Technical Brief

April 2011

SEARCH
The National Consortium for Justice Information and Statistics

Model-driven Development of NIEM Information Exchange Package Documentation

By Andrew Owen and Scott Came
SEARCH

Since its earliest days, the National Information Exchange Model (NIEM) Program has supported NIEM users by ensuring the availability of tools. These tools have included applications provisioned by the NIEM Program itself, along with a mix of tools offered by national associations and private companies.

Throughout its history, the NIEM Program has sought to improve the available tools, and has encouraged providers from the community at large to innovate in the tools space. More recently, NIEM Program staff and stakeholders have expressed a strong interest in a tools strategy that emphasizes openness and standards-conformance. In particular, the NIEM Technical Architecture Committee (NTAC) has published a tools strategy and architecture document[1] that calls for tools with open interfaces and standards-based formats for artifacts.

Recently the NIEM Program and NTAC have begun an effort to develop a Unified Modeling Language (UML) Profile for NIEM Information Exchange Package Documentation (IEPD). A UML Profile for IEPDs will bring many benefits to NIEM tools developers and users, including the openness and standards conformance necessary to support interoperability between IEPD-modeling tools. The NTAC effort to develop a UML Profile for IEPDs builds upon an earlier NTAC effort to establish a UML profile for NIEM reference namespaces.

This paper focuses on another benefit of a UML Profile for IEPDs: the ability to support model-driven development of IEPDs, and therefore development of IEPDs that fit within a model-driven architecture.

NIEM and IEPDs

NIEM is a controlled vocabulary that supports information exchange among organizations.[2] As a general vocabulary for information exchange within and among its member domains, NIEM provides the building blocks from which architects and developers build the specifications for individual exchanges. These specifications are called *Information Exchange Package*

[1] http://www.niem.gov/pdf/HLTA.pdf
[2] NIEM began as a partnership between the U.S. Department of Justice and Department of Homeland Security; as such, the content of the vocabulary initially focused on domains within the purview of those organizations. However, NIEM has continued to expand, most recently including the Department of Health and Human Services as a full partner. The NIEM governance process provides for the inclusion of new domains in the future.

Documentation (IEPD). Each IEPD contains a set of XML schemas that together formally specify the structure and semantics of the exchange, and a set of associated documentation that assists stakeholders in understanding and implementing the exchange. The IEPD itself and, in particular, the schemas contained within it are considered "conformant" if they adhere to a set of normative rules developed by the NIEM governance process. These rules are codified in several official NIEM documents, including the NIEM Conformance Specification,[3] the NIEM Naming and Design Rules (NDR),[4] and the NIEM Model Package Description (MPD) Specification.[5]

It is important to note that the NIEM NDR places constraints on the range of valid XML schema constructs that may appear in a NIEM-conformant schema. That is, there are constructs that are valid according to the XML Schema specification but that are not valid in a NIEM-conformant schema. In addition, the MPD Specification defines rules about the number and types of XML schemas and other artifacts that can or must be present in an IEPD, and about metadata that must be contained within an IEPD.

NIEM conformance requires reuse of the structure and semantics within the NIEM reference schemas.[6] The practical implication of this rule is that when an architect or developer wishes to represent a concept in an exchange specification, he or she must choose the NIEM reference schemas' representation of that concept if it exists. As a result, many IEPD developers who create platform-independent models of their exchanges (e.g., in UML) do not model concepts "from scratch," but leverage NIEM concepts in the platform-independent notation. Thus, it is important to distinguish the structure and semantics defined in NIEM (which are reusable in platform-independent models) from the XML schema representation of the structure and semantics.

Model-Driven Architecture and UML

Model-Driven Architecture (MDA) is an approach to defining the functionality of software systems in structured models, from which software developers can generate executable software. The structured models represent the "logical" view of the required functionality in a *platform-independent model* (PIM) that intentionally avoids any dependency on a particular program language, operating system, application platform, or other physical mechanism. The translation of the PIM into a physical form that a computer can execute happens via one or more transformations supported by tooling[7] designed to bridge the logical and physical layers.

The development of tools to support MDA has relied upon the adoption of open standard modeling formats, such as UML. The very nature of tools that bridge separate domains—such as abstract modeling and physical execution—necessitates an approach that is inclusive of multiple vendors, execution models, and programming languages. It is likely that without the foundation

[3] http://www.niem.gov/conformance.pdf
[4] http://www.niem.gov/files/NIEM-NDR.pdf
[5] The MPD Specification has not been published yet to the NIEM web site; a copy is available from the NIEM Program Management Office upon request (NIEMPMO@NIEM.gov).
[6] A reference schema is defined by the MPD Specification, in part, as "the authoritative definition of business semantics for components in its namespace." The set of NIEM reference schemas is the fundamental structural feature of NIEM, and is available for download from http://www.niem.gov/niem/index.html, as well as within a number of NIEM tools, such as the Justice Information Exchange Model (JIEM®, see http://www.search.org/programs/info/jiem/tool/), Wayfarer (see http://www.ncsconline.org/niemwayfarer/), and the NIEM subset schema generation tool (see http://tools.niem.gov/niemtools/ssgt/index.iepd).
[7] The AndroMDA tool (see http://www.andromda.org/docs/contrib/birds-eye-view.html) is a prominent example, in open source form, of tooling that supports model-to-executable transformations.

of open standard modeling notations and metamodels,[8] MDA could never have achieved its current level of adoption and success.

Many architects and developers are familiar with UML in its graphical, diagrammatic form—and it is indeed useful for communication via visual models. However, in an MDA context, perhaps the most helpful feature of UML is that it includes a standards-based representation of the model structure underneath the graphical representation. At the lowest level, this representation consists of the XML Metadata Interchange (XMI) format, which specifies an XML-based structure for representing generic models. The Meta Object Facility (MOF) leverages XMI to establish metamodels for modeling frameworks like UML, and enables a diverse, robust ecosystem of tools to grow around these frameworks. The ultimate result of the standards-based approach is that models exist in a predictable, consistent, vendor-neutral format that allows tools to read, manipulate, and leverage the structure in each model, independent of the graphical representation that is most familiar to human users.

While in theory it may be possible to generate a platform-specific, executable output artifact from an arbitrary UML model, in practice[9] most MDA tools require the use of UML stereotypes, as defined in UML profiles, to guide the generation. UML has long supported the notion of a profile to constrain the UML metamodel in order to meet the requirements of a certain modeling domain or methodology. MDA tools leverage profiles to prevent modelers from defining model constructs that are impossible to represent in the physical layer, and to provide generation mechanisms with the specific information they need to produce correct physical models.

While MDA has traditionally sought to support the development of executable software, the philosophy and concepts of MDA can easily be extended to support exchange specification artifacts, such as IEPDs, that have a physical representation in XML Schema.

A Model-driven Approach to NIEM IEPD Development is Leveraging UML

Initial implementation experience[10] has demonstrated that all the information needed to generate a conformant IEPD can be specified in a properly profiled UML model. The community of NIEM IEPD developers could significantly benefit from further expansion of this model-driven approach to constructing IEPDs.

Leveraging UML makes available a wide array of modeling tools and utilities that are not as readily available for working with XML Schema directly. These tools support capabilities such as:

- Standard visual representation of models (i.e., class diagrams)
- Standard mechanisms for developing model validation rules (e.g., OCL[11])
- Model transformation utilities (e.g., Eclipse M2M[12])
- Toolkits for building model editors and utilities (e.g., Eclipse uml2tools[13])
- Interchange of models between tools from different vendors or open source projects

[8] A *metamodel* is a specification for the structure and semantics of models—a "model for models." The specification for UML, for example, includes a metamodel defined using the Meta Object Facility.
[9] For instance, AndroMDA requires that input models apply an AndroMDA-defined profile. See http://www.andromda.org/docs/contrib/birds-eye-view.html.
[10] The JIEM® Tool, version 5.0, uses a NIEM UML profile for IEPDs to support its IEPD editing capabilities. See http://www.search.org/programs/info/jiem/tool/.
[11] http://www.omg.org/spec/OCL/
[12] http://www.eclipse.org/m2m/
[13] http://www.eclipse.org/modeling/mdt/?project=uml2tools

Regardless of the tools used upstream in the IEPD development process, the ultimate goal of every IEPD developer is to create a conformant IEPD. Thus, developers will require tools that can extract information from the UML model and use the information to create required IEPD artifacts, which include:[14]

- Schemas that conform to the NIEM NDR and MPD specifications
- Appropriate components extracted from the NIEM reference schemas
- A synopsis of the IEPD contents in spreadsheet form (commonly known as a "mapping spreadsheet")
- A catalog (manifest) of the IEPD contents and metadata values for the IEPD
- A change log for the IEPD

For tools to function correctly (i.e., produce conformant IEPDs), the UML model will need to have IEPD- and NIEM-specific metadata at the model level and the level of individual model elements. In addition, the semantics and structure of the UML metamodel will need to be constrained to ensure a clean and complete mapping to IEPD artifacts. Consistent with the UML standard, these extensions and constraints of the core UML metamodel should be specified in a UML profile.

Principal Elements in a UML Profile for IEPDs

A UML profile for NIEM IEPDs should address the following high-level requirements:

- Capture of model-level metadata necessary to populate required and optional elements in the IEPD catalog artifact
- Mapping of UML packages to XML schema namespaces, and distinguishing between exchange, extension, and subset namespaces as defined in the MPD specification
- Mapping of UML classes, data types, properties, and enumerations to their schema equivalents, with capture of metadata necessary to specify and generate the full range of simple content models allowed by the MPD specification and NIEM NDR
- Support for basic metadata items on model elements, such as definitions
- Support for distinguishing association, metadata, and augmentation classes as defined in the NIEM NDR
- Support for globally-defined data elements and substitution groups
- Support for IEPD root data elements

In the end, a developer (using supporting tools) should be able to generate an equivalent conformant IEPD from any UML model that applies the envisioned UML profile properly. Conversely, a developer should be able to create an equivalent profiled UML model from a conformant IEPD. This is not to say construction of the tools to support these transformations will be trivial; rather, the point is that the UML model should contain all the information necessary to generate a conformant IEPD, and vice versa.

Conclusion

This paper has described NIEM, IEPDs, UML, and UML profiles, all in the context of model-driven architecture. Its core argument is that the NIEM community has much to gain from defining a UML profile that constrains UML models in such a way that developers, with appropriate tooling, can generate IEPDs from those models. If such a profile existed, the NIEM community could take advantage of a wide range of tools to create, manage, analyze, and transform UML models. Because these tools are standards-based and, in many cases, open source, they have the potential to increase IEPD developer productivity significantly while

[14] The MPD specification contains the complete list of required IEPD artifacts in Appendix F.

reducing the costs associated with proprietary tools. A profile would make it more attractive for commercial and open source tool providers to embrace the NIEM architecture, while simultaneously improving interoperability with developers in other domains.

This project was supported by Grant No. 2009-DB-BX-K009 awarded by the Bureau of Justice Assistance. The Bureau of Justice Assistance is a component of the Office of Justice Programs, which also includes the Bureau of Justice Statistics, the National Institute of Justice, the Office of Juvenile Justice and Delinquency Prevention, and the Office for Victims of Crime. Points of view or opinions in this document are those of the author and do not represent the official position or policies of the United States Department of Justice.

SEARCH
7311 Greenhaven Drive, Suite 145 • Sacramento, CA 95831
(916) 392-2550 • (916) 392-8440 (fax) • www.search.org