SyncML Meta-Information DTD, version 1.0

Abstract

This document defines a XML Document Type Definition (DTD) as defined in [5]. The DTD represents standard meta-information used in the SyncML Representation Protocol as defined in [3]. The meta-information is included in a SyncML XML document through the declaration of the SyncML Meta-information DTD name space on any element types from this DTD. See [3] for more details.

SyncML Initiative

The following companies are Sponsors of the SyncML initiative:

Ericsson
IBM
Lotus
Matsushita Communications Industrial Co.
Motorola
Nokia
Palm, Inc.
Psion
Starfish Software

Revision History

Revision	Date	Comments
v0.9	2000-05-31	Version 0.9.
v1.0a	2000-08-24	Completed description of each element. Updated format for headers. Included correct Sponsor name. Added WBXML tokens.
V1.0b	2000-11-13	Added MaxMsgSize element. Updated URL in reference section.
V1.0	2000-12-07	Candidate version for the final release.

Copyright Notice

Table of Contents

1 Introduction

This Document Type Definition (DTD) defines a set of mark-up that is used by the SyncML DTD to identify meta-information associated with a SyncML command or data item or collection.

2 Formatting Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY" and "OPTIONAL" in this document are to be interoperated as described in [2].

Any reference to components of the Device Information DTD or XML snippets is specified in this `typeface`.

3 Terminology

Meta-Information

Parameter or attributes about the representation, state or type or content of an object or property.

4 XML Usage

The SyncML **Meta-Information** is represented in a mark-up language defined by [5]. The meta-information is identifiable as an XML name space. The SyncML Meta-Information DTD (Document Type Definition) defines the XML document type used to represent meta-information used in the [3] representation protocol. The SyncML Meta-Information DTD can be found in Section 5.

The SyncML Meta-Information XML documents are specified using well-formed XML. However, they need not be valid XML. That is, they do not need to specify the XML prolog. They only need to specify properly identified name space element types from the SyncML Meta-Information DTD. This restriction allows for the SyncML Meta-Information to be specified with greater terseness than would be possible if a well-formed, valid XML document was required.

This DTD makes heavy use of XML name spaces. Name spaces MUST be declared on the first element type that uses an element type from the name space. Element types from the SyncML Meta-Information DTD can be used in other XML documents, including a SyncML message.

Names in XML are case sensitive. By convention in the SyncML Meta-Information DTD, the element type and attribute list names are specified with a "Hungarian" like notation of the first character in each word of the name in upper case text and remainder of the characters in each word of the names specified in lower case text. For example, `MetInf` for the Sync meta-information root element type or `Type` for the content type tag.

The element types in the SyncML Meta-Information DTD are defined within a namespace associated with the URI `http://www.syncml.org/docs/syncml_metinf_v10_20001207.dtd` or the URN `syncml:metinf`.

SyncML also makes use of XML standard attributes, such as `xml:lang`. Any XML standard attribute can be used in a XML document conforming to this DTD.

XML can be viewed as more verbose than alternative binary representations. This is often cited as a reason why it may not be appropriate for low bandwidth network protocols. In most cases, this DTD uses shortened element type and attribute names. This provides a minor reduction in verbosity. Additionally, the SyncML Meta-Information can be encoded in a tokenized, binary format defined by [4]. The use of [4] format is external to specification of the DTD and should be transparent to any XML application. The combination of the use of shortened element type names and an alternative binary format makes this DTD competitive, from a compressed format perspective, with alternative, but private, binary representations.

One of the main advantages of XML is that it is a widely accepted International recommendation for text document mark-up. It provides for both human readability and machine processability. In addition, XML allows the originator to capture the structure of a document, not just it's content. This is extremely useful for applications such as data synchronization, where not just content, but structure semantics is often exchanged.

5 Element Type Descriptions

5.1 Anchor

Usage: Specifies the synchronization state information (i.e., sync anchor) for the current synchronization session.

Parent Elements: `MetInf`

Restrictions: The optional `Last` element type specifies the synchronization anchor for the previous synchronization session. The required `Next` element type specifies the synchronization anchor for the current synchronization session.

Content Model:

```
(Last?, Next)
```

Attributes: None.

Example:

```
<Anchor xmlns='syncml:metinf'>
  <Last>20000824T133000Z</Last><Next>20000824T221300Z</Next>
</Anchor>
```

5.2 Format

Usage: Specifies the encoding format of the content information in the `Data` element.

Parent Elements: `MetInf`

Restrictions: The value of this element SHOULD BE one of `b64`, `chr`, `int`, `null` or `xml`. If the element type is missing, the default value is `chr`. If the value is `chr`, then the format of the content information is clear-text in the character set specified on either the transport protocol, the MIME content type header or the XML prolog. If the value is `int`, then the format of the content information is numeric text representing an unsigned integer between zero and 2**32-1. If the value is `null`, then there is no content information. This value is used by some synchronization data models to delete the content, but not the presence of the property. If the value is `b64`, then the format of the content information is binary data that has been character encoded using the Base64 transfer encoding defined by [RFC2045]. If the value is `xml`, then the format of the content information is XML structured mark-up data.

The target object is the one in which the meta-information appears.

Content Model:

```
(#PCDATA)
```

Attributes: None.

Example: The following example illustrates how the element type is used within the SyncML DTD to specify format meta-information for data in the Item element type.

```
<Item>
  <Meta>
    <Format xmlns='syncml:metinf'>int</Format>
  </Meta>
  <Data>1024</Data>
</Item>
```

5.3 Last

Usage: Specifies the synchronization state information (i.e., sync anchor) for the last successful synchronization session.

Parent Elements: `MetInf`

Restrictions: The value MUST specify either an UTC based ISO 8601 [1] date/time stamp or a monotonically increasing numeric integer string. If a date/time stamp, then the text MUST be in the complete representation, basic format defined by [ISO8601].

Determination of the ordinal sequence of the version of an existing object in the recipient and the version of the object can be made by comparing the content information of the object with the value on the existing object.

Content Model:

```
(#PCDATA)
```

Attributes: None.

Example:

```
<Anchor xmlns='syncml:metinf'>
  <Last xmlns='syncml:metinf'>20000824T133000Z</Last>
  <Next xmlns='syncml:metinf'>20000824T221300Z</Next>
</Anchor>
```

5.4 Mark

Usage: Specifies a meta-information "mark" on the data object.

Parent Elements: `MetInf`

Restrictions: The content information for this element type SHOULD BE one of `draft`, `final`, `delete, undelete`, `read`, `unread`.

When this meta-information is specified repetitively in a hierarchically of element types (e.g., in a SyncML collection, as well as the items in the collection), then the meta-information specified in the lowest level element type takes precedence.

This element type is used to set the meta-information characteristics of a data object, such as the draft/final, delete/undelete, read/unread marks on a folder item or mail item.

Content Model:

```
(#PCDATA)
```

Attributes: None.

Example: The following example illustrates how the element type is used within the SyncML DTD to specify meta-information about a data object specified by the Item element type.

```
<Update>
  <CmdID>10</CmdID>
  <Item>
    <Source>
      <LocName>jsmith</LocName>
      <LocURI>host1.com-19991208T234504-001</LocURI>
    </Source>
    <Meta>
      <Mark xmlns='syncml:metinf'>unread</Mark>
    <Meta>
  </Item>
</Update>
```

5.5 MaxMsgSize

Usage: Specifies the maximum byte size of any response message to a given SyncML request.

Parent Elements: Root element type.

Restrictions: The element type appears in the `Meta` element in the `SyncHdr` of a SyncML request to specify the maximum size of any subsequent response messages. The element type is usually specified by a SyncML client, but can also be specified by a SyncML server.

This element type value is applicable for the remainder of the synchronization session, unless it is specified again.

The element type value is the text string representation of the maximum, decimal byte size of any response message.

In order to use the elements from the MetInf name space, the root element does not need to be specified.

Content Model:

```
(#PCDATA)
```

Attributes: None.

Example: Normally, the root element type does not appear in a SyncML `Meta` element type.

```
<MaxMsgSize xmlns='syncml:metinf'>1023</MaxMsgSize>
```

5.6 MetInf

Usage: Specifies the root element for the SyncML meta-information document.

Parent Elements: Root element type.

Restrictions: In order to use the elements from the MetInf name space, the root element does not need to be specified. The element type can appear in the `Meta` element of a SyncML document to allow for declaring a default name space.

Content Model:

```
(Format?, Type?, Mark?, Size?, Anchor?, Version, NextNonce?)
```

Attributes: None.

Example: Normally, the root element type does not appear in a SyncML `Meta` element type.

```
<mi:MetInf xmlns:mi='syncml:metinf'>
  <mi:Type>text/calendar</mi:Type>
  <mi:Format>chr</mi:Type>
  <mi:Size>877566</mi:Size>
  <mi:Version>20000714T082300Z</mi:Version>
</mi:MetInf>
```

5.7 Next

Usage: Specifies the synchronization state information (i.e., sync anchor) for the current synchronization session.

Parent Elements: `MetInf`

Restrictions: The value MUST specify either an UTC based ISO 8601 [1] date/time stamp or a monotonically increasing numeric integer string. If a date/time stamp, then the text MUST be in the complete representation, basic format defined by [ISO8601].

Determination of the ordinal sequence of the version of an existing object in the recipient and the version of the object can be made by comparing the content information of the object with the value on the existing object.

Content Model:

```
(#PCDATA)
```

Attributes: None.

Example:

```
<Anchor xmlns='syncml:metinf'>
  <Last xmlns='syncml:metinf'>20000824T133000Z</Last>
  <Next xmlns='syncml:metinf'>20000824T221300Z</Next>
</Anchor>
```

5.8 NextNonce

Usage: Specifies the nonce string to be used in any subsequent communication.

Parent Elements: `MetInf`

Restrictions: The nonce string MUST be further re-formatted using the Base64 algorithm.

This element type is used to specify the next nonce string that is to be used in any subsequent SyncML message. For example, a SyncML server specifies this element type to tell the SyncML client to change it's nonce to a new value.

Nonce strings are used in the SyncML "MD5 Digest" scheme of authentication credentials.

Content Model:

```
(#PCDATA)
```

Attributes: None.

Example:

```
<Meta>
  <NextNounce
xmlns='syncml:metinf'>QWxhZGRpbjpvcGVuIHNlc2FtZQ==</NextNounce>
<Meta>
```

5.9 Size

Usage: Specifies the byte size of a data object.

Parent Elements: `MetInf`

Restrictions: The byte size is specified as the numeric text equivalent of the byte count of the data object.

Content Model:

```
(#PCDATA)
```

Attributes: None

Example: The following example illustrates how the element type is used within the SyncML DTD to specify meta-information about the byte size of the Item element type.

```
<Item>
  <Target><LocURI>4</LocURI>
  <Meta>
  <Size xmlns='syncml:metinf'>10</Size>
  </Meta>
  <Data>John Smith</Data>
</Item>
```

5.10 Type

Usage: Specifies the media type of the content information in the `Data` element.

Parent Elements: `MetInf`

Restrictions: If this element is missing, then the default content-type is `text/plain`. The content information for this element type SHOULD BE a registered MIME content-type. Alternatively, a URN can be used to specify the media type.

Content Model:

```
(#PCDATA)
```

Attributes: None

Example: The following example illustrates how the element type is used within a SyncML message to specify meta-information about the media type of the content information in the Item element type.

```
<Item>
  <Target><LocURI>3</LocURI></Target>
  <Meta>
    <Type xmlns='syncml:metinf'>text/directory;profile=vCard</Type>
  </Meta>
  <Data>BEGIN:VCARD
VERSION:3.0
FN:Jim Smith
N:Smith;Jim
TEL;TYPE=WORK,VOICE,FAX:+1-919-555-1234
EMAIL;TYPE=INTERNET,WORK:Jim_Smith@mail.host.com
END:VCARD
  </Data>
</Item>
```

5.11 Version

Usage: Specifies the revision identifier of a data object.

Parent Elements: `MetInf`

Restrictions: The value MUST specify either an UTC based ISO 8601 [1] date/time stamp or a monotonically increasing numeric integer string. If a date/time stamp, then the text MUST be in the complete representation, basic format defined by [ISO8601]. Determination of the ordinal sequence of the version of an existing object in the recipient and the version of the object can be made by comparing the content information of the object with the value on the existing object.

If not present, then the object doesn't currently have a revision version identifier. When the element type is missing, this SHOULD NOT be interpreted as meaning the original version of the data object.

Content Model:

```
(#PCDATA)
```

Attributes: None.

Example: The following example illustrates how the element type is used within the SyncML DTD to specify meta-information about the synchronization version of the Item element type.

```
<Item>
  <Target><LocURI>4</LocURI>
  <Meta>
  <Version xmlns='syncml:metinf'>20000301T133000Z</Version>
  </Meta>
  <Data>John Smith</Data>
</Item>
```

6 DTD Definition

```
<!--Copyright Notice

Copyright (c) 2000 Ericsson, IBM, Lotus, Matsushita Communications
Industrial Co., Motorola, Nokia, Palm, Inc., Psion, Starfish Software. All
Rights Reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions are
met:

1. Redistributions of source code must retain the above copyright notice,
this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright
notice, this list of conditions and the following disclaimer in the
documentation and/or other materials provided with the distribution.

3. The end-user documentation included with the redistribution, if any,
must include the following acknowledgement: This product includes software
developed by The SyncML Initiative.

Alternatively, this acknowledgment may appear in the software itself, if
and wherever such third-party acknowledgment normally appears.
```

```
4. The name of the authors, jointly or severally, must not be used to
endorse or promote products derived from this software without specific
prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHORS ``AS IS'' AND ANY EXPRESS OR
IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES
OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE OR NON-
INFRINGEMENT ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHORS BE LIABLE FOR
ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER
CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
SUCH DAMAGE.

-->

<!-- This DTD defines a sequence of meta-information that is used within
the SyncML DTD.

This DTD is to be identified by the URI string "syncml:metainf".

Single element types from this name space can be referenced as follows:

    <element xmlns='syncml:metinf'>blah, blah</element>

-->

<!-- Root or Document Element and -->

<!ELEMENT MetInf (Format?, Type?, Mark?, Size?, Anchor?, Version?,
NextNonce?, MaxMsgSize?)>

<!-- Format or encoding type -->

<!ELEMENT Format (#PCDATA)>

<!-- Element specific type specification -->

<!ELEMENT Type (#PCDATA)>

<!-- Mark -->

<!ELEMENT Mark (#PCDATA)>

<!-- Byte count -->

<!ELEMENT Size (#PCDATA)>

<!-- Data versioning info -->

<!ELEMENT Anchor (Last?, Next)>

<!ELEMENT Last (#PCDATA)>

<!ELEMENT Next (#PCDATA)>
```

```
<!ELEMENT Version (#PCDATA)>

<!ELEMENT NextNonce (#PCDATA)>

<!ELEMENT MaxMsgSize (#PCDATA)>

<!-- End of DTD →
```

7 WBXML Definition

The following tables define the token assignments for the mapping of the Meta-Information DTD element types into WBXML as defined by [4].

7.1 Code Space and Code Page Definitions

The element types from the Meta-Information DTD (i.e., name space) are only intended to be used within a SyncML document. Hence, the elements from the Meta-Information DTD are always maped into the single SyncML WBXML code space.

DTD Name	WBXML PUBLICID Token (Hex Value)	Formal Public Identifier
SyncML	FD1	-//SYNCML//DTD SyncML 1.0//EN

The SyncML DTD is assigned the WBXML document public identifier (i.e., the "publicid" WBXML BNF production) associated with the `FD1` token.

The element types from the Meta-Information DTD utilize the code page x02 (two) within the SyncML Code Space.

DTD Name	WBXML Code Page Token (Hex Value)	Formal Public Identifier
SyncML	00	-//SYNCML//DTD SyncML 1.0//EN
MetInf	02	-//SYNCML//DTD MetInf 1.0//EN

7.2 Token Definitions

The following WBXML token codes represent element types (i.e., tags) form code page x02 (two), Meta-Information DTD.

Element Type Name	WBXML Tag Token (Hex Value)
Anchor	05
Format	06
Last	07
Mark	08
MetInf	09
Next	0A
NextNonce	0B
Size	0C
Type	0D
Version	0E
MaxMsgSize	0F

The WBXML token codes from code page x01 (one) represent the DevInf DTD. These token definitions are defined in the DevInf DTD specification.

The WBXML token codes from code page x02 (two) represent the MetInf DTD. These token definitions are defined in the MetInf DTD specification.

8 Static Conformance Requirements

Static conformance requirements (SCR) specify the features that are optional, mandatory and recommended within implementations conforming to this specification.

Simple tables are used to specify this information

In these tables, optional features are specified by a "MAY", mandatory features are specified by a "MUST" and recommended features are specified by a "SHOULD".

The following specifies the static conformance requirements for the SyncML Meta-Information element types for devices conforming to this specification.

Element Type	Support of Synchronization Server		Support of Synchronization Client	
	Sending	Receiving	Sending	Receiving
Anchor	MUST	MUST	MUST	MUST
Format	MUST	MUST	MUST	MUST
Last	MUST	MUST	MUST	MUST
Mark	MAY	MAY	MAY	MAY
MaxMsgSize	MAY	MUST	MAY	MUST
MetInf	MUST	MUST	MUST	MUST
Next	MUST	MUST	MUST	MUST
NextNonce	MUST	MUST	MUST	MUST
Size	MAY	MAY	MAY	MAY
Type	MUST	MUST	MUST	MUST
Version	MUST	MUST	MAY	MAY

9 References

[1] Data elements and interchange formats - Information interchange - Representation of dates and times, ISO.

[2] Key words for use in RFCs to Indicate Requirement Levels, IETF.

[3] SyncML Representation Protool DTD, SyncML.

[4] WAP Binary XML Content Format Specification, WAP Forum.

[5] Extensible Mark-up Language (XML) 1.0, W3C.