Claude Nanjo

cnanjo@gmail.com

Abstract

The vMR and QIDAM take different approaches towards modeling a fact model for CDS and CQI. Yet, the goal is towards convergence into one logical model rather than divergence into two separate models. Some of the points listed below aim to stimulate discussion about how to achieve this convergence and build a better overall model.

Converging towards one Logical model

Some discussion points

# Logical Clinical Model

The vMR is a static model characterized by a set of concepts, relationships, and model constraints.

vMR = (K, R, T)

K represents the set of normative clinical concepts including core datatypes. They represent the basic semantic building blocks of the model.

R represents the set of normative relationships between concepts in the model. It consists essentially of the set of K x K x v where v represents a term in a vocabulary V that best represents the relationships. The relationship type, v, is typically associated with an IRI such as Code System + Code System Version + Code. Note this IRI may be fully specified or implied (in the case of class attributes). In the implied case, an IRI could be specified based on a scheme that would include the model’s namespace, version, and other contextual characteristics of the attribute.

T is the set of all constraints placed on the model. (More on this later)

# Concepts

K represents a concept in the model such as an observation, an ingredient, or a documentation concept. Note that many base datatypes apart from raw primitives are concepts containing attributes.

K= (M, D)

M represents metadata about the concept such as its name, definition and whether or not it is abstract.

D represents the set of attribute definitions for the concept.

Many concepts in the vMR have the notion of a concept code (e.g., ObservationFocus) that provide the semantics *for an instance of the concept.* The concept semantics/code itself – i.e., I am an observation and not a car – is implicit in the model.

If one were to specify a unique identifier for the concept, one could do so implicitly as: urn:hl7-org:vmr:r2:observation for instance. A concept with this identifier would represent the set of instances that are clinical observations.

# Attributes

Attributes specify properties of the concept and are often categorized based on their value types. Attributes that point to primitives or ‘core’ types and attributes that point to ‘full-fledged’ concepts. The latter essentially defines a relationship between two concepts whose relationship semantics is understood based on the definition of the attribute within the context of the concept. For instance, one may define an attribute called Problem.supportingObservation of type ObservationResult. This attribute could be interpreted as follows:

Attribute Source (subject): Problem (or subclasses thereof)

Attribute Target (object): ObservationResult (or subclasses thereof)

Relationship Semantics (predicate): “A supporting observation that aided in the diagnosis.”

Relationship Code (implicit): More generally as urn:hl7-org:vmr:r2:supportingObservation or more specifically as urn:hl7-org:vmr:r2:problem:supportingObservation

The presence of the attribute in the concept K is specified using an attribute definition. If we let C represent the set of possible cardinality constraints, then an attribute definition D can be expressed as:

D = K x C

That is, an attribute has a type k and may occur c times in the enclosing concept.

# Constraints

The logical model for CDS is intended to be a semantically accurate model. However, it is intended to be more general than a detailed clinical resource model in order to support computable operations on clinical knowledge that can be leveraged across related concepts. To that effect, the logical model specifies a set of core and normative semantic concepts and relationships that are minimally constrained. Through both composition and inheritance, the model then proceeds from more general to more specific concepts in order to support rules that can operate at both levels of expressivity. It is important to note, however, that the model only aims for a certain level of details as it strives to achieve balance between generalizability and specificity.

Furthermore, the specification of concepts and relationships does not suffice to ensure that a logical clinical model is clinically actionable. In order to be actionable, the model must also specify a set of constraints that supplement limitations in UML expressivity. The most important such constraints consists of terminology constraints imposed on coded fields.

The model provides two mechanisms to reach levels of clinical specificity beyond what is supported by the core set of concepts and relationships:

1. Constraints on the model called Templates (HL7 V3)
2. Extensions to the model (FHIR Kernel approach)

While the first mechanism above may be normative if the template is part of the set of templates specified by the normative model, the second mechanism is non-normative and provided for point-to-point use cases. Extensions, however, may be incorporated into the normative model over time.

It is proposed here that templates may only refine constraints already imposed on concepts and relationships present in the model such as tightening an attribute cardinality constraint, further restricting a more general type or constraining a coded type.

Changes that require increased concept expressivity (e.g., new attributes added to a concept), new concepts altogether, and new relationships should be handled using a formal model extension mechanism still to be defined.

# When to use templates

Templates may be used when a general *concept* *graph* defined in the logical model must be constrained to represent a family of specific clinical concept graphs. Doing so in a UML model without the use of templates would result in many very similar classes that differ primarily in the cardinality of their attributes, the value assigned to their coded fields, or in empty subclasses named appropriately without additional attributes and with notes about constraints. Such general concept graphs ensure model stability but should be used with care and only in cases where the semantics are fairly close (see diagram below):



The following illustrate a template that constrains a coded field and the subtype for the details field:



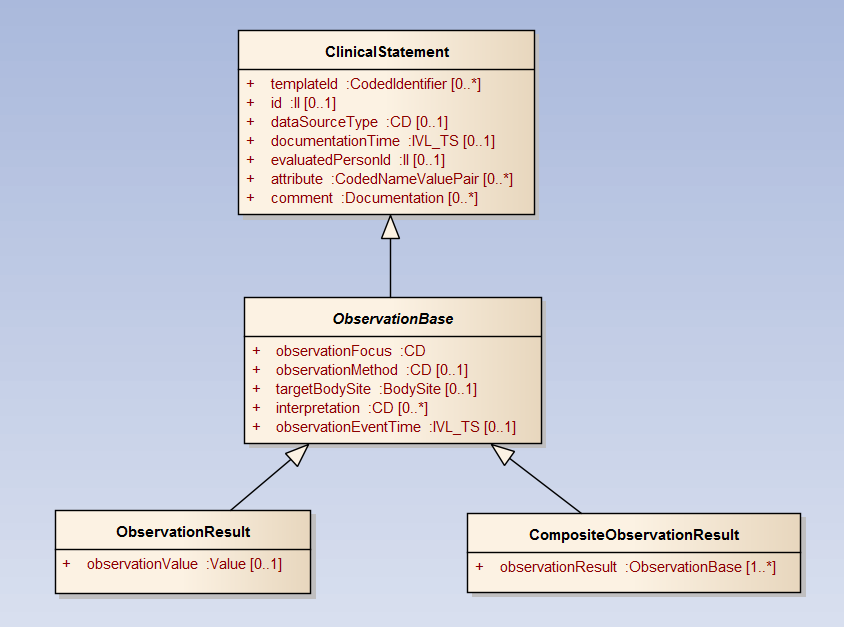
Note that in the above case, one of the constraints could be eliminated if one took a slightly different approach directly in the model:



This brings up two important observations:

1. In some cases, it is more desirable and intuitive to address a constraint directly within the model *provided the semantics are acceptable and/or equivalent.*
2. Coded attribute constraints are a limitation of UML and must therefore be addressed through constraints external to the model. (Though Galen argues that this may not entirely be true – OCL?)

[Add an example to illustrate the compositional generalizability of the LM. The model provides ‘building blocks’ which can be used to assemble a number of clinical concepts rather than providing explicit representations of these clinical concepts – e.g., panels made up of composite and simple observations. Need a word about the fact that the vMR is, in fact, a hybrid model which distinguishes it between CEM or OpenEHR on the one hand and FHIM on the other end of the spectrum]



# Extending the Model

The normative logical model must be extensible. That is, while the LM may strive to represent the 20% of concepts that occur of 80% of the time in a CDS use case, it must support ways to address the remaining 80% of concepts when needed. Towards this end, the LM must provide a way for users of the model to define the set of C’, R’ and T’ consisting of additional concepts, relationships, and constraints to meet a particular CDS or CQI use case.

Model extensions are typically presented as being of three types:

1. A new concept is required such as Care Provider
2. A new attribute is required such as ‘specialty’
3. A new relationship is required such as ‘supporting observation’

Extension type 2 and 3 are both attribute extensions. In one case, the attribute type is a common clinical data type such as a coded field, an extended type (e.g., frequency) or a primitive. In the other case, the attribute type is a clinical concept such as a clinical statement or clinical entity. In the vMR, type 3 attribute extensions can be performed using a relatedClinicalStatement or a relatedEntity relationship. The semantics of the attribute is specified by the role type and the target of the relationship represents the attribute type. Type 2 attribute extension is performed in the vMR using a coded NVP where the coded name represents the attribute semantics and the value represents its type.

Type 2 and type 3 extensions currently must define codes to specify attribute semantics. At this time, this represents a drawback as many terminologies are currently inadequate and in flux. Hence, the following approach is proposed (Aziz, Ken, Claude to discuss together):

1. Avoid current attribute and relationship extension mechanism for common types of relationships. Rather, explicitly define attributes in the respective concepts to capture such relationships. (E.g., adding an attribute in ClinicalStatement called supportingInformation of type ClinicalStatement).
2. Provide a formal way to add attributes and concepts to the model that are compatible with the medium used to represent the model. As the vMR is currently a UML model, the approach would be to:
   1. Specialize an existing class when adding a new concept within the UML model.
   2. Add an attribute to a concept in the model where it lacking.

If such an approach is taken, a set of constraints should be observed:

1. One cannot extend the LM by constraining it further in the UML model. (E.g., making an optional attribute mandatory or constraining it out of the model altogether). This should be done instead via constraints on the LM.
2. One cannot add new concept hierarchies. All proposed concepts should derive from existing ones. This is to support proper inferencing.

The vMR may wish to consider the following extensibility approach:

1. All extensions are done in UML.
2. Extensions must solely be additive. That is, they must consist solely of the addition of new attributes and subclasses. The core normative logical model cannot be altered. (e.g., removing concepts, adding new hierarchies, impacting model granularity through splitting or merging concepts).
3. Extensions may not alter the semantics nor the intent of the normative logical model.
4. Extensions are to be published using the OMG XMI serialization format and must start from the normative XMI serialization.
5. Extensions may be submitted and reviewed periodically for inclusion into the normative LM.
6. Extensions are not normative and may only be used for point-to-point communications. Systems that must understand extensions to the model may not be interoperable systems.

All changes that alter the original LM must be proposed and balloted and are not considered ‘extensions’ to the model.

Opportunities:

1. If the technology to represent the model is UML and the technology to serialize the model is XMI, then we should explore OCL as a constraining language.
2. One may wish to explore and see if one could add an OCL expression to handle terminology constraints.

One consideration when discussion model extensibility is its impact on existing libraries and rule definitions. While a CDS system can build an object model based on the normative logical model, it cannot reasonable anticipate the wide number of extensions that may be required for specific use cases and clinical communities. The vMR supports two extension schemes designed to be highly general in nature in order to accommodate extensions without requiring the recompilation of the object model or the redefinition of existing rules. This generic extension mechanism – Coded name value pairs and related Clinical Statements/Entities – provide a general way to handle new datatype/object properties.

Another approach is described below. This is a compositional approach to extensibility whereby every concept defines an abstract extension point which may be rendered concrete via specialization.



## Attribute Extension

Attribute Semantic Identifier

* New attribute: <http://my.org.com/clinical_concept/property/date_reviewed>
* Reusing existing one: ??

*Should attribute extension support specialization and reuse? E.g., I wish to add a comment attribute here.*

Target Type – DateTime

Cardinality – ??

## Concept Extension

Parent Concept Semantic Identifier - urn:hl7-org:vmr:r2:ObservationResult

Concept Semantic Identifier – http://my.org.com/clinical\_concept/blood\_pressure\_measurement

Attribute Definition Set – Systolic/Diastolic