Comments on DOL Revised Submission (ad/2015-08-01)
Ed Seidewitz / 15 October 2015

4. Terms and Definitions

4.3 Structured OMS

1. Page 13, definition of "closed world assumption": This is not a definition, but a statement that the "closed world assumption" is the default. An actual definition of what this assumption is should be given here. (Also, "status in unknown" should be "status is unknown".)

4.4 Mappings Between OMS

1. Page 14, definition of "correspondence": The initial definition, before the note, should mention that the correspondence is given "with a confidence level".

4.6 Logic

1. Page 16, definition of "institution": It isn't clear to me what it means to provide a formal "interface" for notions like signature. mode, sentence and satisfaction.
2. Page 16, definition of "plain mapping": What are "infrastructure axioms" in this context?
3. Page 17, definition of "exact mapping" and "weakly exact mapping": "compatible with certain DOL structuring constructs" begs the question of which constructs these are. Can you make a forward reference to clarify this?

4.11 OMS Annotation and Documentation

1. Page 19, definition of "annotation", second note: In "According to note 4.11…", what is "note 4.11"?

6. Additional Information

6.2 How to Read This Specification

1. Annex K is not mentioned in this subclause.

7. Goals and Usage Scenarios

7.2 Use Case Onto-2: Ontology Integration by Means of a Foundational Ontology

1. Page 24, first paragraph of subclause, last sentence: "A sample orchestration of interactions…is depicted in Figure 8.1 below." I do not believe that this is what is shown in Figure 8.1 in Clause 8. Is there a missing figure that should have been in subclause 7.2?
2. Page 24, second paragraph of subclause, first sentence: "[Alignment1,2]" seems to be a misformatted reference citation.

7.4 Use Case Onto-4: Interoperability Between Closed-World Data and Open-World Metadata

1. Page 26, End of second paragraph of subclause: "[OBDA]" should be replaced with a proper numeric reference citation.

7.5 Use Case Onto-5: Verification of Rules Translating Dublin Core Into PROV

1. Page 26, First paragraph of subclause, end of first sentence: "(c.f. Use Case Onto-1)" should be replaced with a reference to subclause 7.1.

7.6 Use Case Spec-1: Modularity of Specifications

1. Page 27, DOL example: The comment "%% refinement from abstract sorting to insert sort" should be removed, since the refinement is not actually shown in this snippet.

7.7 Use Case Spec-2: Specification Refinements

1. Page 27, second paragraph of subclause, first sentence: "[V-model]" should be replaced with a proper numeric reference citation.
2. Page 29, last line of subclause: "`refinement R3''' = R1 then R2`" The refinement "R1" referenced here does not seem to be defined in the example.

7.8 Use Case Model-1: Consistency Among UML Diagrams of Different Types

1. Subclause 4.8 has some good definitions differentiating concrete from abstract syntax. However, this does not seem well reflected in the discussion of Use Case Model-1. UML *diagrams* are *concrete* syntax, but UML semantics are defined on the UML *abstract* syntax. Diagrams are mapped to various modeling constructs defined in the UML abstract syntax metamodel, and there are already many

interconnections and constraints within the context of that metamodel. It needs to be clear that what is important for this use case is using DOL to ensure *semantics* consistency between parts of a UML model, not to duplicate the *syntactic* consistency constraints already captured in the UML metamodel.

2. Note also that the XMI for a UML model (as used here) is a serialization of the abstract syntax representation of the model, not the concrete syntax of the diagrams (the diagrams can also be serialized separately using Diagram Interchange XMI, but that is presumably not what is being used here). Typically the underlying model for a set of diagrams such as shown in Figure 7.2 will all be serialized together in a single XMI file, so that would be probably be a better assumption to make than that there are individual XMI files for each diagram. But even if individual XMI files are used, there would still be cross-file references representing the links between elements shown on multiple diagrams, so this is really just a multi-file serialization of the single underlying UML model. Again, it needs to be clear in the discussion here that the issue isn't the syntactic consistency across XMI files, but the semantic consistency of the underlying UML model (as represented in its abstract syntax, regardless of serialization).
3. In Figure 7.2:
    a. Since the sequence diagram and protocol state machine show the interaction between the ATM and the Bank, it would be useful to show the "BankIn" and "BankOut" interfaces for this interaction, in addition to the UserIn and UserOut interfaces. Presumable one would find the "verify", "reenterPIN" and "verified" operations on the Bank interfaces.
    b. The precondition "trialsNum >=3" cannot be made on the UserIn::keepCard operation, because the "trialsNum" attribute is defined on the ATM class, not the UserIn interface. The "trialsNum >= 3" constraint is probably better seen as an invariant on the ATM class, since it is a constraint for this specific ATM class and different ATM implementations with the same interfaces could potentially allow different numbers of trials or allow it to be configurable or some such.

7.9 Use Case Model-2: Refinements Between UML Diagrams of Different Types

1. This section should be about refinements between different types of UML *models,* not different types of *diagrams.*
2. Unfortunately, the example refinement of the protocol state machine to the ATM state machine is not clear. This is because the protocol state machine is a specific of the interaction *between* the ATM and the Bank, while the ATM state machine only gives the behavior of the ATM, which is just one side of this interaction. While the ATM state machine

can be reduced to an abstract state machine with the two states "Idle" and "Verified" in the way shown, it is not clear how the protocol state machine can be semantically refined to the resulting reduced ATM state machine in a sensible way, because the *events* on the transition between the states in the protocol state machine only partially match the events in the transitions between the states in the reduced ATM state machine. In particular, if one hides "card" and "PIN", then there is no way to transition from Idle to Verifying in the reduced ATM state machine. This is because the ATM does not *react* to the "verify" event, but, rather, *sends* a "verify" message to the Bank and it is the Bank that reacts to this event. Thus, rather than saying that the ATM behavior state machine refines the protocol state machine. It would be better to say that it is the behavior of *both* the ATM and the Bank that together provide a refinement of the protocol state machine. This concept of the need for the *joint* behavior of both parties in an interaction protocol to conform to the behavior specified by the protocol is very important, and the ability to capture this is, I think, a strong example of the value of DOL. So I would suggest that it is worth re-working this example in some detail to make it clearer and more correct.

7.10 Use Case Model-3: Coherent Semantics for Multi-Language Models

1. Page 31, second bullet: Change "UML diagrams" to "UML models".

8. Design Overview

8.1 DOL in a Nutshell

1. Page 34, first bullet: Change "UML class diagrams" to just "UML". Use Cases Model-1 and Model-2 demonstrate the possibility for using DOL for more of UML than just "class diagrams".

8.2 Features of DOL

1. Page 34, first paragraph in subclause: "DOL is freely available for unrestricted use." I don't think it is necessary to state this explicitly here. This is generally true of OMG specifications (particularly if the "non-assert" IP option is adopted).

Comments on DOL Revised Submission (ad/2015-08-01)
Ed Seidewitz / 16 October 2015

9. DOL Syntax

General

1. The following general concerns with the metamodel have been noted previously:
    a. Primitive types like String and Double should not be modeled as metaclasses but, instead, as UML primitive types String and Real should be used.
    b. Primitive types and enumerations should not be targets of associations but should only be used as the types of attributes.
    c. It would be better if there were a clear composition hierarchy in the metamodel. The guideline is that composition should be used if the composite part should be deleted if its “owner” is deleted. Otherwise a non-composite reference should be used.
2. Rather than being divided into multiple XMI files, the DOL metamodel should be serialized in a single XMI file, but be structured into a reasonable set of packages (one level should be sufficient). The diagrams could then be aligned with those packages. XMI 2.5 (the latest version) should also be used instead of XMI 2.1.

9.1 MOF Metaclasses

1. As has been previously noted, there are issues with the meaning of the models in subclause 9.1 as currently given. A better approach would be to use SMOF multiple classification as a means for integrating existing metamodels with the DOL metamodel. It should also be clear that the integration of the specific languages in the appendices is not part of the normative DOL metamodel.

9.2 Documents

1. Page 39, first paragraph on page, last sentence: “Moreover, Annex K informatively introduces QueryRelatedDefinition.” This sentence indicates that “QueryRelatedDefinition” is non-normative, but it appears on the subsequent diagram, making it seem like it is part of the normative metamodel. Instead of appearing here, the specification of QueryRelatedDefinition as a subclass of Definition should be given in Annex K, and QueryRelatedDefinition should not appear in the serialization of the normative DOL syntax metamodel (it could be provided as a very small non-normative extension to the normative metamodel).

9.3 OMS Networks

1. Page 40, diagram: It seems to me that it would be better to have PathReference::source and target, and NetworkElement::elementRef, just be IRIs themselves, rather than being ElementRefs to IRIs. Reusing the ElementRef class as both a kind of ExcludedElement and as a basic reference seems confusing, because it presumably not acting as an ExcludedElement when used as, e.g., a NetworkElement::elementRef. I would suggest renaming ElementRef here to ExcludedElementRef and replacing uses of ElementRef in other contexts with just IRI.

9.4 OMS

9.4.1 Abstract Syntax

1. Page 42, diagram: QualifiedOMS::qualification is shown as having multiplicity 0..*. Shouldn't there always be at least one qualification (multiplicity 1..*)?
2. Page 43, second bullet under "Using ExtendingOMS, further OMS can be built": "a closure of an OMS with a Closure" Neither the diagram on this page or the concrete syntax later shows a Closure as being related to an ExtendingOMS.
3. Page 43, footnote: "DOL's module sublanguage should be given preference over the module sublanguage" What does "give preference" really mean here? Especially in a specification document, it is important that it is clear what statements like this mean (even if they only have the force of "should"), or they will just be ignored.

9.4.2 Concrete Syntax

1. Page 44, first sentence of subclause: "While in most cases the translation from concrete to abstract syntax is obvious (the structure is largely the same)…" I found several cases (identified in following comments), beyond the few bullet points given, in which the relationship of the concrete to abstract syntax seemed to me not to be so obvious. It would be much better to have a more explicit specification of how the abstract syntax is synthesized from the parsing of the concrete syntax. This will be even more important as the metamodel is revised to be a better MOF model, rather than being derived as just an abstraction of the concrete syntax.
2. Page 45, ConservitivityStrength, ExtConservitivityStrength, Conservative: These concrete syntax productions make it clear that there are certain conservativity strengths that can only be used in certain contexts. It would be best if these restrictions were also recorded as constraints in the abstract syntax metamodel.

3. Page 45, OMSRef: Presumably this maps to the abstract syntax class OMSReference, so it should be called OMSReference in the concrete syntax, not OMSRef.

9.5 OMS Mappings

9.4.1 Abstract Syntax

1. Page 46, third paragraph of subclause: Would it be possible to formalize the constraints in this paragraph in the abstract syntax metamodel? These constraints are important, because it is noted that the semantics in Clause 10 assume they hold.
2. Page 50, diagram:
   a. The metaclass GeneralizedTerm seems unnecessary. SingleCorrespondence could just have a generalizedTerm association directly to IRI. (I don't think GeneralizedTerm is used anywhere else.)
   b. I could not find any further definition of the AlignmentCardinalityBackword and AlignmentCardinalityForward classes. Based on the concrete syntax, these are seemingly both just supposed to be AlignmentCardinalities. Perhaps AlignmentCardinality should be an enumeration in the abstract syntax metamodel, with AlignmentCardinalityPair::alignmentCardinalityBackward and alignmentCardinalityForward being attribute with this enumeration as their type.

9.4.2 Concrete Syntax

1. The structure of the BNF for OMS mappings seems to be more divergent from the structure of the corresponding abstract syntax than in other areas.
2. Page 51, production for InterpretationDefinition:
   a. The first two clauses could be combined into a single clause (with "'=' LanguageInterpretation* [SymbolMap] 'end'" being optional).
   b. It would be clearer if the third clause were separated into its own "RefinementDefinition" production (particularly in light of the comment bleow on InterpretationKeyword).
3. Page 51, InterpretationKeyword: It is not clear how the use of this keyword in the concrete syntax maps to the abstract syntax. Only the first two clauses of the InterpretationDefinition production (as given) would seem mappable to InterpretationDefinition in the abstract syntax, and should thus always use the "interpretation" keyword, while only the third clause would be mappable to RefinementDefinition, and should thus always use the "refinement" keyword. And what is a "view"?

4. Page 51, RefMap: This production should be divided into separate productions for OMSRefinementMap (first two clauses) and NetworkRefinementMap (last clause), since the former applies only to SimpleOMSRefinements, while the later applies only to SimpleNetworkRefinements.
5. Page 52, Correspondence: How does “*” map? To DefaultCorrespondence?
6. Page 52, Relation: There should be an explicit specification of the mapping of the symbols given in the production for Relation to literals in the StandardRelationValues enumeration, and from an IRI to RelationReference.

9.6 Identifiers

9.6.2 Abbreviating IRIs using CURIEs

1. Page 53, last paragraph, second sentence: “Informatively, the CURIE grammar supported by DOL can be restated as follows:” The CURIE grammar given informatively in 9.6.2 is repeated again (identically) in 9.6.4, where it is presumably normative. It needs to be clarified whether this grammar is normative or informative. In any case, it should only be given in one place (probably 9.6.4).

9.6.4 Concrete Syntax

1. Page 55, CURIE: The production ends in “-“. Is there something missing here, or is this a typo? If the latter, what is the difference between a CURIE and a MaybeEmptyCURIE?

9.8 Integration of Serializations of Conforming Languages

1. Page 57, list item “Standard markup conformance”: This sentence is stated as a normative requirement (using “shall”), but the requirements is for “standard mechanisms *like*” a couple of examples given. This is weak for a normative requirement, since it actually leaves largely open to interpretation what a “standard mechanism” is. Can this requirement be more strongly worded, at least giving XML contexts, say, in which XPointer or IETF/RFC 5147 *must* be used?