colectica

DDI 3.2 INTEROPERABILITY AND BEST PRACTICES

THE STORY BEHIND DDI 3.2

EDDI 2014 : December 2nd 2014

In This Talk

- 3.2 Technical Committee Goals
  - A focus on consistency
  - Design and content guidelines
- Automated Testing
- Identification improvements
- 3.2 Serialization Improvements
  - New serialization container
  - Single serialization location for items
  - Aligned with DDI 4 xml serialization and views

A focus on consistency

- For 3.2, the technical committee outlined a set of design and content guides
- These practices were already used in 3.1, but not followed throughout the schema
- Used the 3.2 development process to make sure practices were followed
- This was the TC's main work following the public review

Key focus on

- Ensure items and properties were uniquely named
- Make sure elements were reused where appropriate
- Remove most mandatory properties
  - Cardinality was relaxed to 0..1 or 0..n
- Ensure child items were always able to be referenced
  - 3.1 sometimes required inline inclusion

Automated testing of the schema

- Many improvements were made through manual investigation of the schema, but completeness is required
- A tool was created to perform consistency checks on the schema
- Ensured that the technical committees consistency goals were realized
- Open Source
  - https://github.com/DanSmith/DDISchemaCheck/

Automated testing checks (I)

- Check compilation of the schema as an XML Schema Set.
  - Versionables and Maintainables allowing inline or reference usage.
  - Versionables and Maintainables are in a xs:Choice.
  - Versionables and Maintainables in a xs:Choice contain two elements.
  - Versionables and Maintainables in a xs:Choice contain a xxxReference.
- FragmentInstance contains all Versionables and Maintainables.
- Type of Object for references
  - Duplicate Element names detected for referenceable types.
  - Element names detected without a TypeOfObject defined.

Automated testing checks (II)

- Spell checking
  - Element names
  - Attribute names
  - XSD annotations/documentation
  - Breaking apart CamelCasedWords
  - Allows words to be added to dictionary
  - Uses en-US
  - Highlighting of misspellings in generated reports.
- Proper spelling in the standard and documentation gives a professional feel

Automated testing checks (III)

- Example Report
  - DDI Schema Test Report Sample

Identification updates

- In DDI 3.1, identification of some items were tied to the identification of a parent item in the information model
  - Made reuse of items overly complicated
- In 3.2, identifier scoping is well defined
  - Unique within an agency scope
- Backwards compatible with 3.1 system
  - Previously unique within parent identifiers result in a concatenated id
  - Also unique within an agency scope

The Results

- Five years of development results in DDI 3.2
- The most consistent version of the standard
  - Adherence to defined patterns
  - Automated testing
- Many small **consistency improvements** combine to enable **better serialization** of the information model
- Lets compare the XML serialization capabilities of DDI 3.1 and 3.2

Old Serialization Style

- In Instances, where are items concretely located?
- Variables
  - DDIInstance\ResourcePackage\VariableScheme\Variable
  - DDIInstance\ResourcePackage\LogicalProduct\VariableScheme\Variable
  - DDIInstance\StudyUnit\LogicalProduct\VariableScheme\Variable
  - DDIInstance\Group\LogicalProduct\VariableScheme\Variable
  - DDIInstance\Group\StudyUnit\LogicalProduct\VariableScheme\Variable
  - DDIInstance\Group\SubGroup\StudyUnit\LogicalProduct\VariableScheme\Variable
  - DDIInstance\Group\SubGroup\SubGroup\StudyUnit\LogicalProduct\VariableScheme\Variable
  - DDIInstance\Group\SubGroup\*\SubGroup\StudyUnit\LogicalProduct\VariableScheme\Variable
  - DDIInstance\LocalHoldingPackage\LocalAddedContent\LocalGroupContent\LogicalProduct\VariableScheme\Variable
  - DDIInstance\LocalHoldingPackage\LocalAddedContent\LocalGroupContent\StudyUnit\LogicalProduct\VariableScheme\Variable
  - DDIInstance\LocalHoldingPackage\LocalAddedContent\LocalGroupContent\SubGroup\StudyUnit\LogicalProduct\VariableScheme\Variable
  - DDIInstance\LocalHoldingPackage\LocalAddedContent\LocalGroupContent\SubGroup\*\StudyUnit\LogicalProduct\VariableScheme\Variable
  - DDIInstance\LocalHoldingPackage\LocalAddedContent\LocalStudyUnitContent\LogicalProduct\VariableScheme\Variable
  - DDIInstance\LocalHoldingPackage\LocalAddedContent\LocalResourcePackageContent\LogicalProduct\VariableScheme\Variable
  - DDIInstance\LocalHoldingPackage\LocalAddedContent\LocalResourcePackageContent\VariableScheme\Variable
- With SubGroups, the number is actually unlimited.

Old Serialization Style (2)

- This issue occurs for all item types in DDI
- Tight coupling of the Information Model and the Serialization Format created the problem
- Profiles?
  - Allows users to document their usage
  - Must be exchanged, and implemented
  - Do not address the main serialization issue within the standard, which still leaves interoperability challenges

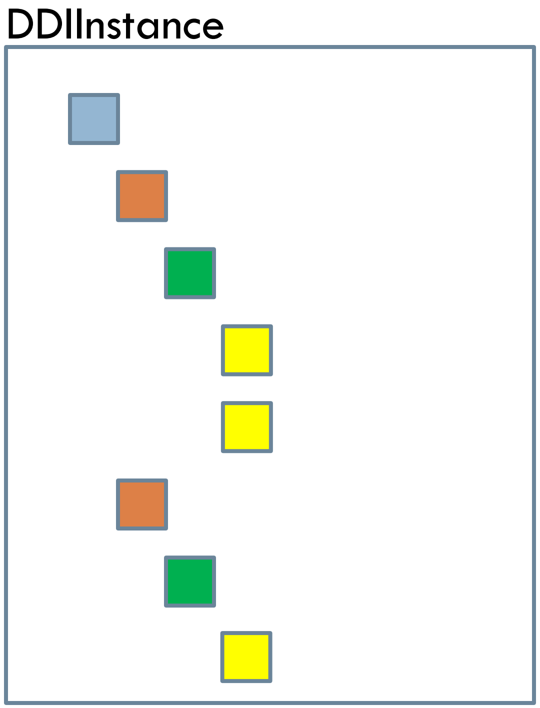
Three Key Enablers for Serialization

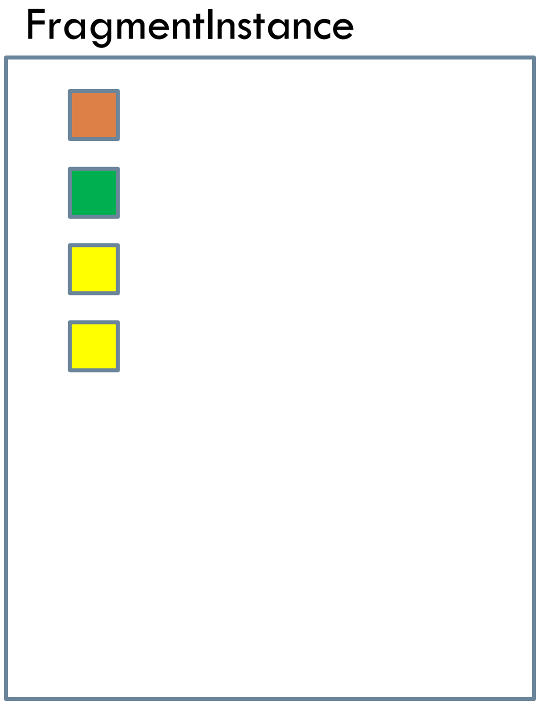
- Concise Bounded Descriptions
  - http://www.w3.org/Submission/CBD/
  - In 3.1 where items were nested inline, they are now also available via reference
  - 3.2 schema was programmatically checked before release to ensure reference availability
- Min Occurs is zero
  - Most elements are now optional
- Resolution of DDI identification issues
  - Decouple item hierarchy from item identity
  - Allows agency scoped ids for all items

Serialization Solution in 3.2

- Eliminate the tight coupling of the Information Model and the Serialization
- Solution – 3.2 FragmentInstance
  - Limit locations for concrete items, and use references
  - Implemented in DDI 3.2 as alternative container
  - Create views using TopLevelReferences
  - Only implementation pattern in DDI 4
- Each item type has a single location (Xpath)
  - FragmentInstance\Fragment\Variable

Serialization Solution in 3.2 (2)





Nesting mixes object
model and serialization

Uniform item serialization

Serialization Solution in 3.2 (3)

- Each item type has a single location (Xpath)
  - FragmentInstance\Fragment\Variable
  - Improvement in application interoperability
- Each serialized item (maintainable or versionable) includes its child items using a reference
  - Implements Concise Bounded Descriptions
  - References include item type
  - Reference resolution is simplified, only one possible location for each concrete item type

colectica

Thank you

Web	colectica.com
Blog	blogs.colectica.com
Twitter	@Colectica
YouTube	youtube.com/colectica