News Architecture decisions

Based on the “Expressing the IPTC Standards Architecture using W3C XML Schema” document

Document history			[Document URN: urn:iptc:workdoc:nar:0503:3]	
Revision	Issue Date	Pages	Author (revised by)	Remark
unrevised	2005-03-05	7	Laurent Le Meur	-
2	2005-03-10	8	Laurent Le Meur	Adds inputs from Johan and Jay
3	2005-03-11	8	Laurent Le Meur	Edited during confcall, 2005-03-11

1 Overview

This document summarises the decisions made by the NAR WP regarding the decision points at the end of the document DRAFT-NAR_1.0-doc-ArchitectureDiscussionDocument_10.pdf (from Jay Cousins /Ulf Wingstedt, coded *ADD* in this document) and exposed at the end of the presentation made by the consultants during the IPTC 2005 Spring meeting in San Diego.

It uses SWOT diagrams (S= Strength, W = Weakness, O = opportunity, T = threats) for clarity.

Extensibility of content

- Policy for when to use generic or explicit schema type approach for validation of controlled vocabularies?

Note: the question can be understood has: what should be the mechanism to validate a values against a list of allowed values (in a controlled vocabulary), especially when several controlled vocabularies can be used at the same place (e.g. an IPTC recommended list and a provider created list).

The NMDF works on the handling of metadata, and envisages the use of {scheme,code} pairs to identify metadata values in a flexible manner. The “scheme” (or codeList) would indicate which vocabulary the property value belongs to (vocabulary controlled by the IPTC or an individual provider). The exact syntax of scheme-code pairs and its extension as a resource URI are out of scope of this discussion, and will be addressed by the NMDF.

The schema based validation of values is useful in some occasions: for the qualification of a feed (by a provider or a recipient), on a test channel (provider) or before processing specific metadata where bad values could break the processing of items (e.g. management metadata).

The consultants propose (see ADD 3.7.1) two approaches:

<u>Generic approach</u>

Sample:

```
<ProductCode codeList=”company.se:ProdCodes”  value=”SPIK”/>
<ProductCode codeList=”company.com:ProdCodes”  value=”NAIL”/>
```

S	W
- Simplicity of the design - Benefits of a fixed data model coupled with the provision of a common processing model	- No schema based validation - Need of a specific processing for validation

- No ‘hard-wiring’ of semantics in to element structures - Resilient to change in the ctrld values (-> independent of the schema)	
O - Good when the number of values is large, when they change often, when IPTC has no control on the ctrl.voc - Adopted in Atom 0.5 for the Category element	T - Some implementers will not implement validation processing.

<u>Explicit schema based approach:</u>
Sample:

```
<ProductCode xsi:type=”SwedishProductCodes”>SPIK</ProductCode>
<ProductCode xsi:type=”EnglishProductCodes”>NAIL</ProductCode>
```

An enumerated type is created in the schema. A metadata element may then be given that type in the schema, by the IPTC or by a provider extending the schema (if allowed). Alternatively a provider can use the specific xml schema instance attribute (xsi:type) to force the type of a metadata element (a method called type substitution).

S - The provider of the item can tweak the validation constraints “on the fly”	W - Need for another namespace declaration (xsi) at the top of each item - If the recipient doesn’t have the type definition in his version of the schema, validation will fail - Works with xml schema, not with dtd - Impacted by schema updates
O - Good for set of values that are critical for the processing of items - Validation of values integrated in the xml schema world - Can be implemented as an extension schema created by users	T - Newscomers and basic users will be afraid of this weird notation

<u>Decision</u>:
The generic approach will be used in most cases (e.g. descriptive metadata).
For some metadata elements, where a schema based validation is needed, the model will use “Plug-in schemas” (controlled vocabularies defined by providers, also called user defined extensions) as it is the case for SportsML. But it will not use the added flexibility of the xsi:type notation.
The members ask the consultants to detail the recommendation for the implementation of plug-in schemas, detail what they call the ‘fixed-generic’ approach (ADD 3.7.2), and add somewhere a recommendation for the resolution of schemes-codes pairs as references for topic items (i.e. a URI

identifying the chunk of xml, managed as other items, that carries information about a concept like a person or event).

Extensibility of the data model

- Location of structural extensions?
  - ☐ Allow wild card attribute and elements across components - wherever a user organization would like, or,
  - ☐ Allow only in specific contexts – an extension element of xs:any?

Wild card attribute and elements

S - Total flexibility for the providers	W - Loads the XSD schema with a lot of wild cards
O	T - Lowers the perception of this work as a "standardization" work

Extension element

S - Extensibility with good control by IPTC	W - Location of extension may evolve as IPTC members show have new needs - The extended information may not be managed like the components derived from AnyComponent (e.g extensions may have no local id etc.)
O - Atom allows this kind of extension by the addition of new elements to an entry, in a different namespace than the core.	T - May be already giving too much flexibility for the core level - May be dangerous for management metadata (for the sake of total interoperability of processing).

Decision:
Location of structural extensions is allowed only in specific contexts.

Additional decision:
Providers won't be allowed to modify the syntax and semantics (i.e. the model) of an existing component (core or aggregate), e.g. to modify the type of an element (from date to string for example).

Request to consultants:
The consultants are asked to provide some information on the "cost" of allowing the addition of attributes from #other namespace to existing components, i.e. the possible drawbacks/problems that would have to be managed in order to get the benefits from such an approach.

For IPTC discussion:

Extension could be supported by aggregate components only.
The COCO WG will choose which aggregate components can be extended.
Note: an alternative is to give to the "specialised" WGs (e.g. EventML WG) the responsibility to choose which constructs can be extended. But users of multiple item classes would be surprised to see that what can be extended in a given item cannot in another.
The NMAN WG will choose if the Management component can be extended this way.
These components will get a "#other" extension point at their end (extension with elements from another namespace).
Maybe extension will be allowed only at the power conformance level (keeping the core level straightforward to implement); this will be reviewed later by the NSTR WG.
If information has been added to a construct, the recipient processor – e.g. the xml schema processor – will apply a mustIgnore rule ("if you don't know, just ignore").
The added information will not be managed like native IPTC information (no update, no assignment, no value control).
The extensibility of the items (derived from AnyItem) themselves will be reviewed later by the NSTR WG. A possible solution is to allow providers to create their own components and include them inside items of any kinds. This would true for all standards, maybe at the power level only. Being IPTC components, they would be managed like other components (update, assignment, value control)

Conformance

- Conformance
    - ☐ Is there a need for schema enforced conformance rules or not?

From the ADD: "A single schema can reflect requirements from different conformance levels by providing alternative content models as choice, substitution groups or type substitution (xsi:type). This approach is best implemented when it is possible to clearly modularise the content model for different conformance levels." (ADD 3.10.8).

About a single schema:

S - Maintenance is simplified for the IPTC - A provider manages only one schema.	W - Basic users see a complex schema - Conformance levels are applied at the level of the application, not enforced via schema
O	T

Alternative: a "core" schema is used stand-alone, or included in a "power" schema that extends the features of the elements classes.

About a model based on the inclusion of a core schema in a power schema:

S - Basic users see a simple schema	W - IPTC has to maintain 2 schemas for each standard = burden - Could be complicated at the instance level

O	T

Decision
Conformance levels should be schema enforced (i.e. different data structures that support different conformance levels should be clearly distinguished in the schema).

Added decision:
Our preference is the inclusion of a core schema in a power schema. This would be transparent for the users, as the declared namespaces would be the same for all conformance levels. Doing so, implementers of the core conformance level would not have to bother about extra features.
The consultants are asked to check if this preferred approach is manageable (e.g. if the elements of the core and power profiles had to be declared in a different namespace, this preferred approach does not fly).

Note for IPTC:
The specification of the power level is an extension of the core specification: it should be reflected in the specification documents (-> two sections = 'core' and 'power extensions').
This discussion only deals with the structural facet of the model, but conformance levels are treated also in the processing model.

Versioning policy

- ☐ Does the proposed policy fulfil business requirements?

Summary of the proposed model:
A major version is associated with a specific namespace. The same namespace is kept for minor revisions (backward compatible).
The major/minor version is indicated in instances of documents via attribute(s) (e.g. schemaVersion), and schemaLocation may be given also if recipient validation is an option.
Major/minor version is also indicated in the schemas, as a xs:schema/@version attribute.

S - No need to modify the recipient processing model because one element has been added (or other bkw compatible change) - The recipient processor knows what version it is without relying on the schema.	W - Schema version has to be added in each instance
O - This model is followed by several initiatives, and has been recommended in different articles on the web	T - This model is disputed by some experts - The IPTC has previously adopted a namespace change for minor versions also.

Decision:
We follow the recommendations of the consultants.

Added decision:
The consultants are asked to modify the samples in order to use the usual notation of IPTC (major version as a positive integer, minor version as a positive integer) e.g. 1.0.3 becomes 1.3.
Note: The IPTC policy is that editorial versioning is treated separately.

Common Components - namespaces

- ☐ Should there be separate namespaces for the individual types of common components?
    - ☐ For common component layers – data type, property, construct?
    - ☐ For the different types of construct created?

If there is one namespace only (per major version) for all common components:

S - Management of CC remains simple.	W - If one element of the library is strongly modified (non backward compatible transformation), the whole library must be given a new major version
O -	T -

Note: An alternative approach would be one namespace per major common constructs (management, description, rights, publication, signature), and the use of a ‘common’ namespace for embedded constructs and properties. But it would introduce another namespace level with the result that instead of having a single namespace for all common components, we would have a namespace for common components and namespace(s) for some specific major construct common components.

Decision:
There will be one namespace for the whole CC library.

Note for IPTC:
The IPTC needs to discuss the release procedure of a news version of the CC library. (Shall we have a “patch day” policy for individual items included in standards?)..

Common Components - naming

- ■ Is the proposed structure, definition and naming of the common components appropriate?
    - ☐ Data type, basic component and aggregate component

Decision:
As no consensus is currently achieved amongst the COCO WG members, the consultants can keep the current names.

Note for IPTC:
The COCO WG will provide agreed names to the NSTR and NMAN WG for proper inclusion in the conceptual and processing model, and these names will be also used in the names of common components.

Common Components - specialization

- Should specialization of common components be allowed in items (standards) or not?

Use case:
1/ The COCO WG creates a PersonComponent. Can the Event WG derive from it a ParticipantComponent with added properties?
2/ An IPTC WG (e.g. SportsML WG) takes a component that has fixed enumerated values (e.g Management/Status). If Status values are not sufficient for the WG: can the WG add values in its standard?

If specialization is allowed:

S - IPTC WG can adapt components to specialised needs.	W
O	T - IPTC WG could end up specializing all components. Then there would be no more common library.

Decision:
Extension of common components by IPTC WGs is allowed by using the extensibility mechanism built in the common components.

Note for IPTC:
A good policy is to ask first to the COCO WG an adaptation of the component, with optional elements. If the additional elements can be used by other classes of items, the extension of the library is preferable. If it does not make sense at all, IPTC WGs will be able to add to common components using the extension mechanism in a distinct namepace. And if a WG has specialised a component by adding elements, and if at a later stage the common component is updated with the same elements, the WG will be faced to a deprecation of its specialization when upgrading to the latest version of the common library.
The NMAN WG will decide whether to allow extension of the Management Component.

Item construction

- ☐ Can an item be made up of only aggregate components or a combination of aggregate and basic components?

From Johan: “`I think we should allow items to use both properties/basics and aggregates/constructs. If not we might only force constructions of dummy aggregates/constructs to hold the property/basic needed.`”

If both are allowed:

S - .	W
O	T .

If only aggregate components are allowed:

S - .	W - There might be dummy aggregates only created to hold one basic property.
O	T .

Decision:
An item can be made up of a combination of aggregate and basic components.

Note for IPTC:
We need to discuss what is a basic component exactly, and whether a set of attributes is a basic component.

CC Description Template

- ☐ Is the provided template approach a good base for further work?

From Johan: "`I think it looks as a good base. But if the Common Components group will handle it specifically as an excel form in that format is another question. But the list of information is a good base to start with.`"

Decision:
The approach is approved.

Added decision:
The consultants are asked to provide a description of the specific columns of the template.

Validation of the NewsMessage

- ☐ Enforce validation of NewsMessage with payload, or only the NewsMessage level?

Decision:
Validation of the NewsMessage will disregard the payload.

=== END of document ===