MDA Journal

February 2005

David S. Frankel
David Frankel Consulting

df@DavidFrankelConsulting.com

Introduction

If you mention XML and MDA in the same breath to most MDA aficionados, they will usually point to the OMG's XML Metadata Interchange (XMI®) standard as the embodiment of the application of MDA to XML.

This article explores some additional initiatives afoot in the computer industry that apply model-based principles to the use of XML, and examines the relationship of XMI to these initiatives. It also takes a look Microsoft's approach to the use of XML in model-driven systems.

The Importance of XML

XML is undoubtedly an important technology in today's data processing systems. It has become popular as a means of serializing data among disparate systems, particularly when data has to traverse firewalls in electronic commerce scenarios. XML has many practical uses and is here to stay.

MDA views XML as one of a number of information conveyance technologies. It could take many years for XML to displace EDI formats such as EDIFACT. We all discovered as the year 2000 approached that legacy systems hang around longer than we ever thought they would. Other serialization formats such as ISO 8583 and various internal formats used by financial transaction networks are not going away so fast, since converting trillions of annual dollars of commerce to new technologies cannot be undertaken suddenly, and the high bandwidth consumption that follows from XML's ASCII basis makes it prohibitive for use in super high volume transaction environments, at least for now. Furthermore, new information conveyance technologies could emerge in the future.

MDA also takes the position that an XML schema should be not be produced in a vacuum. Rather, it should be an outgrowth of the development of a formal, logical data model that is independent of XML, EDIFACT, ISO 8583, and so on.

XMI

Technically, XMI is a mapping of the OMG's Meta Object Facility (MOF™) to XML. It tells you how to take a model expressed in the MOF language and derive an XML schema or DTD from the model. The schema or DTD becomes a format for encoding instances of the model.

The MOF language is a subset of UML, and it is used as language for defining metamodels. In essence, therefore, MOF metamodels are UML class models. Pragmatically speaking, therefore, the XMI rules can be applied to derive an XML schema from a UML class model, as long as the UML class model doesn't use a few class modeling constructs that MOF doesn't support.

For example, if suppose we create a UML class model of the entities in a bank, with the classes Account, Customer, and so on. When we apply XMI to the model we produce an XML schema or DTD (the engineer can choose which one) that serves as a format for encoding instances of Account, Customer, and so on in XML documents.

Early versions of XMI produced rather verbose XML. Furthermore, the first major use of XMI was to apply it to the UML 1.x metamodel to produce an XML DTD for exchanging UML models. The UML 1.x metamodel was large and had a lot of unnecessary cross dependencies. Therefore, the XMI DTD resulting from the application of the XMI translation rules was large and had a lot of unnecessary cross dependencies, and the overall verboseness mentioned above exacerbated the messiness of the XMI DTD for UML.

Also, the popularity of UML led many people to the mistaken impression that XMI was nothing more than an XML DTD designed for exchanging UML models. They did not realize that XMI was a mapping that, in this particular case, had been applied to the UML metamodel.

Furthermore, the early versions of XMI offered a one-size-fits all mapping. The rules were inflexible. Consider, for example, the mapping to XML of an attribute in the class model. There are basically two ways to do it. One way is to map the containing class (i.e. the class that contains the attribute) to an XML element, and map the class attribute to an XML attribute of the XML element. The other way maps the class to an XML element in the same fashion, but maps the class attribute to another XML element that is nested within the XML element that corresponds to the class. There are pros and cons for each way, which I do not wish to explore here. The point is that forcing everyone to adopt one and only of these approaches in all cases proved too inflexible to be sold to industry.

As a result, XMI 2.0 is a parameterized mapping. There are a host of options, each of which has as default that an XML engineer can override. Table 1 and Table 2 are reproduced, with permission of the Object Management Group, from the XMI 2.0 specification;[1] they list all of the parameters and their defaults. The specification also contains detailed explanations of each of the parameters.

XMI 2.0 is an important part of the increasingly pervasive Eclipse ecosystem. Eclipse implements only the default values of the XMI mapping parameters. But the XMI specification itself has this flexibility. It is only a matter of time before the dynamic Eclipse community produces an implementation of the parameterization.

This is where XMI stands today. We shall return to the subject of XMI after looking at some of the other model-driven XML initiatives.

[1]Table 1 and Table 2 are copyrighted by the Object Management Group (OMG). (c) 2003

Tag Name	Value Type	Default value	Description
Naming tags			
xmiName	string	nil	Provides an alternate name from the MOF name for writing to XMI. Useful in cases where the MOF name has characters that conflict with XML. This value is used rather than the MOF name.
idName	string	xmi:id	The value is the name of the id attribute.
nsURI	string	nil	The namespace URI of the MOF package.
nsPrefix	string	nil	The namespace prefix of the MOF package; this is used in schemas. (Any legal XML prefix may be used in documents.)
XML Syntax tags			
serialize	boolean	true	If false, suppresses serialization of the MOF construct. Typically applied to derived features.
attribute	boolean	false	If true, serializes the MOF construct as an XML attribute.
element	boolean	false	If true, serializes the MOF construct as an XML element.
remoteOnly	boolean	false	If set on one end of a bidirectional relationship, only serializes that end if it is remote.
href	boolean	false	If true, use the href attribute rather than the idref attribute for links within a document.
Ordering			
superClassFirst	boolean	false	If true, serialize the super class content first.
ordered	boolean	false	If true, serialize object content in the order it is defined in a MOF metamodel.

Table 1: XMI Mapping Parameters, Part 1

ISO 20022 (UNIFI)

The ISO 20022 standard, named the *UNIversal Financial Industry message scheme* (UNIFI)[2], was created by TC68, ISO's technical committee for financial services. SWIFT, the organization that manages standards and a network used globally for cross border electronic communication among financial institutions, has been a major driver of this new standard.

ISO 20022 defines a UML-based methodology for modeling business transactions, message flows, and messages, and defines a set of rules for deriving XML schemas from the UML models.

SWIFT is also active, along with some of the world's major banks, in the *International Standards Team (IST) Harmonization Group*[3], which is using ISO 20022's UML modeling approach and UML-to-XML mapping rules to produce XML schemas for next generation electronic payment systems. The banks active in this group include ABN AMRO, Bank of America, Citibank, Deutsche

Tag Name	Value Type	Default value	Description
Content			
includeNils	boolean	false	If false, do not serialize nil values.
XML Schema Production			
enforceMaximumMultiplicity	boolean	false	If true, enforce maximum multiplicities; otherwise, they are "unbounded."
enforceMinimumMultiplicity	boolean	false	If true, enforce minimum multiplicities; otherwise, they are "0."
useSchemaExtensions	boolean	false	If true, use schema extensions to represent inheritance in the MOF metamodel.
schemaType	string	nil	The name of a datatype defined in the XML Schema Datatype specification.
contentType	string	empty	Defines the schema content type. Other valid values are: complex, any, mixed, complex, and simple.
processContents	string	strict	If the contentType is any, this tag is used to specify the value of the processContents attribute of the any element. Other valid values are: lax, skip.
form	string	nil	Specifies the value of the form attribute for attributes. Other valid values are qualified and unqualified.
defaultValue	string	nil	Specifies the default value for attributes.
fixedValue	string	nil	Specifies the fixed value for attributes.

Table 2: XMI Mapping Parameters, Part 2

Bank, HSBC, JPMorgan Chase, Nordea, Standard Chartered, and Wells Fargo. Standards organizations active in the finance industry that are collaborating in the IST Harmonization Group in addition to SWIFT include IFX, OAGi, and TWIST.

As I said earlier, legacy transaction formats will not disappear overnight, and XML is not appropriate for all kinds of transactions streams. Nevertheless, it is significant that this array of heavy hitters in the finance industry is using a model-driven approach to producing its XML schemas for the future.

ebXML and ISO 15000-5

UN/CEFACT and OASIS originally came together to define the ebXML standard, created a series of technical standards that together define a framework for secure, reliable web services. ebXML has not been highly visible in the United States ever since IBM and Microsoft shifted their attention to their version of web services. However it is still very much alive, particularly in the Pacific Rim economies of East Asia and Australia, and an increasing number of European Union countries are adopting it as well. Now UN/CEFACT and OASIS are teaming with ISO TC 154 to codify ebXML in the ISO 15000 standards

Of particular interest to us in this discussion is ISO 15000 Part 5, called *ebXML Core Components*. According to Mark Crawford, Senior Research Fellow for

the non-profit LMI Government Consulting company firm and Chair of UN/CEFACT's XML syntax group, this new effort uses the *UN/CEFACT Modelling Methodology* (UMM), which is a set of standardized UML profiles and techniques that apply the many features of UML in a consistent fashion. ISO 15000-5 leverages these UML profiles and ISO 11179[4] data naming and definition approaches to define standard data constructs that form the basis of data architectures and information exchanges. These 15000-5 constructs are defined as both syntax-neutral *core components* and syntax-specific *business information entities* (BIEs), which describe the content of e-commerce message payloads for various business domains.

The OASIS Universal Business Language Technical Committee and the UN/CEFACT XML syntax group have developed complimentary naming and design rules (NDRs) that transform the ISO 15000-5 BIE constructs into standardized XML. The NDRs were developed by some of the best and brightest in the XML world.

The net effect of this effort is a series of UML models, data architectures, information structures, and XML schemas that are harmonized from beginning to end, and are applicable not only within single organizations, but across organizations.

ISO 20022, ISO 15000-5, and XMI Profiles

ISO 20022, ISO 15000-5, and XMI all have defined UML-to-XML mapping rules.[5] Obviously it would be helpful if these rules converge to the greatest extent possible. The parameterized nature of the XMI mapping could hold the key to such convergence.

Accordingly, preliminary discussions have started to investigate whether choosing a specific set of values for the XMI mapping parameters would yield the ISO 20022 UML-to-XML mapping, and whether yet another set of values would yield the IS0 15000-5 mapping. We can call each set of values an *XMI profile*. The OMG made XMI 2.0 a parameterized mapping precisely so that industry could profile it to fit their requirements. If it is not possible to define XMI profiles that yield the two ISO mappings, then the OMG should looking into revising XMI so that it is flexible enough to achieve this objective.

This appears to me to be a possible path to convergence. We shouldn't expect every group to use the same exact mapping rules. But XMI might be able to serve as the master mapping template, with XMI profiles defining the specific mappings. If we can achieve that, then an engineer could readily configure an XMI mapping machine to be an ISO 20022 XML production machine or an ISO 15000-5 XML production machine.

Microsoft's Approach

Microsoft takes a totally different approach to XML usage. Its model-driven systems initiative, spearheaded by the Visual Tools Enterprise Tools Group that is populated by some fine talent that Microsoft recruited from the UML world,

[2] [ISO 20022]
[3] [IST HARMONIZATION]
[4] See [MDA JOURNAL DEC 2004] for more on ISO 11179
[5] As stated earlier, technically XMI is a MOF-to-XML mapping, but MOF is a subset of UML, so we can consider it a kind of UML-to-XML mapping for purposes of this discussion.

believes that XML serialization formats should not be derived by mappings but, rather, should be designed directly by engineers.[6] For example, when you create a metamodel with Microsoft's new metamodeling framework, the XML format for serializing models that conform to the metamodel is hand crafted; it is not produced by applying mapping rules to the metamodel.

There has been some debate about Microsoft's approach on the pages of MDA Journal before (see the January and February 2003 issues). Rather than argue the merits of the automatic mapping versus hand crafting approach, I think it important to step back and look at what our goal with model driven systems is.

We are trying to raise the abstraction level and automate as much of software production as is pragmatically possible. As a corollary, we should prefer not to have to hand code from scratch new serialization code—be it Java or C# code—for each new kind of model that we want to serialize. If we want to hand craft an XML format, that does not mean that we then have to hand craft a serializer for that format. How can we avoid that? Even with a hand crafted serialization format, there is a mapping from the metamodel (or model) to the serialization format. For example, if we define a metamodel for some specialized modeling language, and hand craft an XML serialization format for it, then we of necessity must define a mapping from the metamodel to the serialization format. Now, if this mapping can be encoded in a transformation modeling language that is more abstract than Java or C# code—such as the MOF Query, View Transformation (QVT) language that the OMG is working on—then we can construct model-driven serializers that either read the transformation models dynamically or use them to generate the Java or C# code for custom serializers.

Conclusion

The approach of generating XML schemas from more abstract models is taking hold in some important electronic commerce forums, particularly in the influential finance industry. The main streams of activity are:

- OMG's XMI 2.0, which is a UML (or MOF)-to-XML mapping
- ISO 20022's UML-XML mapping and the related IST Harmonization Group's application of 20022 in the finance industry
- ISO 15000-5's codification of ebXML Core Components and establishment of UML-XML mapping rules

XMI 2.0 is a parameterized mapping. I have defined an XMI profile as a set of values for the mapping parameters. Hopefully, the IS0 20022 and 15000-5 mappings can be expressed as XMI profiles. If not, the OMG should consider fixing XMI so that this is possible.

Microsoft has a markedly different approach, preferring to handcraft XML schemas. However, Microsoft is embracing model-driven approaches to software development. In many of its newer environments, a handcrafted XML schema will be logically derived from a model. By using models of the transformation between the model and the handcrafted XML schema, Microsoft could still follow its preferred approach while staying within an overall model-driven approach.

[6] [GREENFIELD ET ALL 2004]

References

[GREENFIELD ET AL 2004] Jack Greenfield and Keith Short, with Steve Cook and Stuart Kent, *Software Factories*, John Wiley and Sons, 2004

[ISO 20022] *ISO 20022 UNIversal Financial Industry message scheme*, http://www.iso20022.org

[IST HARMONIZATION] *World Wide Payment Harmonization Project: Backgrounder*, http://www.openapplications.org/wg/PaymentHarmonization/200311107-Gartner/Background.htm

[MDA JOURNAL DEC 2004] *The DRM, the Semantic Web, and MDA*: *Comments on the U.S. Federal Government's Data Reference Model*, MDA Journal, December 2004

[XMI 2.0] *XMI 2.0 Specification*, OMG Document formal/03-05-02, May 2003