An OCL-compliant GELLO engine

Jing MEI, Haifeng LIU, Guotong XIE, Shengping LIU, Baoyao ZHOU
GELLO, you know…

- **GELLO: an expression language for clinical decision support**
  - It has been approved as an HL7/ANSI normative standard since 2005

- **GELLO & OCL**
  - The GELLO language is based on the Object Constraint Language (OCL) that applies to an object-oriented data model

- **GELLO & vMR**
  - The underlying data model for GELLO was called a “virtual medical record” (vMR)
  - The HL7 CDS Work Group embarked on the development of an HL7 vMR standard based on a multi-national, multi-institutional analysis of CDS data needs
GELLO sample

– in OCL style with underlying data model of vMR

package vMR

class Patient

def: DM: CD = self.factory.CD('73211009', 'SNOMED CT', 'diabetes mellitus')

def: obs : Sequence(ProblemObservation) =
  self.isAssociatedWith.problemObservation

obs -> exists(problemCode.imply(DM))
GELLO sample

– in OCL style with underlying data model of vMR

```ocl
package vMR
context Patient

def: DM: CD = self.factory.CD('73211009', 'SNOMED CT', 'diabetes mellitus')

def: obs : Sequence(ProblemObservation) = self.isAssociatedWith.problemObservation

obs -> exists(problemCode.imply(DM))
```

- **package**: to specify the underlying data model being used for the following GELLO expressions
- **context**: to specify the model entity for which the following GELLO expressions are defined
- **factory**: a mechanism for creating instances of classes
GELLO implementation, you may not know…

- **Opt 1**: translate GELLO expressions into another language by means of a compiler producing executable codes
  - It requires ad-hoc translations due to a variety of target languages

- **Opt 2**: build a generic GELLO engine to evaluate native GELLO expressions (preferable)
  - **Opt 2.1**: regard GELLO as a totally new language syntactically, and implement a standalone GELLO BNF parser
    - The GELLO authoring tool developed by Medical Objects is such a representative
  - **Opt 2.2**: not to reinvent the wheels, but to leverage a range of well-developed OCL tools, considering that GELLO is fully compliant with OCL
    - That’s the objective of our work
An OCL-compliant GELLO implementation

- Clinical data
- GELLO expression

Service layer:
- vMR transformer
- OCL engine

Configuration layer:
- schema registry
- ontology repository

Model layer:
- vMR model implementation
  - function implementation
  - ontology reasoner
- vMR model
Three layers

- **Model layer: provide the vMR model implementation**
  - There is a core module of function implementation for manipulating HL7 data types where an ontology reasoner is employed to compute the implication relationship of concept descriptors

- **Configuration layer**
  - Ontology repository: load relevant terminologies such as SNOMED CT for observation codes
  - Schema registry: accommodate clinical data in non-vMR form such as CDA XML schema

- **Service layer**
  - Input: clinical data and GELLO expressions
    - OCL engine: parse the input GELLO expressions
    - vMR transformer: transform the input clinical data into the form of vMR
  - Output: evaluation results
Model layer

- **vMR model implementation**
  - Referring to the HL7 vMR domain analysis model
    - Version 2010-03-22
  - Leveraging EMF to generate the vMR model code packages automatically
    - EMF (Eclipse Modeling Framework) : a modeling framework and code generation facility for building tools and other applications based on a structured data model

- **Function implementation**
  - For example, PQ.greaterThan, the comparison between measurements with different units for physical quantity

- **Ontology reasoner**
  - CD.imply: the implication of concept descriptor
Configuration layer

- **Ontology repository**
  - For those terminologies appearing in GELLO expressions, the engine is able to reason on them with the access to the repository of the referenced ontology

- **Schema registry**
  - If the clinical data as input to the service layer is not in vMR form, the registry is configured to assist the engine in understanding the schema of the input clinical data and transforming them into vMR
Service layer

- **Input**
  - Clinical data
  - GELLO expressions

- **Output**
  - evaluation results

- **Components**
  - vMR transformer
    - If the input clinical data is in other format rather than vMR, it performs the transformation task, where the schema of the input clinical data has been registered via the configuration layer
  - OCL engine
    - To parse the input GELLO expressions, where all functional computations are passed to the underlying vMR model implementation
    - We used the Eclipse MDT OCL, but our methodology is vendor-independent and other OCL tools could be applicable as well
Experimental results

- **Implementation**
  - Our OCL-compliant GELLO engine has been implemented in a clinical guideline-based decision support system for chronic disease management.
  - We computerized a diabetic guideline (as defined by consulting an expert and referring the literature) as a clinical decision process which represents decision conditions with GELLO expressions and assists clinicians:
    - To raise health alerts to patients, such as a continuously high blood glucose alert.
    - To provide prescription advices, for oral glucose control and insulin therapy.
    - To make referral suggestions, for transferring patients from primary to secondary care, and vice versa.

- **Deployment**
  - The prototype system has been successfully deployed to Peking University People’s Hospital (one of the largest health providers in China), for managing diabetes patients.
Discussion

**Advantages**
- An OCL-compliant GELLO engine can profit from the utility of well-developed OCL tools
- It takes minimal effort, special for updating of the HL7 vMR model
  - Our model-driven development facilitates the code re-generation, and the manual part (for function implementation) will not be overrode
- An ontology reasoner is plugged in, making the implication of concept descriptor logically sound and complete

**Future work**
- Providing full support for all HL7 data type functions
- Enriching our GELLO engine as a vMR query engine