Canonical Modeling: NIEM and Beyond

19 September 2012

Canonical Modeling: NIEM and Beyond
Priscilla Walmsley
Datypic, Inc.
1. About canonical models
2. NIEM as an example
3. Challenges
4. Other scenarios and tools
About Canonical Models
Definitions

Forrester Research:

• A canonical information model is a model of the semantics and structure of information that adheres to a set of rules agreed upon within a defined context for communicating among a set of applications or parties.

digitalML:

• A canonical model is an enterprise design pattern which provides a common set of definitions and values for all data in motion. Canonical models are abstracted models not related to any applications. They tend to be based on simple but extensible XML Schema and provide a single view of core business entities.

Wikipedia:

• A Canonical Model is any model that is canonical in nature, i.e. a model which is in the simplest form possible based on a standard, common view within a given context.
Features of Canonical Models

• Data definitions that are intended to be reused
• Centralized source/repository for discovery
• Typically describing data in motion rather than data at rest
• Accessible to business analysts, not just techies
• Ability to subset the model for specific contexts
  • It is too big to be used as is to described an exchange.
• Ability to extend the model for specific contexts
  • It is not intended to cover every possible data element.
Why Canonical Models?

• Better data definitions
  • More organized
  • Better documented
  • More standardized names, structures
• Better communication between business and IT
• Increased interoperability
• Reduced development time
Competing Goals

Universal Vocabulary
- Reusable Components
- Broad Applicability
- Consistency Across Implementations

Strictly Specified Exchanges
- Better Validation
- Easier for Implementers
- Better Performance
NIEM as an Example
What is NIEM?

- National Information Exchange Model (niem.gov)
- A U.S. **National Standard** that facilitates information sharing:
  - Across organizational and jurisdictional boundaries
  - At all levels of government
- A **Data Model** providing:
  - Agreed-upon terms, definitions, and formats for various business concepts
  - Agreed-upon rules for how those concepts fit together
  - Independence from how information is stored in individual agency systems
- A **Structured Approach** for:
  - Development tools, processes, and methodologies

Source: NIEM Practical Implementer’s Course Available under the Creative Commons License at http://www.niem.gov/training.php
NIEM at 50,000 Feet

Domains
- Infrastructure Protection
- Immigration
- Intelligence
- Screening
- Criminal Justice

NIEM Core
- People
  - Person
  - Organization
- Places
  - Location
- Events
  - Activity
- Items
  - Substance
  - Vehicle
  - Equipment
- Emergency Management
- International Trade

Source: NIEM Practical Implementer’s Course Available under the Creative Commons License at http://www.niem.gov/training.php
The NIEM Model

- Based on XML Schema (plus annotations)
- NIEM has a "meta model" on top of XML Schema
  - defines things like objects, properties, associations, roles
- This meta model is used by both the NIEM model itself, and must be used by any extensions
NIEM Objects and Properties

• The NIEM model consists most fundamentally of objects and properties
  • Example: "Person" is an object, "PersonHairColorCode" is a property.
• Objects are represented in XML as complex elements (elements with children)
• Properties are generally represented as children of the objects.

```xml
<nc:Person s:id="Per1">
  <nc:PersonHairColorCode>PUR</nc:PersonHairColorCode>
</nc:Person>
```
Object Inheritance

• NIEM objects can extend other objects
• The base object has the type ComplexObjectType, from which all other objects are (directly or indirectly) specialized
• XML Schema complex type extension is used to represent this...
Substitutions

• NIEM properties that are semantically the same but have different physical representations can substitute for each other

• XML Schema substitution groups are used to represent this
Associations and References

- Two objects can be related using an association element.
- An association contains references to the related objects, and possibly other information.

```xml
<nc:ResidenceAssociation>
  <nc:AssociationBeginDate>
    <nc:Date>2000-01-01</nc:Date>
  </nc:AssociationBeginDate>
  <nc:AssociationEndDate>
    <nc:Date>2007-01-01</nc:Date>
  </nc:AssociationEndDate>
  <nc:PersonReference s:ref="Per1"/>
  <nc:LocationReference s:ref="Loc1"/>
  <nc:ResidenceDescriptionText>duplex</nc:ResidenceDescriptionText>
</nc:ResidenceAssociation>
```
Roles

• Roles can be used to indicate the role an object plays in another type or in an exchange
• Avoids creating conflicting specializations of the same object
  • for example, having a VictimType and a WitnessType, when a single person could play both roles

```xml
<nc:Person s:id="Per1">
  <nc:PersonName>....</nc:PersonName>
</nc:Person>

...elsewhere...
<j:Witness>
  <nc:RoleOfPersonReference s:ref="Per1"/>
  <j:WitnessWillTestifyIndicator>true</j:WitnessWillTestifyIndicator>
</j:Witness>
```
Augmentations

- Reusable bundles of properties in particular contexts
- For example, the Justice domain has a set of person-related properties that it bundles together for exchanges to reuse
- An exchange might define its own augmentations

```xml
<lexsdigest:Person>
  <nc:PersonBirthDate>...</nc:PersonBirthDate>
  <nc:PersonName>...</nc:PersonName>
  <j:PersonAugmentation>
    <j:PersonFBIIdentification>...</j:PersonFBIIdentification>
    <j:PersonEarShape>...</j:PersonEarShape>
  </j:PersonAugmentation>
  <lexsdigest:PersonRegisterNumber>...</lexsdigest:PersonRegisterNumber>
</lexsdigest:Person>
```
Metadata

- Information about the data
  - Source, quality, language, reliability, etc.
- Can be shared by multiple objects

```xml
<nc:Metadata s:id="M1">
  <nc:CommentText>Reported by suspect</nc:CommentText>
  <nc:DistributionText>SBU</nc:DistributionText>
  <nc:LastVerifiedDate>
    <nc:Date>2004-01-01</nc:Date>
  </nc:LastVerifiedDate>
</nc:Metadata>

...elsewhere...
<nc:Person>
  <nc:PersonBirthDate s:metadata="M1">...</nc:PersonBirthDate>
</nc:Person>
```
Strict Naming and Design Rules (NDR)

- Naming and documentation rules (ISO 11179)
- Specialized XML Schema annotations
  - Target type of references, metadata, augmentations
- Schema design pattern
  - Garden of Eden (global element declarations, named types)
- Namespace strategy
- Disallowed features of XML Schema
Typical NIEM Exchange Development Process

1. Exchange Content Modeling
   - Develop an exchange model diagram, depicting objects, properties, associations, etc.
   - Use UML, spreadsheets, etc. -- NIEM does not care
   - Helpful to think of things in terms of the NIEM meta-model (roles, augmentations, etc.)
2. Mapping

- Map the model to NIEM components to find the overlap
  - typically done with a spreadsheet called a Component Mapping Template
3. Subsetting NIEM

**NIEM contains:**
- ~6000 elements
- very loose cardinalities (everything is optional and repeating)
- multiple representations of the same semantics

**The subset contains:**
- only elements and types relevant to your exchange
- stricter cardinalities
- more constrained types (e.g. code lists)
- a wantlist (manifest)
4. Define an *extension schema* for anything that was missing from NIEM

5. Define an *exchange schema* for the root element(s)

6. Assemble the schemas, along with other artifacts into an *IEPD*
Challenges
Challenge #1: Looseness via Reuse

- Reuse/extension can result in overly loose models
- For example:
  - Org Unit required for Law Enforcement
  - Tax ID prohibited for Gang
- Solutions:
  - Constraint schemas with XML Schema restrictions:
    - brittle
    - require significant refactoring
  - Business rules (Schematron, new techniques)
Challenge #2: Interoperability

• Two NIEM subsets are not necessarily compatible
  • Different representations of the same semantics (code vs. text)
  • Different levels of semantic specificity
  • Different levels of structure
  • Entirely different properties chosen
  • Different customizations/substitutions

<table>
<thead>
<tr>
<th>IEPD 1 Subset</th>
<th>IEPD 2 Subset</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Person</strong></td>
<td><strong>Person</strong></td>
</tr>
<tr>
<td>• Full Name</td>
<td>• First Name</td>
</tr>
<tr>
<td>• Hair Color Text</td>
<td>• Last Name</td>
</tr>
<tr>
<td>• Identification</td>
<td>• Hair Color Code (FBI)</td>
</tr>
<tr>
<td>• Eye Color Text</td>
<td>• Drivers License Number</td>
</tr>
<tr>
<td>• IEPD 1-Specific Attribute</td>
<td>• Passport Number</td>
</tr>
</tbody>
</table>
LEXS: One Approach to the Interoperability Issue

- Digest is common area contains that is exactly the same for every IEPD
- Extensions for individual IEPDs are separated into a payload with references back to the digest
- Useful if interoperability is important (e.g. federated search scenario)

(More at http://lexsdev.org)
Challenge #3: Versioning

• Reliance on a common model complicates versioning

• If the canonical model changes, it can have a ripple effect on the exchanges that use it

• Approaches to ease versioning:
  • Defining a clear versioning policy
    • Differentiation between minor and major releases based on backward compatibility
    • Deprecation policy
  • Repository that tracks versions and can show diffs
  • No forced upgrades for exchanges that don't need it
Challenge #4: Governance and Harmonization

- Who governs the model?
  - NIEM "domain" concept allows governance to be somewhat decentralized
  - IEPDs are even more decentralized
- Harmonizing the model within and across domains is a process that:
  - Takes a lot of time
  - Has to balance competing interests
  - Requires volunteers or generous sponsors
Harmonization Tools (e.g. OpenII)
Challenge #5: XSD as Modeling Language

• XSD is not an ideal modeling language
  • Some positives:
    • easily parseable for use in tools
    • directly tied to XML structure: no need to keep in sync
    • some features of XSD do help you understand semantics
      • Type extensions
      • Substitutions
    • NIEM-specific annotations also help where XSD is limited
  • Some negatives
    • Cannot express all data constraints
    • Perception that is only for XML (JSON, RDF)
• Alternatives...UML? Spreadsheets?
Is UML the Right Representation?
Challenge #6: Level of Effort

• Myth: "NIEM (because of its tools, documentation, etc.) should make it easy to implement XML."
  • Reality: Creating a one-off exchange is easier. Sharing is hard.

• Solutions:
  • Continuously improving tools
    • Simplified presentation of the model
    • Straightforward UML to XSD mapping/conversion
  • Better documentation of best practices
Despite the Challenges, Strong Advantages...

• Shared semantics
  • Despite the challenges, it really does help
• Forces some rigor in modeling
• Tools
• Community
  • NIEM gets different groups of people in a room together and gets them talking about information sharing
Other Scenarios and Tools
Other Canonical Model Scenarios

• Other industry standards that offer different subsets/"views", e.g.
  • FpML
    • Four different views (subsets) are automatically generated from a master schema set
    • Extension is possible via substitution groups or type substitution
  
• Intra-enterprise canonical models
  • Typically large organizations implementing SOA
    • banking, insurance
  • Like an "Enterprise Data Model" but:
    • for data in motion rather than data at rest
    • directly used rather than being for documentation purposes
Other Tools

- Home-grown tools written in-house
- CAM
- Commercial tools
  - igniteXML
  - Altova Schema Agent
  - Progress DataXtend
CAM

- CAM = Content Assembly Mechanism
  - OASIS WG/Standard
  - CAM Toolkit
    - Open source reference implementation
      - http://sourceforge.net/apps/mediawiki/camprocessor
  - Design time and runtime components
  - Define an exchange from a set of source schemas via "templates"
igniteXML Approach

Physical Models

Organizers (Model Management)

Logical Models

Metadata Overlay

Consumers

Business Analyst

Integration Developer

(Web more at http://ignitexml.com)

Licensed under a Creative Commons Attribution-Noncommercial-Share Alike 3.0 Unported License

www.xmlsummerschool.com
Questions? Comments?

Thank you for your attention.

Priscilla Walmsley
http://www.datypic.com
pwalmsley@datypic.com