AdsML® Framework for E-Commerce Business Standards for Advertising

Document Names and Identifiers – Guidelines and Examples

Document Authors: AdsML Technical Working Group

Document ID: AdsML2.0-DocumentNames-AD-6

Document File Name: AdsML2.0-DocumentNames-AD.pdf

Document Status: Approved

Document Date: 1 October 2006

Draft Number: 6

Table of Contents

1 AdsML Standard Documentation

1.1 Document status and copyright

This is the Approved documentation of AdsML Document Names and Identifiers – Guidelines and Examples.

This is an internal AdsML document for use by AdsML Members and other interested parties. It may be updated, replaced, or made obsolete by other documents at any time.

1.2 Document Number and Location

This document, AdsML2.0-DocumentNames-AD-6, is available to members of the AdsML Consortium and other interested parties. It will be located in the public area of the AdsML website at http://www.adsml.org/.

1.3 Abstract

This document sets out guidelines and examples for the naming, versioning and packaging of AdsML standards and supporting documentation.

1.4 Audience

This is an internal working document that is designed to provide guidance to the AdsML Technical Working Group. Some familiarity with existing AdsML processes and deliverables is assumed.

Comments on this document should be addressed to the Technical Working Group of the AdsML Consortium (technical.wg@adsml.org).

1.5 Purpose of this document

This document is intended to provide guidance to developers of the AdsML Framework.

1.6 Accompanying documents

There are no accompanying documents.

1.7 Change History

Version	Date	Changes	Editor
AD 6	1 October 2006	• Updated AdsML references to reflect Registered Trademark status	TS

AD 5	1 June 2006	• Updated example schema element to show current versions of referenced namespaces. • Revised text and examples to reflect our new naming convention of “spec part 1” and “spec part 2” for all normative specifications • Clarified the rules about the relationship between schema and specification versions for a given standard	JC / TS
AD 4	25 November 2005	• Removed reference to private type libraries which no longer exist in our architecture • Corrected errors in the schema numbering example	UW/TS
AD 3	14 September	• Added explicit coverage of schema “usage” documents which are handled the same as schema specifications	TS
AD 2	1 July	• Fixed errors in section 1.1, document status and copyright	TS
AD 1	20 May	• First approved draft	TS
WD 5	10 May 2005	• Minor changes based on responses from the team	TS
WD 4	26 April 2005	• Major revisions based on input from the Technical Working Group • Incorporated the schema naming conventions and assigned the same guidelines to the Specification documents • Created “PD” and “AD” statuses for non-normative documentation	TS
WD 3	9 March 2005	• Fixed typos	TS
WD 2	5 March 2005	• Modified the Release Versions naming and numbering scheme • Defined folder structure • Added explicit Errata document • Created “quick start examples” section	TS
WD 1	11 February 2005	• Initial release	TS

1.8 Acknowledgements

This document is a product of the AdsML Technical Working Group.

Primary authorship and editing was performed by:

- Tony Stewart (RivCom) - tony.stewart@rivcom.com

1.9 Definitions of key words used in this document

The key words "**MUST**", "**MUST NOT**", "**REQUIRED**", "**SHALL**", "**SHALL NOT**", "**SHOULD**", "**SHOULD NOT**", "**RECOMMENDED**", "**MAY**", and "**OPTIONAL**" in this document are used as described in IETF RFC 2119. When any of these words do not appear in upper case as above, then they are being used with their usual English language sense and meaning.

1.10 Open issues and To-Do

This section contains notes and to-do items that have not been placed inside one of the sub-sections of the document.

1. Define the format of the Errata document (see Israel's example of the JDF version)
2. Define the format of the readme file, including
    a. release notes (1 para)
    b. file list with hyperlinks to specific files or folders
    c. release stack (notes for prior releases)
        i. note: this is the only document containing hyperlinks to specific files on disk
        ii. it is updated with each framework release, no matter how small
3. Define the contents of the FrameworkOverview file, including:
    a. Extended textual introduction to the current Framework.
        i. For Framework 2.0 (last fall) this would be the text of the "Phase 2 Overview" document pasted right into the HTML.
    b. List of business problems and filesets that address them.
4. **Suggestion**: merge the AdsML Documentation Set into the readme.html file, so that we eliminate the apparent duplication between them.
5. Develop repeatable procedures for packaging the zip
6. Files that require substantial renaming/reworking:
    a. AdsML-Processing-Model -> AdsML-Envelope-Processing-Model
    b. AdsML-Specification -> AdsMLEnvelope-Specification
    c. .. more here
7. Define guidelines for internal metadata (e.g. document number, title) for:
    a. Structured Descriptions spreadsheets

2 Framework concepts

This section provides background information about the design choices embodied in the naming rules and guidelines.

2.1 Framework releases

The AdsML standards are marketed as the “AdsML Framework for E-Commerce Business Standards for Advertising”, or the “AdsML Framework” for short. Each release of the Framework consists of a set of artifacts – standards documents, schemas, etc. – that work together to solve a set of business problems. We apply version numbers to the Framework for clarity in external communications such as press releases: “AdsML Framework 1.0”, “AdsML Framework 2.0”, etc.

It is our goal to keep the number of releases of the Framework relatively small, and to be able to tie each Framework release to a public announcement about the new functionality delivered in that release. Therefore, a numbering and versioning scheme has been developed for the Framework, as well as for the artifacts within it.

The Framework numbering scheme consists of two parts, a version number and a “release identifier”. The version number is a 2-level number such as “1.0” or “2.3”, e.g. “AdsML Framework 1.0”. The release identifier consists of a textual “status” indicator plus a number, for example: “Beta 1”, “Proposed 2” or “Release 1”. The combination of these two concepts allows us to issue multiple uniquely-identified releases of the Framework without changing the version number. For example: “AdsML Framework 1.0 Proposed 1”, “AdsML Framework 1.0 Proposed 2”, etc. This should simplify the marketing of a release – its number doesn’t change – while still providing necessary clarity to implementers about exactly which file set they are working with.

2.2 Contents of the Framework

Each release of the Framework is packaged in a single zip file containing the artifacts (documents, schemas, etc.) that comprise that release. In any given Framework release, some of these artifacts will be newly created or updated, while others will have been carried forward unchanged from an earlier release. This means that each Framework release is additive, carrying within it various older standards as well as any newly-developed or newly-changed ones.

Each Framework release MUST contain a “readme” file that includes a manifest of the files in the release (with hyperlinks to them) plus notes about any changes that have occurred since the last release of the Framework.

A Framework release SHOULD only include the results of the technical work that has been done so far – i.e. the standards, schemas and supporting documentation - not the documents that guided and constrained the work. In particular, it SHOULD NOT include documents describing the Scope, Requirements and Procedures that were followed.

2.3 Artifact names and numbers

The artifacts within the Framework have their own naming and versioning schemes, which are somewhat different from that of the Framework. Each time one of these artifacts is updated, its version information is incremented. However, only some of the version information is apparent in the document's filename. In particular, the "smallest" digit of the document's version number is not included in its filename, and can only be seen by an examination of the document's internal properties, title page, or equivalent. (It can also be inferred from the date of the document, although this is not a foolproof method and is not encouraged.)

We have chosen this approach in order to facilitate the creation and packaging of Framework releases, in particular, the maintenance of internal cross-references between the objects in the Framework (for example, "include" operations in the schema, or hyperlinks in the readme.html file.) Because the filenames within a version of the framework do not change with each minor update, cross-references to those filenames from other documents do not have to be changed. This should allow us to package new versions of the Framework with less effort than has previously been the case.

In broad terms, there are two different naming and numbering conventions in the AdsML Framework:

- A relatively formal numbering scheme for the schemas and their matching Specification documents
- A different approach for all other supporting documentation

2.3.1 Schemas and specifications

Schema and specification numbering is based on the traditional concept of a three-level "version" for a given schema, for example version 1.2.3. This numbering begins at 1.0.0 for the initial release of one of our standards, and then increments whenever a new version of that standard is released. There is no connection between this kind of numbering and the numbering of the Frameworks in which these files are contained. For example, version 1.0.0 of a standard might be created in Framework 2.0

Each specification document consists of two halves: Part 1 Usage Rules and Guidelines, and Part 2 Specification & Schema. Their filenames reflect this.

Two examples of artifacts conforming to this naming convention are:

- AdsMLEnvelope-1.0-SpecP1Usage-AS.pdf
- AdsMLBookings-1.0-Main-PS.xsd

2.3.2 Supporting documentation

For all other supporting documentation, the file naming scheme does conveys two important pieces of information: the start of each filename indicates the version of the AdsML Framework in which that artifact was created or updated, and the end of the filename indicates whether the status of that artifact is "Working Draft", "Proposed Specification" or "Approved Specification". The filename prefix allows users to see at a glance which documents in a Framework release are new, and which are legacy documents that have been carried forward from a previous release.

For example, you can see from the following two filenames that one is an Approved ("AS") document carried forward from the 1.0 Framework, while the other is a Proposed ("PS") document that is being distributed in Framework 2.0 for evaluation and comment:

- AdsML1.0-EnvelopeProcessingModel-AS.pdf

- AdsML2.0-EcommerceOverview-PS.pdf

These concepts are explored in more detail below.

2.4 Artifact IDs and internal metadata

Each AdsML document or schema is assigned a unique ID. The ID is a concatenation of the same components that are in the artifact's filename, plus the additional versioning information that was omitted from the filename. The ID is stored as metadata inside the artifact, and is displayed on its first page.

Many of the components of the ID are also broken out and stored separately as attributes in a schema file or properties in a Word document.

3 Directory structure and zip guidelines

Each release of the AdsML Framework is packaged as a directory folder sub-tree and delivered in a zip file containing that entire structure.

Both the root folder of the sub-tree and the name of the containing zip file have the same name as that release of the framework.

3.1 Example

ZIP NAME: AdsML-Framework-2.0-Proposed-1.zip
ROOT FOLDER: AdsML-Framework-2.0-Proposed-1

- **Readme.htm** (release notes for the latest framework version/release)
- **Errata.htm** (errata file for latest framework version/release)
- **AdsML2.0-ReleaseNotes-PD.doc** - overview containing notes about the current release of the framework
- **AdsML1.0-FrameworkOverview-AS.pdf** – this is our overall overview of the AdsML framework, which should remain relatively static
- **AdsML2.0-EcommerceOverview-AD.pdf** – a general description of how e-commerce works in the AdsML world – not yet written
- **AdsML1.0-Glossary-AD.pdf**
- **AdsML1.0-AdvertisingComponentInteractions-AS.pdf**
- SUBFOLDER AdsML-Specifications-and-Technical-Documentation
    - SUBFOLDER General-Technical-Documentation
        - **AdsML1.0-ArchitectureOverview-AD.pdf**
        - **AdsML1.0-EnvelopeProcessingModel-AS.pdf**
        - **AdsML1.0-ControlledVocabularies-AS.pdf**
    - SUBFOLDER Schema-Specifications
        - **AdsMLEnvelope-1.1-SpecP1Usage-AS.pdf**
        - **AdsMLEnvelope-1.1-SpecP2Schema-AS.pdf**
        - **AdsMLBookings-1.0-SpecP1Usage-PS.pdf**
        - **AdsMLBookings-1.0-SpecP2Schema-PS.pdf**
        - **AdsMLMaterials-1.0-SpecP1Usage-PS.pdf**
        - **AdsMLMaterials-1.0-SpecP2Schema-PS.pdf**
        - **AdsMLFinancials-1.0-SpecP1Usage-WD.doc**
        - **AdsMLFinancials-1.0-SpecP2Schema-WD.doc**
        - **AdsMLStructuredDescriptions-1.0-SpecP1Usage-PS.pdf**
        - **AdsMLStructuredDescriptions-1.0-SpecP2Schema-PS.pdf**
        - etc.
    - SUBFOLDER Schemas
        - **AdsMLBookings-1.0-Main-WD.xsd**

        - **AdsMLBookings-1.0-PublicTypeLibrary-WD.xsd**
        - **AdsMLMaterials-1.0-Main-WD.xsd**
        - **AdsMLMaterials-1.0-PublicTypeLibrary-WD.xsd**
        - **AdsMLControlledVocabularies-1.0-WD.xsd**
        - **AdsMLEnvelope-1.1-Main-WD.xsd**
        - **AdsMLStructuredDescriptions-1.0-PublicTypeLibrary-PS.xsd**
        - **AdsMLTypeLibrary-1.0-WD.xsd**
- SUBFOLDER AdsML-Samples
    - (documents that apply to all of the samples go here)
    - SUBFOLDER AdsML-Structured-Description-Vocabularies
        - files
    - SUBFOLDER AdsML-Bookings-Message-Samples
        - files
    - SUBFOLDER AdsML-Content-Delivery-Message-Samples
        - files

3.2 Notes

1. The root directory name is the same as the zip file name.
2. The root directory and zip file names reflect the Framework version number and status, so two successive release zips will not overwrite each other.
3. All other folder names are independent of release or version numbers, so it is trivial to paste all or part of a new structure onto an old one without breaking links.
4. The rules for assigning filenames (including the 2-letter “document status” codes) are described in section 5 of this document. Available document status codes are: WD (Working Draft); PS (Proposed Specification); PD (Proposed documentation); AS (Approved Specification); AD (Approved documentation).

4 Framework version numbers and statuses

4.1 Example

This example shows a typical progression of Framework Versions and Statuses over time. In this case there were 5 beta releases before the first Proposed version.

- AdsML Framework 1.0 Beta 1
- AdsML Framework 1.0 Beta 2
- …
- AdsML Framework 1.0 Beta 5
- AdsML Framework 1.0 Proposed 1
- AdsML Framework 1.0 Proposed 2
- AdsML Framework 1.0 Release 1
- AdsML Framework 1.0 Proposed 1.1
- AdsML Framework 1.0 Release 2
- AdsML Framework 1.1 Proposed 1
- AdsML Framework 1.1 Release 1
- AdsML Framework 2.0 Proposed 1
- AdsML Framework 2.0 Proposed 2

- AdsML Framework 2.0 Release 1

4.2 Status categories

There are three statuses of Framework: “Beta”, “Proposed”, and “Release”. For example:

- AdsML Framework 1.0 Beta 1
- AdsML Framework 1.0 Proposed 1
- AdsML Framework 1.0 Release 1

4.2.1 Beta status

“Beta” indicates a version of the Framework that is still being developed and is not yet ready for open public evaluation.

- A “Beta” version of the Framework MAY contain any combination of Working, Proposed and Approved artifacts (documents, schemas, etc.) within it.
- Documents in a Beta release can be in their native editable format

4.2.2 Proposed status

“Proposed” indicates a version of the Framework that has been initially tested and is being offered to the public for evaluation, pilot projects, etc. “Proposed” Frameworks must be approved by the AdsML Plenary before they can be released for public evaluation.

- All of the artifacts in a “Proposed” version of the Framework SHOULD have either Approved or Proposed status.
- All narrative textual documents in a Proposed or Release version of the Framework except for working drafts MUST be in .pdf format.

4.2.3 Release status

“Release” indicates a version of the Framework that has been fully tested and approved by an AdsML Plenary and is thought to be ready for production implementation.

- All of the artifacts in a “Release” version of the framework MUST have reached Approved status. Proposed or Working Draft artifacts SHALL NOT be included in a Released framework.
- All narrative textual documents in a Proposed or Release version of the Framework must be in .pdf format.

4.3 Incrementing the Framework version number

This section provides guidelines for determining when and by how much to increment the 2-level version number that is included in the name of every Framework release, for example, the “2.0” in “AdsML Framework 2.0 Release 1”.

1. If a release of the Framework contains the initial implementation of a new message group (e.g. Bookings, Content, Financials, etc.) we MUST increment the Framework to the next major version number. (We can also choose to do this for other reasons if we wish.)

2. If a release of the Framework adds functionality to a message group that has already been started or changes the technical structure of the schemas in a way that affects implementations, we MUST increment the

Framework at least to the next minor version number. We can choose to increment it more if we wish. For example:

- o Adding a few capabilities to existing message types would only justify a .1 level increase, e.g. from 2.1 to 2.2.
- o Adding Insert or Online capabilities to the Bookings schema would probably justify a jump the next .5 level, e.g. from 2.1 to 2.5 or from 2.6 to 3.0.
- o Adding a new message type to an existing message group would certainly justify a jump to the next .5 level

3. If a release of the Framework merely fixes bugs or adds documentation to a feature set that already exists, without changing the schema structures or semantics, we SHOULD NOT increment the Framework version number, but SHOULD increment its Release number instead, e.g. from “Release 1” to “Release 2”.

4.4 Incrementing the Framework status number

This section describes the mechanism for assigning the status version that appears at the end of every Framework release, for example the “1” at the end of “AdsML Framework Release 2.0 Beta 1”.

The status version is indicated by a combination of a word and a number. The number is calculated differently depending on whether the status is “release” or one of the other two possible statuses.

4.4.1 Release statuses

“Release” statuses are numbered using an integer (e.g. “Release 1” or “Release 2”). The status number is set to “1” with the initial Release of a given version of the Framework, and incremented by 1 with each successive Release of that version. For example:

- AdsML Framework 1.0 Release 1
- AdsML Framework 1.0 Release 2
- AdsML Framework 1.1 Release 1 (reset to “1” because the Framework version changed)
- AdsML Framework 1.1 Release 2
- etc.

4.4.2 Beta and Proposed statuses

“Beta” and “Proposed” statuses are numbered using either an integer or a 2-level decimal (e.g. “1” or “1.1”), depending on whether this version of the Framework has already achieved “Release 1” status. After a “Release 1” has occurred, the first digit of the Status number indicates the highest release level that has previously been achieved. For example:

- AdsML Framework 1.0 Beta 1
- AdsML Framework 1.0 Proposed 1
- AdsML Framework 1.0 Release 1
- AdsML Framework 1.0 Proposed 1.1 (two digits now shown)
- AdsML Framework 1.0 Release 2
- AdsML Framework 1.0 Proposed 2.1
- AdsML Framework 1.1 Beta 1 (back to an integer because the Framework Version number has changed)
- etc.

The second digit of the status number is set to 1 each time the status changes, e.g. from “Proposed” to “Release” or from “Release” back to “Proposed”. It is incremented only when two or more sequential releases share the same Framework Version number and status type. For example:

- AdsML Framework 1.0 Beta 1
- AdsML Framework 1.0 Beta 2 (incremented because the framework version and status remain “1.0 Beta”)
- AdsML Framework 1.0 Beta 3 (ditto)
- AdsML Framework 1.0 Proposed 1 (reset to 1 because the status has changed to “Proposed”)
- etc.

Note: In theory a version of the Framework could drop back from Release to Beta status, but this is unlikely in practice. If a new Beta review of a Framework is needed, we would be likely to increment the Framework version number, which would force the Release numbers to restart at “1”.

4.5 Master example of framework names and version numbers

The following sequence of Framework release names provides examples of all of the above rules:

1. AdsML Framework 1.0 Beta 1 (internal zip of the Framework that is only released to beta testers)
2. AdsML Framework 1.0 Proposed 1 (first release for industry consideration)
3. AdsML Framework 1.0 Proposed 2 (some bugs fixed or modest enhancements and re-released for industry review)
4. AdsML Framework 1.0 Release 1 (approved package of standards)
5. AdsML Framework 1.0 Proposed 1.1 (adding some documentation of existing features or perhaps a minor fix but no significant new functionality)
6. AdsML Framework 1.0 Release 2 (documentation or changes have been approved)
7. AdsML Framework 1.1 Proposed 1 (adding some functionality or additional messages within the existing message groups)
8. AdsML Framework 1.1 Release 1 (changes have been approved)
9. AdsML Framework 2.0 Proposed 1 (adding major new functionality or starting to develop a new message group)
10. AdsML Framework 2.0 Proposed 2 (changes made based on industry review)
11. AdsML Framework 2.0 Release 1 (approved package)

5 Documentation filenames and internal metadata

5.1 Schemas, Normative Specifications and Usage Documents

The normative definition for each AdsML message format consists of one or more schemas (.xsd) which work together as described in our architectural guidelines. Each message format is also accompanied by a two-part Specification document: Part One, Usage Rules & Guidelines, documents how that type of message is meant to be used in an ecommerce environment; Part Two, Specification & Schema, provides a textual description of the elements and attributes used in that format. Therefore, there may be several schema files for a given standard (Main and PublicTypeLibrary), but there will always be only one normative specification document and one usage document. (The AdsML Envelope currently does not have a usage document. The Envelope Processing Model performs a slightly different purpose and is not named or packaged as a Usage document.)

Schemas and their related specification and usage documents share the same naming and numbering conventions. These are described in this section.

There may well be other textual documents that support implementation of a given schema. These are packaged in a different directory and follow a different naming convention.

5.1.1 Examples

Filename	**Document ID**
AdsMLBookings-1.0-Main-AS.xsd	AdsMLBookings-1.0.0-Main-AS-1
AdsMLBookings-1.0-PublicTypeLibrary-WD.xsd	AdsMLBookings-1.0.0-PublicTypeLibrary-AS-1
AdsMLEnvelope-1.0-Main-AS.xsd	AdsMLEnvelope-1.0.1-Main-AS-2
AdsMLBookings-1.0-SpecP1Usage-AS.doc	AdsMLBookings-1.0.0-SpecP1Usage-AS-1
AdsMLBookings-1.0-SpecP2Schema-AS.doc	AdsMLBookings-1.0.0-SpecP2Schema-AS-1
AdsMLEnvelope-1.0-SpecP1Usage-AS.pdf	AdsMLEnvelope-1.0.1-SpecP1Usage-AS
AdsMLTypeLibrary-1.1-WD.xsd	AdsMLTypeLibrary-1.1.3-WD-3

5.1.2 Naming patterns

Filename pattern: [Main Schema Name] – [2-level Schema Release Version] – [Document Type] – [Current Document Status]

Document ID pattern: [Main Schema Name] – [complete Schema Release Version] – [Document Type] – [Current Document Status] – [Draft Number]

Hyphens separate the components of the filename or ID, but must not appear within the name components themselves. Multi-word components such as Document Type should be constructed using camelCase (for example, “PublicTypeLibrary”).

5.1.2.1 Exceptions

1. Schemas and specifications with a status of “Approved Specification” do not have a Draft Number.
2. The schema for the AdsML Type Library does not include a [Document Type] component in its filename or internal ID.

5.1.3 Filename and ID components

Schema Name: The name of the root element of the main schema of the cluster of schemas to which this document applies, e.g. “AdsMLBookings” or

"AdsMLEnvelope". (For the AdsML Type Library, this is the phrase "AdsMLTypeLibrary".)

Schema Release Version: The full release version of the schema that we are working towards using three dot separated digits, like "1.0.0" for the first version of AdsMLBookings. Only the first two levels appear in the filename.

Document Type: A text string describing the contents of this particular document. Options are: "Main", "Specification", or "PublicTypeLibrary".

Current Document Status: The document's current status. Status can be expressed in both long- and short-forms. The short form is included in the filename; the long form is used in the internal ID and in dedicated internal metadata properties or attributes. The possible statuses for a normative schema or specification are:

Long Status	Short Status
Working Draft	WD
Proposed Specification	PS
Approved Specification	AS

Draft Number: This is our internal "work in progress" version number. The Draft Number is a running positive integer that is restarted to "1" when a document first reaches Approved status. The Draft Number should be used together with the Schema Release Version, e.g. it is the "15th draft for 1.0.0"

5.1.4 Notes

1. Schema and specification filenames include the first two levels of their Schema Release Version and the Status (e.g. WD or PS) of the current document. However, they do not indicate the third level of the Schema Release Version or the Internal Draft Number of the current document. These are only visible in metadata contained within.
2. Unlike our other documents, schema and specification filenames and IDs do not indicate the version of the Framework in which they were last updated.
3. A WD (Working Draft) specification document can be distributed in its native .doc format, but a PS or AS text document must be packaged as either HTML or PDF
4. The name of the root element of each "master" schema file must always begin with "AdsML" (e.g. "AdsMLBookings", "AdsMLEnvelope"). These are the only elements whose names should begin with AdsML.

5.1.5 Internal metadata

The name components are also carried as internal metadata.

5.1.5.1 Schema metadata

In a schema, the metadata is contained in attributes of the Schema element.

Name Component	Metadata location
Schema Name	Contained in @id
Schema Release Version	@version
Document Type	Contained in @id
Current Document Status	@adsml-sx:status
Draft Number	@adsml-sx:internalVersion

Document ID	@id

For example, here is the opening tag of the Schema element in a draft of the main Bookings schema (AdsMLBookings-1.0.0-Main-PS.xsd), as rendered in Internet Explorer:

```
- <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns="http://www.adsml.org/adsmlbookings/1.0" xmlns:adsml-
  sd="http://www.adsml.org/adsmlstructureddescriptions/1.0"
  xmlns:adsml="http://www.adsml.org/typelibrary/1.1" xmlns:adsml-
  cv="http://www.adsml.org/controlledvocabularies/2.0" xmlns:adsml-
  ma=" http://www.adsml.org/adsmlmaterials/1.0" xmlns:adsml-
  sx="http://www.adsml.org/schema-extensions/1.0"
  targetNamespace="http://www.adsml.org/adsmlbookings/1.0"
  elementFormDefault="qualified" attributeFormDefault="unqualified"
  version="1.0.0" id="AdsMLBookings-1.0.0-Main-PS-16" adsml-
  sx:status="ProposedSpecification" adsml-sx:internalVersion="16">
```

In this case the Schema metadata is contained in the last few attributes, after the long list of namespace declarations. (The sequence of attributes is not significant.)

5.1.5.2 Specification metadata

In a specification document, the metadata is contained in custom Word Properties some of which are displayed in the document's title page.

Name Component	Word Property	Title page?
Schema Name	Contained in the Document number	In the document's Title
Schema Release Version	Contained in the Document number	As part of the document ID
Document Type	Contained in the Document number	The phrase "Specification & Schema" or "Usage Rules & Guidelines" is included in the document title
Current Document Status	Document status	Yes
Draft Number	Draft number	Yes
Document ID	Document number	Yes

5.1.6 Schema Release Version (numbering) guidelines

5.1.6.1 Rules for assigning the release version number to the Main schema for a standard

1. Schema revisions that include non-backwards-compatible changes to a previously approved schema **MUST** increment at least the second digit of the Schema Release Version number.
2. Schema revisions that add new message types to a previously approved schema **MUST** increment at least the second digit of the Schema Release Version number.

3. The third digit of the Schema Release Version number **MUST ONLY** be incremented in the case of a bug-fix release for which all of the following statements are true:
   - The changes are fully “backwards-compatible”, meaning that all possible document instances that were valid according to the previous version will still be valid with the new version
   - No new capabilities have been added to the affected message formats
   - Any documentation changes consist of fixing typos or adjusting the documentation so it correctly describes the newly-fixed schemas.

5.1.6.2 Structural rules

1. Schema files and their associated specification documents are versioned sequentially over the lifetime of that schema, without regard to the version(s) of the Framework in which the work was done.
2. Within a given release of the framework, the Main schema of a given standard and its matching Specification document **MUST** share **the same first two levels** of schema release number. For example, if the schema release number for AdsMLBookings Main is 1.1.x, then the schema release for AdsMLBookings Specification **MUST** also begin 1.1. (However, the third level can be different.)
3. Whenever significant new functionality is added to a Main schema, including (but not limited to) all cases when the first digit of its version number is incremented, the public type library associated with that schema **MUST** be updated and assigned the same version number as the main schema.
4. Each level of the Schema Release Version is an integer that can be incremented indefinitely. In theory there is no limit on the number of digits that can be included in each level of the Version number. However, in practice the second level **SHOULD** be kept to a single digit.
5. After an Approved Specification has been issued, when work starts on the next version of that standard its Schema Release Version **MUST** be incremented by at least the minor digit. There **MUST NOT** be two different drafts of an approved schema that share the same Schema Release Version.
6. The first two levels of the Schema Release Version number of a specification document **MUST** be the same as the first two levels of the version number of its associated schema.
7. A minor update to a specification which is not associated with a change in its schema is indicated by incrementing the Draft number of that specification document but not its schema version number.

5.2 Normative textual documentation *other than* specifications and usage documents

The guidelines in this section apply to textual standards documents *other than* schema specifications which have been submitted (or are expected to be submitted) for plenary approval; for example, the Advertising Component Interactions Analysis and the Envelope Processing Model.

5.2.1 Examples

Filename	Document ID
AdsML1.0-AdvertisingComponentInteractions-	AdsML1.0-AdvertisingComponentInteractions-WD-

WD.doc	2
AdsML1.0-AdvertisingComponentInteractions-AS.pdf	AdsML1.0-AdvertisingComponentInteractions-AS-22
AdsML1.0-EnvelopeProcessingModel-AS.pdf	AdsML1.0-EnvelopeProcessingModel-AS-6

5.2.2 Naming pattern

Filename pattern: [AdsML Framework Version] - [Document Name] - [Current Document Status]

Document ID pattern: [AdsML Framework Version] - [Document Name] - [Current Document Status] - [Draft Number]

Hyphens separate the components of the filename or ID, but must not appear within the name components themselves. Multi-word components such as Document Name should be constructed using camelCase (for example, “AdsML1.0” and “AdvertisingComponentInteractions”).

5.2.3 Filename and ID components

AdsML Framework Version: The word “AdsML” followed by the version number of the AdsML Framework for which this document was intended or actually released. (It is possible that a document will begin being developed as part of one Framework but not be approved until another. In that case the Framework Version will change during the life of the document, with earlier drafts reflecting the originally intended Framework Version, and later drafts reflecting the actual Framework in which it was approved.)

Document Name: A text string containing up to three concatenated words taken from the title of the document. The word “AdsML” should not be included.

Current Document Status: The document’s current status. Status can be expressed in both long- and short-forms. The short form is included in the filename; the long form is used in the internal ID and in dedicated internal metadata properties or attributes. The possible statuses for normative documents are:

Long Status	Short Status
Working Draft	WD
Proposed Specification	PS
Approved Specification	AS

Draft Number: This is our internal "work in progress" version number. The Draft Number is a running positive integer that is restarted when the Framework Version changes. The Draft Number should be used together with the AdsML Framework Version, e.g. it is the "3rd draft for Framework 1.0".

5.2.4 Notes

1. Document filenames indicate the version of the Framework in which they were last updated and the status they have achieved (WD, AS or PS), but not their latest incremental version number or the Release number of the Framework.
2. A WD (Working Draft) text document can be in its native .doc format, but a PS or AS text document must be packaged as either HTML or PDF.

3. Although some of these documents are conceptually related to a particular schema, they are included in this group because their contents are only loosely bound to the precise version of that schema. For example, we would not expect to update the Envelope Processing Model each time we update the AdsMLEnvelope's Schema and Specification, and a change to the Processing Model would not require a change to the Envelope's schema. Therefore, we name and version the Envelope Processing Model as part of the overall Framework rather than including it in the AdsML Envelope specification set.

5.2.5 Internal metadata

The metadata is contained in custom Word Properties, some of which are displayed in the document's title page.

Name Component	Word Property	Title page?
AdsML Framework Version	Contained in the Document ID	In the document number and filename
Document Name	Contained in the Document ID	In the title
Current Document Status	Document status	Yes
Draft Number	Draft number	Yes
Document ID	Document number	Yes

5.2.6 Sample title page

AdsML Framework for E-Commerce Business Standards in Advertising

Advertising Component Interactions Analysis

Document Authors: AdsML Technical Working Group
Document ID: AdsML1.0-AdvertisingComponentInteractions-AS-4
Document File Name: AdsML1.0-AdvertisingComponentInteractions-AS.pdf
Document Status: Approved Specification
Document Date: xxx
Draft Number: 4

5.3 Supporting textual documentation

The guidelines in this section apply to textual documents that are developed by the Technical Working Group as non-normative supporting documentation. These might or might not be submitted to the plenary for approval.

Note that readme and errata files are named and numbered differently than the other types of supporting documentation.

5.3.1 Examples

Filename	Document ID
Readme.html	AdsML2.0-Proposed1-Readme-RD-1
Errata.html	AdsML2.0-Release1-Errata-RD-22
AdsML2.0-ReleaseNotes-WD.doc	AdsML2.0-ReleaseNotes-WD-1
AdsML2.0-ReleaseNotes-PD.pdf	AdsML2.0-ReleaseNotes-PD-2

AdsML2.0-DocumentNames-WD.doc	AdsML2.0-DocumentNames-WD-4
AdsML2.0-FrameworkOverview-AD.pdf	AdsML2.0-FrameworkOverview-AD-3

5.3.2 Naming patterns

5.3.2.1 Naming Pattern for Readme and Errata files

Filename pattern: [Document Name]

Document ID pattern: [AdsML Framework Version] – [Framework Release number] - [Document Name] - [Current Document Status] - [Draft Number]

Hyphens separate the components of the filename or ID, but must not appear within the name components themselves.

5.3.2.2 Naming Pattern for all other supporting textual documentation

Filename pattern: [AdsML Framework Version] - [Document Name] - [Current Document Status] - [Draft Number]

Document ID pattern: [AdsML Framework Version] – [Framework Release number] - [Document Name] - [Current Document Status] - [Draft Number]

Hyphens separate the components of the filename or ID, but must not appear within the name components themselves.

5.3.3 Filename and ID components

AdsML Framework Version: The word “AdsML” followed by the version number of the AdsML Framework for which this document was intended or actually released.

Framework Release Number: The specific Framework Release with which this document is associated. Only used for Readme and Errata files.

Document Name: A text string containing up to three concatenated words taken from the title of the document. The word “AdsML” should not be included.

Current Document Status: The document’s current status. Status can be expressed in both long- and short-forms. The short form is included in the filename; the long form is used in the internal ID and in dedicated internal metadata properties or attributes. The possible statuses for normative documents are:

Long Status	Short Status
Working Draft	WD
Proposed	PD
Approved	AD

Draft Number: This is our internal "work in progress" version number. It is a running positive integer, which is restarted when the Framework Version changes. In this case the Draft Number should be used together with the AdsML Framework Version, e.g. it is the "3rd draft for Framework 1.0".

5.3.4 Notes

1. A WD (Working Draft) textual supporting document can be distributed in its native .doc format, but a PD or AD textual supporting document must be packaged as either HTML or PDF.
2. The naming convention of including the Framework Version number in the filename and ID applies even to documents (such as these naming guidelines) that are not inherently related to the framework in which they were created.

5.3.5 Internal metadata

The name components are also carried as internal metadata.

5.3.5.1 Readme and Errata file metadata

In the Readme and Errata files, the metadata is contained in a large Title and smaller Document Status and Document ID fields at the top of the file. For example:

Readme for AdsML Framework 2.0 Release 1
Version: 12
Status: Approved
Document ID: AdsML2.0-Release1-Readme-AD-12

5.3.5.2 Internal metadata for supporting textual documents

In most other supporting documentation, the metadata is contained in custom Word Properties, some of which are displayed in the document’s title page.

Name Component	Word Property	Title page?
AdsML Framework Version	Contained in the Document ID	In the document number and filename
Release Number	n/a	n/a
Document Name	Contained in the Document ID	In the title
Current Document Status	Document status	Yes
Draft Number	Draft number	Yes
Document ID	Document number	Yes

5.3.5.3 Sample title page

AdsML Framework for E-Commerce Business Standards in Advertising

Advertising Framework Overview

Document Authors: AdsML Technical Working Group
Document ID: AdsML1.0-FrameworkOverview-AD-3
Document File Name: AdsML1.0-FrameworkOverview-AD.pdf
Document Status: Approved
Document Date: xxx
Draft Number: 3

5.4 Names of files <u>not</u> included in a Framework release

5.4.1 Example

- AdsML2.0-DocumentNames-WD-4.doc

5.4.2 Explanation

Many documents are created and modified prior to the release of a formal “Framework” package. Typically but not always these are working drafts (“WD”).

Authors of such documents are encouraged but not required to follow the naming conventions for Supporting Textual Documentation, to the extent possible.

When documents are released outside the context of a Framework, it will usually be necessary to add the document’s draft number to the end of its filename to ensure that all recipients can identify the current version from the filename. For example: AdsML2.0-DocumentNames-WD-4.doc, rather than AdsML2.0-DocumentNames-WD.doc as the normal guidelines require.

Once a document is packaged into a Framework release, it becomes critical to rename the file according to our guidelines.

(end)