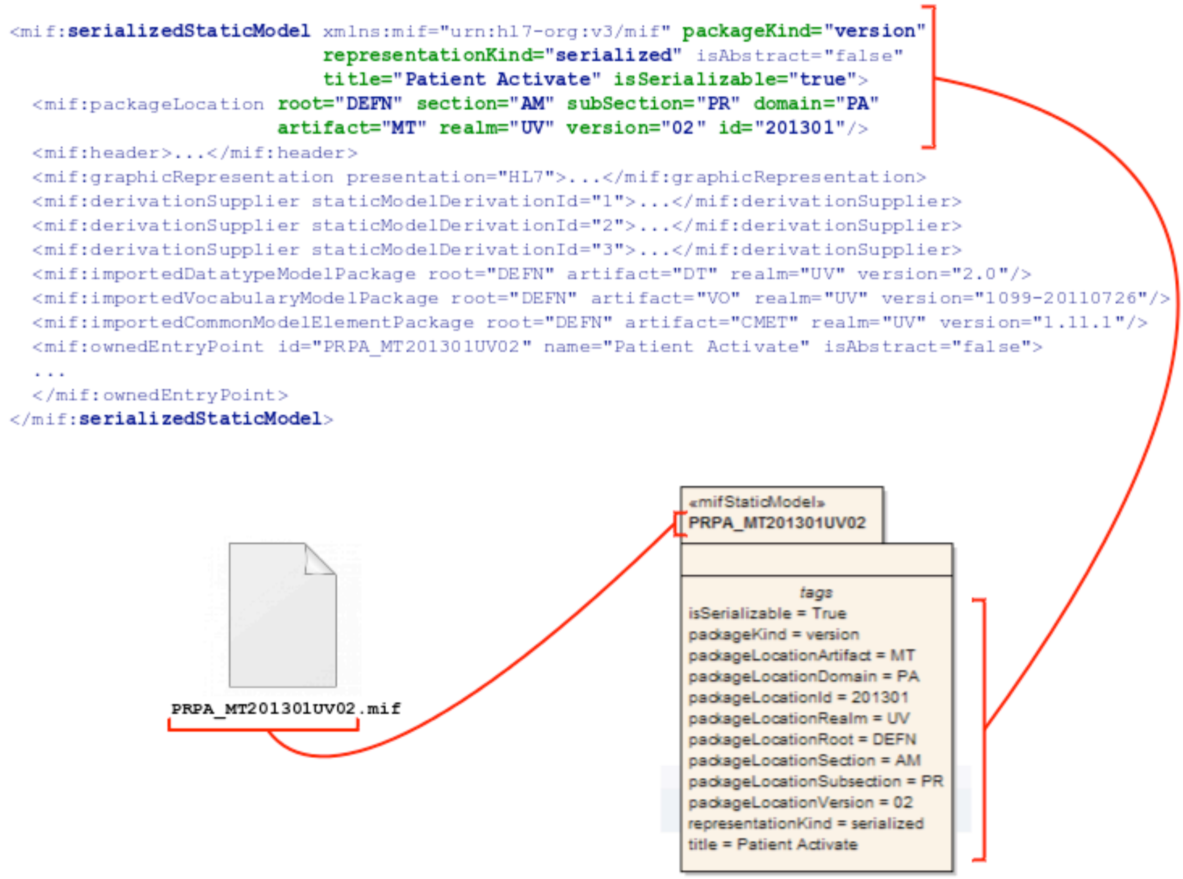
This section provides orientation to the transformation between the structure and content of Model Interchange Format (MIF) based models and the equivalent UML models that make use of the UML profile for MIF static models. The MIF files located in the CD2 (/CD2_2012/Edition2012/processable/MIF/) of the HL7 V3 2012 Normative Edition are the basis in this specification. The transformation aspects described here can be easily adapted to other MIF versions.

9.2 Static Model

A static model describes the structure of a fragment of a system. In HL7 standards, static models are the main artifact used to specify the contents and structure of messages and documents that are shared between several endpoints in a healthcare environment. Each MIF file (containing one serialized static model) is mapped to one UML package.

Figure 44 details the mapping of a MIF static model to an equivalent UML package. The name of the package equals to the name of the MIF file that contains the serialized static model. The new UML package contains the stereotype «mifStaticModel» as defined in 8.4.9. The attributes and values for this stereotype are highlighted in the XML MIF version of the serializedStaticModel element.

The UML package that results from the transformation must contain all the elements described in the MIF file within the serializedStaticModel element.



```
<mif:serializedStaticModel xmlns:mif="urn:hl7-org:v3/mif" packageKind="version"
                           representationKind="serialized" isAbstract="false"
                           title="Patient Activate" isSerializable="true">
  <mif:packageLocation root="DEFN" section="AM" subSection="PR" domain="PA"
                       artifact="MT" realm="UV" version="02" id="201301"/>
  <mif:header>...</mif:header>
  <mif:graphicRepresentation presentation="HL7">...</mif:graphicRepresentation>
  <mif:derivationSupplier staticModelDerivationId="1">...</mif:derivationSupplier>
  <mif:derivationSupplier staticModelDerivationId="2">...</mif:derivationSupplier>
  <mif:derivationSupplier staticModelDerivationId="3">...</mif:derivationSupplier>
  <mif:importedDatatypeModelPackage root="DEFN" artifact="DT" realm="UV" version="2.0"/>
  <mif:importedVocabularyModelPackage root="DEFN" artifact="VO" realm="UV" version="1099-20110726"/>
  <mif:importedCommonModelElementPackage root="DEFN" artifact="CMET" realm="UV" version="1.11.1"/>
  <mif:ownedEntryPoint id="PRPA_MT201301UV02" name="Patient Activate" isAbstract="false">
  ...
  </mif:ownedEntryPoint>
</mif:serializedStaticModel>
```

«mifStaticModel»
PRPA_MT201301UV02

tags
isSerializable = True
packageKind = version
packageLocationArtifact = MT
packageLocationDomain = PA
packageLocationId = 201301
packageLocationRealm = UV
packageLocationRoot = DEFN
packageLocationSection = AM
packageLocationSubsection = PR
packageLocationVersion = 02
representationKind = serialized
title = Patient Activate

Figure 44. Transformation of MIF static model to UML.

9.3 Entry Point

Each static model has one Entry Point. Entry Points designate the class (or classes, if it is connected to a Choice) from which the message/document begin for the particular model. Entry points are graphically represented on the HL7 diagrams as clear boxes with thick black borders. A thick black arrow originates from the Entry Point box and points to a class which is considered the root, or focal class for one or more static model. Each Entry Point is named, carries the artifact ID of the static model which originates from it, and contains a brief description.

Figure 45 provides a sample Entry Point indicating that the central element in a particular static model is the Patient role class.

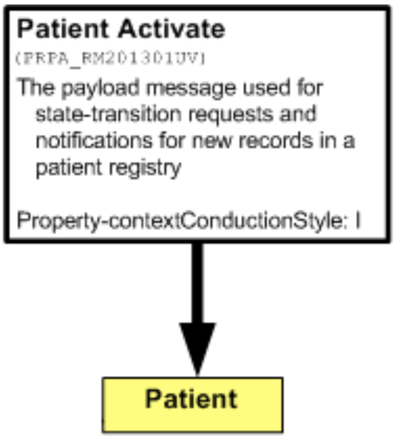

Patient Activate
(PRPA_RM201301UV)
The payload message used for state-transition requests and notifications for new records in a patient registry
Property-contextConductionStyle: I
Patient

Figure 45. Graphical representation of an entry point in HL7.

In the MIF format, the entry point is the first element contained within the serializedStaticModel node of the XML structure. The XML node for the entry point is called ownedEntryPoint and basically contains three XML subnodes: annotations, graphicRepresentation, and specializedClass. Figure 46 includes the transformation of the MIF structure representing an entry point to an equivalent UML class with the same semantics. The new class must contain the stereotype «mifEntryPoint» to indicate the MIF element that represents.

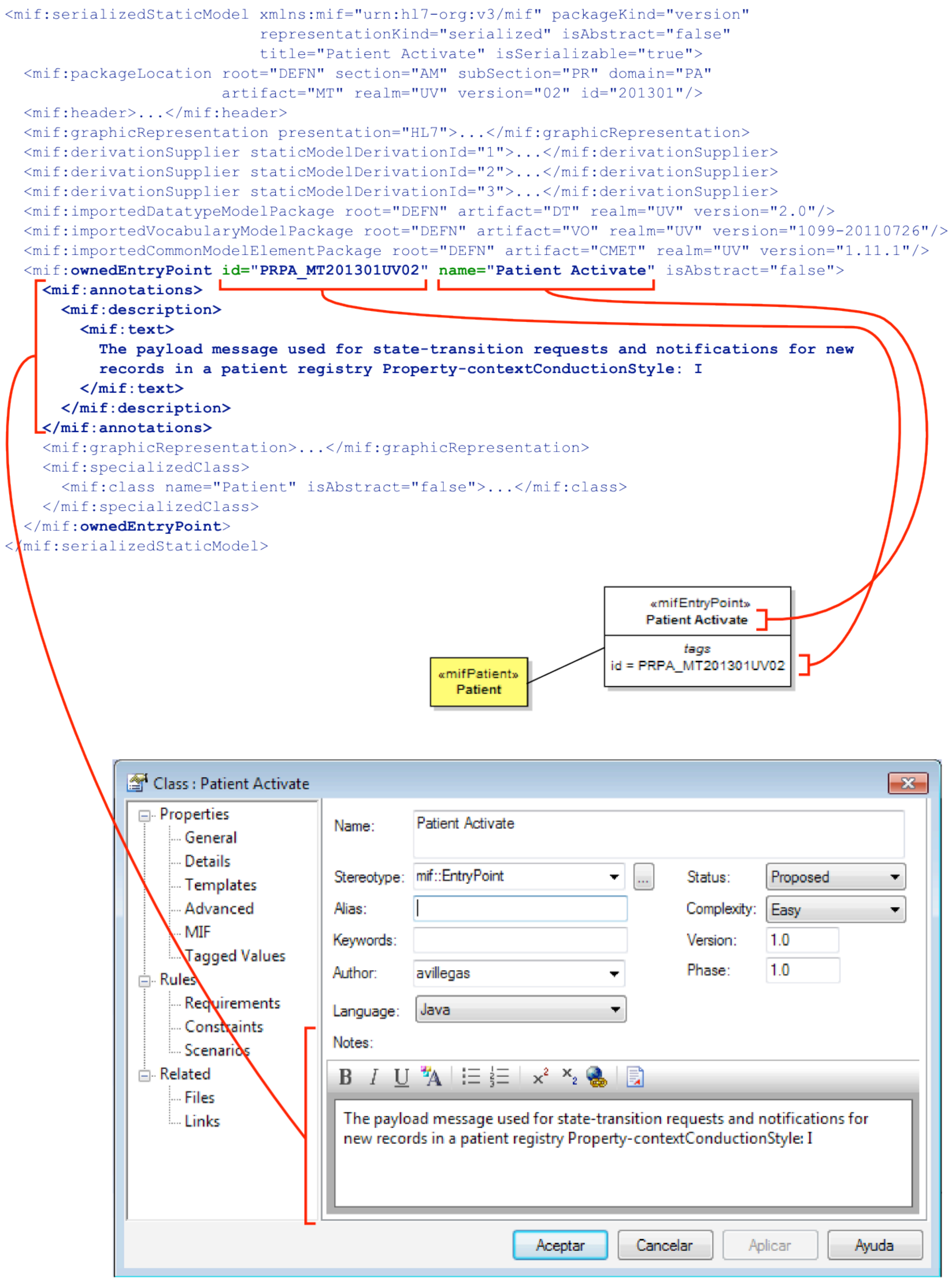
The name and id are attributes of the ownedEntryPoint node that must be mapped into UML. The name of the UML class that represents the entry point equals to the name attribute of the MIF XML node. In addition to this, the value of the id attribute is the value of the attribute id of the «mifEntryPoint» stereotype.

The annotations node defined in the MIF structure includes definitions, usage notes, constraints, and any other textual comment that provide additional information about the entry point. In fact, any XML node within the MIF structure may contain annotations to clarify or constraint the semantics of the node. For the case of the ownedEntryPoint node, the annotations include the description of the entry point. The text within this node is included as an annotation in the new UML class that represents the entry point. Since this specification uses Enterprise Architect, the description text must be included in the Notes textbox of the properties dialog for the entry point class.

The graphicRepresentation node is a platform specific information that contains information about the drawing shape of the parent node in the MIF structure. This node is ignored in the UML profile for MIF static models since the graphical notation in UML is different from the graphical notation in HL7.

The specializedClass node in the MIF structure indicates the central element in the static model that must be connected with the entry point. In the UML equivalent representation, the class with the «mifEntryPoint» stereotype contains only one association with the element referenced by the specializedClass node. In Figure 46, the class Patient Activate, which represents the entry point, is connected with the Patient class, which is the specialized class in the static model.

Note that the graphical representation in Figure 45, the MIF representation in XML, and the graphical representation in Figure 46 of the Patient Activate entry point are all semantically equivalent.



```
<mif:serializedStaticModel xmlns:mif="urn:hl7-org:v3/mif" packageKind="version"
                           representationKind="serialized" isAbstract="false"
                           title="Patient Activate" isSerializable="true">
  <mif:packageLocation root="DEFN" section="AM" subSection="PR" domain="PA"
                       artifact="MT" realm="UV" version="02" id="201301"/>
  <mif:header>...</mif:header>
  <mif:graphicRepresentation presentation="HL7">...</mif:graphicRepresentation>
  <mif:derivationSupplier staticModelDerivationId="1">...</mif:derivationSupplier>
  <mif:derivationSupplier staticModelDerivationId="2">...</mif:derivationSupplier>
  <mif:derivationSupplier staticModelDerivationId="3">...</mif:derivationSupplier>
  <mif:importedDatatypeModelPackage root="DEFN" artifact="DT" realm="UV" version="2.0"/>
  <mif:importedVocabularyModelPackage root="DEFN" artifact="VO" realm="UV" version="1099-20110726"/>
  <mif:importedCommonModelElementPackage root="DEFN" artifact="CMET" realm="UV" version="1.11.1"/>
  <mif:ownedEntryPoint id="PRPA_MT201301UV02" name="Patient Activate" isAbstract="false">
    <mif:annotations>
      <mif:description>
        <mif:text>
          The payload message used for state-transition requests and notifications for new
          records in a patient registry Property-contextConductionStyle: I
        </mif:text>
      </mif:description>
    </mif:annotations>
    <mif:graphicRepresentation>...</mif:graphicRepresentation>
    <mif:specializedClass>
      <mif:class name="Patient" isAbstract="false">...</mif:class>
    </mif:specializedClass>
  </mif:ownedEntryPoint>
</mif:serializedStaticModel>
```

Class : Patient Activate

Name: Patient Activate

Stereotype: mif::EntryPoint

Alias:

Keywords:

Author: avillegas

Language: Java

Status: Proposed

Complexity: Easy

Version: 1.0

Phase: 1.0

Notes:

The payload message used for state-transition requests and notifications for new records in a patient registry Property-contextConductionStyle: I

Figure 46. Transformation of MIF entry point to UML.

9.4 Class

A class is an abstraction of things or concepts that are subjects of interest in a given application domain. All things or concepts subsumed under a class have the same properties and are subject to and conform to the same rules. Classes are the people, places, roles, things, and events about which information is kept. Classes have a name, description, and sets of attributes and relationships.

Graphically, classes in HL7 and in UML are similarly represented as boxes containing a name and several attributes. Figure 47 shows the class ControlActProcess, which is an Act, in HL7.

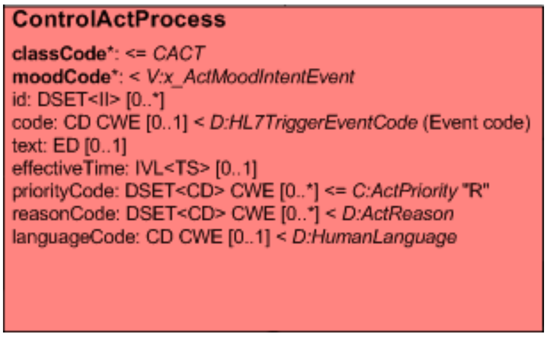

ControlActProcess
classCode*: <= CACT
moodCode*: < V:x_ActMoodIntentEvent
id: DSET<II> [0..*]
code: CD CWE [0..1] < D:HL7TriggerEventCode (Event code)
text: ED [0..1]
effectiveTime: IVL<TS> [0..1]
priorityCode: DSET<CD> CWE [0..*] <= C:ActPriority "R"
reasonCode: DSET<CD> CWE [0..*] < D:ActReason
languageCode: CD CWE [0..1] < D:HumanLanguage

Figure 47. Graphical representation of a class in HL7.

The graphical representation of Participation and ActRelationship classes in HL7 follows a different guideline. Figure 48 shows scheduleRequest and referenceOrderm which are ActRelationship classes, and subject, which is a Participation class. All of them are represented as arrow boxes connecting acts or acts and roles, respectively. In UML, the representation of ActRelationship and Participation classes is the same as a common class, i.e., a rectangular box.

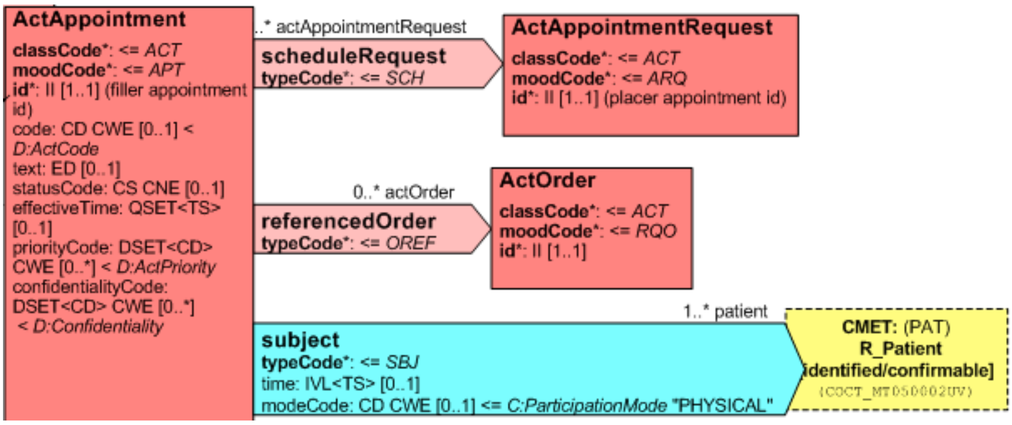


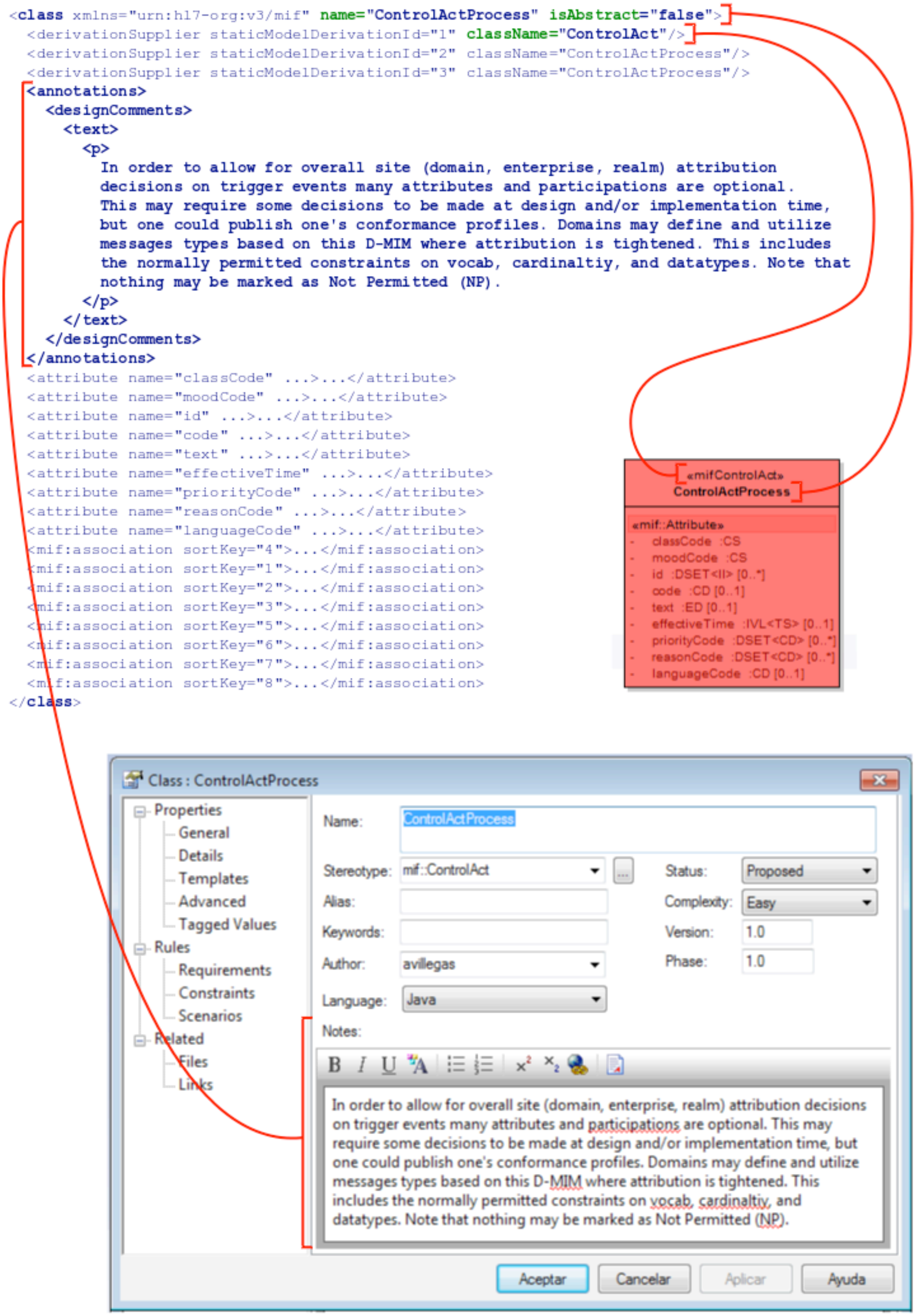
Figure 48. Graphical representation of Participation and ActRelationship classes in HL7.

The MIF XML representation of a class node contains several attributes and subnodes that provide semantics to the class. The node attributes name and isAbstract indicate the name of the class and whether it is abstract or not. If the class is abstract, the name of the class in UML appears in italics.

The first derivationSupplier subnode of the MIF class indicates the RIM class from which it is derived. Figure 49 shows that ControlActProcess is a class that is derived from the ControlAct RIM class, and therefore the stereotype used in the UML representation must be «ControlAct» from the Foundational Area of the UML profile for MIF static models.

The annotations node includes definitions, usage notes, constraints, and any other textual comment that provide additional information about the class. The text within this node is included as an annotation in the equivalent UML class. Since this specification uses Enterprise Architect, the description text must be included in the Notes textbox of the properties dialog for the class.

The class node also includes the subnodes representing attributes and associations owned by the class. These subnodes are described in the following subsections of this specification.



```
<class xmlns="urn:hl7-org:v3/mif" name="ControlActProcess" isAbstract="false">
  <derivationSupplier staticModelDerivationId="1" className="ControlAct"/>
  <derivationSupplier staticModelDerivationId="2" className="ControlActProcess"/>
  <derivationSupplier staticModelDerivationId="3" className="ControlActProcess"/>
  <annotations>
    <designComments>
      <text>
        <p>
          In order to allow for overall site (domain, enterprise, realm) attribution
          decisions on trigger events many attributes and participations are optional.
          This may require some decisions to be made at design and/or implementation time,
          but one could publish one's conformance profiles. Domains may define and utilize
          messages types based on this D-MIM where attribution is tightened. This includes
          the normally permitted constraints on vocab, cardinaltiy, and datatypes. Note that
          nothing may be marked as Not Permitted (NP).
        </p>
      </text>
    </designComments>
  </annotations>
  <attribute name="classCode" ...>...</attribute>
  <attribute name="moodCode" ...>...</attribute>
  <attribute name="id" ...>...</attribute>
  <attribute name="code" ...>...</attribute>
  <attribute name="text" ...>...</attribute>
  <attribute name="effectiveTime" ...>...</attribute>
  <attribute name="priorityCode" ...>...</attribute>
  <attribute name="reasonCode" ...>...</attribute>
  <attribute name="languageCode" ...>...</attribute>
  <mif:association sortKey="4">...</mif:association>
  <mif:association sortKey="1">...</mif:association>
  <mif:association sortKey="2">...</mif:association>
  <mif:association sortKey="3">...</mif:association>
  <mif:association sortKey="5">...</mif:association>
  <mif:association sortKey="6">...</mif:association>
  <mif:association sortKey="7">...</mif:association>
  <mif:association sortKey="8">...</mif:association>
</class>
```

Class : ControlActProcess

Name: ControlActProcess

Stereotype: mif::ControlAct

Status: Proposed

Alias:

Complexity: Easy

Keywords:

Version: 1.0

Author: avillegas

Phase: 1.0

Language: Java

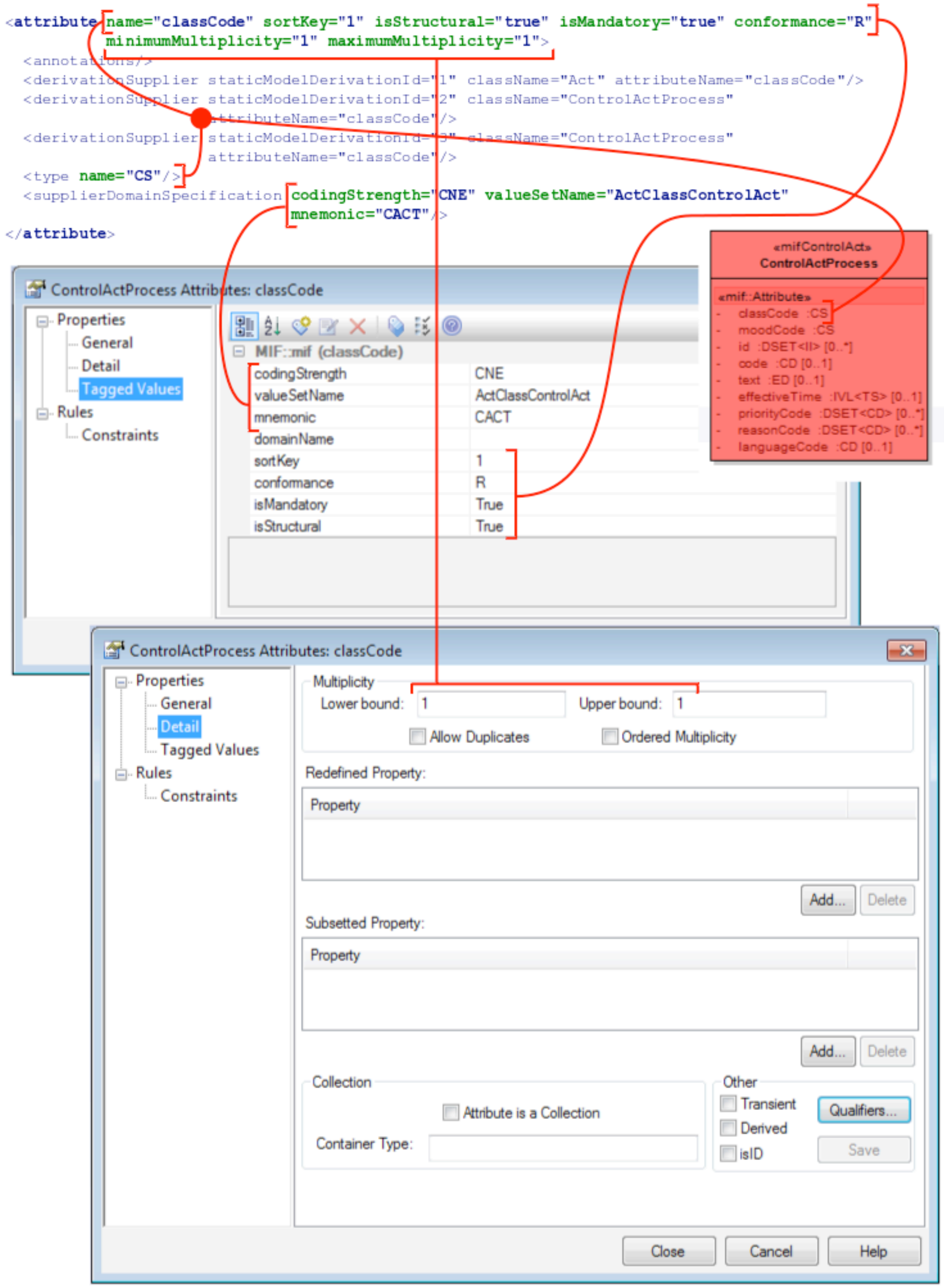
Notes:

In order to allow for overall site (domain, enterprise, realm) attribution decisions on trigger events many attributes and participations are optional. This may require some decisions to be made at design and/or implementation time, but one could publish one's conformance profiles. Domains may define and utilize messages types based on this D-MIM where attribution is tightened. This includes the normally permitted constraints on vocab, cardinaltiy, and datatypes. Note that nothing may be marked as Not Permitted (NP).

Figure 49. Transformation of MIF class to UML.

9.5 Attribute

Class attributes are the core components of the information model. The attributes are the source for all the information content of HL7. The attributes in a MIF static model depict aspects of classes that are important for communication between healthcare systems.



```
<attribute name="classCode" sortKey="1" isStructural="true" isMandatory="true" conformance="R"
    minimumMultiplicity="1" maximumMultiplicity="1">
  <annotations/>
  <derivationSupplier staticModelDerivationId="1" className="Act" attributeName="classCode"/>
  <derivationSupplier staticModelDerivationId="2" className="ControlActProcess"
                      attributeName="classCode"/>
  <derivationSupplier staticModelDerivationId="3" className="ControlActProcess"
                      attributeName="classCode"/>
  <type name="CS"/>
  <supplierDomainSpecification codingStrength="CNE" valueSetName="ActClassControlAct"
                               mnemonic="CACT"/>
</attribute>
```

ControlActProcess Attributes: classCode

- Properties
  - General
  - Detail
  - Tagged Values
- Rules
  - Constraints

MIF::mif (classCode)

codingStrength	CNE
valueSetName	ActClassControlAct
mnemonic	CACT
domainName	
sortKey	1
conformance	R
isMandatory	True
isStructural	True

«mifControlAct»
ControlActProcess

«mif::Attribute»
- classCode :CS
- moodCode :CS
- id :DSET<II> [0..*]
- code :CD [0..1]
- text :ED [0..1]
- effectiveTime :IVL<TS> [0..1]
- priorityCode :DSET<CD> [0..*]
- reasonCode :DSET<CD> [0..*]
- languageCode :CD [0..1]

ControlActProcess Attributes: classCode

- Properties
  - General
  - Detail
  - Tagged Values
- Rules
  - Constraints

Multiplicity
Lower bound: 1   Upper bound: 1
□ Allow Duplicates   □ Ordered Multiplicity

Redefined Property:
Property

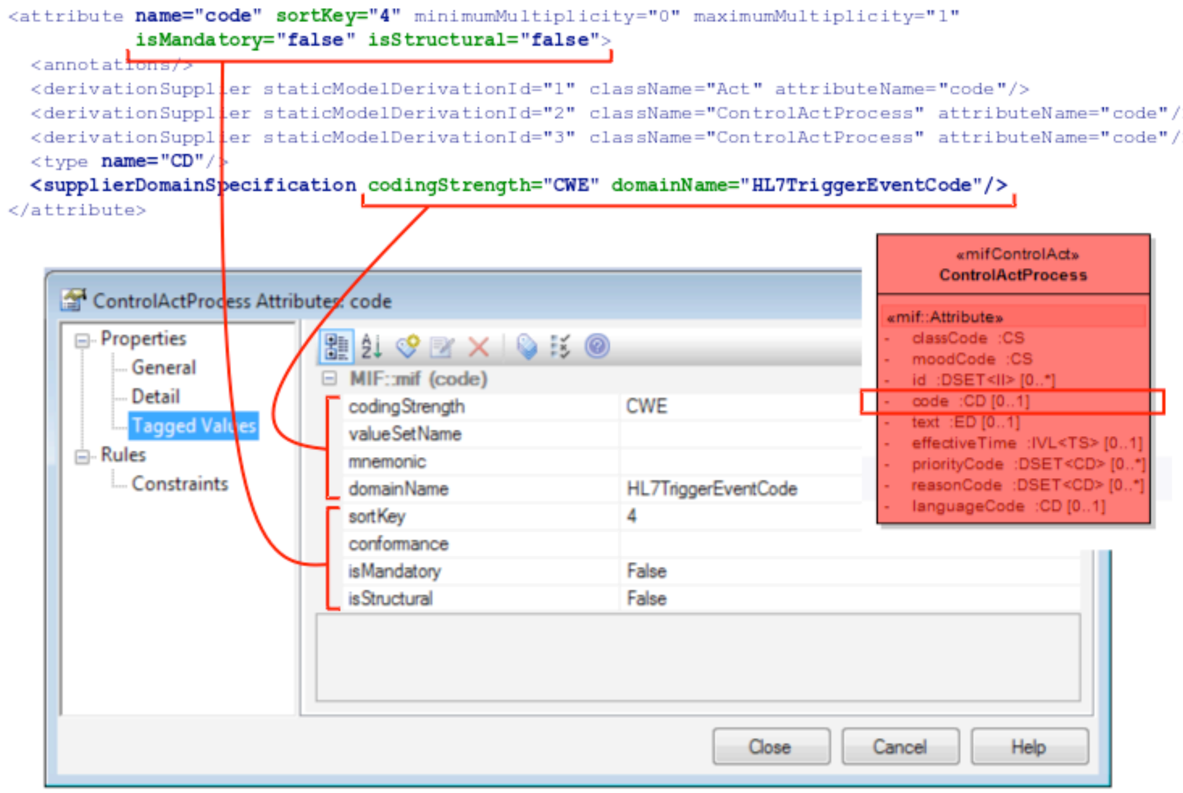
Subsetted Property:
Property

Collection
□ Attribute is a Collection
Container Type:

Other
□ Transient
□ Derived
□ isID

Figure 50. Transformation of MIF attribute to UML.

In UML, an attribute has a name, a type, and a multiplicity. On the other side, attributes in HL7 contain additional characteristics that must be taken into account when transforming to UML. In order to keep the same semantics in UML, all the attributes must have the stereotype «mifAttribute» that defines properties where to keep the additional information of MIF attributes.



```
<attribute name="code" sortKey="4" minimumMultiplicity="0" maximumMultiplicity="1"
           isMandatory="false" isStructural="false">
  <annotations/>
  <derivationSupplier staticModelDerivationId="1" className="Act" attributeName="code"/>
  <derivationSupplier staticModelDerivationId="2" className="ControlActProcess" attributeName="code"/>
  <derivationSupplier staticModelDerivationId="3" className="ControlActProcess" attributeName="code"/>
  <type name="CD"/>
  <supplierDomainSpecification codingStrength="CWE" domainName="HL7TriggerEventCode"/>
</attribute>
```

ControlActProcess Attributes: code

MIF::mif (code)	
codingStrength	CWE
valueSetName	
mnemonic	
domainName	HL7TriggerEventCode
sortKey	4
conformance	
isMandatory	False
isStructural	False

Figure 51. Transformation of MIF attribute to UML (II).

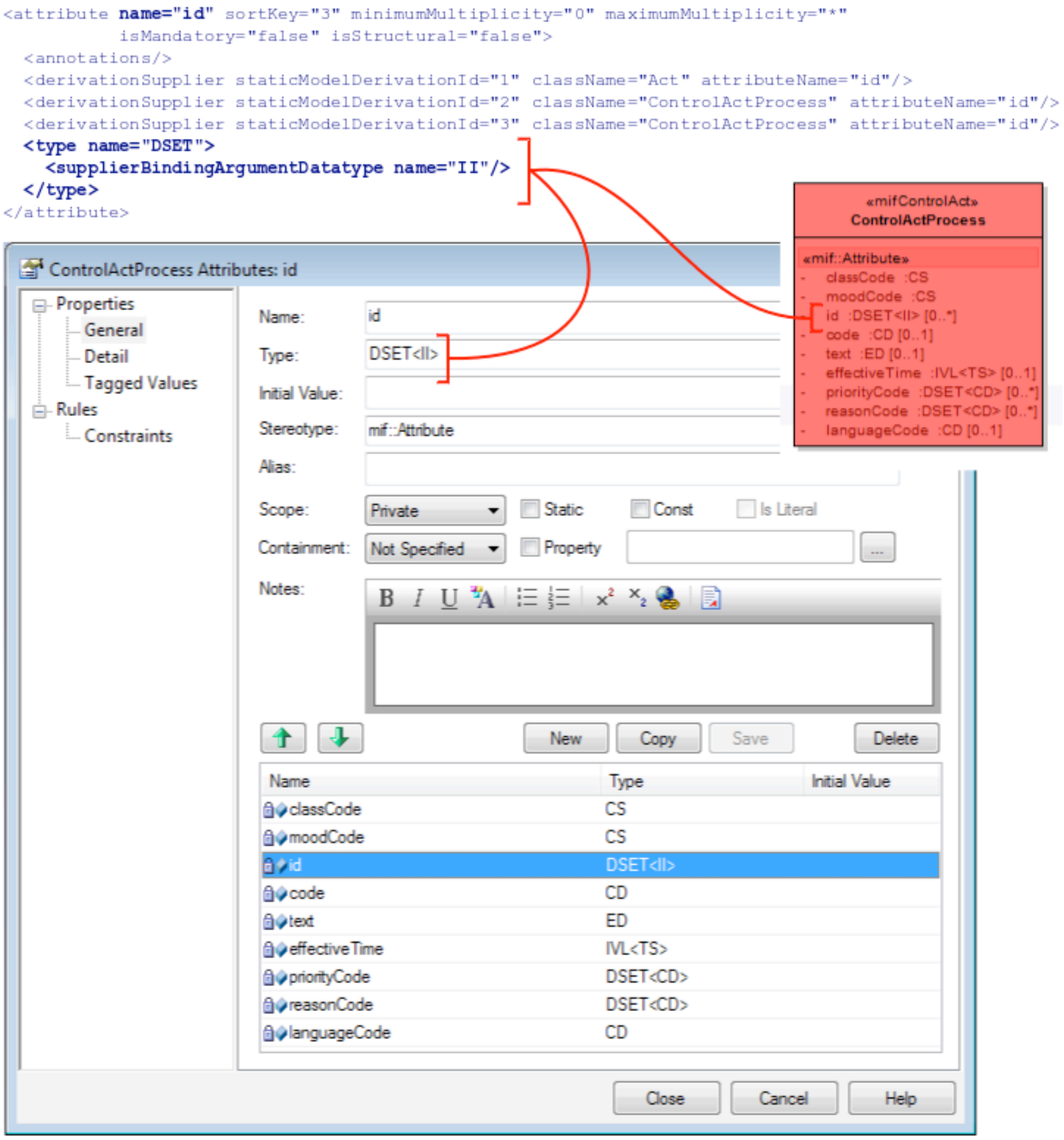
A number of attribute conventions may appear in MIF static model to convey information that would not otherwise be apparent. Typically, these conventions are used only if the default values are not used, although they may be used even if the default values are used. Conventions applied to a particular attribute appear beneath the particular attribute to which they apply and are slightly indented. The conventions that can be used and their mapping to UML attributes with the «mifAttribute» stereotype are described below:

- Mandatory clause: Figure 50 shows the MIF XML specification of the attribute classCode owned by ControlActProcess class. The attributes isStructural and isMandatody specify the mandatory conventions of an attribute. The values for this attributes are stored in the properties isStructural and isMandatody of the «mifAttribute» stereotype.

- Conformance clause: The conformance indicator constraint specifies how a model element must be handled by sending or receiving applications. Figure 50 shows the MIF XML specification of the attribute classCode owned by ControlActProcess class. The attribute conformance specify the conformance indicator of the attribute. The value for this attributes is stored in the property conformance of the «mifAttribute» stereotype.

- Data type: The type node in the MIF XML attribute specification denotes the name of the data type. It must be the name of a data type from the HL7 Abstract Data Types Specification. The type node must contain supplierBindingArgumentDataType subnodes for complex data types. Note that for complex types the hierarchy of subnodes defines the name of the data type by means of angle brackets. For instance, Figure 52 shows the attribute id whose type is DSET<II>.

- Domain: The vocabulary domain associated with a particular attribute instance is designated by supplierDomainSpecification node in the MIF XML specification. This characteristic is defined mostly when the data type of the attribute is a code data type. The domain specification must be either a domain name defined in the vocabulary tables, or a single code value from the appropriate domain. MIF format specifies the attributes within the supplierDomainSpecification node of an attribute node in several ways. For instance, Figure 50 shows an attribute of type CS (coded simple) whose supplierDomainSpecification contains a valueSetName and a mnemonic. These values are stored in the properties valueSetName and mnemonic of the «mifAttribute» stereotype. Alternatively, Figure 51 shows an attribute of type CD (concept descriptor) whose supplierDomainSpecification contains a domainName. This value is stored in the property domainName of the «mifAttribute» stereotype. There is also the possibility to get a MIF file with supplierDomainSpecification containing a codeSystemName attribute. In this case, the value is stored in the property valueSetName of the «mifAttribute» stereotype.



```
<attribute name="id" sortKey="3" minimumMultiplicity="0" maximumMultiplicity="*"
         isMandatory="false" isStructural="false">
  <annotations/>
  <derivationSupplier staticModelDerivationId="1" className="Act" attributeName="id"/>
  <derivationSupplier staticModelDerivationId="2" className="ControlActProcess" attributeName="id"/>
  <derivationSupplier staticModelDerivationId="3" className="ControlActProcess" attributeName="id"/>
  <type name="DSET">
    <supplierBindingArgumentDatatype name="II"/>
  </type>
</attribute>
```

ControlActProcess Attributes: id

Name: id

Type: DSET<II>

Initial Value:

Stereotype: mif::Attribute

Alias:

Scope: Private □ Static □ Const □ Is Literal

Containment: Not Specified □ Property

Notes:

Name	Type	Initial Value
classCode	CS	
moodCode	CS	
id	DSET<II>	
code	CD	
text	ED	
effectiveTime	IVL<TS>	
priorityCode	DSET<CD>	
reasonCode	DSET<CD>	
languageCode	CD	

«mifControlAct»
ControlActProcess

«mif::Attribute»
- classCode :CS
- moodCode :CS
- id :DSET<II> [0..*]
- code :CD [0..1]
- text :ED [0..1]
- effectiveTime :IVL<TS> [0..1]
- priorityCode :DSET<CD> [0..*]
- reasonCode :DSET<CD> [0..*]
- languageCode :CD [0..1]

Figure 52. Transformation of MIF attribute with complex type to UML.

- Vocabulary strength: Vocabulary strength may be denoted by the use of CWE (coded with extension) or CNE (coded, no extension). The strength is defined when the data type of the attribute is a code data type. The attribute codingStrength of an attribute node in a MIF XML representation indicates the strength of the attribute. The value for this attributes is stored in the property codingStrength of the «mifAttribute» stereotype.

- Ordering: The order of the attributes is important in HL7. The attribute sortKey of an attribute node in a MIF XML representation indicates the position of the attribute in the list of owned attributes of a particular class. The value for this attributes is stored in the property sortKey of the «mifAttribute» stereotype.

- Multiplicity: Attribute cardinalities are denoted with the minMultiplicity and maxMultiplicity attributes of the attribute node in the MIF XML representation. The information of the multiplicity of an attribute is also defined for UML attributes. Since this specification uses Enterprise Architect, the multiplicity values are stored in the details section of the Properties dialog for the attribute, as depicted in Figure 50.

9.6 Association

An association defines a relationship between objects. The objects may be instances of two different classes or of the same class (reflexive association). Just as classes have instances, associations have instances too. An association instance is a connection between objects and is defined by an association that connects classes.

Associations in the HL7 information models have two ends. Each end of the association instance connects with one and only one object. However, one object may be associated with more than one object of the same class by the same association. In this case, multiple association instances exist, each connecting exactly two objects. The number of instances of an association that can connect to one object is regulated by the multiplicities of the association.

An association multiplicity specifies the minimum and maximum number of objects of each class participating in the association. The multiplicity is expressed as a pair of cardinal numbers (i.e., non-negative integers). The lower bound is a non-negative integer, usually zero or one. The upper bound is an integer greater or equal to min, usually one, or unlimited, indicated by an asterisk ("*").

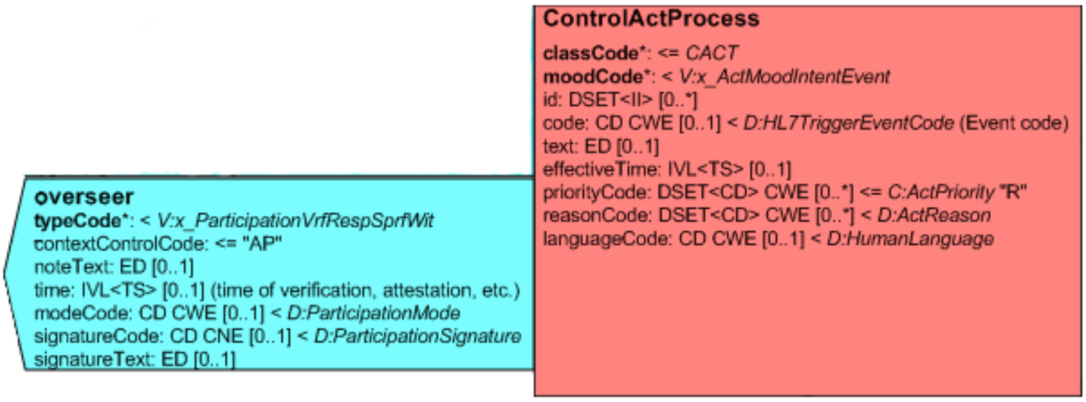
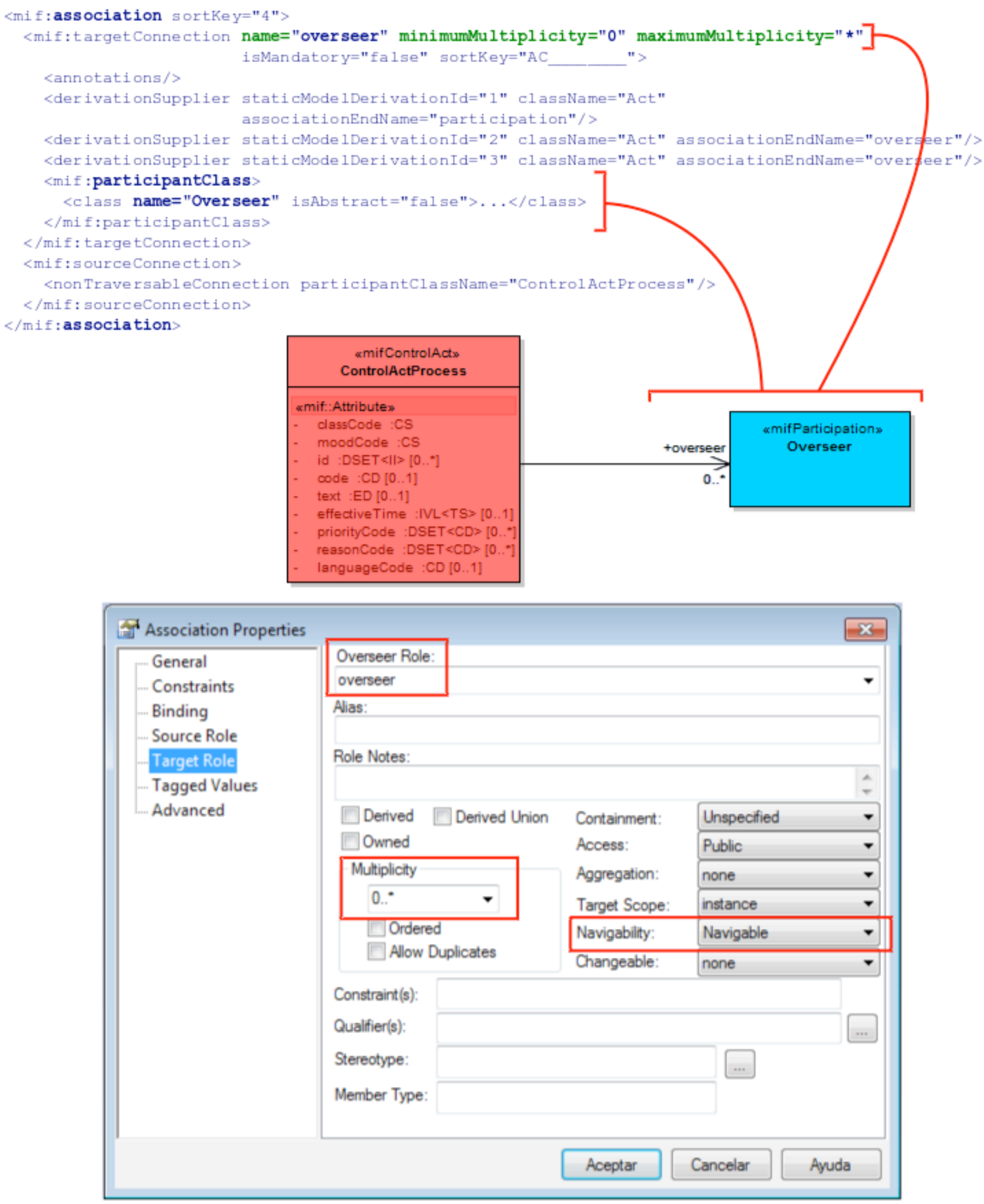

ControlActProcess
classCode*: <= CACT
moodCode*: < V:x_ActMoodIntentEvent
id: DSET<II> [0..*]
code: CD CWE [0..1] < D:HL7TriggerEventCode (Event code)
text: ED [0..1]
effectiveTime: IVL<TS> [0..1]
priorityCode: DSET<CD> CWE [0..*] <= C:ActPriority "R"
reasonCode: DSET<CD> CWE [0..*] < D:ActReason
languageCode: CD CWE [0..1] < D:HumanLanguage
overseer
typeCode*: < V:x_ParticipationVrfRespSprfWit
contextControlCode: <= "AP"
noteText: ED [0..1]
time: IVL<TS> [0..1] (time of verification, attestation, etc.)
modeCode: CD CWE [0..1] < D:ParticipationMode
signatureCode: CD CNE [0..1] < D:ParticipationSignature
signatureText: ED [0..1]

Figure 53. Graphical representation of an association between an act (red) and a participation (blue) in HL7.

HL7 represents associations in two different ways. Figure 53 shows the ControlActProcess act and the overseer Participation. Both classes are directly linked without requiring an explicit line indicating that there is an association between them. That situation happens between participation and act classes, between role and role link classes, and between act and act relationship classes. The salmon colored boxes represent ActRelationship classes, the blue arrowed boxes represent Participation classes, and the yellow arrowed classes represent RoleLink classes. In all of these unions, there is a "source" for the union, and a "target." The tail of the arrow is anchored to the source class, while the head of the arrow points to the target class.

Figure 55 presents two common explicit relationships linking the role AssignedPerson with the entities Person and Organization.



```
<mif:association sortKey="4">
  <mif:targetConnection name="overseer" minimumMultiplicity="0" maximumMultiplicity="*"
                        isMandatory="false" sortKey="AC________">
    <annotations/>
    <derivationSupplier staticModelDerivationId="1" className="Act"
                        associationEndName="participation"/>
    <derivationSupplier staticModelDerivationId="2" className="Act" associationEndName="overseer"/>
    <derivationSupplier staticModelDerivationId="3" className="Act" associationEndName="overseer"/>
    <mif:participantClass>
      <class name="Overseer" isAbstract="false">...</class>
    </mif:participantClass>
  </mif:targetConnection>
  <mif:sourceConnection>
    <nonTraversableConnection participantClassName="ControlActProcess"/>
  </mif:sourceConnection>
</mif:association>
```

Overseer Role: overseer

Alias:

Role Notes:

□ Derived □ Derived Union

□ Owned

Multiplicity: 0..*

□ Ordered

□ Allow Duplicates

Containment: Unspecified

Access: Public

Aggregation: none

Target Scope: instance

Navigability: Navigable

Changeable: none

Constraint(s):

Qualifier(s):

Stereotype:

Member Type:

Figure 54. Transformation of MIF association to UML.

Figure 54 includes the specification of an association node in a MIF XML file. This node contains two subnodes indicating the participants in the relationship. The sourceConnection node presents the element from which the relationship starts, while the targetConnection indicates to which element the association ends. The source element is always the element node (class, entry point, CMET or Choice) that contains the association node. The UML representation of a MIF relationship is a common UML binary relationship linking the sourceConnection element with the targetConnection element.

The targetConnection node contains the attributes name, minMultiplicity and maxMultiplicity that indicate the characteristics of the association end placed in the side of the targetConnection element. For the case of Figure 54, the targetConnection element defines the association end overseer, with multiplicity zero to many (0..*) in the side of the participation class named Overseer. The sourceConnection element indicates that the opposite association end to the overseer end is the act class ControlActProcess. Since this specification uses Enterprise

Architect , the multiplicity and role name of the targetConnection element are stored within the Target Role section of the Association properties dialog. Also, the value for the Navegability property must be changed to "Navigable".

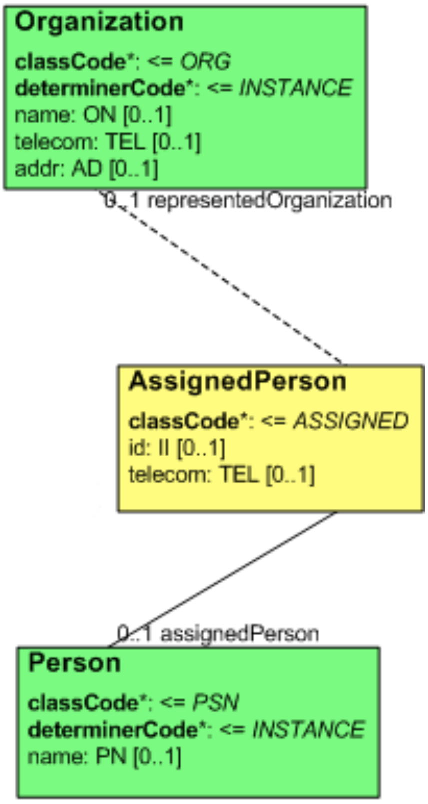


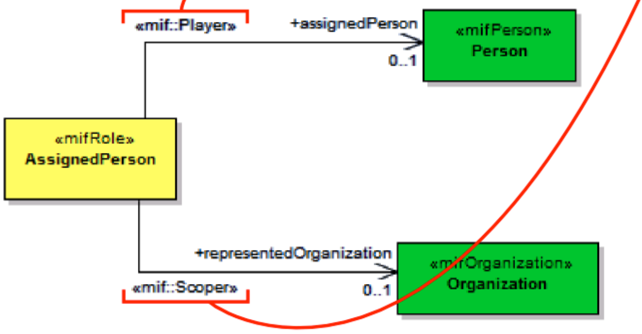
Figure 55. Graphical representation of player/scoper associations between role (yellow) and entity (green) in HL7.

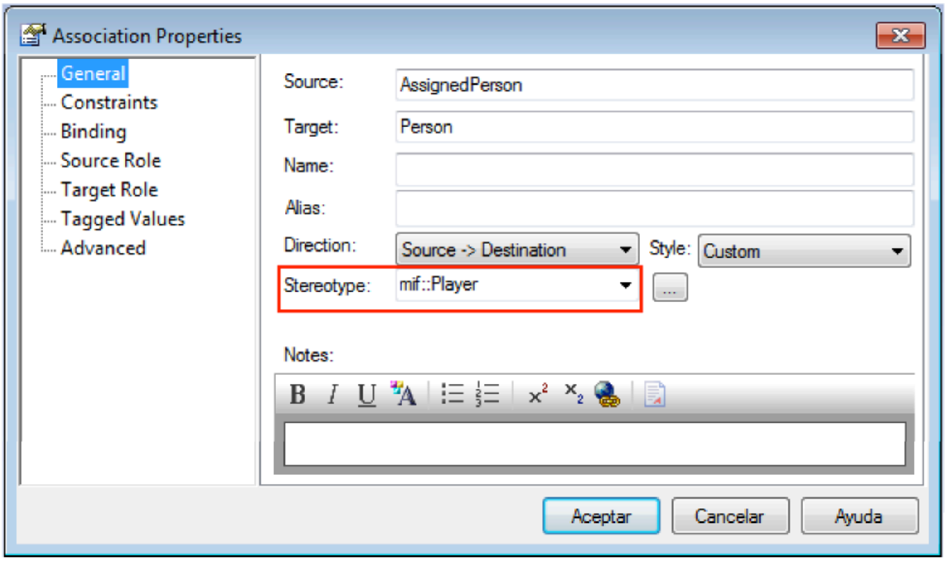
Looking at the Role (yellow boxes) and Entity (green boxes) classes on a static model like the fragment shown in Figure 55, it is frequent to see solid and dotted lines connecting them. One yellow Role class may be connected to up to two green Entity classes. These lines represent the relationships between the Role and Entity classes. One Entity is the player of the role (e.g., person) while the other represents the Entity that scopes (recognizes or assigned) the role (e.g., a healthcare organization). The solid line for a role identifies playing Entity (e.g. the Person acting as AssignedPerson), while the dashed line represents the scoping Entity (e.g., the organization that assigned the role of AssignedPerson). Look at the relationships defined for the AssignedPerson, Person and Organization classes in Figure 55. The classCode attribute in the yellow AssignedPerson role class identifies that the role being played is ASSIGNED (assigned person). The id attribute in the Patient role will contain the assigned person identification number. The Entity recognizing the assigned person number for that Person (and who the person is a assigned person of) is the Organization that scopes the role.

The cardinality of the associations is shown adjacent to the lines. When missing, the default cardinality is zero to many (0..*). As shown in Figure 55, Organization (green box) scopes zero or more AssignedPersons (i.e., the Organization assigns the role of AssignedPerson to zero or more Assigned Persons) while the AssignedPerson is scoped by zero or one Organization. On the playing end, the AssignedPerson role is played by zero or one Person while the Person entity plays zero or more AssignedPerson roles (the person can play the role of AssignedPerson zero or more times with respect to several provider organizations).

For the case of the relationships in Figure 55, the "player" relationship between AssignedPerson and Person is marked with the stereotype «mifPlayer», while the "scoper" relationship between AssignedPerson and Organization is marked with the stereotype «mifScoper». This way the semantics between the MIF representation of player/scoper associations is maintained in UML, as shown in Figure 56.

```
<mif:association sortKey="21">
  <mif:graphicRepresentation>...</mif:graphicRepresentation>
  <mif:targetConnection name="assignedPerson" minimumMultiplicity="0" maximumMultiplicity="1"
                      isMandatory="false" sortKey="AZ">
    <mif:derivationSupplier staticModelDerivationId="1" className="Role" associationEndName="player"/>
    <mif:derivationSupplier staticModelDerivationId="2" className="AssignedPerson"
                            associationEndName="assignedPerson"/>
    <mif:derivationSupplier staticModelDerivationId="3" className="AssignedPerson"
                            associationEndName="assignedPerson"/>
    <mif:participantClass>
      <mif:class name="Person" isAbstract="false">...</mif:class>
    </mif:participantClass>
  </mif:targetConnection>
  <mif:sourceConnection>
    <mif:nonTraversableConnection participantClassName="AssignedPerson">
                                                        ...</mif:nonTraversableConnection>
  </mif:sourceConnection>
</mif:association>
<mif:association sortKey="22">
  <mif:graphicRepresentation>...</mif:graphicRepresentation>
  <mif:targetConnection name="representedOrganization" minimumMultiplicity="0" maximumMultiplicity="1"
                      isMandatory="false" sortKey="BZ">
    <mif:derivationSupplier staticModelDerivationId="1" className="Role" associationEndName="scoper"/>
    <mif:derivationSupplier staticModelDerivationId="2" className="AssignedPerson"
                            associationEndName="representedOrganization"/>
    <mif:derivationSupplier staticModelDerivationId="3" className="AssignedPerson"
                            associationEndName="representedOrganization"/>
    <mif:participantClass>
      <mif:class name="Organization" isAbstract="false">...</mif:class>
    </mif:participantClass>
  </mif:targetConnection>
  <mif:sourceConnection>
    <mif:nonTraversableConnection participantClassName="AssignedPerson">
                                                        ...</mif:nonTraversableConnection>
  </mif:sourceConnection>
</mif:association>
```





Source: AssignedPerson

Target: Person

Name:

Alias:

Direction: Source -> Destination

Style: Custom

Stereotype: mif::Player

Figure 56. Transformation of MIF associations player and scoper to UML.

9.7 Choice

An HL7 MIF static model may contain Choice elements in the model. The Choice element is treated as if it were a class itself. The Choice element contains a finite number of other classes. Choices do not contain attributes; however both the Choice and any class contained in the Choice may participate as the source or target of associations.

Figure 57 shows the graphical representation of the EntityChoiceSubject choice in HL7. It contains two Entity classes: Person and NonPersonLivingSubject. The EntityChoiceSubject may have a number of relationships that apply to any of the classes in the Choice. In addition, some of the classes in the Choice may have associations that apply only to that class and not to the other classes in the Choice. On the other hand, a choice may contain other choices.

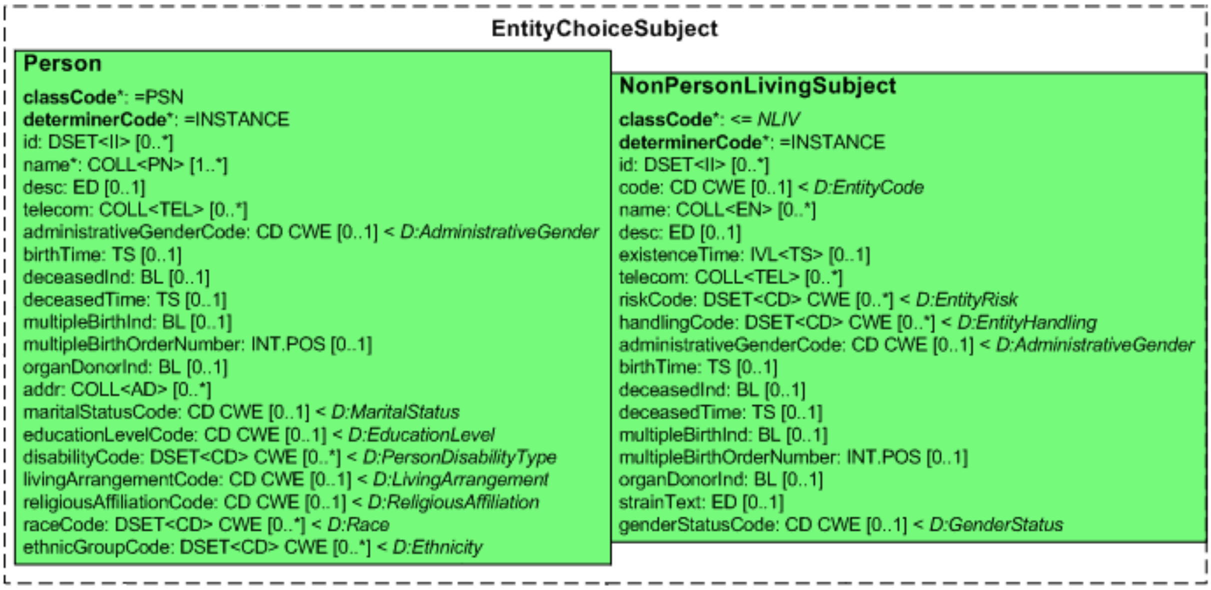

EntityChoiceSubject
Person
classCode*: =PSN
determinerCode*: =INSTANCE
id: DSET<II> [0..*]
name*: COLL<PN> [1..*]
desc: ED [0..1]
telecom: COLL<TEL> [0..*]
administrativeGenderCode: CD CWE [0..1] < D:AdministrativeGender
birthTime: TS [0..1]
deceasedInd: BL [0..1]
deceasedTime: TS [0..1]
multipleBirthInd: BL [0..1]
multipleBirthOrderNumber: INT.POS [0..1]
organDonorInd: BL [0..1]
addr: COLL<AD> [0..*]
maritalStatusCode: CD CWE [0..1] < D:MaritalStatus
educationLevelCode: CD CWE [0..1] < D:EducationLevel
disabilityCode: DSET<CD> CWE [0..*] < D:PersonDisabilityType
livingArrangementCode: CD CWE [0..1] < D:LivingArrangement
religiousAffiliationCode: CD CWE [0..1] < D:ReligiousAffiliation
raceCode: DSET<CD> CWE [0..*] < D:Race
ethnicGroupCode: DSET<CD> CWE [0..*] < D:Ethnicity
NonPersonLivingSubject
classCode*: <= NLIV
determinerCode*: =INSTANCE
id: DSET<II> [0..*]
code: CD CWE [0..1] < D:EntityCode
name: COLL<EN> [0..*]
desc: ED [0..1]
existenceTime: IVL<TS> [0..1]
telecom: COLL<TEL> [0..*]
riskCode: DSET<CD> CWE [0..*] < D:EntityRisk
handlingCode: DSET<CD> CWE [0..*] < D:EntityHandling
administrativeGenderCode: CD CWE [0..1] < D:AdministrativeGender
birthTime: TS [0..1]
deceasedInd: BL [0..1]
deceasedTime: TS [0..1]
multipleBirthInd: BL [0..1]
multipleBirthOrderNumber: INT.POS [0..1]
organDonorInd: BL [0..1]
strainText: ED [0..1]
genderStatusCode: CD CWE [0..1] < D:GenderStatus

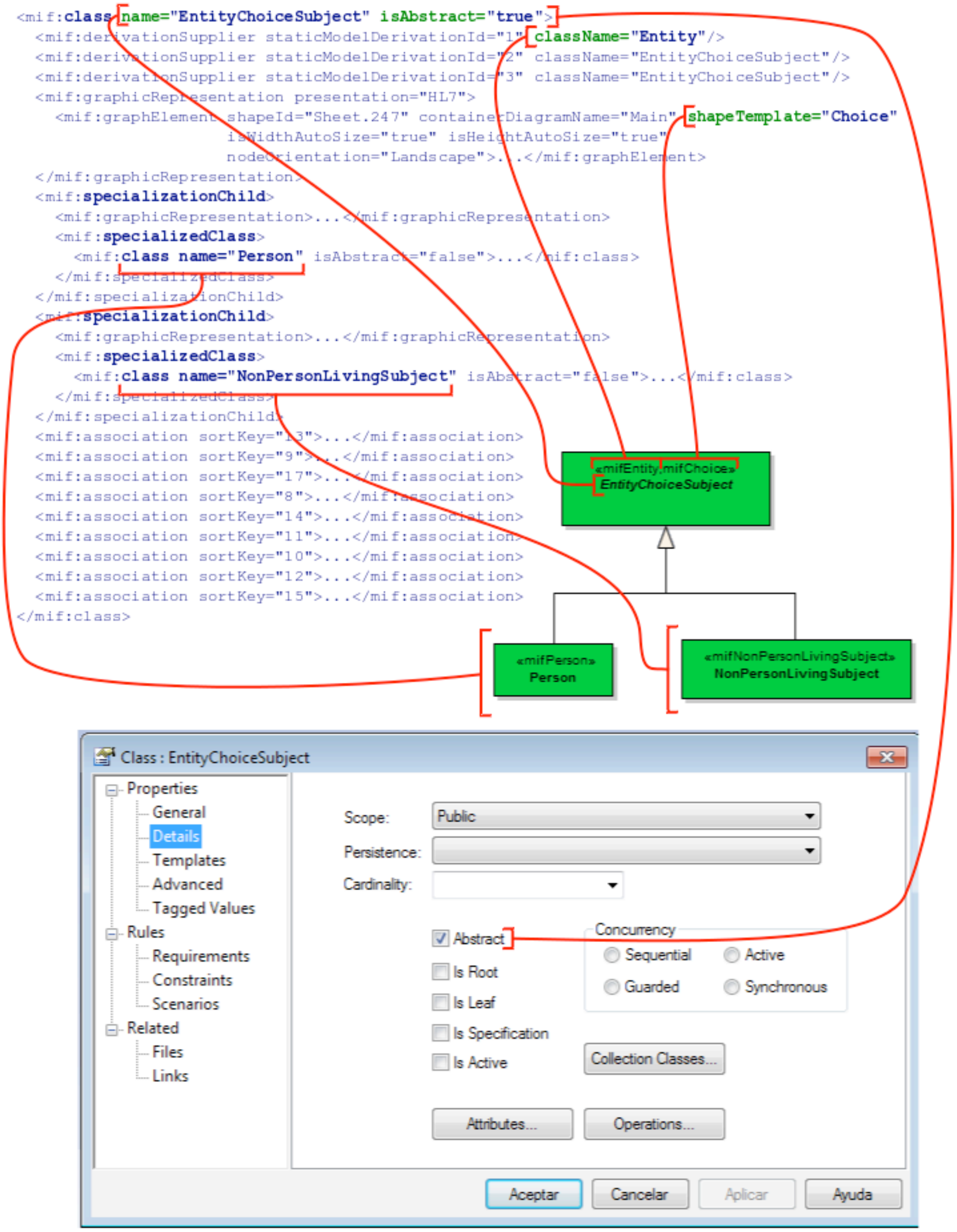
Figure 57. Graphical representation of a choice in HL7.

A Choice element is represented as a class with some particularities in the MIF XML specification. Figure 58 (top) shows the MIF XML excerpt that describes the EntityChoiceSubject choice. It is a class node with a name attribute and the isAbstract attribute set to true. In addition, Choices contain the attribute shapeTemplate with value “Choice” inside their graphicRepresentation subnode. Also, class nodes representing Choices contain one specializationChild subnode including the definition of several specializedClass nodes indicating the definition of those classes that are included within the Choice.

Choices are represented in UML as generalization hierarchies of classes, as depicted in Figure 58 (note that the attributes of the classes are hidden in Figure 58 for the sake of readability). The Choice class is the top-level class in the UML generalization hierarchy. It contains the name indicated in the name attribute of the class node in the MIF XML representation and it is abstract. Since this specification uses Enterprise Architect, the value to indicate that the Choice class is abstract must be included in the Details section of the properties dialog for the class, as depicted in Figure 58.

The Choice class must have two stereotypes. The first stereotype is «mifChoice» which indicates that the class is a Choice. The second stereotype is indicated by the first derivationSupplier subnode of the MIF class, which represents the RIM class from which the Choice class hierarchy is derived. Figure 58 shows that first derivationSupplier subnode contains the className attribute whose value is Entity. Thus, the Choice class must have the stereotype «mifEntity» and all the included subclasses must have also Entity-based stereotypes from the Foundational area of the UML Profile for MIF static models.

Subsequently, each of the members of the specializationChild node is constructed according to this MIF-UML transformation reference and linked to the Choice class with a generalization relationship indicating the hierarchy between the parent, i.e. the Choice class, and its descendants.



```
<mif:class name="EntityChoiceSubject" isAbstract="true">
  <mif:derivationSupplier staticModelDerivationId="1" className="Entity"/>
  <mif:derivationSupplier staticModelDerivationId="2" className="EntityChoiceSubject"/>
  <mif:derivationSupplier staticModelDerivationId="3" className="EntityChoiceSubject"/>
  <mif:graphicRepresentation presentation="HL7">
    <mif:graphElement shapeId="Sheet.247" containerDiagramName="Main" shapeTemplate="Choice"
                      isWidthAutoSize="true" isHeightAutoSize="true"
                      nodeOrientation="Landscape">...</mif:graphElement>
  </mif:graphicRepresentation>
  <mif:specializationChild>
    <mif:graphicRepresentation>...</mif:graphicRepresentation>
    <mif:specializedClass>
      <mif:class name="Person" isAbstract="false">...</mif:class>
    </mif:specializedClass>
  </mif:specializationChild>
  <mif:specializationChild>
    <mif:graphicRepresentation>...</mif:graphicRepresentation>
    <mif:specializedClass>
      <mif:class name="NonPersonLivingSubject" isAbstract="false">...</mif:class>
    </mif:specializedClass>
  </mif:specializationChild>
  <mif:association sortKey="13">...</mif:association>
  <mif:association sortKey="9">...</mif:association>
  <mif:association sortKey="17">...</mif:association>
  <mif:association sortKey="8">...</mif:association>
  <mif:association sortKey="14">...</mif:association>
  <mif:association sortKey="11">...</mif:association>
  <mif:association sortKey="10">...</mif:association>
  <mif:association sortKey="12">...</mif:association>
  <mif:association sortKey="15">...</mif:association>
</mif:class>
```

Class : EntityChoiceSubject

Scope: Public

Persistence:

Cardinality:

☑ Abstract

□ Is Root

□ Is Leaf

□ Is Specification

□ Is Active

Concurrency

○ Sequential ○ Active

○ Guarded ○ Synchronous

Figure 58. Transformation of MIF choice to UML.

9.8 CMET

CMETs (Common Model Element Types) are common references that are likely to be made by multiple models and where there is a desire for the references to all be consistent within a given domain. Examples include such concepts as Patient, Provider, Location, Annotation, etc. CMET references can exist for Act, Entity, Role and "Other" classes. The reference identifies the name of the CMET. CMET names are distinguished from the names of "standard" classes by being prefixed by "A_", "E_", "R_" or "O_", depending upon which base class is associated with the CMET.

CMETs are intended to express a common, reusable pattern. When a CMET is referenced, or used in another diagram, it is shown with a special notation, a box with dashed edges. It contains the name of the CMET, its artifact identification code, its class code and its level of attribution. Notice it is also color-coded in a manner consistent with its root class according to the RIM colors. Figure 59 shows an example of CMET referencing another static model. Note that the name R_PatientIdentified-confirmable indicates that the main class in the static model represented by the CMET is a Role.

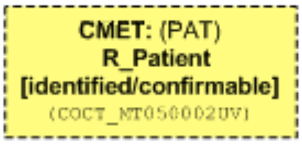
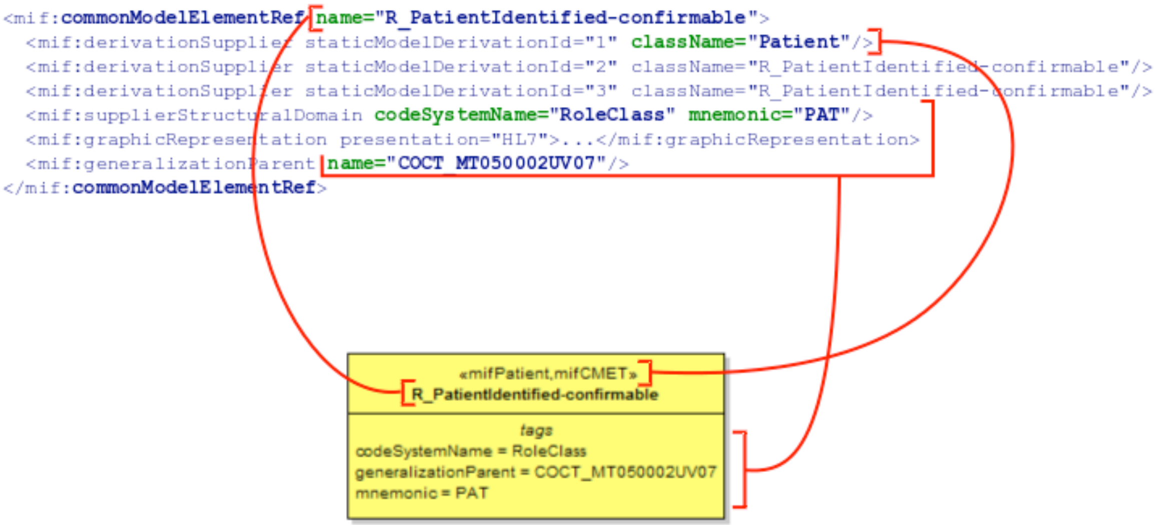


Figure 59. Graphical representation of a CMET in HL7.

In addition, Figure 60 shows the MIF XML representation of the R_PatientIdentified-confirmable CMET from Figure 59. A CMET is represented as a common class in UML with the stereotype «mifCMET». Also, the UML class representing the CMET must contain another stereotype from the Foundational area representing the kind of the main class referenced by the CMET. The first derivationSupplier subnode contains the className attribute that stores this information. Thus, the R_PatientIdentified-confirmable CMET must contain the stereotype «mifPatient».



```
<mif:commonModelElementRef name="R_PatientIdentified-confirmable">
  <mif:derivationSupplier staticModelDerivationId="1" className="Patient"/>
  <mif:derivationSupplier staticModelDerivationId="2" className="R_PatientIdentified-confirmable"/>
  <mif:derivationSupplier staticModelDerivationId="3" className="R_PatientIdentified-confirmable"/>
  <mif:supplierStructuralDomain codeSystemName="RoleClass" mnemonic="PAT"/>
  <mif:graphicRepresentation presentation="HL7">...</mif:graphicRepresentation>
  <mif:generalizationParent name="COCT_MT050002UV07"/>
</mif:commonModelElementRef>
```

«mifPatient,mifCMET»
R_PatientIdentified-confirmable
tags
codeSystemName = RoleClass
generalizationParent = COCT_MT050002UV07
mnemonic = PAT

Figure 60. Transformation of MIF CMET to UML.

Consequently, the artifact identification code and class code of the CMET are indicated within the supplierStructuralDomain and generalizationParent subnodes from the MIF XML representation. The supplierStructuralDomain node contain the attributes codeSystemName and mnemonic, which are the class code values of the CMET. The values for these attributes are mapped to the attributes codeSystemName and mnemonic of the «mifCMET» stereotype in UML. In the same way, the artifact identification code is indicated as the value for the name attribute in the generalizationParent node. It is mapped to the generalizationParent attribute of the «mifCMET» stereotype in UML, as shown in Figure 60.