XMI as a Work Product

Maximizing the Utility of the DDI Model

Arofan Gregory, Joachim Wackerow

# Overview

This document presents the business case for the publication of the DDI model in the form of XMI, so that the model itself can be leveraged in implementations using a variety of tools. XMI is a standard XML format which is designed to exchange UML models.

There is a significant case to be made that publishing an XMI version of the model would provide value to developers working with DDI. There are also significant challenges. This document reviews both of these, so that the DDI Alliance can make a better-informed decision about how to proceed.

DDI has solved many problems by moving to a model-driven approach. There are also, however, potential benefits to further leveraging the existence of the model to make DDI a more useful standard. These dynamics are also discussed here.

# Background

XMI - the XML Metadata Interchange standard – is a mature XML standard published by the same standards organization which developed UML itself, the Object Modeling Group (OMG). XMI has been implemented by many different UML tools, with the idea that modelers working with different tools can exchange their models.

XMI is a critical part of the current DDI production process: the Drupal collaborative development platform exports an XMI version of the DDI model, which then goes through a series of transformations to programmatically create the XML Schemas, RDF Vocabularies, and documentation with which the DDI will be implemented. The existence of an XMI expression of the model is a significant asset to the DDI Alliance in this capacity.

# The Model-Driven Approach

The DDI Alliance decided to switch to a model-driven approach for many reasons: it solved many resource and logistical problems presented by hand-crafting XML Schemas; it allowed for a re-design of the XML products to better suit developer needs; it allowed for the creation of an RDF version of the standard which was aligned with the XML; it increased the transparency of the development process to a high degree; it provided a platform for better documentation; and it allowed for the revision of the development process to solve several problems. While the model is still in development, it appears that many of these goals will be achieved.

With the model in hand, one major driver for increased efficiency is automation: replacing the hand-crafted nature of DDI in the past, it now becomes possible – even with the limited resources of the DDI Alliance – to increase the set of useful deliverables we offer our user community, so long as these are work products are produced automatically from the DDI model.

The development dynamic of increasing our set of work products is clear, however, based on their programmatic production from the model. An initial investment must be made to put the automation in place, after which the costs of maintaining and producing the new work products falls dramatically. The model itself becomes the focus of maintenance, along with some additional quality assurance resource demands. (It should be noted that with the rule-based approach now being implemented as part of the DDI Moving Forward production framework, the quality assurance itself can become much more efficient than it has in the past, again through automation.)

# Why XMI is a Valuable New Work Product

The benefits of publishing an XMI version of the DDI model to the user community are many: there are a large set of mature UML-based tools, some of which are free, and many of which are not only used for developing models, but also to exploit models once they exist. Also, the existence of an information model has, for some other standards, proven to be extremely popular among implementers, a good example being the Statistical Data and Metadata Exchange (SDMX) standard, ISO -13769.

The model-driven approach allows DDI to have two expressions - in RDFS/OWL and in XSD. This provides for a potential expansion of the DDI community. If the publication of an XMI version of the model is performed, and the user base of DDI increases further, then we also benefit from having a larger set of people responding to reviews and reporting bugs, which is a channel for improving the standard. Given that the XMI publication has the potential to expand the user base and the types of users of DDI, this feedback channel will become richer than it has been in the past.

Further, the XMI format is an excellent one for some developers as a documentary channel, making the model easier to understand through tools which are familiar to them, even if not necessarily popular among the existing DDI user community. It also makes DDI an easily usable model for system architects, analysts, and systems designers, a community which has not in the past had very good support concerning DDI work products.

### Common XMI Tools:

* [Eclipse Modelling Tools](https://eclipse.org/modeling/tools.php) like [MDT UML2](https://projects.eclipse.org/projects/modeling.mdt.uml2). Eclipse tools are popular. Therefore they play a similar role nowadays as years ago IBM Rational Rose.
* [Enterprise Architect](http://www.sparxsystems.com/products/ea/) (popular in the DDI and GSIM community)
* [IBM Rhapsody](http://www-03.ibm.com/software/products/en/ratirhapfami) (successor of Rational Rose)
* [Altova UModel](http://www.altova.com/umodel.html)
* [MagicDraw](http://www.nomagic.com/products/magicdraw.html)

Some of these tools are open-source, notably the Eclipse tools. Enterprise Architect is popular among the existing DDI community.

It should be mentioned that many of these modeling tools are able to perform many functions beside simply creating data models in UML. They are also good for doing many other UML tasks such as formally defining use cases, defining process interactions, designing software systems, and so on. They are also often able to perform implementation features such as generating database schemas or application programming structures (for Java, C#, etc.) automatically.

One of the benefits of publishing an XMI version of the DDI model is that we can leverage the many capabilities of these tools, rather than producing work deliverables directly (the generation of relational database schemas is a good example of something we can get from existing tools once we publish the XMI.) This is important because it has a multiplicative effect in terms of what the user community can do – publishing the XMI provides greater value than if the DDI Alliance were to directly publish the relational database schemas directly (a work product which has been discussed in the past).

There are several challenges related to the publication of XMI, however, because not all of the tools mentioned support the same versions or flavors of the standard. There are two issues here: one is the selection of the XMI standard to use. The best supported modern version is XMI 2.4.1. There exist some common profiles of XMI, and some work would be involved in determining which of these profiles would make the most sense to publish. It is possible we might wish to support more than one profile, and possibly in more than one version.

There is strong feeling that DDI should publish a flavor of XMI which is useable within the Eclipse tools, as all of them have a common flavor which may be supported also by some other tools. These tools are very useful, and are open-source. Support also for Enterprise Architect would be good, as this is a popular tool within our community. However, Enterprise Architect uses a fairly proprietary flavor of XMI, which could present issues.

# Notes on Publication of XMI as a Work Product

The starting point for the publication of XMI as a work product would naturally be the XMI which we are using for our own production. However, additional information and changes would need to be made for a publication version to support specific tools. Thus, we would recommend treating these separately: our production XMI would be the basis for a separate flavor (or, potentially, flavors) of XMI useful for tools.

An additional requirement is that, while our XMI for production does not contain layout information for diagrams, published flavors of XMI would ideally include this information. This extra information would need to be included in the transformations producing publishable XMI. Further exploration of tools support for different flavors and versions of XMI would be needed.

# Conclusions

There is a significant benefit to the publication of an XMI version of our model to our user community. There is also the potential for broadening our user base. The relative cost of pursuing this path is relatively low, but when compared to other possible additional work products (such as recommended relational database schemas or object-oriented application models in Java, C#, etc.) XMI would produce more value, as it would allow users to generate these implementation artefacts for themselves. Further, implementation of DDI would be made easier because the tools which consume our model also support other design and implementation functions within the organizations which use them.

Some up-front exploration and prioritization would need to be made to determine which flavor of XMI to support, but once the up-front investment has been made, the benefits would be large. The publication of the DDI model in XMI is a good way for us to best leverage the existence of the DDI model itself.