Dictionary Driven Exchange Content Assembly Blueprints

Concepts, Procedures and Techniques

Author:
David RR Webber
Chair OASIS CAM TC
January, 2010

http://www.oasis-open.org/committees/cam

(CAM – Content Assembly Mechanism Specification)
Agenda

• Today’s XSD Schema-based Exchanges
  • Current accepted practice – pitfalls and challenges
  • How to do this faster, simpler, more reliably?
  • Accelerated process overview

• Blueprints and Dictionaries
  • Constructing your exchange with Blueprint templates
  • Leveraging re-use – standard domain dictionaries
  • Creating your own domain dictionary from XSD or UML

• Generating Exchange Artifacts
  • NDR evaluation, Exchange schema, mapping crosswalk, XML instances, realistic data use, business rules documentation

• Summary
Today’s XSD Schema-based Exchanges

Current Practice – Pitfalls and Challenges
How to do this faster, simpler, more reliably?
Accelerated Process Overview
Current Practice – Conceptual

Diagram:

- Business Modeling
- Charter
- Work Group

Artifacts:
- Exchange Model
- Mapping Artifact
- XML Schemas

Tools:
- Business Model
- Modeling/Diagraming Tools
- Searching Tools
- Import Schema Collection
- XML Editor

Local Requirements

Workflow:

1. Project Inception
2. Exchange Content Modeling
3. Mapping
4. Schema Building
5. Packaging
Current Practice - Mechanics

Exchange Inception → Data Modeling → Standards Mapping → Schema Building → Test and Inspect → Packaging & Posting

Flows/Use Case Model → UML Domain Model → Crosswalk Mapping → Document Schemas → Sample XML Instance → Support Docs

Business Rules → Pick list spreadsheets → Extension Schemas → Sample Style Sheet → Metadata → Package Sharing to Partners

Exchange Model → Business Model → Existing Schema → Import schema Collection → XML Editor → XML Artifacts

Modeling Tool → Search Tool → Mapping Tool → Forms → SW Dev Tools → XML Artifacts

Data Ref Model → Modeling Tool → Search Tool → Import schema Collection → Forms → SW Dev Tools

Business Model → Existing Schema → Import schema Collection → XML Editor → XML Artifacts

Existing Schema → Import schema Collection → XML Editor → XML Artifacts

Import schema Collection → XML Editor → XML Artifacts

XML Editor → XML Artifacts

XML Artifacts

Business Rules

Pick list spreadsheets

Extension Schemas

Sample Style Sheet

Support Docs

Metadata

Package Sharing to Partners
### Current Practice – Team Matrix

**FTE = full time equivalent**

<table>
<thead>
<tr>
<th>Skillset</th>
<th>Qualifications</th>
<th>Experience</th>
<th>Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exchange practitioner / project lead</td>
<td>- Proficient in modelling methodology and exchange development requirements</td>
<td>Prior Information Exchange project work</td>
<td>1 FTE</td>
</tr>
<tr>
<td></td>
<td>- Proficient in complex XSD syntax writing. Familiar with developer support</td>
<td>One to two years actively writing XSD schema</td>
<td>1 FTE</td>
</tr>
<tr>
<td></td>
<td>tooling and constraints</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W3C XSD schema expert</td>
<td>- Familiar with project requirements and business applications and also</td>
<td>One year or more in application area</td>
<td>1 FTE</td>
</tr>
<tr>
<td></td>
<td>developing XML based exchanges</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Domain business analyst</td>
<td>- Use of UML diagramming and models. Information modelling</td>
<td>Prior UML based modelling</td>
<td>1 FTE</td>
</tr>
<tr>
<td></td>
<td>- Knowledge of SW tools available for target environment.</td>
<td>SW tools training and XML development</td>
<td>1 FTE</td>
</tr>
<tr>
<td></td>
<td>- Creating test environments, working with XML test cases, test data</td>
<td>Data analysis and XML content creation</td>
<td>1 FTE</td>
</tr>
<tr>
<td>UML/ data modelling practitioner</td>
<td>generation</td>
<td>Technical writer</td>
<td>1 FTE</td>
</tr>
<tr>
<td>SW dev tooling user</td>
<td>- Writing documentation and spreadsheets</td>
<td></td>
<td></td>
</tr>
<tr>
<td>XML testing and development</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Documentation resources</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Pitfalls and Challenges

• Significant amount of manual labor needed to develop all the exchange documenting artifacts and XML related end products
• Multi-discipline team and supporting cast of exchange / XML savvy developers needed
• Disconnect between the software delivery teams’ schedule and process and the exchange development team and process; production system not matching what the delivery doc says it does
• Alignment to existing domain Enterprise Data Model (EDM)
• Varying quality of hand checked results and no consistency of technical approach to schema development techniques and re-use of domain components
• Process not repeatable and predictable
• Scalability - differing production XML details across teams, often incompatible across implementations and platforms
## Delivery Level of Effort Estimates

<table>
<thead>
<tr>
<th>Component</th>
<th>Tasks</th>
<th>Timings</th>
<th>Constraints</th>
</tr>
</thead>
<tbody>
<tr>
<td>Collect exchange needs</td>
<td>Model information needs</td>
<td>Weeks</td>
<td>Spiral analysis</td>
</tr>
<tr>
<td>Perform XSD schema development with EDM alignment</td>
<td>XSD syntax writing</td>
<td>Weeks</td>
<td>Complex with steep learning curve and limited practitioners.</td>
</tr>
<tr>
<td>Documentation of each element</td>
<td>Excel spreadsheet</td>
<td>Weeks</td>
<td>Manual preparation and review</td>
</tr>
<tr>
<td>Document domain dictionary mapping (pick list)</td>
<td>Excel spreadsheet</td>
<td>2 to 5 days</td>
<td>Manual preparation and review</td>
</tr>
<tr>
<td>Create test cases and examples</td>
<td>Sets of XML instances</td>
<td>Weeks</td>
<td>Manual hand editing of XML from XSD</td>
</tr>
<tr>
<td>Perform interoperability testing</td>
<td>Build test environments</td>
<td>Weeks</td>
<td>Test harnesses vary</td>
</tr>
<tr>
<td>Create exchange documentation</td>
<td>Word documentation</td>
<td>Weeks</td>
<td>Manual preparation</td>
</tr>
</tbody>
</table>

Currently 800+ hour process for 300+ node exchange
Improving the Process

- Resolving the issues and challenges
- Ensuring consistent results that can be easily reviewed
- Leverage existing dictionary work and repositories of components that the enterprise already has
- Reduce the learning curve and need for specialized skills
- Business analysts not excluded from design, review and implementation by technical barriers
- Lock-step the development process to the exchange
- Customizable and configurable so can adapt to changing requirements
Faster, Simpler, Predictable

- Tooling automates much of the manual tasks; ensures predictable quality of results
- Reduce need for specialized technical knowledge of XSD and XML
- Provide consistent approach that leverages best-practice techniques with built-in smarts and knowledge
- Tooling checks for common pitfalls, applies NDR checks
- Allow business analyst to complete much of the design work and crosscheck application details
- Leverage reuse of domain component dictionaries and blueprints
- Lockstep development to exchange artifacts and their delivery
- Accelerate development tasks (test cases, testing, schema writing)
- Produce result that are neutral to developer tooling platforms
- Process repeatable and replicatable when requirements / versions change
Using Dictionaries & Blueprints

• **Dictionaries** provide reference sets of components to be used in exchanges; three possible sources:
  • Dictionaries imported from existing industry schema
  • Domain dictionary built from an Enterprise Data Model schema
  • Reverse engineered out from existing exchange schema

• **Blueprint**
  • Is the outline of the structure components to be used in an exchange schema
  • Can import components from one or more domain dictionary collections
  • Sketches out the desired information exchange with re-use of existing exchange component structures, plus any local additions / extensions / exclusions

• **Expander** tool reads the blueprint, references the dictionary, and constructs the complete exchange schema
Accelerated Process Overview
Blueprints and Dictionaries

Leveraging re-use – dictionaries from industry standards
Creating your own domain dictionary from XSD or UML
Constructing your exchange and blueprints
Building Domain Dictionaries

**Option 1 – From Enterprise Data Model**
- Import XSD and refactor for use with OASIS CAM

1. **EDM**
   - Export Components in XSD syntax
     - Collection of objects from model

2. **Model Components XSD schema**

3. **OASIS CAM template**

4. **NDR Evaluation, Refactor, Renaming Tool**

5. **Generate Standard Components Dictionary XML**

**Option 2 – Derive from existing exchange XSD schema**
- Import each XSD and merge into CAM dictionary

2. **Exchange XSD schema**

3. **OASIS CAM template**

4. **NDR Evaluation, Refactor, Renaming Tool**

5. **Generate Standard Components Dictionary XML**

**Legend**
- Automated
- Manual

**Dictionary of exchange components**

- ebXML CCTS compatible (ABIE, BBIE, ASBIE)
Blueprint Approach Overview

1. Enterprise Data Model
2. Components Definition (XML)
3. Industry dictionaries formatted as XML
4. Local domain dictionary formatted as XML
5. Exchange generator tools (CAM)
6. Exchange Components
7. Exchange Package

LEGEND
- Automated
- Manual

Blueprint Approach Overview Diagram

Enterprise Data Model
- Import and refactor for use with CAM

Components Definition (XML)

Industry dictionaries formatted as XML

Local domain dictionary formatted as XML

Exchange generator tools (CAM)
- Build

Target applications

Diagram showing the process flow from EDM to Exchange Package with various steps and tools involved.
Blueprint Development Tools

1. Blueprint Designer
2. Search Tools
3. Insert Dictionary Parent Components
4. Expander Tool
5. Completed Exchange Template
6. Excel
7. Domain dictionary
8. Industry dictionary
9. Component Definitions
10. Component Definitions
Blueprint Expander Example

Structure Details Expanded

Dictionary Lookups

Expander Tool

EXCHANGE BLUEPRINT

Blueprint follows model

Exchange components outline

COMPLETED EXCHANGE TEMPLATE

Console Log detail

Exchange Template Editor

1. Exchange Structure items
2. Rules for each item
3. Can add new custom domain Items
4. Domain Components Namespaces
Generating Exchange Artifacts

NDR evaluation, crosswalk mapping, Exchange Schema, Subset schema, XML instances, business rules documentation
Exchange Generation Steps

**EXCHANGE TEMPLATE**

**Structure**

Toolkit for exchange artifacts generation

**Rules**

Suite of menu options and tools in desktop CAM toolkit editor

*each run custom xslt scripts on the exchange CAM template xml*

**Documentation**

**CAM Template**

- Run template Evaluation Report
- Compare to industry dictionary
  - create mapping spreadsheet
  - create crosswalk xml
- Generate business rules report
- Generate exchange XSD schema
- Generate XML test instance(s)
NDR Evaluation Report

• Provides scoring, alerts, warnings and potential issues including:
  • Naming and content model conventions
  • Naming and restriction consistency checks
  • Interoperability enablers/inhibitors checks
  • Rules integrity and duplicates
  • Statistics on exchange size
  • Spell checking on component names
Evaluation Report (NDR) example

Part of an example report for LEXS getDataItemRequest template displayed in toolkit HTML viewer

SCORE: 7 out of 10.0
Compare to industry dictionary

- References industry dictionary of names and properties
- Matches on physical names
- Reports mapping details
- Compatible with Microsoft Excel
- Report can be used to do spell checking
- Generates crosswalk xml file
Example cross-reference spreadsheet

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source</td>
<td>Interface</td>
<td>Type</td>
<td>Description</td>
<td>Annotation</td>
</tr>
<tr>
<td>A1</td>
<td>B1</td>
<td>C1</td>
<td>D1</td>
<td>E1</td>
</tr>
</tbody>
</table>

**Formatted view in Microsoft Excel of import of cross-reference report details (from generated XML file)**
Generate Documentation and Schema

- **Documentation:**
  - Create HTML report of exchange schema details and associated content and business rules
  - Report layout and content designed to be reviewed by business analysts

- **Schema:**
  - Generate XSD schema for exchange
  - Customizable exchange folder layout management by namespace for extension, subset and exchange schema components
  - Writes XSD schema in syntax that is clear, simple and compatible with deployment tooling environments
## Business Rules Documentation

### ID: getDataItemRequest

**Taxonomy:** XML

<table>
<thead>
<tr>
<th>XPath locator</th>
<th>Rule(s)</th>
<th>Annotation</th>
</tr>
</thead>
<tbody>
<tr>
<td>lex:s:getDataItemRequest</td>
<td>required item</td>
<td>Definition: LEXS request for a data item.</td>
</tr>
<tr>
<td>lex:s:DataItemRequestMessage</td>
<td>required item</td>
<td>Definition: Request message for a single Data Item. Only one data item may be requested in each message.</td>
</tr>
<tr>
<td>lex:s:SRMessageMetadata</td>
<td>required item</td>
<td>Definition: Metadata about Search/Retrieve message.</td>
</tr>
<tr>
<td>lex:s:LEXSVersion</td>
<td>required item</td>
<td>Definition: Specifies LEXS version used within the document, for example 3.1.1</td>
</tr>
<tr>
<td>@sid</td>
<td>optional</td>
<td>Definition: The id attribute is used to define XML IDs for NIEM objects. These IDs may be targets of reference elements, metadata attributes, and link metadata attributes.</td>
</tr>
<tr>
<td>@s:metadata</td>
<td>optional</td>
<td>Definition: The attribute metadata allows an object to point to metadata that affects itself.</td>
</tr>
<tr>
<td>@s:linkMetadata</td>
<td>optional</td>
<td>Definition: The linkMetadata attribute allows an element to point to metadata that affects the relationship between the context and the value of the object.</td>
</tr>
<tr>
<td>lex:s:MessageDateTime</td>
<td>if string-length(.) &lt;26</td>
<td>Definition: Date and time the message was created.</td>
</tr>
</tbody>
</table>

Part of the example rules for LEXS getDataItemRequest template displayed in toolkit HTML viewer.
Export Exchange to XSD Schema

Export Template to Exchange XSD:

Completed Exchange Structure

Set Exchange Options

Complete set of exchange schemas generated
Exchange Schema Generated

- Each namespace file is imported for those specific type definitions.
- Set of XSD files with filename and namespace suffix.
- Reviewing XSD results in a schema editor tool.
XML Testing Examples Generation

• Support for software development testing process
• Designed to allow creation of concrete realistic examples not just random value based
• Hinting system allows insertion of actual test system values into XML examples
• Can create both valid and invalid examples to support unit testing of application software
• Exclude capability allows generator to create examples that contain only a portion of the entire exchange
• Control over random seed value used allows re-generation of identical test cases
XML example generation wizard

View of CAM toolkit with LEXS getDatItemRequest and dialogue for XML test example generator tool
Running validation rules tests

• Built-in CAM validation engine allows testing of XML instances against actual exchange rules (CAMV).
• Critical to ensure that the exchange validates actual live production example scenarios correctly
• Allows deployed solution to match exchange schema details
• Errors can be reviewed interactively in exchange visual interface
• Post-processing of validation results allows unit regression tests to be created with reporting of errors, warnings and information level notes
Run Exchange Template

Pick XML test case to validate Run validation Review results in visual editor
## Example Exchange Packaging Details

<table>
<thead>
<tr>
<th>Package Artefact</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Exchange Files</strong></td>
<td></td>
</tr>
<tr>
<td>Subset Schema</td>
<td>Subset of the full exchange schema—a reduced set of components only used in this exchange, not every possible component.</td>
</tr>
<tr>
<td>Crosswalk XML</td>
<td>Itemized list of each dictionary component element and attribute included in the exchange.</td>
</tr>
<tr>
<td>Exchange Schema</td>
<td>Base document schema that defines the full XML structure for the exchange and is generally named after the exchange itself.</td>
</tr>
<tr>
<td>Constraint Schema</td>
<td>Optional schema that includes additional constraints and code values for the main exchange schema</td>
</tr>
<tr>
<td>Extension Schema</td>
<td>Specification for extended components—separate local name-spaces of components not contained in dictionary</td>
</tr>
<tr>
<td>Sample XML Instance</td>
<td>Example instance(s) – may reference optional stylesheet.</td>
</tr>
<tr>
<td>Stylesheet</td>
<td>Example stylesheet for display of instance(s).</td>
</tr>
<tr>
<td><strong>Documentation</strong></td>
<td></td>
</tr>
<tr>
<td>Master Documentation</td>
<td>The Master Document is the main document for which all of the context and details around the exchange are explained. This document includes, the overview, as well as details surrounding the exchange, business drivers and requirements</td>
</tr>
<tr>
<td>Exchange model</td>
<td>Exchange model in standard open format (xmi, vsd, zargo) and standard open graphic (jpg, pdf, etc.) preferably a Unified Modeling Language (UML) model.</td>
</tr>
<tr>
<td>Business rules</td>
<td>Business rules in one of the following formats: (1) plain or structured English, (2) written into master documentation, (3) generated by a development tool.</td>
</tr>
<tr>
<td>Mapping to Dictionary</td>
<td>Mapping of domain components, tagged with constraints (i.e., cardinality, etc.) to dictionary components as a spreadsheet.</td>
</tr>
<tr>
<td>Extended components</td>
<td>Components created because they were not in dictionary—may be part of mapping spreadsheet and include structure and definitions of new components.</td>
</tr>
<tr>
<td>Change log</td>
<td>Record of cumulative changes from previous exchange versions. The initial exchange simple records its creation date.</td>
</tr>
<tr>
<td><strong>Catalog</strong></td>
<td></td>
</tr>
<tr>
<td>Catalog XML file</td>
<td>A machine-readable list of artifacts provided in this exchange package.</td>
</tr>
<tr>
<td>Metadata XML file</td>
<td>All metadata of owner and domain to be associated with the exchange.</td>
</tr>
</tbody>
</table>
Summary

Dictionary driven exchanges
Blueprint enabled reuse
Automated exchange package generation
Alignment to NDR Principles and Rules
Testing and validation support
Review

- **Top Down development**
  - Reference dictionary components
  - Create exchange blueprint
  - Run Expander tool
  - Refine desired structure in visual editor

- **NDR Principles and Rules**
  - Best practices for interoperability and schema techniques

- **Dictionary driven reuse**
  - Enterprise Data Model and industry components
  - Ensures consistency of definition and use

- **Automated exchange package generation**
  - Schemas, XML, documentation, mapping crosswalk
  - Test generated example XML with rules validation
Reference Materials

References and Links
Links and Resources

- **DOWNLOADS** -
  - CAM Toolkit download
    - [https://sourceforge.net/projects/camprocessor](https://sourceforge.net/projects/camprocessor)

- **SUPPORTING MATERIALS** -
  - NIEM Naming and Design Rules (NDR) 1.3

- **RESOURCES** –
  - Additional support slides (following)
Blueprint Driven Approach

BUSINESS USERS

Needs
Requirements
Procurement

Agile Dynamic Components
TEMPLATES

Dynamic

Implementation
/Use

Installation

Coding
Design

Maintenance
Test

Static
Conventional
Models, Artefacts, Code
WSDL,XSD,UML,XML

SW DEVELOPERS

Analysis
Specification

Data models
Excel spreadsheet
Blueprint Templates
XML visualization
XML artifacts
XSD schema

Software code
Compilers
Deployment servers
XSD schema
XML artifacts

1. **Template**
   - design time
   - Library
   - Object templates/Components
   - Facets
   - Questions/Data

2. **XML**
   - Domain Dictionary Details Stored

3. **Wizard**
   - runtime configuration

4. **Technology Targeting**
   - Syntax specific production rules

5. **Exchange Interface/Blueprints**
   - Visual editor + review/test/completion steps

6. **Solution Specific Syntax**

OASIS
Domain Exchange Development Steps

• **Adopt formal Naming and Design Rules (NDR)**
  - UN/CEFACT – NDR
  - OASIS UBL – Universal Business Language
  - OASIS EML – Election Markup Language
  - OASIS EM - Emergency Management joint initiative with NIEM

• **Develop data models of core components for use in exchanges**

• **Build Dictionary of Core Components**

• **Provide Principles and Rules guidance to schema team**
  - Use namespaces, Yes / No?
  - Camel case naming convention?
  - Schema constructs and restrictions on use?

• **Information Exchange Package Documentation (IEPD)**
  - Describes formal exchange that conforms to NDR and principles and rules
  - Provides schema, example XML, supporting artifacts
  - Re-uses core components
  - Defines domain specific components
Example Governance Structure

### Required Actions

**DM process & OASIS**
1. Utilize NIEM as development Data Dictionary
2. Utilize NIEM NDR for element/attribute extension requests
3. Utilize and Update the NIEM EM Domain via governance established by NIEM

**NIEM**
1. Provide final Naming and Design Rules (NDR)
2. Provide governance for change management to include processes for addition/deletion/modification of EM Domain elements and attributes
3. Provide IEPD Repository

**NIMS**
1. Evaluate proposed standards for adoption into “NIMS Approved” Messaging Standards
2. Require NIEM compliance for all “NIMS Approved” Messaging standards.
3. Publish to NIEM IEPD Repository with XML Messages for community reuse.
### OASIS Content Assembly Mechanism (CAM) & Integration Technologies Guide

#### WHAT?
- Provides lexicon of information content
- Describes structure constructs
- Arranges groups of information
- Simple content typing
- Software tooling interfaces

#### HOW? WHY?
- Provides actual use patterns (templates)
- Supports context handling and rules
- Rendering outputs and documentation for verification
- Enables integration testing and certification
- Versioning

#### WHO?
- Alignment of meaning and terms
- Consistent domain definitions
- Modelling methods and practice
- Business information content building blocks

#### WHERE?
- Shared resources of semantic definitions
- Code lists
- Dynamic rendering
- Distributed versioning control
- Role and access security management

#### WHEN?
- Alerts
- Process control
- Workflow
- Automated interfacing
- Business Intelligence