Chapter 7
The HL7 V3 RIM

7.1 Origins

Even its supporters accept that HL7 V2 was developed in an ad hoc and unplanned way. For example, when an additional element is needed, it is added in the next available spot. Perhaps more importantly, V2 provides multiple ways of doing the same thing, leading to the well-founded jibe: "when you have seen one implementation of V2, you have seen one implementation; every one is different."

Work on HL7 Version 3 began in 1992 with the establishment of the HL7 Version 3 Task Force. As with many things, many of its characteristics are best understood by considering its origins as the planned successor to Version 2. The HL7 web-site explains the rationale for V3 as follows:

> Offering lots of optionality and thus flexibility, the V2.x series of messages are widely implemented and very successful. These messages evolved over several years using a "bottom-up" approach that has addressed individual needs through an evolving ad hoc methodology. There is neither a consistent view of that data that HL7 moves nor that data's relationship to other data.

HL7 Version 2's success is also largely attributable to its flexibility. It contains many optional data elements and data segments, making it adaptable to almost any site. While providing great flexibility, its optionality also makes it impossible to have reliable conformance tests of any vendor's implementation and also forces implementers to spend more time analyzing and planning their interfaces to ensure that both parties are using the same optional features.

Version 3 addresses these and other issues by using a well-defined methodology based on a reference information (data) model. Using rigorous analytic and message building techniques and incorporating more trigger events and message formats with very little optionality, HL7's primary goal for Version 3 was to offer a standard that would be definite and testable and provide the ability to certify vendors' conformance.

Version 3 uses an object-oriented development methodology and a Reference Information Model (RIM) to create messages. The RIM is an essential part of the HL7 Version 3 development methodology, as it provides an explicit representation

T. Benson, *Principles of Health Interoperability HL7 and SNOMED*, HI,
DOI 10.1007/978-1-84882-803-2_7,

of the semantic and lexical connections that exist between the information carried in the fields of HL7 messages.[1]

In summary, HL7 V3 is designed to be comprehensive in scope, complete in detail, extensible as requirements change, up-to-date and model-based, conformance testable, and technology-independent. It is based on the RIM (Reference Information Model). The RIM was conceived as a universal reference model for healthcare interoperability, covering the entire healthcare domain. Each message specification would be a view into this model. The RIM is at the core of HL7 Version 3; you cannot understand V3 without understanding the RIM.

The effort to develop the RIM took place in two distinct phases.

During the first phase, from about 1992 to 1999, a large complex class model with more was developed comprising more than a hundred classes and several hundred attributes and associations, supported by extensive documentation. In many ways, this was just a rationalized super-set of the content of HL7 V2. However, many people considered this model to be just too large to learn and use.

During 1998–1999, a radical approach, known as the Unified Service Action Model (USAM), was proposed to simplify the problem (Schadow et al. 2000). After a heated debate, HL7 resolved to adopt USAM with effect from January 2000. USAM is based on two key ideas, which lead directly to the structure of the RIM as we know it today.

The first idea is that most healthcare documentation is concerned with "happenings" and things (human or other) participate in these happenings in various ways. Furthermore, happenings have a natural life cycle such as the concept itself, an intent for it to happen, the happening, and the consequences of its happening. These are like the moods of a verb.

The second idea is to recognize that the same people and things can perform different roles when participating in different types of happening. For example, a person can be either a care provider or the subject of care.

7.2 Overview

HL7 Version 3 is a lingua franca used by healthcare computers to talk to other computers, to help provide information when and where needed. Healthcare communication is complex and any language needs to accommodate this complexity and also handle future needs. HL7 Version 3 is designed to handle most if not all healthcare communications in an unambiguous way, using a relatively small set of constructs, which can be learnt relatively easily.

The HL7 RIM (Reference Information Model) specifies the grammar of V3 messages and, specifically, the basic building blocks of the language (nouns, verbs, etc.), their permitted relationships, and Data Types. The RIM is not a model of health care, although it is healthcare-specific, nor is it a model of any message, although it is used in messages.

[1] HL7. *HL7 Standards – HL7 Version 3.* www.hl7.org

At first sight, the RIM is quite simple. The RIM backbone has just five core classes and a number of permitted relationships between them. However, it presents quite a steep learning curve. The good news is that once you reached the plateau the ground becomes much less steep.

The RIM defines a set predefined Attributes for each class and these are the only ones allowed in HL7 messages. Each attribute has a specified Data Type. These Attributes and Data Types become tags in HL7 XML messages. Message specifications, to do a particular task, use a subset of the available RIM Attributes, listing each element used and how many repeats are allowed. This is known as refinement. Each Data Type is constrained to the simplest structure that meets the requirements of the task.

HL7 V3 uses a graphical representation, called Refined Message Information Model (RMIM) to display the structure of a message as a color-coded diagram. Most RMIMs can be shown on a single sheet of paper or PowerPoint slide and these RMIM diagrams are used to design messages and to explain what each HL7 message consists of. The actual interchange (the wire format) is usually XML.

All of the XML tags and attributes used in V3 messages are derived from the HL7 Reference Information Model (RIM) and the HL7 V3 Data Types. The structure of each HL7 message is set out in an XML schema, which specifies which tags and attributes are needed or allowed in the message, their order, and the number of times each may occur, together with annotations describing how each tag shall be used. HL7 message schema are lengthy, detailed, and verbose.

The RIM itself is shown in Fig. 7.1. The next part of this chapter sets out to explain how it works.

7.3 The RIM Backbone

The V3 RIM is based on a simple backbone structure, involving three main classes, Act, Role, and Entity, linked together using three association classes: ActRelationship, Participation, and RoleLink.

In HL7 V3, every happening is an Act, which is analogous to a verb in English. Each Act may have any number of Participations, which are Roles, played by Entities. These are analogous to nouns.

Each Act may also be related to other Acts, via Act-Relationships.

Act, Role, and Entity classes have a number of specializations. For example, Entity has a specialization called Living Subject, which itself has a specialization called Person. Person inherits the attributes of both Entity and Living Subject (Fig. 7.2).

7.3.1 Structural Attributes

Structural Attributes are a device, which is used to reduce the size of the original RIM from over 100 classes to a simple backbone of six main classes. Structural attributes are used to specify what each RIM class means when used in a message.

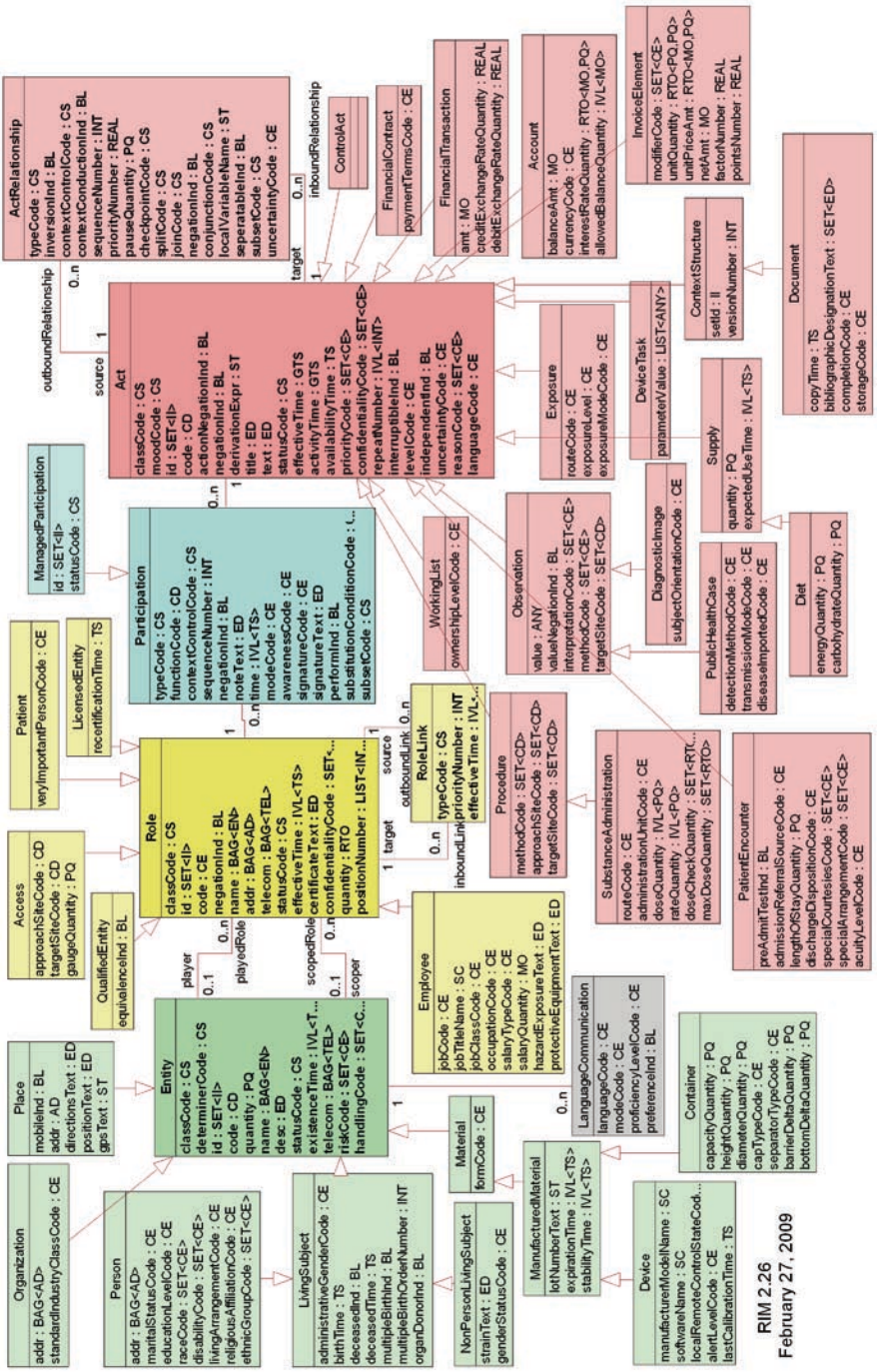


Fig. 7.1 HL7 V3 RIM normative content

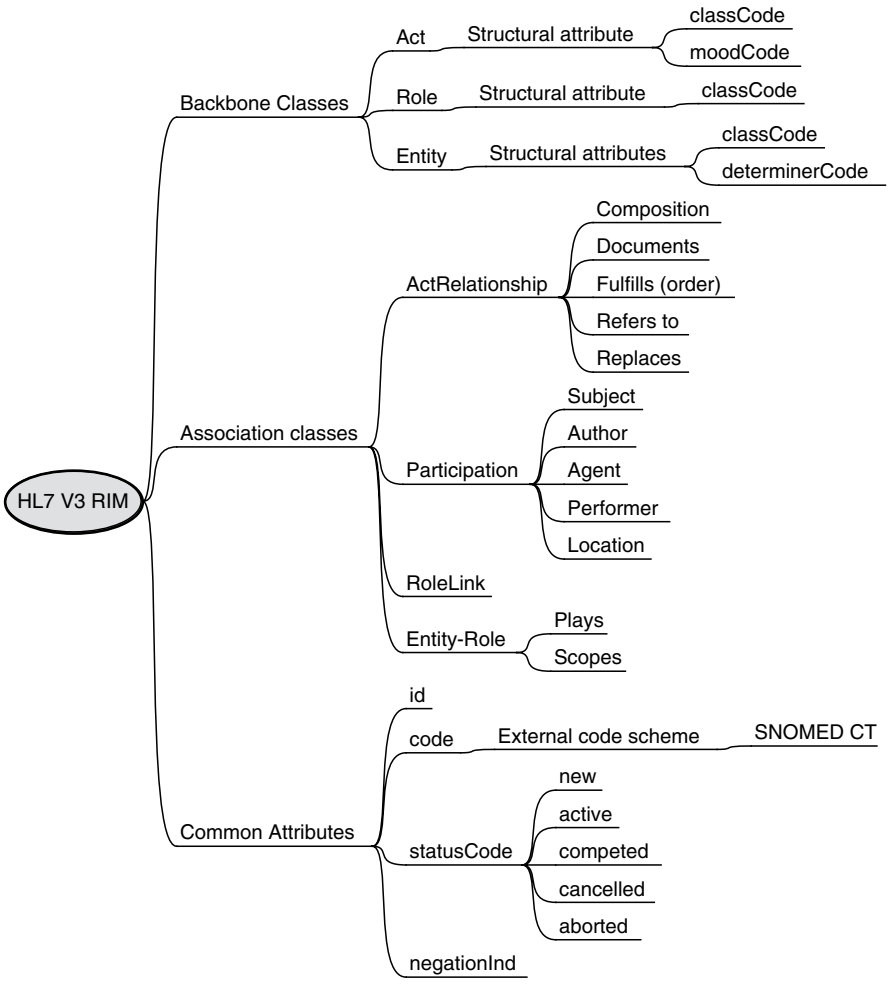


Fig. 7.2 HL7 V3 Reference information model backbone

For example, every Act has a *classCode* and a *moodCode*. The *classCode* states what sort of Act this is, such as observation, encounter, or administration of a drug.

The primary use of structural attributes is in the design of messages or other services, which are then implemented in computer applications. Message designers select the values of structural attributes when designing messages, or groups of related messages. These values are then frozen and may not be changed by application programmers or anyone else downstream.

In a very real sense, each class is named by its structural attributes. The semantic meaning of every class in an HL7 V3 message specification is specified by its structural attributes. The actual name of the class does not really matter; what matters is the meaning of its structural attributes.

The full set of structural attributes for each of the backbone RIM classes is:

- Act: classCode, moodCode, negationInd, levelCode
- Entity: classCode, determinerCode
- Role: classCode, negationInd
- ActRelationship: typeCode, inversionInd, contextControlCode, contextConduction Ind, negationInd
- Participation: typeCode, contextControlCode
- RoleLink: typeCode

7.3.2 Specializations

Each of the main backbone classes (Act, Role, and Entity) has a number other classes linked to it, using a line with an open triangle arrowhead at the backbone-class end. This is the UML symbol for specialization. The class that is pointing is a specialization of the class that is being pointed toward, which is a generalization.

The specialization inherits all of the properties of the generalization, while adding any specific attributes of its own. For example, the class Patient, at the top center of the RIM, is a specialization of Role with the addition of the optional attribute veryImportantPersonCode. The convention is that only a class, which has one or more additional attributes specific to itself, is shown on the RIM.

A number of frequently used attributes are found in more than one class. These include: *id, code*, and *status code*.

The conventional way to denote each attribute is to prefix it with its class name, so *Act.id* is the attribute *id* in Act, and *Role.id* is an *id* in Role.

id is used to identify classes and has the II (Instance Identifier) Data Type, which may be a universally unique identifier (UUID) or/and object identifier (OID).

Codes and identifiers avoid ambiguity in messages. The *id* attribute is used to give unique identity to people, persons, organizations, things, and information objects. There are also two main types of code. The first type covers the specialized codes used for structural attributes and are defined by HL7 itself. The second covers externally defined terms and codes such as SNOMED CT (Clinical Terms).

While *classCode* is a structural attribute used to indicate the name of an Act, Role, or Entity, the *code* attribute, is used to specify precisely what the class means at a leaf level of granularity. Unlike *classCode*, the code attribute is not a mandatory nor is it a structural attribute.

The *classCode* and *code* attributes are related in so far as *code* should always be a specialization of *classCode*. For example, if *Act.classCode* is a procedure, the *Act. code* must be a type of procedure and may not be anything else.

The *code* attribute is usually populated from an external coding scheme. External coding schemes are identified using an OID. The combination of the OID and *code* value is unique.

Each class may only contain a single *code*. If it is necessary to apply several attributes to a class, which are best done with *code*, then each code has to be in a separate class, which must be linked to the parent class using an ActRelationship.

The *negationInd* can be used to reverse the meaning of a class.

We now discuss the three core classes, Act, Entity, and Role in more detail.

7.4 Act

Act is a record of something that has happened or may happen.

Full representation of an Act identifies the kind of act (what happens), the actor who performs the deed, and the objects or subjects (e.g., patients) that the act affects. Additional information may be provided to indicate location (where), time (when), manner (how), together with reasons (why), or motives (what for).

Acts can be related to many other Acts using the ActRelationship class. For example, one Act may contain, cause, lead to, update, revise, or view information about other Acts.

Information in a document is treated as an Act – the act being the creation of the document content. Each transaction is a kind of act. An account is a record of a set of transactions.

7.4.1 Act classCode

Acts include an enormous range of happenings: events, such as encounter, visits, and appointments; observations such as tests, diagnoses, and examination findings; notifications such as alerts, confirmation, and consent; the supply and administration of medicines and other consumables; clinical, administrative, and financial procedures. The *classCode* field determines whether an Act is an observation, an encounter, or a procedure.

7.4.2 moodCode

Act has another important structural attribute called *moodCode*, which is similar to the tense of a verb. The term mood is a grammatical term representing a category of verb use typically expressing: fact (indicative mood), command (imperative mood), question (interrogative mood), wish (optative mood), or conditionality (subjunctive mood). *moodCode* indicates whether an Act has happened (an event), is a request for something to happen, a goal, or a criterion. For example, “weight = 100kg” is an observation event; “measure weight daily” is a request; “reduce weight to 80Kg” is a goal and “if weight is greater than 80Kg” is a criterion.

EVN	Event (occurrence)	A service that actually happens, may be an ongoing service or a documentation of a past service
RQO	Request	A request or order for a service is an intent directed from a placer (request author) to a fulfiller (service performer)
PRMS	Promise	An intent to perform a service that has the strength of a commitment. Other parties may rely on the originator of such promise that said originator will see to it that the promised act will be fulfilled
PRP	Proposal	A nonmandated intent to perform an act. Used to record intents that are explicitly not Orders
DEF	Definition	Definition of a service

moodCode is used to distinguish between an order (RQO), which is something you want to happen, and a report (EVN), which is something that has happened.

In clinical guidelines, *moodCode* can be used to distinguish between the definition of the guideline as originally authored (DEF), the intent that it should be followed for a particular patient (PRP) and the actual compliance (EVN).

7.4.3 StatusCode

statusCode specifies the state of an Act, such as:

New	Act is in preparatory stages and may not yet be acted upon
Active	The Act can be performed or is being performed
Completed	An Act that has terminated normally after all its constituents have been performed
Canceled	The act has been abandoned before activation
Aborted	The act has been terminated prior to the originally intended completion

The full state-machine diagram for the Act class is shown in Fig. 7.3.

7.4.4 Times

The Act class has two important time attributes: *activityTime* and *effectiveTime*, which have rather different meanings. The *activityTime* states when the Act itself occurs, but *effectiveTime* states the clinically relevant time of the Act. The difference is best explained by examples.

- The *activityTime* for an appointment booking is the time of making the appointment, while the *effectiveTime* is the appointment date/time.
- For a laboratory request, the *activityTime* is the time the request is made, while the *effectiveTime* is the time that the sample is requested to be taken (in

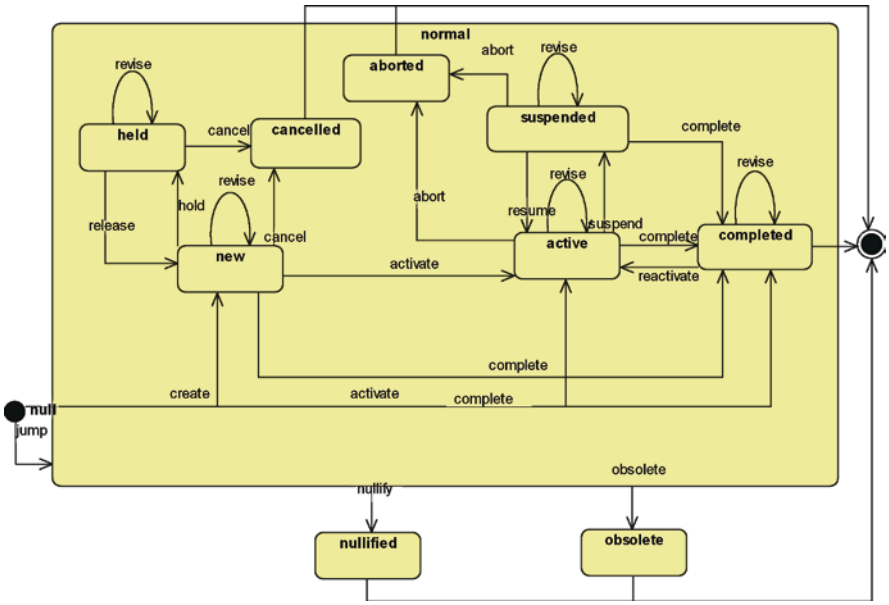


Fig. 7.3 State-machine diagram for Act class

hospitals, doctors often order blood tests with instructions for the sample to be taken at some future time).

- For a laboratory test result, the *activityTime* is the time the test was performed, but the *effectiveTime* is the time the sample is taken from the patient.
- For a contract, the *activityTime* is the date of the contract itself, while the *effectiveTime* is the time that the contract holds good.
- The activity time for a prescription is the date of the prescription, while the *effectiveTime* is how long the medication is to be taken.

7.4.5 Act Specializations

The most important Act specializations are Observation, Procedure, Substance Administration, Supply, and *PatientEncounter*.

7.4.5.1 Observation

Observation is defined as an Act of recognizing and noting information about the subject, and whose immediate and primary outcome (postcondition) is new data

about a subject. Observations often involve measurement or other elaborate methods of investigation, but may also be simply assertive statements, such as a diagnosis.

Many observations are structured as name-value pairs, where the Observation.code (inherited from Act) is the name and the Observation.value is the value of the property. The Observation class is always used to hold name-value pairs.

Observation.value contains the information determined by the observation action. It is unique in the RIM in that it has the Data Type ANY, which is to say it can be any Data Type, although in messages, the Data Type is usually constrained to a specific Data Type, such as physical quantity (PQ) or a code. This works in a way, which is similar to the OBX segment in V2, which can also contain any Data Type.

7.4.5.2 Procedure

A Procedure is defined as an Act whose immediate and primary outcome (postcondition) is the alteration of the physical condition of the subject.

Note that this definition of procedure is more limited than the definition of procedure used in SNOMED CT (see Chapter 12), although it includes most surgical procedures and physical treatment such as physiotherapy. It does not cover imaging or laboratory investigations, administrative procedures, counseling, or medication.

7.4.5.3 Substance Administration

Substance Administration is defined as the act of introducing or otherwise applying a substance to the subject. This class is used when prescribing a medicine, with a mood code of Intent, because the intent of a prescription is to administer medication.

Substance Administration is also used in Event mood to record that a medication has been administered to a patient.

7.4.5.4 Supply

Supply is defined as an act that involves provision of a material by one entity to another.

For example, dispensing a medicine is a Supply Act, while prescribing and administration are both *SubstanceAdministration* Acts.

Supply and *SubstanceAdministration* Acts require a participation link to identify the material or medicine involved.

7.4.5.5 *PatientEncounter*

PatientEncounter is defined as an interaction between a patient and a care provider for the purpose of providing healthcare-related services. Examples of *PatientEncounter* include inpatient and outpatient visits, home visits, and even telephone calls.

Appointments, which have been booked, are *PatientEncounters* in the mood Promise, until they have taken place, when the mood is changed to Event.

Details of the HL7 V3 Act Class are summarized in Fig. 7.4.

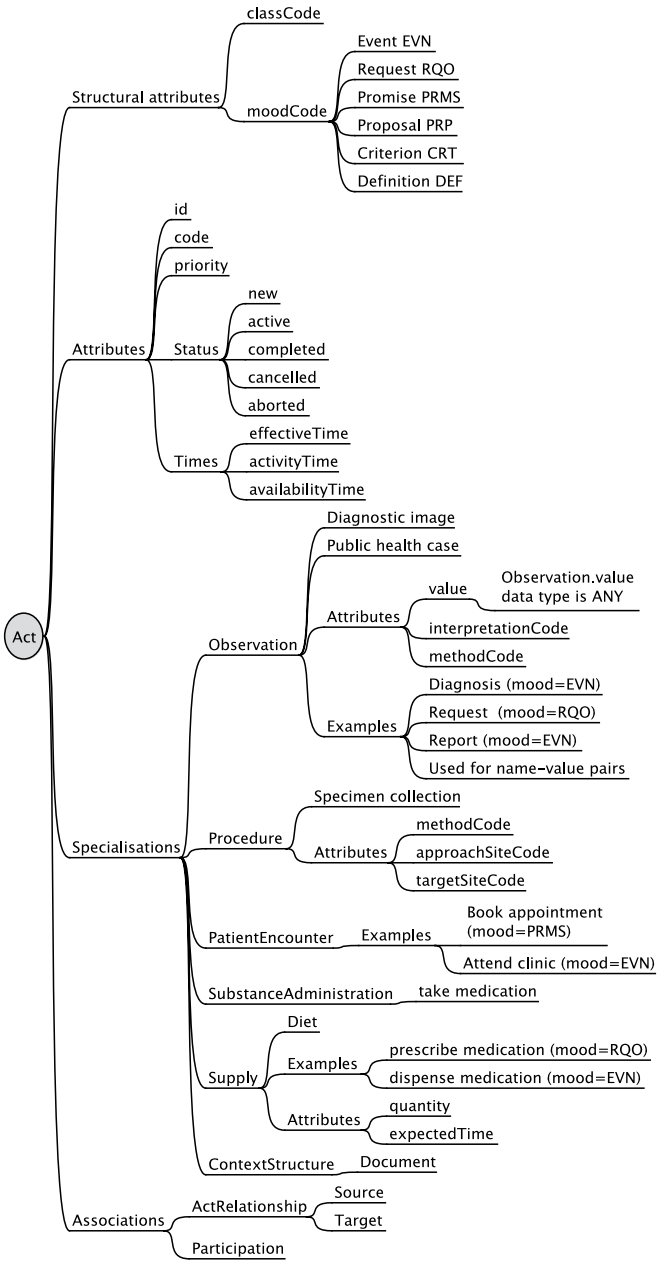


Fig. 7.4 HL7 V3 RIM – Act

7.5 Entity

Entity is the second main backbone class in the RIM. Entity is any living or nonliving thing, such as a person, animal, organization, or thing, which has or will have existence. It can also represent a group of things. An Entity can also be a group of things or a category or kind of thing.

Entities cover the whole universe of

- Living things, such as people, animals, plants, and microorganisms
- Nonliving things such as places, manufactured items, and chemical substances
- Abstract things such as organizations

Entity has a *classCode*, which states whty type of thing it represents. It also has a second structural attribute, *determinerCode*, Which is used to help distinguish between an individual instance of an Entity, such as a person, a collection of instances, such as a herd or the generic class of that entity, such as a particular type of micro-organism.

Entities may either play a Role directly, or may provide the "scope" for a Role. For example, Dr Smith plays the Role of Doctor, but this Role may be scoped by the organization she works for, such as St Mary's Hospital. Similarly, the scope link may also be used to note the manufacturer of a medicine. An Entity may perform any number of Roles, but every instance of Role is only played by a single Entity.

7.5.1 Entity Specializations

Entity has four main specializations, *LivingSubject* (incuding Person), Material, Place, and Organization (Fig. 7.5).

Person is a specialization of Living Subject, which is a specialization of Entity. Person has the attributes inherited from Entity and Living Subject as well as its own.

For example, *name* is an attribute of Entity, while sex (*administrativeGenderCode*), date of birth (*birthTime*), and date of death (*deceasedTime*) is each an attribute of LivingSubject.

LivingSubject has a second specialization, *NonPersonLivingSubject*, which is mainly used for veterinary subjects (animals, birds, fishes, etc.), but also includes bacteria, plants, fungi, etc.

Some attributes of an Entity, are also found in the Role class. These include *id, code, name, addr* (address), *telecom, statusCode, and quantity*. The primary rule for determining whether to use an Entity attribute or a Role attribute is whether or not an attribute value is permanent. If it is permanent, then use Entity, if it is not permanent, and in particular, if it is related to how a thing is used or what a person does, then it is a Role attribute.

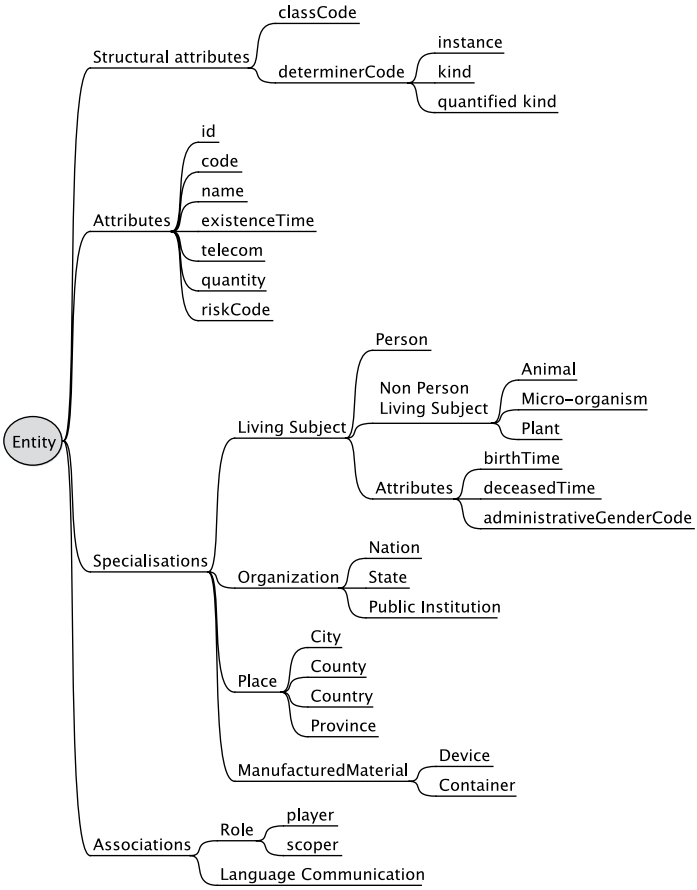


Fig. 7.5 HL7 RIM – Entity

7.6 Role

Role is the third main backbone class. A Role is defined as a competency of an Entity playing the Role. Roles for people are usually positions, jobs, or "hats," which they are qualified to do. Roles for inanimate objects such as places and machines are what they are normally used for.

There is also a wide variety of Roles, which can be played by:

- People, such as patient, practitioner, or employee
- Places, such as hospital, home, clinic, or place of birth
- Organizations, such as care provider, employer, or supplier

- Things, such as drug, instrument, or computer system
- Responsible entities, such as parent, employer, or manufacturer

7.6.1 Role Specializations

The most important Role specialization is Patient. However, it is important to remember that specializations are only shown explicitly in the RIM when they add additional attributes to the general class (Fig. 7.6).

Patient is defined as a Role of a person (player) as a recipient of healthcare services from a an Organization (scoper).

7.7 Association Classes

This simple backbone structure of Act, Role, and Entity is sufficiently flexible to cover almost anything you may want to say. However, we also need explicit connectors between each of these classes.

The *contextControlCode* specifies how the source contributes to the context of a target Act, and whether it may be propagated to descendent Acts whose association allows such propagation.

7.7.1 ActRelationship

ActRelationship is a relationship between two Acts, used to link Acts together, from a source Act to a target Act. There are various types of link, including composition, documentation, fulfillment, etc. Every ActRelationship has a source and a target to which it points. An Act can have any number of ActRelationships, which may be organized as a hierarchy.

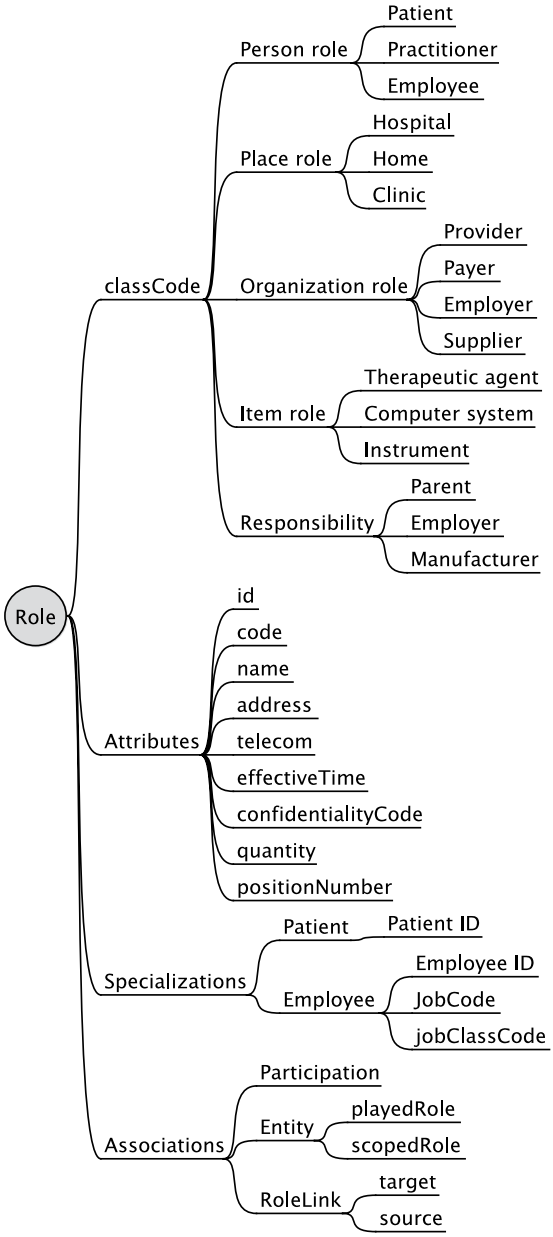
ActRelationship *typeCode* describes the type of association between Acts:

- Composition *comprises* entries
- Discharge summary *documents* a hospital visit
- Test report *fulfills* a test request
- Discharge summary *refers* to a referral
- Final report *replaces* a preliminary report

7.7.2 Participation

Participation defines the involvement of a Role in an Act. It shows how an Entity, in a particular Role, functions during the scope of an Act. Participants take part in Acts as either actors or targets in the Act. Actors do things, while targets are

Fig. 7.6 HL7 RIM – Role



essentially passive. Participation is specific to a single Act. When the Act stops, the Participation ceases.

A particular Entity in a particular Role can participate in an Act in many ways. Thus, a person in the role of surgeon may participate in an act as primary surgeon or as assistant surgeon.

Participation Type Code describes the type of association between an Act and each participating Role:

- Performer of act (surgeons, observers, practitioners)
- Subject of act, such as the patient
- Location of act
- Author, informant, addressee, or information recipient

7.7.3 Role Link

Role Link is a relationship between two Roles. It provides a simple way of linking Roles together such as between jobs in an organization chart, family members, or between members of a medical team.

7.8 HL7 Version 3 Data Types

In Version 3, Data Types have a similar role as in Version 2, providing fine detail.

7.8.1 Basic Data Types

Basic Data Types include:

BL Boolean – true or false
BIN Binary
ST Character String – unformatted text string
ED Encapsulated Data – text data that is primarily intended to be read by a human. ED can include format information
INT Integer number – any positive or negative integer
REAL Real number
PQ Physical Quantity – a measure quantity with units
MO Money – a currency amount

7.8.2 Codes and Identifiers

7.8.2.1 Instance Identifier (II)

The II (Instance Identifier) data type has two main flavors: UUIDs and OID-based identifier.

Universally Unique Identifiers (UUID) are software-generated identifiers, created on the fly to identify information artefacts uniquely. UUIDs are 16-byte (128 bit) numbers. The number of theoretically possible is large (more than ten followed by 37 zeroes).

The standard way of displaying a UUID is as 32 hexadecimal digits, displayed in five groups separated by hyphens in the form 8-4-4-4-12, such as:

550e8400-e29b-41d4-a716-446655440000

UUIDs are usually used when the identifier in question is generated by a software application without human assistance.

The second type of identifier is that held in some type of register. Here we use an identifier for the register itself and each item which is registered is allocated an identifier that is unique within that register. The convention is HL7 is to use an OID (object identifier) to identify the register itself.

An OID is a node in a hierarchical tree structure, with the left-most number representing the root and the right-most number representing a leaf. Each branch under the root corresponds to an assigning authority. Each of these assigning authorities may, in turn, designate its own set of assigning authorities that work under its auspices, and so on down the line. Eventually, one of these authorities assigns a unique (to it as an assigning authority) number that corresponds to a leaf node on the tree. The leaf may represent an assigning authority (in which case the root OID identifies the authority), or an instance of an object. An assigning authority owns a namespace, consisting of its sub-tree.

While most owners of an OID will "design" their namespace sub-tree in some meaningful way, there is no generally applicable way to infer any meaning on the parts of an OID.

HL7 has its own OID 2.16.840.1.113883 (iso.country.us.organization.hl7) and maintains an OID registry with around 3,000 nodes.

The combination of an OID and a value is intended to be globally unique.

One way to obtain an OID in the UK is to use the company registration number. All companies registered in England and Wales may append their company registration number to the 1.2.826.0 root to obtain an OID that is unique to the company without further formality or charge. The hierarchy is:

- Top of OID tree
- 1 – ISO assigned OIDs
- 1.2 – ISO member body
- 1.2.826 – Great Britain (GB/UK)
- 1.2.826.0 – UK National registration

For example, 1.2.826.0.1.4224538 means Abies Ltd.

7.8.2.2 Code Data Types

HL7 Version 3 has four code Data Types (CS, CV, CE and CD) listed in increasing order of complexity:

CS is just a simple code value, optionally accompanied by a display name. CS does not include any coding system identifier, which means that the meaning is determined by the context in which it occurs. CS is only used only for codes such as structural attributes, which are defined by HL7 itself and are always required.

CV includes a code value, an optional *displayName*, a *coding system*, identifier, enabling externally defined coding schemes to be used, and original text used. CV can be qualified as being CNE (coded no exceptions) or CWE (coded with exceptions).

CE is an extension of CV, which allows a term to be coded in more than one way. It includes the original code (such as a local code used in the sending system) to be sent along with a translation into a code required by the receiving system, which may have courser granularity.

CD is the most complex code Data Type, providing the functionality of CE as well as qualifiers to enabling complex postcoordinated expressions to be exchanged. For example the term "compression fracture of neck of femur" can be represented as a postcoordinated SNOMED CT expression using compositional grammar as follows:

71620000|fracture of femur|: 116676008|associated morphology|=21947006|compression fracture|,363698007|finding site|=29627003|structure of neck of femur|

This expression can be represented in HL7 using the CD Data Type as:

<code code="71620000" codeSystem="2.16.840.1.113883.6.96"
displayName="fracture of femur">
<qualifier>
<name code="363698007" displayName="finding site"/>
<value code="29627003" displayName="structure of neck of femur"/>
</qualifier>
<qualifier>
<name code="116676008" displayName="associated morphology"/>
<value code="21947006" displayName="compression fracture"/>
</qualifier>
</code>

7.8.3 Date/Time

Dates and times are represented by a hierarchy of Data Types:

TS Point in time
IVL <TS> Interval of time
PIVL Periodic interval of time
EIVL Event-related periodic interval of time
GTS General Timing Specification

The time format is similar to that used in Version 2, based on ISO 8601 (e.g., YYYY MMDDhhmmss).

Most, practical needs are met by TS, which may specify either a date or date/time. Interval of Time may be expressed with start and end dates/times, or as a duration or as an open range, with only the start or end date specified.

The more complex specifications were developed to meet the potential requirements of complex medication regimes.

7.8.4 Name and Address

The Data Types used in V3 for names and addresses are similar to those used in Version 2. Each name or address can be structured or unstructured and may include codes to specify its type and use and a date range for validity dates.

Types of name and address include:

TN Trivial name (unstructured)
ON Organization name
EN Entity name (any thing)
PN Person name in a sequence of name parts such as family given name(s), family and given, together with name use (legal, maiden name, former name, alias)
AD Postal Address as a sequence of address parts, such as house, street, city, postal code, country, and the address use (home, temporary, etc.)
TEL Telecommunication Address is specified as a Universal Resource Locator (URL), which covers telephone numbers (voice, fax, or mobile), e-mail addresses, and web pages.

7.8.5 Generic Collections

Multiple repeats can be specified in four ways:

SET is an unordered collection of values without any repeats
LIST is a sequence, containing values in a defined sequence (repeats not allowed)
BAG is an unordered collection of values, which are allowed to repeat
IVL is a set of consecutive values of an ordered base type such as time or physical quantity

7.9 Communication Infrastructure

The communication infrastructure is a collection of subject areas that define the technical infrastructure of HL7, including messaging and structured documents such as CDA. Structured documents are discussed in Chapter 9, Clinical Document Architecture; Messaging is discussed further in Chapter 10, HL7 Dynamic Model.

7.9.1 Infrastructure Root

All classes in the RIM are regarded as being specializations of the Infrastructure Root, which has four optional fields, which can be used in any RIM class or clone to support special communications needs. For example, the *templateId* field is used to specify the identity of the template being applied to any class. These special fields are:

- *nullFlavor* – When valued in an instance, this attribute signals that the class instance is null, and that the remainder of the information for this class and its properties will not be communicated. The value of this attribute specifies the flavor of null that is intended. Flavors of null include: no information (default), unknown, asked but unknown.
- *realmCode* – signals the imposition of realm-specific constraints. The value of this attribute identifies the realm in question.
- *TypeId* – When valued in an instance, this attribute signals the imposition of constraints defined in an HL7-specified message type. This might be a common type (also known as CMET in the messaging communication environment), or content included within a wrapper. The value of this attribute provides a unique identifier for the type in question.
- *templateID* – When valued in an instance, this attribute signals the imposition of a set of template-defined constraints. The value of this attribute provides a unique identifier for the templates in question (Fig. 7.7).

7.10 Use of the RIM

The Health Level Seven (HL7) Reference Information Model (RIM) is a static model of health and healthcare 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 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 HL7 V3 standard development process defines the rules governing the derivation of domain information models from the RIM and the refinement of those models into HL7 standard specifications. The rules require that all information structures in derived models be traceable back to the RIM and that their semantic and related business rules not conflict with those specified in the RIM.

The RIM is only one model of healthcare information needs. The abstract style of the RIM and the ability to extend the RIM through vocabulary specifications

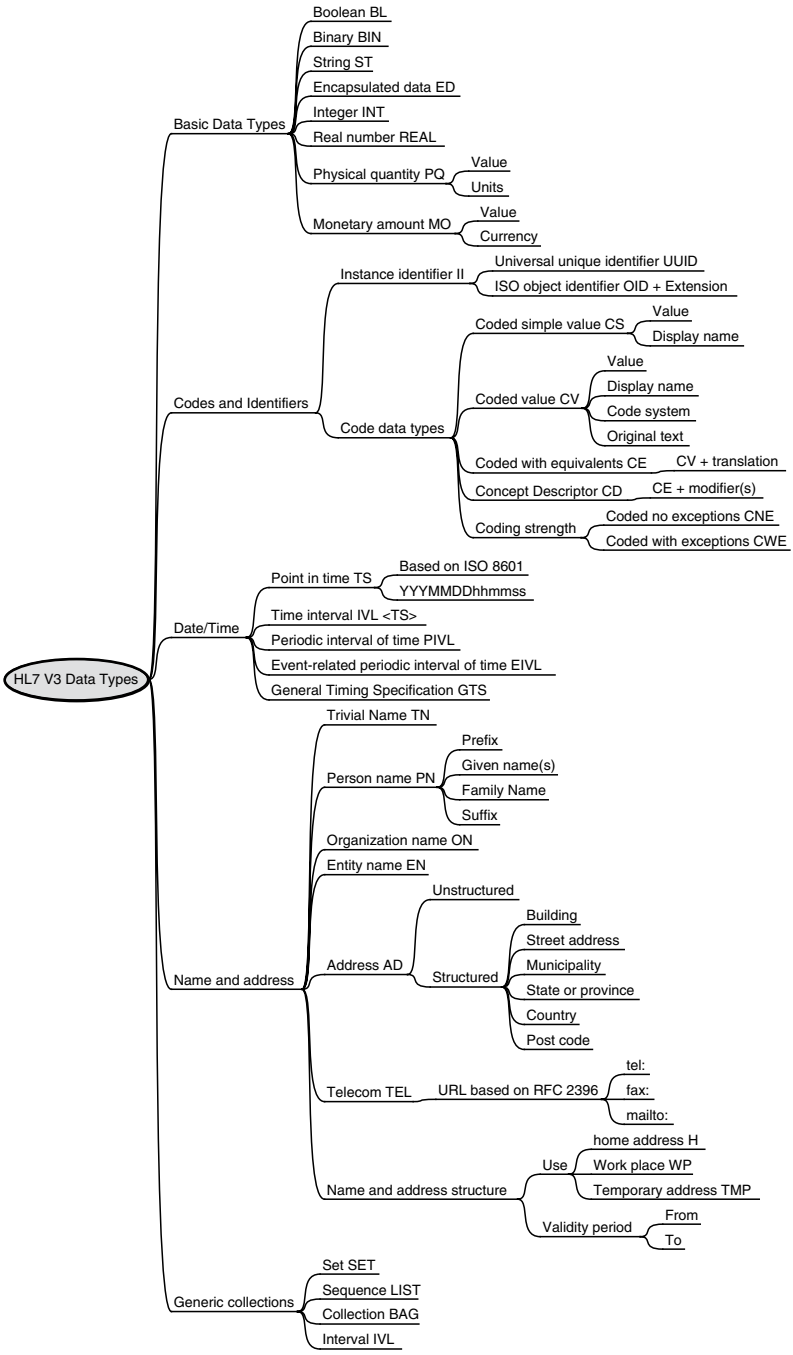


Fig. 7.7 HL7 Version 3 Data types

make the RIM applicable to any conceivable healthcare system information interchange scenario. In fact, it is conceptually applicable to any information domain involving entities playing roles and participating in acts.

The universal applicability of the RIM makes it particularly useful for an organization like HL7 that has to consider the needs of a large and diverse membership. The style of the RIM makes it extremely stable, which is another important characteristic for HL7.

The HL7 standards development process calls for the creation of domain-specific models derived from the RIM and the incremental refinement of those models into design models that are specific to the problem area. These problem-area-specific design models narrow the abstractness of the RIM and include constraints on attribute values and class relationships that are use-case-specific. External organizations considering using the HL7 RIM are advised to adopt a similar process of deriving design models as a transformation of the RIM.

In summary, the RIM has six backbone classes: Act, *ActRelationship*, Participation, Role, *RoleLink*, and Entity. The meaning of each class is determined by one or more structural attributes, such as classCode and moodCode. Each class has a predefined set of possible attributes and may have specializations, which provide additional attributes for specialized classes. Each attribute has a *Data Type*.