Requirements Traceability Interpolation For HL7 Specification Generation

Introduction

Health Level Seven(HL7) is the predominant interoperability-related global healthcare standard in operation today. Introduced in 1987 by the HL7 International Inc. a non-profit body, the standard has evolved to its current version 3. Presently, approximately 500 corporate members representing 90 percent of healthcare-related information systems vendors use it, and it has also been adopted by the International Organization for Standardization(ISO).

HL7 however has its issues. The current version v3 which has been promoting *Semantic Interoperability*, which is two or more computer systems communicating information with homogenous understanding, has been found to be difficult to implement and maintain. Further, core germinal issues exist such as the inability to trace requirements from specification segments, to the original antecedent domain documentation. Finalized specifications *elucidated* with *Reference Information Model* (RIM)-oriented vocabulary, cannot be back-linked to corresponding inceptive progenitoral domain requirements. This underlying deficiency in the HL7 specifications generation process and its negative effects are multi-phase. The objective of this paper is to present a sound, reliable, secure solution to the requirements traceability issue. This solution would enable finalized, terminal specifications to be backlinked summarily and seamlessly to germinal domain requirements and vocabulary, affording uniform, solution-oriented communication and consensus amongst all stakeholders.

Materials and Methods

Typically, *Domain Analysis Models* (DAMs) are developed to capture domain requirements. The *Object Management Group's* (OMG's) *Unified Modelling Language* (UML) is used to typify *DAM* structures. This solution proposes the representation of all *DAM* artifacts in the newly-devised *Unified Data Atom(UDA⁺)* representation, either first-hand or as a single-step transliteration. It is proved that the *DAM(UML)* to *UDA⁺* transformation is trivial. The seven *DAM* structures as described in [1] are *Data Element*, *Classes and Attributes*, *State Machines*, *Storyboards*, *Activities*, *Interactions*, and *Use Cases*. This solution transforms all seven *DAM* structures mentioned above to the *UDA⁺* representation, ensuring uniformity in domain-captured representations laterally across all *DAM* structures, and longitudinally overarching all phases, as the *DAM trundles* to finalized terminal specifications.

Results

If UDA^{+} signifies the set of transliterated, target DataAtoms $\{u_1, u_2, u_3, u_4, \ldots\ldots\ldots\ldots, u_k\}$ as a result of the *Equivalence* relation T^{ω} acting on the source *UML* informational schema *U*, then

$T^{\omega} : U \longrightarrow UDA^{+}$ $\quad U \sqsubseteq U$ and $UDA^{+} \sqsubseteq UDA^{+}$ where U - Problem domain *UML* super schema and UDA^{+} - Problem related target UDA^{+} super schema

$UDA^{+} \in \bigcup(u_i \leftrightarrow u_j)$ where $\{i, j, = 1, 2, 3, , \ldots, k\}$

where UDA^{+} : set of target DataAtoms with implicit, complete interconnectivity.

$\bigcup$: union of bidirectionally inter-connected, target DataAtom pairings

The following definitions are made with regard to *UML-originated* target UDA^{+} DataAtom types.

62/74

Simple UML Element (*SUE*) – *Entity Demarcator*, *Property.Demarcator*, *Restriction Demarcator*.

Regular UML Element (*RUE*) –*LabelDataAtom*, *ValueDataAtom*, *TagDataAtom*, all other *DataAtom* types.

Hence the transformed target UDA^{+} set $S = \mathrm{U}\{(SUE) \cup (RUE)\}$

where $\mathrm{U}(SUE) \sqsubseteq S$, $\mathrm{U}(RUE) \sqsubseteq S$

S is *unordered* with respect to the target UDA^{+} and its intrinsic *Nesting*.

Thus $\hat{\mathrm{Q}}\, S \equiv UDA^{+}$ where $\tilde{\mathrm{Q}}$ signifies the function ordering the target S and preserving the original syntax, semantics , and nesting.

$$\Rightarrow \mathrm{U}(SUE) \sqsubseteq UDA^{+}, \mathrm{U}(RUE) \sqsubseteq UDA^{+}$$

The table below illustrates the modality of the transformation. Two *DAM* structures, ie., the *Data Element* and *Classes and Attributes* related UDA^{+} transformations are shown in the table below. Indeed this exercise is extrapolatable to all seven *DAM* structures.

Figure 1 : <u>DAM to *UDA*⁺ Transformationand Syntax Mapping</u>

DAM' Structure	Representation	Component	Target UDA^{+} *(Syntax Mapping- Satisfied by T^{ω})*
(1)Data Element	Named, Typed, Valued attribute	Named, Typed, Valued attribute.	DataAtom-Matching Type *(yes)*
(2)Classes and Attributes	Class Diagram	Class Name	Entity Demarcator *(yes)*
		Attribute	LabelDataAtom, *(yes)* ValueDataAtom *(yes)*
		Inter-Class Multiplicity/Cardinality Inter-Class Association Label	Property Demarcator (Value and Name) *(yes)*
		General Constraint - DataType	Intrinsic DataAtom DataType *(yes)*
		Vocabulary Constraint	Restriction Demarcator *(yes)*
		Generalization Relationship	Property Demarcator (Value field indicates direction of generalization) *(yes)*

		Aggregation Relationship	Property Demarcator (Value field indicates *containing* class} (*yes)*

Discussion

As has been shown above, all *UML* structures can be trivially be *transliterated* in UDA^+ representation. Further, all domain requirements can also be captured in UDA^+ in the first instance, as well.

This solution presents a tri-threaded *requirements-traceability-ready* specifications generation process with optimal efficacy and efficiency in mind. The three streams of execution for domain requirements capture are:

1. Storyboards } → *UML* Diagrams → UDA^+
   Other textual record modes, eg., questionnaires
2. Transcribe to *UML* Diagrams → UDA^+ representation
3. Transcribe directly to UDA^+ representation (FastTrack).

The proposed solution uses intermediately-placed, state(phase)-related annotation-posts in order to capture and record in-place every state mutation, eg., a *RIM*-related annotation. These annotation-posts act as incremental, transition-related *Scoreboards* to the specification generation process. In essence these scoreboards instantiate the principle coordinates for onward and backward navigation between the source domain and target specification artifacts.

Conclusion

The *DAM*-related *UML* is subjected to a veritable metamorphosis in its journey to functional specifications; *RIM* annotations, transliterations, and insertions. The proposed solution using a system of state(phase)-related *annotation-posts* to afford the synchronous capture of state mutations, avails the precise identification of the corresponding *quintessence* domain requirement and vocabulary. Indeed, these posts are *Scoreboards* to the specification generation process, and indeed assist in effecting a complete onward and backward transformation between the set of domain requirements and allied vocabulary, and the target terminal specifications. Of immense significance is the fact that *analysis and design interoperability* amongst all parties is also derived as a filip by this solution, affording uniform, solution-oriented stakeholder communication and consensus.

References

[1] Shakir, A., Walker, M., Walden, A., *"Clinical Domain Analysis Models"*, DAM Construction Manual, v0.8, February 2011.

[2] Schadow, G., Mead, C., N., Walker, D., M., *"The HL7 Reference Information Model Under Scrutiny"*.

[3] Dillert, J., *"HL7 RIM – An Introduction for Non-Technical Professionals"*, PhUSE Paper CD04, 2013.

[4]*"Quantifiers in Regular Expressions"*, .NET Framework 4.5, http://msdn. microsoft.com.

[5] *"HL7 Version 3 Standard : Security and Privacy Ontology, Release 1"*, Health Level Seven International, September 2013.

64/44