COMBINE Archive Specification

Nicolas Le Novère
n.lenovere@gmail.com
Babraham Institute
Babraham Campus Cambridge
Cambridge , UK

Frank T. Bergmann
fbergmann@caltech.edu
Computing and Mathematical Sciences
California Institute of Technology
Pasadena, CA, US

Version 1, Draft

April 2, 2014

This is a working draft of the specification for the COMBINE archive and not a normative document.
Please send feedback to the COMBINE mailing list at combine-discuss@mbine.org.

This release of the specification is available at
http://co.mbine.org/documents/archive

Contents

1 Introduction and motivation 3

1.1 Document conventions 3

2 Background 4

2.1 Past work on this problem or similar topics 4

3 Proposed syntax and semantics 5

3.1 The archive format 5

3.2 COMBINE archive extensions 5

3.3 Content of the archive 5

3.4 Namespace URI and other declarations necessary 5

3.5 Primitive data types 6

3.6 The `manifest.xml` file and the **OmexManifest** class 7

3.7 The **Content** class 7

3.8 Advised format for the archive metadata 8

4 Illustrative examples of the syntax 11

5 Future development 12

5.1 Linking to external documents 12

5.2 Cross References between entries 12

5.3 Alternative versions of the archive metadata 12

Acknowledgments 13

References 14

1 Introduction and motivation

Computational modeling is an increasingly interdisciplinary field, different aspects come together that need to be stored in a cohesive unit. When exchanging a model, it increasingly becomes an issue that not all relevant files are exchanged along with it. Many different approaches have been taken to solve this issue, such as folder based project structures, or special versions of version control systems. Unfortunately these approaches are not as easy to support for tool authors as a single file based solution would.

This specification describes the "COMBINE archive" format. A *COMBINE archive* is a single file containing the various documents (and in the future, references to documents), necessary for the description of a model and all associated data and procedures. This includes for instance, but not limited to, simulation experiment descriptions in SED-ML, all models needed to run the simulations in SBML and their graphical representations in SBGN-ML.

The COMBINE archive aims to augment these efforts by standardizing a meta file format (or manifest) that describes what files all belong to computational models, as well as a description format and a convention for bundling the information together.

Details of earlier independent proposals are provided in Section 2.

1.1 Document conventions

Following the precedent set by other COMBINE specification documents, we use UML 1.0 (Unified Modeling Language; Eriksson and Penker 1998; Oestereich 1999) class diagram notation to define the constructs provided by this package.

We also use the following typographical conventions to distinguish the names of objects and data types from other entities; these conventions are identical to the conventions used in the other COMBINE specification documents:

AbstractClass: Abstract classes are classes that are never instantiated directly, but rather serve as parents of other object classes. Their names begin with a capital letter and they are printed in a slanted, bold, sans-serif typeface. In electronic document formats, the class names defined within this document are also hyperlinked to their definitions; clicking on these items will, given appropriate software, switch the view to the section in this document containing the definition of that class.

Class: Names of ordinary (concrete) classes begin with a capital letter and are printed in an upright, bold, sans-serif typeface. In electronic document formats, the class names are also hyperlinked to their definitions in this specification document.

`SomeThing`, `otherThing`: Attributes of classes, data type names, literal XML, and generally all tokens *other* than UML class names, are printed in an upright typewriter typeface. Primitive types defined begin with a capital letter; the COMBINE archive also makes use of primitive types defined by XML Schema 1.0 (Biron and Malhotra, 2000; Fallside, 2000; Thompson et al., 2000), but unfortunately, XML Schema does not follow any capitalization convention and primitive types drawn from the XML Schema language may or may not start with a capital letter.

For other matters involving the use of UML and XML, we follow the conventions used in other COMBINE specification documents.

2 Background

2.1 Past work on this problem or similar topics

The COMBINE Archive has been formed out of the SED-ML Archive, that was first introduced in the SED-ML mailing list in 2009, and briefly described in SED-ML Level 1 Version 1 Waltemath et al. (2011). The SED-ML Archive, addressed the need of prividing a way to include all relevant models used in a SED-ML simulation experiment along with the experiment specification. The SED-ML archive is basically the convention to provide inside a ZIP file (with extension .sedx) a SED-ML document with the same base name as the extension. That was sufficient for SED-ML.

The COMBINE Archive extends this approach by a metafile, that allows to include provenance information with the archive, and a specification of the content types of files included in the COMBINE Archive (the manifest).

This approach also makes it easy to extend approaches using a distributed version control system (like PMR2) to export COMBINE archives, by generating a manifest file during export.

This specification has been created out of the many discussions that took place on the combine-discuss mailing list. The primary threads are located here:

- http://listserver.ebi.ac.uk/pipermail/combine-discuss/2011-October/thread.html
- http://listserver.ebi.ac.uk/pipermail/combine-discuss/2011-November/000016.html
- http://listserver.ebi.ac.uk/pipermail/combine-discuss/2012-January/thread.html

The discussion continued at the last Hackathons, HARMONY 2012 where for the first time a larger audience discussed a preliminary proposal put together by Richard Adams, Frank T. Bergmann and Nicolas Le Novère. Further discussions were held at HARMONY_2013.

3 Proposed syntax and semantics

In this section, we define the syntax and semantics of the COMBINE Archive. We expound on the various data types and constructs defined, then in Section 4 on page 11, we provide complete examples of using the constructs in an example archive.

3.1 The archive format

The COMBINE archive is a "zip" file Wikipedia (2014). Zip is a file format used for data compression and archiving. A zip file contains one or more files that have been compressed, to reduce file size, or stored as is. The technical specification of the ZIP format is available from the PKWARE website PKWARE Inc. (2012).

3.2 COMBINE archive extensions

The extension for the COMBINE archive is `.omex`, for "Open Modeling EXchange format". If a COMBINE archive with that extension is encountered, an application should present the user with the *active document* (see also Section 3.7).

Additional extensions are being used, so that applications can quickly open a specific file from the COMBINE archive. This provides users with a consistent behavior in that each time such a file is opened, the corresponding same main file opens. The extensions in use are:

- `.sedx` - SED-ML archive
- `.sbex` - SBML archive
- `.cmex` - CellML archive
- `.neux` - NeuroML archive
- `.phex` - PharmML archive

3.3 Content of the archive

The archive contains:

1. a manifest file, called `manifest.xml`, always located at the root of the archive, that describes the location and the type of each data file contained in the archive (including itself).

   The location of those files is defined by a relative URI. In the current version of the COMBINE archive, all the files described must be included in the archive itself. It is envisioned that in the future the manifest could list files located elsewhere, using valid and resolvable http URIs.

2. a metadata file, called `metadata.*` (where `*` means the suitable file extension) containing clerical information about the various files contained in the archive, and the archive itself.

3. all the remaining files necessary to the model and simulation project.

3.4 Namespace URI and other declarations necessary

The COMBINE archive defines a namespace URI that allows to uniquely identify the manifest. The following is the namespace URI for this version of the COMBINE archive manifest:

“`http://identifiers.org/combine.specifications/omex-manifest`”

3.5 Primitive data types

The COMBINE archive uses the XML Schema 1.0 data types (Biron and Malhotra, 2000). More specifically we make use of `integer`, `double`, and `string`.

3.6 The manifest.xml file and the OmexManifest class

At the root of the COMBINE archive stands one file, with the prescribed name manifest.xml. This file contains an instantiation of the **OmexManifest** class.

It contains a number of **Content** children, one of which represents the manifest itself.

Note that a valid manifest needs to have at least one entry, that of the manifest itself, but may contain as many entries as needed.

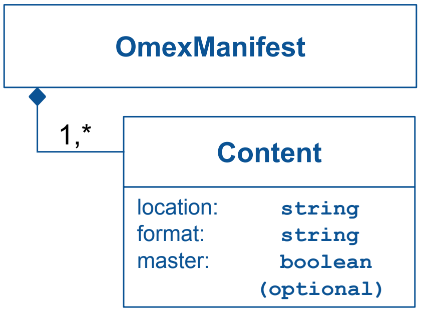

Figure 1: *A UML representation of the Manifest. Each manifest contains a number of Content elements.*

3.7 The Content class

The **Content** class represents an entry in the **OmexManifest** and by extension a file in the *COMBINE archive*. It consists of two required attributes: location and format and the optional attribute master

The location *attribute*

The location attribute is a required attribute of type string. It represents a relative location to an entry within the archive. The root of the archive is represented by a dot '.'.

The format *attribute*

The format is a required attribute of type string. It indicates the file type of the **Content** element. The values of the format attribute fall in two categories. Either the format denotes one of the COMBINE standards, in which case the format will begin with its identifiers.org url. Otherwise the format will represent a MIME type.

Using identifiers.org allows to unambiguously define the COMBINE standard, and even its level and version. For example, the identifier: http://identifiers.org/combine.specifications/sbml would denote the **Content** element as being encoded in the SBML format. That is usually sufficient, as tools supporting one level of SBML usually support others as well. However, if the software exporting the COMBINE archive wanted to be more precise, it could specify that it is an SBML Level 2 document with

http://identifiers.org/combine.specifications/sbml.level-2

or even declare its Version with

http://identifiers.org/combine.specifications/sbml.level-2.version-3.

The master *attribute*

The master is an optional attribute of type boolean. It represents a hint, that a certain file is to be used first when processing the content of an archive. Are top model description in a composed model, calling the various submodels; simulation description, calling the different model descriptions and data sources used in the experiment. At most one content element per archive may have its master attribute set to true. For example in the snippet below, it is the SED-ML element with `location="./simulation.xml"` that a software program should first present to their users.

```
<?xml version="1.0" encoding="utf-8"?>
<omexManifest xmlns="http://identifiers.org/combine.specifications/omex-manifest">
    <content location="./manifest.xml"
             format="http://identifiers.org/combine.specifications/omex-manifest"/>
    <content location="./model/model.xml"
             format="http://identifiers.org/combine.specifications/sbml"/>
    <content location="./simulation.xml" master="true"
             format="http://identifiers.org/combine.specifications/sedml"/>
    <content location="./article.pdf"
             format="application/pdf"/>
    <content location="./metadata.rdf"
             format="http://identifiers.org/combine.specifications/omex-metadata"/>
</omexManifest>
```

To simplify the identification of the active document the file extensions (see also Section 3.2) can also be used. For the example above, where a SED-ML document is being marked as the master document, the recommended extension would be `.sedx`.

3.8 Advised format for the archive metadata

One can include any type of file in a COMBINE archive, and therefore any type of metadata format. However, in the interest of interoperability, and to ease the development of software support for metadata, a recommended format is provided as part of the specification of the archive.

The recommended format is based on several standards developed by other organisations:

- The Resource Description Format of the W3C, in particular its terms:
  - `RDF`,
  - `Description`,
  - `Bag`,
  - `li`.
- vCard 4 (Perreault (2011)), a file format standard for electronic business cards, in particular its terms:
  - `hasName`,
  - `family-name`,
  - `given-name`,
  - `hasEmail`,
  - `organization-name`.

  More information on how to use vCard in RDF can be found on the W3C website[1].
- Metadata terms[2] of the Dublin Core Metadata Initiative, in particular the terms:
  - `description`,
  - `creator`,
  - `created`,
  - `modified`,
  - `W3CDTF`.

[1] http://www.w3.org/TR/vcard-rdf/

[2] http://dublincore.org/documents/dcmi-terms/

More information on the use of Dublin Core in RDF can be found on the Dublin Core website[3].

More information is available about the definition of the date format used within `dcterms:created` and `dcterms:modified` elements[4].

[3] http://dublincore.org/documents/dc-rdf/
[4] http://www.w3.org/TR/NOTE-datetime

Users of the COMBINE standards may already be familiar with this approach, as it is also taken by the Systems Biology Markup Language. Note however that the format here slightly differs from the controlled annotations of SBML. The differences have been made to address inconsistencies in the SBML vCard specification and to follow W3C recommendations [4] . The changes are:

- "n" becomes "hasName"
- "Family" becomes "family-name"
- "Given" becomes "given-name"
- "EMAIL" becomes "hasEmail"
- "Orgname" becomes "organization-name"

A *COMBINE archive* can include multiple metadata elements. To identify that a particular **Content** element is being annotated, the `rdf:about` attribute should use the same value that is also used in `location` of the **Content** element.

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:dcterms="http://purl.org/dc/terms/"
         xmlns:vCard="http://www.w3.org/2006/vcard/ns#">
   <rdf:Description rdf:about="./simulation.xml">
   ...
   </rdf:Description>
</rdf:RDF>
```

The example above signifies that a content element with `location="./simulation.xml"` is described. A complete example of the metadata related to a simulation description contained in a COMBINE archive is described in Section 4.

[4] http://www.w3.org/TR/vcard-rdf/

4 Illustrative examples of the syntax

This section contains a worked example showing the encoding of a model and associated data in a COMBINE archive. First the **OmexManifest** , that includes five entries. One of these entries is the **OmexManifest** itself, it has a fixed location `location="./manifest.xml"`. Additionally the archive includes an SBML model, a SED-ML description. It also includes a PDF file that is specified through its MIME type. Finally associated meta information for clerical data is included in `location="./metadata.rdf"`.

```
<?xml version="1.0" encoding="utf-8"?>
<omexManifest xmlns="http://identifiers.org/combine.specifications/omex-manifest">
    <content location="./manifest.xml"
             format="http://identifiers.org/combine.specifications/omex-manifest"/>
    <content location="./model/model.xml"
             format="http://identifiers.org/combine.specifications/sbml"/>
    <content location="./simulation.xml"
             format="http://identifiers.org/combine.specifications/sedml"/>
    <content location="./article.pdf"
             format="application/pdf"/>
    <content location="./metadata.rdf"
             format="http://identifiers.org/combine.specifications/omex-metadata"/>
</omexManifest>
```

Here a complete example, on how the clerical data could be encoded:

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:dcterms="http://purl.org/dc/terms/"
         xmlns:vCard="http://www.w3.org/2006/vcard/ns#">

   <rdf:Description rdf:about="./simulation.xml">
      <dcterms:description>SED-ML Description Representing a 1D Steady Scan
         experiment carried out on the Heinrich Oscillator model.
      </dcterms:description>
      <dcterms:creator>
         <rdf:Bag>
            <rdf:li rdf:parseType="Resource">
               <vCard:hasName rdf:parseType="Resource">
                  <vCard:family-name>Bergmann</vCard:family-name>
                  <vCard:given-name>Frank</vCard:given-name>
               </vCard:hasName>
               <vCard:hasEmail rdf:resource="fbergman@caltech.edu">
               <vCard:organization-name>
          California Institute of Technology
         </vCard:organization-name>
            </rdf:li>
         </rdf:Bag>
      </dcterms:creator>
      <dcterms:created rdf:parseType="Resource">
         <dcterms:W3CDTF>2014-01-20T19:52:11Z</dcterms:W3CDTF>
      </dcterms:created>
      <dcterms:modified rdf:parseType="Resource">
         <dcterms:W3CDTF>2014-01-20T19:54:05Z</dcterms:W3CDTF>
      </dcterms:modified>
   </rdf:Description>
</rdf:RDF>
```

5 Future development

In this section we highlight some open issues not addressed in this version of the COMBINE archive specification.

5.1 Linking to external documents

It was often discussed to also allow the `location` elements of a **Content** element point to an external document. However, in this first version we restrict them to local files, so as to make it easier to adopt in software tools. That way tools could focus on the primary use case of bundling up several local resources, rather than worry about retrieving information from online resources that may not always be available or may be more complex.

5.2 Cross References between entries

At HARMONY we spend some time discussing whether cross references between the individual entries in the archive ought to be in this first version of the specification. However, it was decided to leave the cross referencing to the individual standards for now, rather to impose them ad-hoc.

5.3 Alternative versions of the archive metadata

It was suggested to allow different versions of the archive metadata format. The manifest already provides a way for referencing alternate versions, all that would need to be changed, would be the format identifier to point to a different specification rather than:

`http://identifiers.org/combine.specifications/omex-metadata`

However, as of the time of this writing no such format was proposed.

Acknowledgments

We would like to thank all the people who contributed in various ways to the development of both the original proposal and this specification.

For financial/travel/technical and moral support we thank especially (in alphabetical order): Michael Hucka (CalTech, USA) and Ursula Kummer (Heidelberg University, Germany).

We also would like to thank Richard Adams, and the attendees of dedicated sessions at HARMONY2012 / HARMONY2013 and the members of the combine-discuss mailing list and all others who contributed to discussions on various occasions.

References

Biron, P. V. and Malhotra, A. (2000). XML Schema part 2: Datatypes (W3C candidate recommendation 24 October 2000). Available via the World Wide Web at http://www.w3.org/TR/xmlschema-2/.

Eriksson, H.-E. and Penker, M. (1998). *UML Toolkit*. John Wiley & Sons, New York.

Fallside, D. C. (2000). XML Schema part 0: Primer (W3C candidate recommendation 24 October 2000). Available via the World Wide Web at http://www.w3.org/TR/xmlschema-0/.

Oestereich, B. (1999). *Developing Software with UML: Object-Oriented Analysis and Design in Practice*. Addison-Wesley.

Perreault, S. (2011). vCard Format Specification. http://www.ietf.org/rfc/rfc6350.txt. Updated by RFC 6868.

PKWARE Inc. (2012). APPNOTE.TXT - .ZIP File Format Specification. http://www.pkware.com/documents/casestudies/APPNOTE.TXT. [Online; accessed 13-January-2014].

Thompson, H. S., Beech, D., Maloney, M., and Mendelsohn, N. (2000). XML Schema part 1: Structures (W3C candidate recommendation 24 October 2000). Available online via the World Wide Web at the address http://www.w3.org/TR/xmlschema-1/.

Waltemath, D., Bergmann, F. T., Adams, R., and Le Novère, N. (2011). Simulation experiment description markup language (sed-ml) : Level 1 version 1. Available via the World Wide Web at http://co.mbine.org/specifications/sed-ml.level-1.version-1.pdf.

Wikipedia (2014). Zip (file format) — Wikipedia, the free encyclopedia. http://en.wikipedia.org/w/index.php?title=Zip%20(file%20format)&oldid=588974210. [Online; accessed 13-January-2014].