Validation in the DDI–Views

Production Flow

# Overview

This document explores the different ways in which the DDI Views production process could incorporate different forms of validation. Below is a high-level diagram of the overall production flow. In each area, the potential validation tools for each type of deliverable are considered.



# Validation Points

## Business Modeling (1)

The business modeling groups (Data Capture, Data Description, Codebook, etc.) work with the Modeling Team (in charge of foundational sections of the model such as datatypes, patterns for collections and process, etc.) to produce the UML model in Drupal. This model can be expressed in XMI (the XML form of UML for exchange between tools, etc.) What is produced by this work is both focused constructs in the model library for supporting a specific business function, and a *functional view* which provides a focused look at what classes in the model are most important for that function.

Documentation also comes out of this work – for each class and property in the model at a detailed level, for the functional views being created, and - especially for the Modeling Team – for how the overall model works, both for end users and for internal use by modelers and others involved in developing DDI work products.

For this work, validation is mostly making sure that the modeling in specific portions of the model is correct, and will support the intended business functionality. There are two different types of validation here:

1. **Completeness and Consistency:** Is everything which needs to be in the model present, in an up-to-date, correct form? Are examples and definitions of sufficient quality and detail? Much of this work can be automated using reports run against the XMI form of the model or checks performed within UML modeling tools. Barriers to this today are those surrounding how to get our “flavor” of XMI supported by useful tools, and the absence of comprehensive reporting tools (XSLT or Schematron could be run against the XMI expression of the model to generate reports.)
2. **Functional Quality:** Does the model support the desired use cases in a way that is effective/efficient and understandable? Are the examples and documentation appropriate to the expected use of the functional views? This level of validation is primarily a manual/intellectual activity which does not lend itself to automated validation tooling.

A comprehensive list of guidelines for business modeling is articulated at [https://ddi-alliance.atlassian.net/wiki/spaces/DDI4/pages/37552132/Modeling+Guidelines+for+Business+Modelers](https://ddi-alliance.atlassian.net/wiki/spaces/DDI4/pages/37552132/Modeling%2BGuidelines%2Bfor%2BBusiness%2BModelers)

## Hand-Over to Integration Modelers (2)

It is incumbent on the business modelers to deliver what they consider to be a complete and high-quality model within their functional area, and to pass this to the Modeling Team for integration into the model library. It should be noted that in some cases this process will be internal within the Modeling Team, as this team is functioning both as a business modeling group (for foundational constructs, and as a support team for other business modeling teams) and as the group charged with performing the integration modeling.

At this stage of the production, the key focus is the submission and approval of the functional view and the (possibly updated) constructs it uses. Validation at the point of hand-over is mostly to determine that the submitted material meets the criteria established for the business modelers. As noted above, some of this can be automated (consistency and completeness) and these checks will rely on the same tooling described above (e.g. checks built into UML tools, XLST/Schematron reports).

## Integration Modelling (3)

This activity is conducted by the Modeling Team. The scope of the work here expands beyond a specific set of functionality as organized around a view into the model to include the entire model library and the existing set of functional views. Is the submission consistent as a part of the larger whole? Are there constructs in the submitted material which duplicate existing ones? Have the modeling guidelines been followed? Is everything consistent and complete?

The validation needed for this activity is very similar to that needed for business modelling and hand-over, but must include the expanded scope of the activity. Thus, validation provided by UML tools from the XMI format of the model will be very useful, as will reports generated using XSLT/Schematron (or other technologies, as appropriate).

One additional aspect of this work is the validation of the XMI expression of the model itself. This is a deliverable in of itself, and should be assessed by the Modeling Team from this perspective. Is there sufficient documentation for the constructs in the model, aimed at an external audience using the model in its UML form? Are target UML tools supported by the XMI version of the DDI model? Validation here will largely consist of testing the XMI-formatted model with various UML tools.

***[QUERY: Is this the right place for these checks? The needed skills and knowledge exist primarily within the modeling team as opposed to the production team, at least notionally.]***

## Hand-Over to Documentation Production (4)

This activity is the transfer of modeling artefacts (including XMI and any documentation required) to a sub-group of the Modeling Team, charged with designing and producing the human-readable documentation in HTML and PDF form for all DDI work products. Some of the inputs needed for this can be programmatically derived from the modeling artefacts created upstream. Other text and material must be generated in document form for inclusion in the documentation.

Much of the validation here cannot be automated, especially regarding the quality of the input documentation text. However, at the point of hand-over there are checks which can be made specifically to guarantee that at least the structural and documentary information carried in the upstream modeling artefacts is consistent and complete/present.

*The production framework should export from the model for availability in documentation, all available information about the model at a class level and relevant mappings to previous versions of the DDI standard and to support transformations to bindings e.g. RDF mapping. For functional views, sufficient information to describe the purpose and intended usage of the view should be provided.*

## Hand-Over to Bindings Production (5)

This is the transfer of modeling artefacts to a subgroup of the modeling team focused on the production of the RDF and XML bindings. There are several items involved in these bindings (multiple XSD schemas, multiple OWL vocabularies) but all are generated from the XMI expression of the model programmatically. While there are several techniques for automatic validation of the binding, the simplest is to verify that the XMI as transferred will produce output bindings, without fatal errors. Errors at this stage would likely involve both modelers and production team members in their resolution.

Because a full validation of the bindings is not a simple matter of submitting the XMI formatted model to the transformations which produce the bindings (see below), but also involves some non-automated processes, the validation at hand-over is restricted to making sure that bindings can be produced for further validation.

The mapping to RDF should be complete, i.e. should contain both a vocabulary and a criteria for mapping at the property level.

## Documentation Production (6)

The output from XMI to rst into Sphinx should build without errors.

## Bindings Production (7)

The Production Team is a subgroup of the Modeling Team, charged with the generation of all bindings. Having an XMI expression of the model which can be used as input into the automated binding production process is the minimum criteria for hand-over. There are still other types of validation which can be applied. Will the OWL vocabularies import into Protégé and other appropriate tools without errors? Do the XML schemas import into tools which we expect to consume them? (This will include not just XML editors, but also – and perhaps more important – tools which generate code or other artefacts from XML schemas such as JAX-B, etc.).

Another aspect of validation is the determination of whether the subset bindings describing specific functional views are in fact valid subsets of the library-level bindings. This will likely involve the generation of test cases consisting of XML instances/RDF graphs which can be tested against the XML schemas/OWL vocabularies. (Note that, if we create RDF validation using ShEx and/or SHACL, the validation tools at the library and view level will likely also be provided as part of the binding delivered to the end user.)

Round-tripping between two expressions of the model must also be tested. This is still an area which is being explored, but it is likely that this will also rely on having test cases expressing specific instances of the XML schemas and RDF graphs for the OWL.

***[QUERY: What will testing for round-tripping look like?]***

## Documentation Submission for Quality Assurance (8)

Once the production is complete, documentation is submitted to the Technical Committee for quality assurance. For hand-over, validation will focus on the completeness of the submitted documentation.

In many ways, this stage of Quality Assurance is repeating checks which may already have been performed upstream, and may be able to benefit from the use of the same validation tools. This work may also involve (re-)validation of upstream artefacts from the modeling work.

Alignment of use cases, test cases and higher level documentation which spans classes will need particular attention, whilst some of these issues might be picked up, subtle changes will be more difficult to keep instep with class level version changes.

Another area which will need attention is the differentiation between documentation of the model and of the bindings, the level of the difference will depend on how far the serialization to bindings varies from the model, especially in relation to abstract classes and patterns more generally.

## Bindings Submission for Quality Assurance (9)

Once production is complete, the bindings are submitted to the Technical Committee for quality assurance. The key validation here is to verify that all submitted bindings are valid examples according to their own syntactic rules. The tools used to assess this are a subset of the tools used during production, and will involve text/example instances (for XML) and RDF graphs (for OWL bindings, etc.) This work may also involve (re-)validation of upstream artefacts from the modeling work.

## Quality Assurance (10)

The Technical Committee is responsible for assuring the quality of the documentation and bindings for all DDI work products on an on-going basis (that is, both during and after the end of the development project for DDI Moving Forward). To the extent possible, this work should be automated through the use of report generators/scripts, although these will need to be developed and maintained by the DDI Alliance (presumably the Production/Documentation subgroups within the Modeling Team).

Some aspects of documentation Quality Assurance will require manual effort, and cannot be automated. Existing guidelines should be examined for those areas where they can be distilled into checklists, which might support automated validation.

## Packaging, Distribution, and Review (11)

All functions related to packaging, distribution, and review are handled by the Technical Committee. Validation here consists in establishing that packages for delivery to users are complete and correct, and conducting testing websites or repositories used in the delivery process. Any guidance provided at the time of reviews would also fall into this function – the Technical Committee being responsible for digesting the comment s resulting from reviews. This type of validation will be manual, and is probably not well suited to automation.

Many of the tools used to perform automated validation and reporting will also be useful to those in the technical committee, supplemented by specific checks which will span the entire breath of the DDI production for a particular delivery target.

A delivery checklist should be produced as an agreed deliverable against which the sign off will be made. This should include pass/fail for any validation tests that have occurred further down the production process.

# Validation Tools as External Deliverables

Some of the tools developed for validation throughout the production of DDI-Views will also be useful to developers and other users external to the production process. Examples of this will include validation tools for specific functional views of the library (in XSD, ShEx/SHACL, etc.) and may also include other validation artefacts (see *Examples and Use Cases* below).

The prioritization of which validation artefacts are created first may be influenced by their usefulness to both internal and external communities.

# Issues/Resource Demands

Many different groups are working now in areas which will potentially impact or be impacted by how well validation tools and approaches are brought into the production environment.

 If production is moved from Drupal and onto a COGS/Git-based platform, production of downstream artefacts (bindings, documentation, etc.) and execution of validation checks could be performed at a granular level and in an on-demand fashion, allowing upstream modelers and others to rapidly see the consequences of changes and new development.

The production of examples, test bed, and use cases (see below) will form the basis of improved validation in some areas of the process, and will in some instances (as with test cases) be directly implemented into validation checks.

The RDF binding discussion within the Production Team is looking specifically at ShEx and SHACL and how these tools might be useful both internally and externally, etc.

The integration of validation into the production flow offers a chance to greatly improve the efficiency of the work, both in terms of individual productivity and in terms of realizing a product at each stage of the process which will require less effort to bring to an acceptable level of quality. Given the resource situation within DDI, increases in efficiency are much to be desired.

# Iterations for Needed Upstream Changes

It should be realized that, at each stage of the production process, issues discovered may require that some upstream changes are made. A problem in the RDF or XML binding might be solved by making changes in the model, for example, rather than implementing changes to the transformation producing the binding, etc.

In such cases, upstream groups will need to work together with those encountering the issues to identify and implement a solution. Thus – although presented here as a simple linear flow – we should understand that a given production process may require several iterations through some portions of the production framework before a final deliverable product is available for dissemination or review.

# Examples, Use Cases, and the Test Environment

Work has begun already on the creation of test cases in the Moving Forward project. Use Cases have also been identified and documented for proving out the information model. Examples have also been highlighted as an important form of documentation which has not always received sufficient attention in the past.

All of this work bears directly on the type of validation which can be integrated into the production process. In some cases, the relationship is a direct one – the act of testing the round-trippability of bindings (RDF-to-XML and back) is itself an important form of validation. Other relationships are perhaps less obvious: a use case for a specific feature of the model might well provide rules which could be implemented in an automated fashion when testing bindings derived from that portion of the model.

The plan for integrating validation checks and tools within the DDI production process should be coordinated with work in these areas.

In the medium to long term, the development of tooling that generates test cases will be required for validation as the building of these is unsustainable in anything more than the short term. The target for this should be reassessed after the release of the June 2018 prototype.