Chapter 14
Constrained Information Models

Abstract HL7 v3 works by constraining the RIM for specific use cases. Several types of constrained models are defined (DMIM, RMIM, HMD, MT and CMET). These all use common types of constraint, including omission, cloning, multiplicities, constrained data types and code binding. HL7 has developed a special graphical notation for specifying constrained information models. The clinical statement pattern is a common pattern used for clinical information in various profiles. Implementation Technology Specifications (ITS) describe how information is expressed as XML on the wire.

Keywords Constrained information models • DMIM • RMIM • HMD • Message type • CMET • Cloning • Multiplicities • Constrained data types • Code binding • Clinical statement pattern • Implementation Technology Specification (ITS) • XML • Documentation

Types of Model

A central idea of the HL7 V3 approach is to limit optionality by constraining or refining a general model for the specific use case being considered. This idea of constraining a general model to create an agreed subset and interpretation of the specification is widespread in the standards world. Constrained specifications are called profiles.

Many standards have multiple optional aspects and if different suppliers do not implement the same subset they will fail to interoperate. The use of profiles is a way to enforce a particular interpretation to ensure interoperability.

Constrained information models create a tree-like hierarchy of possible models. At the root of HL7 V3 lies the RIM. Everything else is a constraint on the RIM.

The following types of constrained model are recognised within HL7 V3, starting with the broadest, proceeding to the narrowest (Fig. 14.1).

DMIM	Domain Message Information Model
RMIM	Refined Message Information Model
HMD	Hierarchical Message Description

T. Benson, G. Grieve, *Principles of Health Interoperability*,
Health Information Technology Standards, DOI 10.1007/978-3-319-30370-3_14

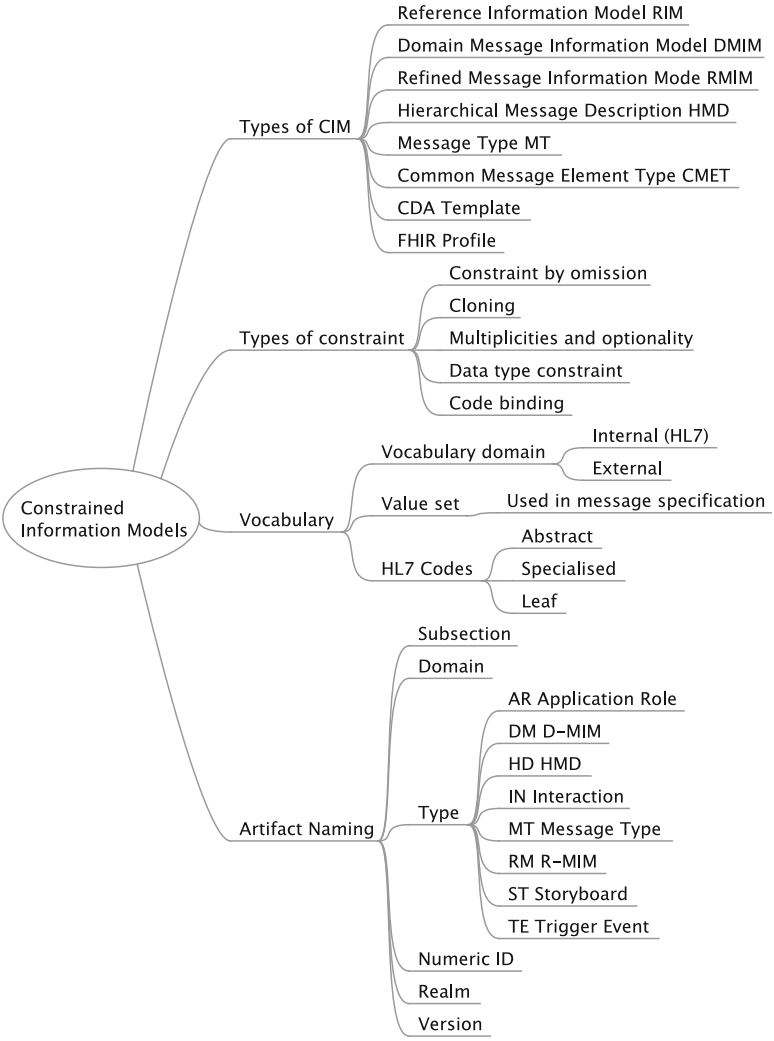

Fig. 14.1 Constrained information models

MT Message Type
CMET Common Message Element Type

DMIM (Domain Message Information Model) is a general model of a domain in HL7 notation from which a related family of message specifications can be derived. DMIMs have been defined for many subject areas. A DMIM may be created top-down from domain experience or bottom-up as a superset of messages in a domain. Once created a DMIM can be used a reference from which further messages may be defined. DMIMs do not have a hierarchical structure and cannot be serialized. A DMIM cannot be implemented as it is but needs to be further constrained as RMIMs.

The primary purpose of a DMIM is to provide a common point of reference to ensure compatibility between all artefacts, such as RMIMs in the same domain. Not all projects use DMIMs.

RMIM (Refined Message Information Model) is the most widely used constrained information model and may be thought of as a diagram of a message specification. RMIMs and DMIMs use the same notation. One important difference is that an RMIM has only one point of entry and can be expressed in a serialized format. Serialization is essential if a message is to be transmitted as a string of bits over a wire.

HMD (Hierarchical Message Description) is effectively an RMIM expressed in a tabular format. HMDs and RMIMs can contain the same information, but most people find that graphical RMIMs are easier to use and understand.

Message Type (MT) is a specific specification of a message, which can be used in a data interchange. Any one RMIM or HMD can be further constrained to create a set of closely related message types, which are then exchanged as a linear string of XML and validated using an XML schema.

Common Message Element Types (CMET) are reusable modules, which can be used in multiple messages, rather like a program sub-routine. Using CMETs can speed up the process of developing messages and increase consistency between different specifications.

Each CMET has two parts. The CMET reference is a special class, which can be added to an RMIM. When a CMET is referenced, or used in another diagram, it is shown with a special notation, a box with dashed edges, which contains the name of the CMET, its artefact id, classCode and level of attribution. This box is color-coded in a manner consistent with its root class. Each CMET has a unique artefact identifier (beginning with `COCT_`), which is the primary link between each CMET reference and its content.

The CMET content itself is defined as a small RMIM, which is stored in the CMET library, which is designed for common use by any HL7 committee. Relevant CMETs are included automatically in messages when they are constructed.

Each CMET has a single entry point, which is the point at which it is attached to any containing message, which references it. CMETs do not have exit points, which means they have to be at the terminal or leaf point in the hierarchical structure of a message.

CDA Templates and FHIR Profiles are also types of constrained model, described further in Chap. 15/CDA Templates and Chap. 21/Profiling Resources.

Types of Constraint

The RIM, DMIMs and RMIMs can be constrained by omission, cloning, multiplicity, optionality, data type constraint and code binding

The simplest form of constraint is by **omission**. Classes or attributes with classes are simply left out. All classes and all attributes (apart from structural attributes) in the RIM are optional, so you only use the ones you need.

The same RIM class can be re-used many times in different ways in DMIMs and RMIMs. This process is referred to as **cloning** and the classes selected for use in constrained models are referred to as clones. The idea is that you take a clone of a class from the RIM and constrain that clone in the constrained information model. Cloning limits the number of classes that need to be defined in the RIM, leading to a small stable RIM. The name of each cloned class in an RMIM is derived from its structural attributes. For example, a test request is represented in the HL7 V3 RIM as an observation request, so its structural attributes are `classCode=OBS` (observation) and `moodCode=RQO` (request or order).

The next form of constraint is to constrain **multiplicities** in terms of repeatability and optionality. Most associations and attributes in the RIM are optional and allow any number of repeats. These can be constrained by making such multiplicities non-repeatable mandatory (1..1) if you need one and only one; or non-repeatable optional (0..1) if you have any you can only have one.

In HL7 Version 3 specifications, the correct verb form for indicating a requirement is SHALL. The verb form for indicating a recommendation is SHOULD. The verb form for an option is MAY. Terminology used in standardization does not recognize the term 'must' and SHALL is always used to indicate a mandatory aspect on which there is no option. The negatives are SHALL NOT, SHOULD NOT, and MAY NOT.

The next type of constraint involves **constraining data types**. The HL7 V3 data types have been designed with a hierarchical structure. For example there are four code data types: CS (code simple), CV (coded value), CE (code with equivalents) and CD (concept descriptor) in increasing order of complexity. A more complex data type, such as CD can be constrained to a simple data type such as CV. Similarly the data type GTS (General Time Specification) can be constrained to IVL<TS>(Time Interval) or to TS (Timestamp). Data types can be further constrained to create data type flavors. For example the TS data type could be constrained to a date (TS.date) or year (TS.year).

The final type of constraint involves **code binding** – specifying what code value sets shall be used. The coding strength of a code may also be restricted to CNE (Coded No Exceptions) or may be specified as CWE (Coded With Exceptions). This may all sound quite complex but is simpler and more intuitive than it sounds. The simple rule is that you only specify what you need, leave out everything else or make it a simple as possible.

Vocabulary and Value Sets

The HL7 V3 standards talk about *vocabulary domains* and *value sets* and it is important to understand the difference between them.

A **value set** is the set of codes that may be used to populate a specific attribute in a message instance. The message designer usually specifies value sets. A value set may be a single code only, for example to specify a structural attribute, a subset of an HL7 defined code, or all or part of an externally defined coding system.

A **vocabulary domain** is the set of codes available to the message designer for a specific attribute. For example, the vocabulary domain for `Act.moodCode` is the set of all `moodCode` values defined and maintained by HL7.

Message users and implementers are concerned with value sets, while message designers need to think about vocabulary domains and select appropriate value sets from these.

The concept of vocabulary domains is most applicable to HL7's own internally defined vocabulary tables, which are quite extensive. These must be used for structural attributes and are widely used within data types. Each HL7-defined concept normally has a mnemonic code which is the code value used, a print name which explains its meaning, a concept ID used for internal reference, a level and type. Mnemonic codes have to be unique within a particular coding scheme. These tables have a hierarchical structure, with each concept being allocated a level, so a level 2 concept is the child of the preceding level 1 concept and so on. The code type may be:

- Abstract (A) which does not have a code but does contain child concepts
- Specialised (S) which has a code and contains child concepts
- Leaf (L) which has a code but no child concepts.

Artefact Names

HL7 V3 artefacts are identified using a common naming scheme, which is at first sight a bit complex. The format is

```
SSDD_AAnnnnnnRRVV
```

The first four characters identify the subsections and domains.

COCT	Common Message Elements
COMT	Common Message Content
FIAB	Accounting & Billing
FICR	Claims & Reimbursement
MCAI	Message Act Infrastructure
MCCI	Message Control Infrastructure

MFMI	Master File Management Infrastructure
POLB	Laboratory
PORX	Pharmacy
PRPA	Patient Administration
PRPM	Personnel Management
PRSC	Scheduling
QUQI	Query Infrastructure
RCMR	Medical Records

This is followed by an underscore character "_" and then the artefact type, identified with a two-character acronym.

AR	Application Role
DM	D-MIM (Domain Message Information Model)
DO	Domain
EX	Example
HD	HMD (Hierarchical Message Descriptor)
IN	Interaction
MT	Message Type
NC	Narrative Content
RM	R-MIM (Refined Message Implementation Model)
ST	Storyboard
ST	Storyboard Narrative
TE	Trigger Event

The artefact type is followed by a six digit identifier allocated by the committer responsible. The final characters are a 2-character Realm Code, where the identifying which international affiliate of HL7 is responsible for this. The default is UV (Universal) followed by a version number in the range (00–99).

For example: PRPA_RM001234UV00 may be interpreted as Patient Administration RMIM, with identifier 001234, used universally, revision 00. It is worth taking the trouble to memorize the main acronyms.

A Simple Example

Figure 14.2 shows a simple RMIM for an investigation report.

Every RMIM has an entry point, which states its name, `Demo Report` in this case, identifier and any descriptive notes that the author has provided.

The entry point or focal class, pointed at by the arrow is `ObservationEvent`. This is the default name for any act with `classCode=OBS` (observation) and `moodCode=EVN` (event). This has three other attributes: a unique identifier `id` (such as a UUID), `code` that states the type of report and `effectiveTime`, which refers to the date/time of the observation.

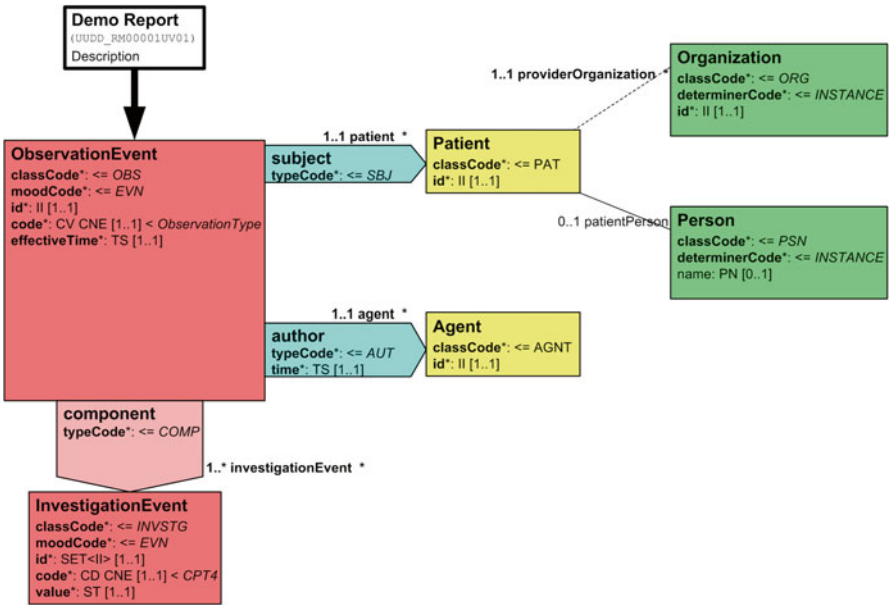

Fig. 14.2 Simple RMIM for an investigation report

This report contains one or more `InvestigationEvent` (`classCode=INVSTG`, `moodCode=EVN`), each of which has an `id` (such as a UUID or a line number), a `code` such as a CPT4 code to indicate what it is, and a `value`, which is a simple text string (ST).

The report has two participations: `subject` and `author`. The way to read the Participations is that the `ObservationEvent` has `subject Patient` and has `author Agent`.

The `subject` is a `Patient`, with an `id`, such as a hospital number. The `Patient` is scoped by an `Organization`, which has an agreed identifier (`id`). The combination of the `Organization/id` and the `Patient/id` should be globally unique.

The patient has an optional name, in the `Person` class (Entity). The playing association (`patientPerson`) between `Person` and `Patient` is indicated as `[0..1]`, and is not in bold font, indicating that this is not mandatory. Similarly the `name` attribute in `Person` is not in bold font and is annotated as `[0..1]`.

The `author` is an `Agent`, which could be a clinician, technician or a machine. The `Agent` has a unique identifier (`id`).

In this RMIM all elements are mandatory (and therefore required), which is why they are all written in bold font and suffixed with the "*" indicator.

R-MIM Notation

HL7 uses a special graphical notation for specifying RMIMs and DMIMs.

Each Act is represented as a red rectangle
Role as a yellow rectangle
Entity as a green rectangle
ActRelationship is usually shown as pink (salmon) arrow-shaped pentagon
Participation as a cyan (light blue) pentagon
RoleLink as a light yellow pentagon.

Each of the arrow-shaped pentagons has a source for the relationship and a target. The direction of the arrow indicates the meaning of the association, but this is not always the way that the diagram should be navigated. The direction of navigation, the way you read the diagram, is indicated by the location of the multiplicity shown just outside the class. This may sound confusing, but the important thing to remember is that the direction of the arrows is not always the way that the diagram should be read.

ActRelationship and RoleLink may be recursive, that is, each may point back to itself. This is indicated by a "pig's ear" box with a notched out corner which fits around one corner the Act or Role.

Each attribute uses exactly the same attribute name as is in the RIM; they cannot be changed. The attributes selected for use in RMIMs are formed by constraining or limiting the attributes as defined in the RIM. This allows checking and validation and is the key reason why the RIM may not be changed.

The attribute name in an RMIM diagram may be in **bold** print. This indicates that this attribute is mandatory, it must always be present, null values are not allowed. This is a responsibility of the sender Application Role.

The attribute name may have a star '*' next to it. This indicates that this attribute is required to be present in messages. If data is not available a 'null' value may be sent.

The multiplicity or cardinality of the attribute is denoted within square brackets [ ] to indicate how many times this attribute may be repeated. [0..1] indicates zero or one; [1..1] indicates exactly one. '*' indicates no upper limit, so [0..*] indicates zero to many.

The attribute's data type is specified after the attribute name, separated by a colon ':'. The specified data type must be either the same as or a valid constraint on the RIM data type for that attribute.

If the data type is a code, then the coding strength may be denoted by adding either CNE (coded no exceptions) or CWE (coded with exceptions) after the data type designator.

The value set or vocabulary domain to be used with each attribute is specified after either an '<=' or '=' symbol. '<=' indicates that the value may be taken from a vocabulary domain or the code specified or any of its descendants in a hierarchy. The equals sign indicates that the value should be as specified. The domain specification must be either a domain name defined in the vocabulary tables, or a single code value from the appropriate domain.

A string in quotes (e.g. "string") indicates a default value for this attribute.

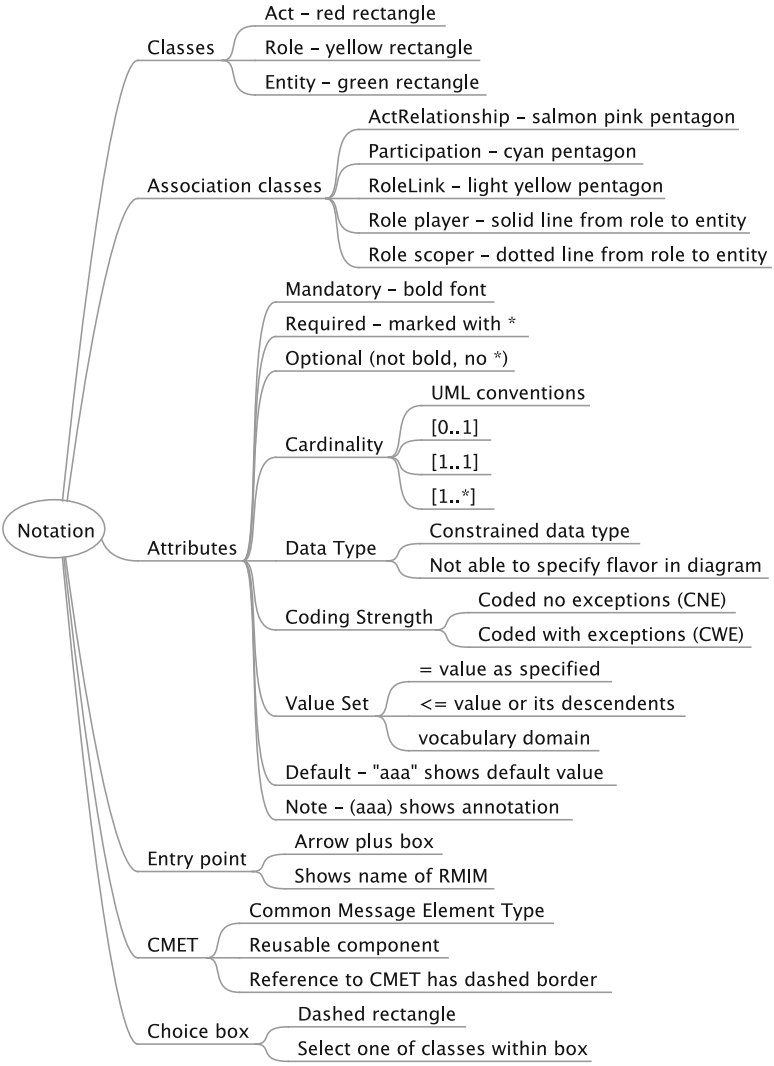

Fig. 14.3 HL7 V3 diagram notation

Finally, a brief description of attributes may be included, enclosed within parentheses, for example (description).

If the attribute information extends beyond one line, then second and subsequent lines are indented.

Choice Box is used in HL7 RMIMs to show alternative options. Each of the options is shown in a box with a dashed line border, from which a single choice is made. Associations may be made either to a specific class within the choice box, or to the outside border of the choice box, in which case that association applies irrespective of choice is selected (Fig. 14.3).

Tooling

RMIMs are built using a special tool-set developed by HL7. The original tools, based on Microsoft Access and Visio are in the process of being replaced by a new generation of tools, which use a slightly modified notation.

The basis of these tools is a set of inter-related XML schema, known as Model Interchange Format (MIF). MIF defines the primary artefacts that can be developed or exchanged as a result of HL7 V3 standards development and implementation.

Templates

HL7 templates are used to constrain and verify conformance to profiled HL7 Version 3 Refined Message Information Models (RMIMs). A template is an expression of a set of constraints on the RIM, which is used to apply additional constraints to a portion of an instance of data expressed in terms of some other Static Model. Templates are used to further define and refine these existing models within a narrower and more focused scope.

Each template is identified with a `templateId`, a globally unique identifier. Templates are used widely in CDA (see Chap. 9).

Clinical Statement Pattern

The HL7 Version 3 Clinical Statement is as a common pattern, which is used for the development of all types of clinical messages. For example, it is used in CDA Release 2 Level 3, for the exchange of complete electronic patient records between GPs, and for highly structured messages such as prescriptions and test reports.

HL7 defines a Clinical Statement as:

> An expression of a discrete item of clinical (or clinically related) information that is recorded because of its relevance to the care of a patient. Clinical information is fractal in nature and therefore the extent and detail conveyed in a single statement may vary.

Any clinical statement may have a number of participants, including subject, author, location, performer, participant and informer.

At the center of the clinical statement pattern is a choice box (ActChoice). A clinical statement to have any of the following specializations:

Observation covers a very broad range of statements relating to history, examination, tests, diagnosis and prognosis. Depending on the value of the moodCode, an observation can be an actual observation (mood = Event), a requested observation (mood = Request) or a goal set for a future observation (mood = Goal). Observation Events are usually reported using code-value pairs, where the code represents what is being observed and value represents the result. Observations may have child

observations. For example a blood count may consist of sub-observations. The Observation class may also be linked to a specimen and to normal range values.

Procedure may refer to a specimen(s) or images and is used for all invasive procedures including surgical procedures and imaging. Procedures can have associated observations. The moodCode is used to distinguish between those that have happened (mood = Event) and those that are planned or proposed. Procedures in HL7 are clinical and exclude administrative events such as admissions, clinic appointments, which are encounters (below).

Encounter which covers most administrative procedures involving an interaction between a patient and a healthcare provider for the purpose of providing a healthcare service. Encounter includes admissions, discharges, transfers of care, appointment scheduling and waiting list management.

Substance Administration may refer to products such as medication mainly used for prescribing and administration of drugs. Depending on the value of the moodCode it can be used for requesting, recommending or administration of medicines. Both substance administration and supply (below) are associated with a product, material or substance.

Supply is mainly used for dispensing drugs or other medical supplies. This can support precise identification of the actual product supplied, such as manufacturer, batch or serial number, using the HL7 Common Product Model.

Organizer is a specialization of the act class designed to support grouping information into clusters or batteries. For example, the components of a full blood count is typically ordered and reported together.

Act is a generic class, which is used if none of the above apply; it is rarely used.

Several types of associations between clinical statements are provided such as containment, cause and effect, problem linkage.

Relationships Between Entries

The Clinical Statement pattern allows for a rich set of relationships between entries, to reflect the structure of clinical information and links between different items. The main relationships are direct, with a source and target, containment and association.

Examples of the types of ActRelationships frequently found in clinical statements include:

CAUS shows that the source caused the target, such as substance administration (e.g. penicillin) caused an observation (e.g. a rash), or observation (e.g. diabetes mellitus is the cause of kidney disease).

COMP is used to show that the target is a component of the source (e.g. hemoglobin measurement is a component of a full blood count).

GEVL (evaluates (goal)) links an observation (intent or actual) to a goal to indicate that the observation evaluates the goal (for instance, a source observation of *walking distance* evaluates a target goal of *adequate walking distance*).

MFST (is manifestation of) is used to say that the source is a manifestation of the target (for instance, source *hives* is a manifestation of target *penicillin allergy*).

RSON (has reason) shows the reason or rationale for a service (for instance, source *treadmill test* has reason *chest pain*).

SAS (starts after start) shows that the source Act starts after the start of the target Act (for instance, source *diaphoresis* starts after the start of target *chest pain*).

SPRT (has support) shows that the target provides supporting evidence of the source (for instance, source *possible lung tumor* has support target *mass seen on chest X-ray*).

HL7 Development Framework

HL7 Development Framework (HDF) describes the methodology for developing HL7 V3 standards [1]. The HDF is written for HL7 members who are developing standards within HL7 committees. However much of what it says is of universal relevance. The HDF adopts a project-oriented approach, based on a product life cycle with the following stages.

The **Project Initiation Process** (PIP) includes initiation, planning, and approval sub-stages, including the development of a detailed Project Scope Statement (PSS) and plan. The project plan identifies the business case and objectives, participants including sponsor committee, project leader, contributors and early implementers and a time schedule.

Domain Analysis Process (DAP) includes analysis and requirements documentation, including the development of a **Domain Analysis Model** (DAM), which includes:

- Business context including documentation using storyboards and identification of relevant actors and interactions

- Use case analysis documenting use cases and actors
- Process model, documented using activity diagrams
- Information model, documented using classes and attributes
- Business rules including trigger events
- Glossary

Specification Design Process (SDP) is the core of the process. It involves mapping the requirements as set out in the Domain Analysis Model to the HL7 RIM, data types and vocabulary to specify the message structures, value sets and dynamic processes.

Profiles

A profile is a set of information used to document system requirements or capabilities from an information exchange perspective and is expressed in terms of constraints, extensions, or other alterations to a referenced standard or another profile. Profiles of HL7 Version 3 are derived from a Version 3 specification, as balloted either by HL7 or by one of its affiliates.

The categories and use of profiles include annotation, constraint, localization implementable and conformance profiles.

Annotation Profiles document the standard exactly but with more information to further explain the base document to educate prospective users and/or implementers.

Constraint Profiles may contain unchanged and constrained elements, reducing the optionality and cardinality of the base specification (i.e., the HL7 V3 standard) in order to make the specification more exact.

Localization Profiles meet the same objectives as a constraint profile, with the addition of some additional elements (extensions). HL7 Version 3 allows localization of some parts of the standard but not others. In particular, HL7 does not allow anyone, apart from HL7 itself through a formal process, to change or modify the RIM or any of the Data Types. Localization can make full use of the constraint mechanisms and make certain changes to RMIMs, Data Types, Message Types, CMETs and Vocabularies.

Implementable Profiles are the most constrained constraint profiles and eliminate all optionality in the base specification (the HL7 V3 standard) in order to make the specification exact and approach plug-and-play interoperability. Optionality is

eliminated when the conformance indicator for every attribute and association is either `Required` or `Not Permitted` and every vocabulary domain is bound to a value set.

A **conformance profile** indicates the set of interactions that a computer system (or application role) supports. It implies a commitment to fulfill all of the responsibilities of the interactions specified and to implement faithfully the artefacts that constitute the interactions and any further constraints or extensions. Conformance statements set out a computer system's conformance claim to a set of interactions.

Implementation Technology Specification (ITS)

The XML implementation technology specification describes how individual instances of message types shall be rendered in XML for serial transmission over the network and the structure of schemas used to validate each instance. Note that the HL7 generated XML schemas are not able to test all of the constraints defined in a HL7 message definition.

The generation of schemas and message representation is done automatically. Those not directly involved in that process do not need to understand the technical details. The key points are as follows:

- One XML element is defined to correspond to each attribute or association in the RMIM, with the exception of structural attributes, which are expressed using XML attributes.
- Each data type has a defined XML representation. The ‘restriction base’ feature in an XML schema is used extensively to define how data types are implemented.
- Schema files for CMETs are supplied separately and then used by each message schema as required.
- V3 data types and data type refinement use the W3C schema restriction element. Additional standard schema sections support RIM classes and the HL7-defined vocabulary definitions. These schema sections can be selectively combined with a specific message schema through the include function in the XML schema standard.
- HL7 messages share the same XML namespace. Message version information is conveyed as attributes within the message rather than by changes to the namespace identifier (Fig. 14.4).

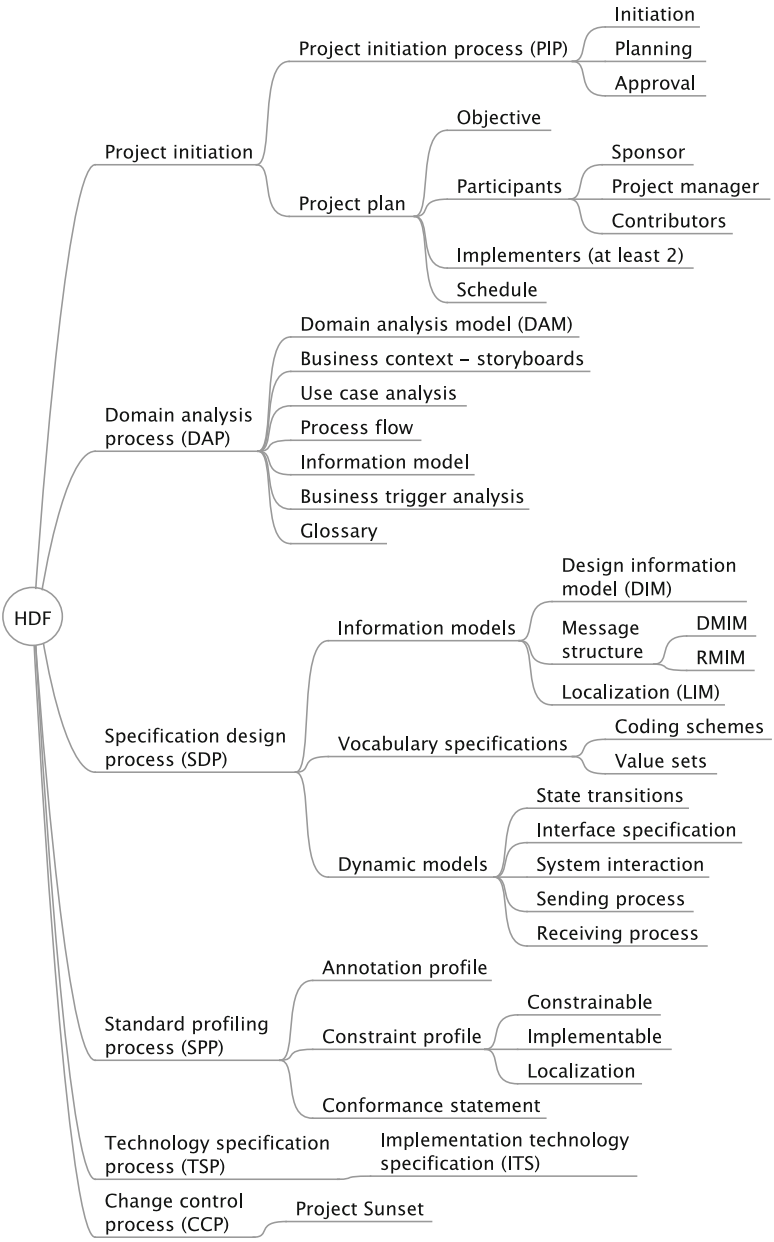

Fig. 14.4 HL7 development framework

Documentation

HL7 V3 documentation is voluminous. The full set of HL7 V3 standards is published annually and can be downloaded from the HL7 web site (www.hl7.org).

Foundation documents are the basis of the standard.

Note to Readers (7300 words)

Core Principles and Properties of HL7 Version 3 Models (17,200 words)

Refinement, Constraint and Localisation (14,600 words)
Reference Information Model (42,000 words)

Data Types – Abstract Specification (82,300 words)

XML Implementation Technology Specification – Data Types (43,600 words)
Vocabulary (6000 words)

HL7 Common Terminology Services (26,200 words)

Using SNOMED CT in HL7 Version 3: Implementation Guide (36,700 words)

HL7 Development Framework – HDF (21,300 words)
Specification and Use of Reusable Constraint Templates (17,500 words)
Glossary (24,900 words)

In addition HL7 has produced a wide range of domain specific standards covering:

Accounting and Billing
Blood, Tissue and Organ
Care Provision
Claims and Reimbursement
Clinical Decision Support
Clinical Document Architecture (CDA)
Clinical Genomics
Clinical Statement
Common Message Element Types (CMET)
Immunization
Laboratory
Master File/Registry
Medical Records
Medication
Message Control
Observations
Orders
Patient Administration
Personnel Management
Pharmacy

Public Health Reporting
Query
Regulated Products
Regulated Studies
Scheduling
Shared Messages
Specimen
Therapeutic Devices
Transmission

Reference

1. HL7 Development Framework. Version 1.3, 2009