



Taxonomy Strategies

the business of organized information

PRI SM and PAM

Presentation for News Summit at XML 2003
Dec. 8, 2003

Ron Daniel, Jr.

rdaniel@taxonomystrategies.com

Outline

- ◆ PRISM (Publishing Requirements for Industry Standard Metadata)
 - Working Group History
 - Specification Overview
- ◆ PAM (PRISM Aggregator Message)
 - Business Goals and Requirements
 - Technical Walkthrough

PRISM History

- ◆ 1999 – Group Formed
 - Founding members: Adobe, Artesia, Cahners Business Information, Condé Nast, Getty Images, L A Burman Assoc., MarketSoft, Quark, Time Inc., Vignette, Wavo
 - Purpose: Metadata for magazine publishers
- ◆ 2001 – PRISM 1.0 Specification Released
- ◆ 2003 – PAM 1.0 Specification Released
- ◆ 2004? – Release PRISM 1.2

Overview of PRISM 1.2

General Purpose	Provenance	Dates and Times	Subject Description	Relations	Rights	Controlled Vocabs	Inline Markup
<p>dc: identifier title creator contributor description language format type</p> <p>prism: category</p>	<p>dc: publisher source</p> <p>prism: distributor edition issn issueName number startingPage Volume</p>	<p>prism: creationDate expirationDate modificationDate publicationDate releaseDate receptionDate</p>	<p>dc: coverage subject</p> <p>prism: event industry location person organization section</p>	<p>prism: isCorrectionOf hasCorrection isPartOf hasPart isVersionOf hasVersion isFormatO hasFormat References isReferencedBy isBasedOn isBasisFor isTranslationOf hasTranslation requires isRequiredBy isAlternativeFor hasAlternative</p>	<p>dc: rights</p> <p>prism: copyright expirationTime releaseTime rightsAgent</p> <p>prl: geography industry usage</p>	<p>pcv: broaderTerm code definition Descriptor label narrowerTerm relatedTerm synonym Vocabulary</p>	<p>pim: event industry location objectTitle organization person quote</p>

Additional PRISM Specifications

- ◆ PRISM 1.2 specifies
 - Metadata elements
 - Some conventions and vocabularies for the values of the elements
- ◆ OK if all you want to do is swap metadata.
- ◆ Publishers make money on content, not just metadata about it.
- ◆ PRISM group starting to define new formats that use PRISM elements as part of a larger purpose.

PAM (PRISM Aggregation Message)

- ◆ Publishers need electronic versions for
 - Secondary licensing
 - Magazine website
 - Low cost production of additional materials
- ◆ “Secondary licensing”
 - Magazine publishers license content to database aggregators for redistribution to customers
- ◆ Need to send content and metadata
- ◆ PAM format developed for that purpose

PAM Business Goals

- ◆ Publishers
 - Cut costs – revise current process for easier handling, better searching for future article research
 - Grow revenues – targeted feeds, higher rates from aggregators for easier-to-handle content?
 - Technology refresh
- ◆ Aggregators
 - Reduced costs for adding new feeds
 - Easier handling of publisher-specified rights
 - Technology Refresh
- ◆ Subscribers
 - Richer content – tables, charts, informative graphics, ...
 - More searchable – row/column selection, finer subject granularity
 - Better presentation

PAM Requirements

- ◆ Must be able to send content and metadata.
- ◆ Must be able to send article updates and corrections which follow magazine conventions.
- ◆ Must be relatively easy to implement without massive process changes.
- ◆ Should add table markup for better presentation, search, use of tabular data by customers.
- ◆ Should add in-line markup of people, places, things, ...
- ◆ Nice to allow for images, but not mandatory.
- ◆ Nice to allow for sophisticated DRM, but not mandatory.

Outline of PAM Design

- ◆ Mix PRISM metadata with XHTML for content
- ◆ Add new PAM namespace for 'glue'
 - message wrapper
 - corrections handling
- ◆ For XHTML
 - 'Yes' to tables
 - 'No' to scripts, events, styles, forms, frames, imagemaps, ...

Sample Markup – Overall Structure

- ◆ <pam:message>
- ◆ <pam:article>
- ◆ <xhtml:head>
 - mix of DC, PRISM, PAM
- ◆ </xhtml:head>
- ◆ <xhtml:body>
 - mix of XHTML and PRISM inline markup
- ◆ </xhtml:body>
- ◆ </pam:article>
- ◆ ...

Sample Markup – Elements for Descriptive Metadata

```
<prism:section>Portfolio</prism:section>
<dc:subject>Photography</dc:subject>
<dc:subject>Politics</dc:subject>
<dc:subject>Books</dc:subject>
<prism:person>Diana Walker</prism:person>
<prism:objectTitle>Public and Private: Twenty
  Years Photographing the Presidency</prism:objectTitle>
```

- ◆ Additional elements

```
<dc:description>
<prism:location>
<prism:organization>
<prism:event>
```

Sample Markup – Classes

- ◆Magazines have common idioms, which have numerous variations.

- ◆PAM uses the CLASS attribute for:

deck, byline, dateline,
sidebar, lead-in

which allows `<p>`, ``,
and `<div>` to markup the
underlying information.

- ◆Footnotes also use the CLASS attribute, with values of

fnRef, fnBody, fnKey

```
</head>
<body>
<h1>The Gong Goodbye</h1>
<p class="deck">Was game-show king
<pim:person>Chuck Barris</pim:person> a
hitman or just a hitmaker? The subject of
<pim:objectTitle>Confessions of
a Dangerous Mind</pim:objectTitle>
confronts his past lives.</p>
<p class='byline'>Karen Valby</p>
<p>Chuck Barris can't stand the sight of
himself. Old episodes of The Gong Show, the
daffy '70s talent show he created and hosted
with manic glee, turn his stomach. "I went
nuts up there on the stage to a point where
it was pitiful," he says. "I. Was. So.
Obnoxious."
```

Sample Markup – Article Content Markup

`<body>`

`<p>` Don't suppose that `<pim:person>`Robert Capa`</pim:person>`'s famous advice to photojournalists - `<pim:quote>`If your pictures aren't good enough, you're not close enough`</pim:quote>` - applies only to the battlefield. There are few tanks better armored these days than most celebrities, who are fully prepared to fend off all attempts to see any side of them but the faces they want you to see. And what is the President if not a celebrity operating at the highest levels of consequence? So one thing `<pim:organization>`TIME`</pim:organization>`'s `<pim:person>`Diana Walker`</pim:person>` can tell you is that a successful photographer is one who is close enough in all senses.`</p>`

DTD Structure – Types of Modules

- ◆ Driver
- ◆ For each namespace:
 - QName Module
 - Content Module
- ◆ One extra qname and content module to tie them together

DTD Structure - Driver

- ◆ HTML (and thus, XHTML) is bulky
- ◆ Modular XHTML makes it easy to exclude features
- ◆ PAM is intended for static textual content, so lots of stuff gets excluded
- ◆ Driver has to define two entities so XHTML framework knows where to get things:
 - `<!ENTITY % xhtml-qname-extra.mod ...`
 - `<!ENTITY % xhtml-model.mod ...`

```
<!-- Turn off browser UI stuff, as magazine content is basically static. -->
<!ENTITY % xhtml-events.module 'IGNORE'>      <!-- Events -->
<!ENTITY % xhtml-script.module 'IGNORE'>      <!-- Scripting -->
<!ENTITY % xhtml-csismap.module 'IGNORE'>     <!-- Client-side Image Map -->
<!ENTITY % xhtml-ssismap.module 'IGNORE'>     <!-- Server-side Image Map -->
<!ENTITY % xhtml-style.module 'IGNORE'>       <!-- Style Sheets -->
<!ENTITY % xhtml-inlstyle.module 'IGNORE'>    <!-- Inline Style attrib -->
<!ENTITY % xhtml-basic-form.module 'IGNORE'>  <!-- Basic Forms -->
<!ENTITY % xhtml-form.module 'IGNORE'>       <!-- Full-fledged Forms -->
...
```

DTD Structure – QName Modules

- ◆ QName Modules declare aliases for the element and attribute names we really want to use, so that namespaces can be overridden later.
- ◆ Example: Dublin Core QName Module

```
<!ENTITY % DC.prefix "dc" >
<![%DC.prefixed;[
<!ENTITY % DC.pfx "%DC.prefix;:" >
]]>
<!ENTITY % DC.pfx "" >
<!ENTITY % DC.title.qname "%DC.pfx;title" >
<!ENTITY % DC.contributor.qname
"%DC.pfx;contributor" >
...
```

- ◆ PAM has QName modules for DC, PRISM, PAM, and PIM
- ◆ QName Collection Module brings all the qname modules together into one place so it can be found through the definition of the `xhtml-qname-extra.mod` parameter entity.

DTD Structure – Content Modules

- ◆ Content modules declare the content models, using the aliases established in the QName modules.

```
<!ENTITY % PIM.content "( #PCDATA | %Inline.mix; )*" >
<!ELEMENT %PIM.event.qname; %PIM.content; >
<!ATTLIST %PIM.event.qname;
    %Common.attrib;
    %PIM.xmlns.attrib;
>
```

- ◆ PAM has content modules for DC, PAM, PRISM, and PIM. In addition, an overall content model has to be defined which is where the real mixing of the namespaces happens.
- ◆ The module for the overall content module ends up repeating parts of XHTML.

Repetition in Modular XHTML

```
<!ENTITY % InIPres.class
    "| %b.qname; | %big.qname; | %i.qname; | %small.qname;
    | %sub.qname; | %sup.qname; | %tt.qname;" >

<!ENTITY % Anchor.class "| %a.qname;" >
<!ENTITY % InISpecial.class "" >
<!ENTITY % Inline.extra " %PIM.class; " >
<!-- %Inline.class; includes all inline elements,
    used as a component in mixes -->
<!ENTITY % Inline.class
    "%InIStruct.class;
    %InIPhras.class;
    %InIPres.class;
    %I18n.class;
    %Anchor.class;
    %InISpecial.class;
    %Ruby.class;
    %Inline.extra;"
>
<!ENTITY % Inline.mix
    "%Inline.class;
    %Misc.class;"
>
```

Experiences with Modular XHTML

Pluses

- ◆ Modular XHTML Specification allowed us to mix PRISM, Dublin Core, PRISM Inline Markup, and new PRISM Aggregator Message namespaces and define the new format
- ◆ Most individual parts are simple
- ◆ Much quicker to develop than something of equivalent power
- ◆ Widespread use of HTML browsers

Minuses

- ◆ Result is complex
- ◆ Slow to validate
 - 'De-normalized' version generated automatically
- ◆ *Different tools handled file paths differently!*
- ◆ Need to repeat parts of XHTML declarations is ugly

Where to Get It

PRISM Working Group:

<http://www.prismstandard.org/>

PRISM and PAM Specification and DTDs:

http://prismstandard.org/Pam_1.0/PRISM_1.2h.pdf

http://prismstandard.org/Pam_1.0/

Why a DTD instead of XML Schema?

- ◆ XML Schemas provide many features not in DTDs, so why not start with them?
- ◆ Uses of XML fall into “data” and “documents”
 - XML Schema’s features primarily target the “data” view
 - Aggregation message format is for sending documents with some associated metadata
- ◆ DTDs meet certain requirements better
 - Fairly simple, widely understood - very important if format is to be adopted by a range of publishers and aggregators
 - Can up-convert later to Schemas –
- ◆ XHTML Tables widely understood and usable on client machines