SCM6 Standalone Multimedia Specification for MGH

Introduction

Silverchair has chosen the Book Interchange Tag Suite (BITS) as our standard for loading standalone multimedia XML wrappers into SCM6. BITS is a tag suite based on NLM's popular Journal Article Tag Suite, which is now a NISO standard. Although it was originally created for handling books, the BITS tag library offers more generic options for describing non-standard content.

Because this implementation of BITS is unique to Silverchair, the recommendations below represent a very limited selection of elements from BITS. If you are working with an item that appears to meet all the criteria of standalone multimedia, but do not see a way to represent it within the spec below, please contact us at xmlspecs@silverchair.com.

What is Standalone Multimedia?

Silverchair defines **standalone multimedia** (SAMM) as any image, video, link, or other file format that comes to us independent of another XML-based resource. Examples include metadata for videos hosted by a third-party (such as Brightcove) and XML wrappers for PDF objects. **Standalone multimedia must be loaded as an independent object, and may even constitute a content set that has no other journal, conference, book, or non-standard content counterpart**.

By contrast, **integral multimedia** consists of media that are part of the content of a larger XML resource. For example, a textbook may contain a number of images, which are called out in the text and should be imported as assets of that resource. This is also commonly called 'inline' multimedia.

Standalone multimedia may refer to other pieces of content even though they are independent pieces of content. In other words, they may have other XML-based resources as related objects. Related objects can be declared in the XML wrapper submitted with the multimedia (see "Related content," below).

Unique identifiers

Every file must receive a unique identifier for automatic replacements and related content linking to work. Many clients choose to assign DOIs to their digital objects and use these as the unique identifier, but in the absence of these, Silverchair relies on the publisher to create and manage unique publisher IDs. **MGH does not use DOIs for SAMM.**

Zipline will use the multimedia's DOI or publisher ID to automatically identify duplicate articles and make replacements To replace content, the value of the publisher ID **must be unique and remain stable** in order for the revised content to correctly replace the originals on the site.

Use the following nodes to designate the multimedia's unque ID:

ID	node
DOI	`<book-part-id book-part-id-type="doi">`
publisher ID	`<book-part-id book-part-id-type="publisher-id">`

For content without DOIs, please consult with Silverchair if there is any question about how best to assign IDs to new content.

For MGH, we request that the publisher-ID have a prefix equal to the site's abbreviation. So, a video on Emergency Medicine would start aem.######.

We also ask that, whatever your system of choice for assigning IDs, please use it for assigning IDs to filename and image naming conventions as well.

Basic file structure

Below is a very basic template for structuring a standalone multimedia XML wrapper. Keep in mind that the overall structure of your XML file will depend upon the kind of media you are submitting, and the metadata requirements for that object.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE book-part-wrapper PUBLIC "-//NLM//DTD BITS Book Interchange DTD v1.0
20131225//EN" "http://jats.nlm.nih.gov/extensions/bits/1.0/BITS-book1.dtd">

 <book-part-wrapper
 content-type="multimedia"
 dtd-version="1.0" xml:lang="en"
 xmlns:mml="http://www.w3.org/1998/Math/MathML"
 xmlns:xlink="http://www.w3.org/1999/xlink"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 >

 <book-meta>
  <book-title-group>
   <book-title>Silverchair Multimedia of Demonstrable Diseases</book-title>
  </book-title-group>
 </book-meta>
 <book-part book-part-type="object">
  <book-part-meta>
   <book-part-id book-part-id-type="publisher-id">jdd47252.fakevid</book-part-id>
   <subj-group subj-group-type="category-Multimedia Categories">
    <subject>Bedside and Office-Based Procedures</subject>
    <subj-group subj-group-type="category-Multimedia Categories">
     <subject>Cardiovascular</subject>
    </subj-group>
   </subj-group>
   <title-group>
    <label>Video 43</label>
    <title>Test Video: F18-Fluorodeoxyglucose–Positron Emission Tomography/Computed
Tomography Screening in Li-Fraumeni Syndrome</title>
   </title-group>
   <contrib-group>
    <contrib contrib-type="author">
     <name>
      <surname>Hassing</surname>
      <given-names>Kirsten</given-names>
      <prefix>Ms.</prefix>
      <suffix>Jr.</suffix>
     </name>
     <degrees>PhD</degrees>
     <address>
      <addr-line>1 Jefferson Lane</addr-line>
      <addr-line>Charlottesville, VA 22903</addr-line>
      <email>khassing@example.com</email>
     </address>
     <author-comment content-type="disclosure">
      <p>The author was funded by…</p>
     </author-comment>
     <!-- Cross reference to the author's affiliation. -->
     <xref ref-type="aff" rid="aff1" />
    </contrib>
    <contrib contrib-type="author">
     <name>
      <surname>Keefe</surname>
      <given-names>Lindsey</given-names>
     </name>
     <degrees>PhD</degrees>
     <role>Contributing Author</role>
     <xref ref-type="aff" rid="aff1" />
    </contrib>
   </contrib-group>
   <aff id="aff1">
    <label>1</label>Silverchair Information Systems, Charlottesville, VA 22902.
   </aff>
   <pub-date date-type="pub" publication-format="electronic">
    <day>21</day>
    <month>9</month>
    <year>2012</year>
   </pub-date>
   <supplementary-material content-type="thumbnail">
    <graphic xlink:href="test_video_thumb.jpg" /> <!-- This is the filename of the
packaged asset -->
   </supplementary-material>
   <permissions>
    <copyright-statement>Copyright 2010 American Medical Association. All Rights
Reserved. Applicable FARS/DFARS Restrictions Apply to Government
Use.</copyright-statement>
    <copyright-year>2010</copyright-year>
   </permissions>
   <self-uri xlink:href="http://bcove.me/1xyjm06i" content-type="brightcove" />
   <related-object document-type="book" document-id-type="isbn13"
document-id="978-0-07-179071-0" object-type="chapter" object-id-type="publisher-id"
object-id="Guyatt_ch6" link-type="34"/>
   <abstract abstract-type="caption">
    <p>Test Description: Rotating 3-dimensional F18-fluorodeoxyglucose–positron
emission tomography (FDG-PET) whole-body maximal intensity projection image of an
individual with Li-Fraumeni syndrome. The arrowhead indicates an area of focal
increased FDG uptake at the gastroesophageal junction. The lesion was diagnosed as a
3-cm stage IIA esophageal adenocarcinoma (T3N0M0). There is normal physiological FDG
uptake in the brain, oropharynx, heart, urinary collecting system, and genitalia</p>
```

```
   </abstract>
   <abstract abstract-type="short-description">
    <p>Test Short Descrip: Rotating 3-dimensional F18-FDG-PET whole-body maximal
intensity projection image of an individual with Li-Fraumeni syndrome.</p>
   </abstract>
   <custom-meta-group>
    <custom-meta>
     <meta-name>duration</meta-name>
     <meta-value>00:05</meta-value>
    </custom-meta>
    <custom-meta>
     <meta-name>height</meta-name>
     <meta-value>600</meta-value>
    </custom-meta>
    <custom-meta>
     <meta-name>width</meta-name>
     <meta-value>168</meta-value>
    </custom-meta>
    <custom-meta>
     <meta-name>player</meta-name>
     <meta-value>1377024934001</meta-value>
    </custom-meta>
    <custom-meta>
     <meta-name>playerkey</meta-name>
     <meta-value>AQ~~,AAAAyT9oH9k~,DTe1zot21E1nSpajEYjJB2xxrjlZGznd</meta-value>
    </custom-meta>
    <custom-meta>
     <meta-name>videoid</meta-name>
     <meta-value>1445545584001</meta-value>
    </custom-meta>
   </custom-meta-group>
   </book-part-meta>
  </book-part>
</book-part-wrapper>
```

XML declaration

The following declarations are required at the top of each XML file:

- XML version
- Character encoding
- Document Type Definition

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE book-part-wrapper
  PUBLIC "-//NLM//DTD BITS Book Interchange DTD v1.0 20131225//EN"
  "http://jats.nlm.nih.gov/extensions/bits/1.0/BITS-book1.dtd">
```

XML encoding

XML must be well-formed conformant XML according to the W3C XML Recommendation, fully tagged, and valid to the BITS schema. XML must also meet the requirements in this Silverchair specification.

Character encoding of UTF–8 is required to ensure that characters are translated properly upon import into the SCM6 database. Content XML may also contain numeric character references (e.g. `&#345;` or `&#x159;`) or the character entity references that BITS supports for special characters. Predefined entities for the ampersand, less-than, and greater-than should be used where the intent is to display the actual characters ampersand (&), less-than (<), or greater-than (>) respectively; the predefined entities should not be used where those characters are part of markup or encodings.

Main elements

Every standalone multimedia record should have the root element of `<book-part-wrapper>`. Within the wrapper, there are three main sections, of which two are important for this content type.

- `<book-meta>`: information about the larger content set to which the multimedia belongs
- `<book-part>`/`<book-part-meta>`: information about the multimedia itself
- `<body>`: in BITS, the object’s content. This should remain empty for multimedia.

book-part-wrapper

The root element must be `<book-part-wrapper content-type="multimedia">`. The @content-type attribute is required and the value must be “multimedia”.

book-meta

Even though the object you are submitting meets the definition of “standalone,” it should belong to a general content set for the site (e.g. images, videos, PDFs). This general content set might be a browse-able feature on the site (perhaps there is a “Browse Multimedia” option in the navigation) or it might be accessible only through its declared related content objects. Either way, the `<book-meta>` element is set aside for telling our system which content set the object is a part of.

See the snippet from the example above:

```
         <book-meta>
            <book-id book-id-type="publisher-id">Insert Publisher ID here</book-id>
            <book-title-group>
                <book-title>Title of Content Set</book-title>
            </book-title-group>
        </book-meta>
```

In this example, the `<book-id>` contains a publisher-assigned identifier for the content set. For example, if we were loading a set of standalone images for an image browse feature on silverchair.com, a unique ID for that content set could be “silverchair.images”. It is up the publisher to create and maintain unique identifiers for their content sets. However, Silverchair usually works with clients to find a nomenclature that works best for everyone.

Ideally, the content set would have its own DOI, or “digital object identifier.” If this is the case for your content set, DOI is the preferred value for the `<book-id>`. In that case the sample would be changed to read:`<book-id book-id-type="doi">10.1000/182</book-id>`. Each object that is part of this content set would receive its own DOI based on the publisher’s prefix and, most likely, the suffix of the content set in question (“10.1000/182.001,” to continue the example).

You are welcome to submit multiple identifiers for a given object, but keep in mind that DOI is preferred, and will serve as the unique identifier in our system whenever provided.

The other required element in `<book-meta>` is `<book-title-group>`, and its child element, `<book-title>`. This is simply a naming convention for the overall content set of which the multimedia object is a part. Please keep in mind that, if the content set is meant to be browsed on your site, the value of `<book-title>` **should reflect how you want that content set to appear on your site**. To go back to the previous example of the image browse feature for silverchair.com, the XML might say`<book-title>Images</book-title>`, which is exactly how the user will see that option on the navigation bar.

Summary:

- `<book-id>` Required element, unique identifier of content set
- @book-id-type Required attribute. Possible values: “publisher-id” or “doi”.
- `<book-title-group>`/`<book-title>` Required elements. Name of content set of which the object is a part.

book-part

This element provides information on the individual object being submitted. Since a standalone multimedia wrapper can only accompany one multimedia object, the value of this element should remain the same across your content sets.

```
<book-part book-part-type="object" xml:lang="en">
```

@book-part-type is required, with the value always “object” for standalone multimedia. The @xml:lang is optional, with the default value being “en” for English if not provided. If you are submitting an object in a language other than English, please**follow the the ISO 639–1 codes** for the representation of names of languages.

Summary:

- `<book-part book-part-type="object">`: Required element and attribute for all standalone multimedia wrappers
- `@xml:lang`: Optional attribute. Value should come from ISO 639–1 Codes.

book-part-meta

This element should be the most robust of the standalone multimedia wrapper. It contains all metadata directly related to the object itself.

book-part-id

Just as the larger content set must have a unique, publisher-assigned ID, the object must also carry a unique ID. See Unique identifiers for more detail.

title-group

Every object must contain a title. If your object has no discernible title at all, please work with Silvechair to choose a default value for our system. The `<label>` element is not used for standalone multimedia and will not be imported.

contrib-group

If the object has an author, editor, or creator associated with it, that information should be stored in this element.

permissions

If there is copyright information for the object, it should be stored in the `<permissions>` element. Please see the BITS tag library for examples, and the summary below for a list of accepted elements and attributes. If you have permissions related information that does not fit this mold, please contact your Project Manager.

Example:

```
<permissions>
    <copyright-statement>Copyright © 2008 by Silverchair. All rights reserved.</copyright-statement>
    <copyright-year>2008</copyright-year>
    <copyright-holder>Silverchair</copyright-holder>
</permissions>
```

The `<license>` element contains the set of conditions under which people are allowed to use an article, or other license-related information or restrictions.

- `@license-type` attribute: The access type to be stored in the database. The following licenses will make the content available on the site without authentication. Other license types will default to private.

license type	description
`@license-type="free"`	Free article

- `<license-p>`: A paragraph of text within the description of a `<license>`.

Example:

```
 <permissions>
    …
    <license license-type="free"
xlink:href="http://creativecommons.org/licenses/by/2.0/">
    <license-p>This is an open-access article distributed under the terms of the
Creative Commons Attribution License, which permits unrestricted use, distribution,
and reproduction in any medium, provided the original work is properly
cited.</license-p></license>
</permissions>
```

self-uri

The `<self-uri>` element is the reference to the object being submitted, such as the file name or URL of the multimedia file itself. The @xlink:href value should be exactly the same as the object's filename, if the multimedia is a file submitted with the Zipline package.

For file names, provide only the literal file name relative to the directory specified for the file in the appropriate packaging guidelines. For example, an image stored directly in the Assets directory should use `xlink:href="image.png"` without a file path preceding the file name.

The list of accepted values for the self-uri @content-type:

@content-type	type of value in @xlink:href
image	file name
video	file name
audio	file name
flash	file name

podcast	file name
pdf	file name
excel	file name
powerpoint	file name
word	file name
url	URL
brightcove	URL
amazon	URL
vimeo	URL
YouTube	URL

abstract

Store captions in the `<abstract>` element. Captions are optional, and there can be more than one.

The @abstract-type attribute is required. Possible values are:

- short-description
- caption

Which values to use are site-specific; Silverchair will inform you as to which to use and what to include in them.

Example:

```
 <abstract abstract-type="caption">
    <p>Test Description: Rotating 3-dimensional F18-fluorodeoxyglucose–positron
emission tomography (FDG-PET) whole-body maximal intensity projection image of an
individual with Li-Fraumeni syndrome. The arrowhead indicates an area of focal
increased FDG uptake at the gastroesophageal junction. The lesion was diagnosed as a
3-cm stage IIA esophageal adenocarcinoma (T3N0M0). There is normal physiological FDG
uptake in the brain, oropharynx, heart, urinary collecting system, and genitalia</p>
</abstract>
<abstract abstract-type="short-description">
    <p>Test Short Descrip: Rotating 3-dimensional F18-FDG-PET whole-body maximal
intensity projection image of an individual with Li-Fraumeni syndrome.</p>
</abstract>
```

custom-meta

`<custom-meta-group>` is used for videos that are hosted by third parties, such as Brightcove, Amazon Web Services or Ooyala.`<custom-meta>` is used to provide information about the video’s duration, height and width along with any IDs or keys required to play the video. Within the `<custom-meta-group>` are as many `<custom-meta>` sections as needed to convey required information.

The format is as follows:

```
<custom-meta-group>
            <custom-meta>
                <meta-name>duration</meta-name>
                <meta-value>00:05</meta-value>
            </custom-meta>
            <custom-meta>
                <meta-name>height</meta-name>
                <meta-value>600</meta-value>
            </custom-meta>
            <custom-meta>
                <meta-name>width</meta-name>
                <meta-value>168</meta-value>
            </custom-meta>
            <custom-meta>
                <meta-name>player</meta-name>
                <meta-value>0101010101</meta-value>
            </custom-meta>
            <custom-meta>
                <meta-name>playerkey</meta-name>
                <meta-value>playerkeyvalue</meta-value>
            </custom-meta>
            <custom-meta>
                <meta-name>videoid</meta-name>
                <meta-value>1111111111</meta-value>
            </custom-meta>
        </custom-meta-group>
```

Brightcove

Custom metadata required for Brightcove is:

- duration
- height
- width
- player
- playerkey
- videoid

Other Specific Guidelines

Publication dates

Include the publication date for the multimedia in a `<pub-date>` node in the `<book-part-meta>`. The node must include the`@date-type="pub"` and `@publication-format="electronic"` attributes to be imported.

To supply a date, the following options are available:

- Supply at least a `<month>` and `<year>` in numerical form (`<day>` is optional).
- Supply a date in an `@iso-8601-date` attribute in the format `YYYY-MM-DD`.

Dates can be provided in non-numerical form (e.g. `<month>January</month>`). The `<season>` tag can also be used instead of day and month (e.g. `<season>Fall</season>` or `<season>1st Quarter</season>`). If the date is provided in a form other than numerical month and year, then the `@iso-8601-date` attribute must be included to supply a numerical date. In this case, the day and month or season provided can be used for display, while the `@iso-8601-date` value is used as the underlying article date by the system.

Hierarchical categories

Multimedia can be assigned to categories (i.e. subjects or groupings of any sort) that span each site. Use the`<subj-group subj-group-type="category">` and `<subject>` elements to designate categories for the multimedia. Each category should be placed in a `<subject>` element within the `<subj-group>` element.

SCM6 does not create categories on the fly. They must be set up in SCM6 before importing. This means that if you wish to create a new category that has not been used before on the site, please contact Silverchair through Salesforce and your project manager to make arrangements.

SCM6 supports hierarchical categories, which allows for parent-child relationships between categories. When assigning a child-level category to content, the parent categories must also be supplied in case more than one subcategory shares the same name. Zipline can currently import up to a 2-level hierarchy. The example below shows an article with a category structure of:

- Wireless Sensor Networks
    - Alternative Energy Harvesting Systems

```
 <article-categories>
        <subj-group subj-group-type="category">
            <subject>Wireless Sensor Networks</subject>
            <subj-group subj-group-type="category">
                <subject>Alternative Energy Harvesting Systems</subject>
            </subj-group>
        </subj-group>
</article-categories>
```

Since BITS only allows nested `<subj-group>` elements after the `<subject>` elements, use multiple top-level`<subj-group subj-group-type="category">` nodes to designate multiple parent-child categories.

If multiple facets of organization are required and the same category name exists for multiple facets, append a hyphen and the the category type to the end of the `@subj-group-type` value. For example, if the following categories have been set up in SCM6 for the same site:

- Research (with category type of Book Categories)
- Research (with category type of Multimedia Categories)

then specify the latter in the XML by including:

```
 <subj-group subj-group-type="category-Multimedia Categories">
    <subject>Research</subject>
</subj-group>
```

`Here is a spreadsheet of all the MGH multimedia categories and their hierarchical system. - Multimedia Categories for MGH.xlsx`

Related article links

`<related-article>` is used for linking content to an article. For example, you can connect an author interview to the author’s article. Related articles must be limited to articles within the publisher’s content. Place the `<related-article>` node in the`<book-part-meta>`; `<related-article>` elements in the body of the multimedia will not generate links between articles. The XML containing the `<related-article>` and the target content can be imported in any order in relation to each other.

There are two methods for indicating which article to link to:

1. DOI of the related article
2. Volume and first page number of the related article

Of these, DOI is strongly preferred. Volume and page number can be used for articles that do not have an assigned DOI, but links may not always be created if the system cannot find a match or finds more than one match and cannot determine which is correct. This can happen if there are multiple articles on a single page, e.g. multiple letters to the editor.

To link via DOI, two methods are available:

- Use `@elocation-id`. The following attributes are required:
  - `@related-article-type`
  - `@elocation-id="[DOI of the related article]"`
- Use `@xlink:href`. The following attributes are required:
  - `@related-article-type`
  - `@ext-link-type="doi"`
  - `@xlink:href="[DOI of the related article]"`

To link via volume and page number, the following attributes are required:

- `@journal-id-type="issn"`
- `@journal-id`—set to the ISSN of the journal
- `@vol`
- `@page`—set to first page of the related article
- `@related-article-type`

Related articles that use the `@xlink:href` attribute to identify the target will be ignored by the SCM6 system, unless the target value is a DOI and the element includes the `@ext-link-type="doi"` attribute. Currently supported values for`@related-article-types` are listed below. Contact your Project Manager if other types are needed.

Link types

Link type	JATS definition
commentary	Used in an article to name its associated commentary or editorial.
companion	Used in an article to name a companion (related or sibling) article.
corrected-article	Used in a correction or erratum to name the article being corrected.
letter	Names a letter to the publication or a reply to such a letter.
retracted-article	Used in a retraction to name the article being retracted.

Related content

You may link multimedia to other content or objects within the content.

Use the `<related-object>` element in the `<book-part-meta>` to designate a relation to another piece content. Use the`@document-type`, `@document-id-type`, and `@document-id` attributes to designate the type of content and document to target. To link to an object (e.g. figure, table) within a piece of content, also include the `@object-type`, `@object-id-type`, and `@object-id`attributes with the appropriate values. You must designate both the document and the object when linking to an object.

Links to the following content types can be created from the non-standard content. The XML containing the `<related-object>`and the target content can be imported in any order in relation to each other.

Target document/content type	tagging format
Non-standard	`<related-object document-type="non-standard" document-id-type="publisher-id" document-id="gbos123-12345"/>`
Book (using ISBN 13)	`<related-object document-type="book" document-id-type="isbn13" document-id="1-234-56790-3"/>`
Book (using ISBN 10)	`<related-object document-type="book" document-id-type="isbn10" document-id="7412589636"/>`
Book (using eISBN)	`<related-object document-type="book" document-id-type="eisbn" document-id="1-234-56790-3"/>`

To target objects within non-standard content, additionally include the following attributes:

Target object	tagging format
Main Division (in non-std)	`< … object-type="main-division" object-id-type="publisher-id" object-id="publisherID-value"/>`
Part (in book)	`< … object-type="part" object-id-type="publisher-id" object-id="publisherID-value"/>`

Chapter (in book)	`< … object-type="chapter" object-id-type="publisher-id" object-id="publisherID-value"/>`
Section	`< … object-type="sec" object-id-type="publisher-id" object-id="publisherID-value"/>`
Figure	`< … object-type="fig" object-id-type="publisher-id" object-id="publisherID-value"/>`
Table	`< … object-type="table" object-id-type="publisher-id" object-id="publisherID-value"/>`
Video	`< … object-type="video" object-id-type="publisher-id" object-id="publisherID-value"/>`

For example`<related-object document-type="non-standard" document-id-type="publisher-id" document-id="non-standard-object-123-12345" object-type="fig" object-id-type="publisher-id" object-id="publisherID-value"/>`

However, links to objects within non-standard content will not be maintained if the target content is replaced.

Use the `@link-type` attribute to designate the link type. The link-type must match a value in our system.

If the link-type is directional in nature (e.g. A is a correction for B vs. B is corrected by A), the link-type should follow the logic:

“The content in **this XML** is a/the **[linkType]** of/for the **target** content.”

Image Files

Images should be no larger than 1800 pixels wide or high at 72 ppi, and no more than 2MB. Silverchair’s preferred image formats (in order) are:

- png
- gif
- jpg

Whenever possible, please provide PNGs.

Do not put spaces in the filenames, use underscore tags. Right Ventricle MI should be Right_Ventricle_MI.

As mentioned above in the section on packaging requirements, these files should be stored in a separate **assets** directory in the .zip file submitted for loading.

Packaging Guidelines

In order to be loaded properly in our system, the converted files need to be sent to Silverchair in a .zip file, with two directories:

- **XML** should contain the XML file
- **Assets** should contain all other related files (image files, PDFs)

Note that standalone multimedia that is a URL will not have any assets.

Multiple multimedia belonging to the same content set can be in a single zip file. For instance, if you have five standalone PDFs to load, you can put all five PDFs into a single Assets folder and all five XML files into a single XML folder and zip those up into a single zip file. There is no need to load all five separately.