SyncML Meta-Information DTD, version 1.1

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., Ltd.
Motorola
Nokia
Openwave
Starfish Software
Symbian

Revision History

Revision	Date	Comments
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.
V1.0c	2000-11-24	Candidate for the Gold release. Release, Code page for MetInf changed, New elements; EMI, FreeID, FreeMem, Mem, SharedMem added
V1.0	2000-12-07	The candidate version for the final release.
V1.0.1a	2001-05-23	Errata rolled in.
V1.0.1b	2001-05-30	Minor corrections.
V1.0.1	2001-05-30	Fixed header and copyright dates.
V1.1	2002-02-15	The candidate version for v1.1 release. Update SyncML Initiative members. Update copyright and intellectual property notices. Minor typographic changes. Add new format types. Add MaxObjSize.

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 interpreted as described in [2].

Any reference to components of the Meta 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_v11_20020215.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

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

5.2 EMI

Usage: Specifies the non-standard, experimental meta information (EMI) extensions supported by the device. The extensions are specified in terms of the XML element type name and the value.

Parent Elements: MetInf

Restrictions: The EMI element type MUST specify the extension element name. It may also specify one or more enumerated values. Multiple non-standard extensions can be specified by specifying the EMI element type multiple times. This element type is optional.

Content Model:

```
(#PCDATA)
```

Attributes: None.

Example: The following example specifies a non-standard extension with a value of "Confidential".

```
<EMI xmlns='syncml:metinf'>Confidential</EMI>
```

5.3 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 `bin`, `bool`, `b64`, `chr`, `int`, `node`, `null` or `xml`. If the element type is missing, the default value is `chr`. If the value is `bin`, then the format of the content is binary data. If the value is `bool`, then the format of the content is either true or false. 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 `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 `node`, then the content represents an interior object in the management tree. 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 `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>
```

5.4 FreeID

Usage: Specifies the number of free item identifiers available for adding new items to a local datastore or device.

Parent Elements: Mem

Restrictions: Content information MUST be specified as the decimal integer number of item identifiers that are currently available for new items in the local datastore. If the Mem element type is present, then the FreeID element type is required.

Content Model:

```
(#PCDATA)
```

Attributes: None.

Example: The following is an example of 25 free item identifiers.

```
<FreeID xmlns='syncml:metinf'>25</FreeID>
```

5.5 FreeMem

Usage: Specifies the amount of free memory, in bytes, available in a local database or a device.

Parent Elements: Mem

Restrictions: Content information MUST be specified as the decimal integer number of free bytes of memory currently available in the local database. If the Mem element type is present, then the FreeMem element type is required.

Content Model:

```
(#PCDATA)
```

Attributes: None.

Example: The following is an example for 1022 free bytes.

```
<FreeMem xmlns='syncml:metinf'>1022</FreeMem>
```

5.6 Last

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

Parent Elements: `Anchor`

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

5.7 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>
```

5.8 MaxMsgSize

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

Parent Elements: MetInf.

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>
```

5.9 MaxObjSize

Usage: Specifies the maximum size in bytes of a data object that the device is able to receive.

Parent Elements: MetInf.

Restrictions: The element type appears in the `Meta` element of a SyncML request to specify the maximum size of the largest object it is capable of receiving in any subsequent response messages. This element type value is applicable for the remainder of the synchronization session.

The element type value is the text string representation of the maximum, decimal byte size without leading zeroes of any object.

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: Device that can receive a maximum object of 10K bytes.

```
<MaxObjSize xmlns='syncml:metinf'>10240</MaxObjSize>
```

5.10 Mem

Usage: Specifies the maximum free memory and item identifier for a source (e.g., a datastore or a device.

Parent Elements: MetInf

Restrictions: The element type is optional.

Content Model:

```
(SharedMem?, FreeMem, FreeID)
```

Attributes: None.

Example: The following example specifies a shared datastore memory within a Sync command.

```
<Sync>
  <CmdID>1</CmdID>
  <Target><LocURI>./contacts/james_bond</LocURI></Target>
  <Source><LocURI>./dev-contacts</LocURI></Source>
  <Meta>
    <Mem xmlns='syncml:metinf'>
      <SharedMem xmlns='syncml:metinf'/>
      <FreeMem xmlns='syncml:metinf'>8100</FreeMem>
      <!--Free memory (bytes) in Calendar database on a device -->
      <FreeID xmlns='syncml:metinf'>81</FreeID>
      <!--Number of free records in Calendar database-->
    </Mem>
  </Meta>
...
</Sync>
```

5.11 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?, MaxMsgSize?,
MaxObjSize?, EMI*, Mem?)
```

Attributes: None.

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

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

5.12 Next

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

Parent Elements: `Anchor`

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>
```

5.13 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 its 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>
  <NextNonce
xmlns='syncml:metinf'>QWxhZGRpbjpvcGVuIHNlc2FtZQ==</NextNonce>
</Meta>
```

5.14 SharedMem

Usage: Specifies if the datastore memory is shared. If the memory is shared, the actual memory space is used also by other datastores, and the actual memory space may be more limited than in theory it might be.

Parent Elements: Mem

Restrictions: The content of this element MUST be empty. This element type is used as a flag, and if this element type is present, then the given datastore memory is shared. This element is optional.

Content Model:

```
EMPTY
```

Attributes: None.

Example: The following is an example of shared datastore memory.

```
<Mem xmlns='syncml:metinf'>
  <SharedMem xmlns='syncml:metinf'/>
  <FreeMem xmlns='syncml:metinf'>25601</FreeMem>
  <FreeID xmlns='syncml:metinf'>200</FreeID>
</Mem>
```

5.15 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>
```

5.16 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>
```

5.17 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) Ericsson, IBM, Lotus, Matsushita Communication Industrial Co., Ltd., Motorola, Nokia, Openwave, Palm, Inc., Psion, Starfish Software, Symbian and others (2000-2002). All Rights Reserved.

Implementation of all or part of any Specification may require licenses under third party intellectual property rights, including without limitation, patent rights (such a third party may or may not be a Supporter). The Sponsors of the Specification are not responsible and shall not be held responsible in any manner for identifying or failing to identify any or all such third party intellectual property rights.

THIS DOCUMENT AND THE INFORMATION CONTAINED HEREIN ARE PROVIDED ON AN "AS IS" BASIS WITHOUT WARRANTY OF ANY KIND AND ERICSSON, IBM, LOTUS, MATSUSHITA COMMUNICATION INDUSTRIAL CO. LTD., MOTOROLA, NOKIA, OPENWAVE, STARFISH SOFTWARE, SYMBIAN AND ALL OTHER SYNCML SPONSORS DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT SHALL ERICSSON, IBM, LOTUS, MATSUSHITA COMMUNICATION INDUSTRIAL CO., LTD., MOTOROLA, NOKIA, OPENWAVE, STARFISH SOFTWARE, SYMBIAN OR ANY OTHER SYNCML SPONSOR BE LIABLE TO ANY PARTY FOR ANY LOSS OF PROFITS, LOSS OF BUSINESS, LOSS OF USE OF DATA, INTERRUPTION OF BUSINESS, OR FOR DIRECT, INDIRECT, SPECIAL OR EXEMPLARY, INCIDENTAL, PUNITIVE OR CONSEQUENTIAL DAMAGES OF ANY KIND IN CONNECTION WITH THIS DOCUMENT OR THE INFORMATION CONTAINED HEREIN, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH LOSS OR DAMAGE.

The above notice and this paragraph must be included on all copies of this document that are made.

Attention is called to the possibility that implementation of this specification may require use of subject matter covered by patent rights. By publication of this specification, no position is taken with respect to the existence or validity of any patent rights in connection therewith. The SyncML Initiative is not responsible for identifying patents having necessary claims for which a license may be required by a SyncML Initiative specification or for conducting inquiries into the legal validity or scope of those patents that are brought to its attention.

A patent/application owner has filed a statement of assurance that it will grant licenses under these rights without compensation or under reasonable rates and nondiscriminatory, reasonable terms and conditions to all applicants desiring to obtain such licenses. The SyncML Initiative makes no representation as to the reasonableness of rates and/or terms and conditions of the license agreements offered by patent/application owners. Further information may be obtained from the

```
SyncML Initiative Executive Director.

-->

<!-- This DTD defines a sequence of meta-information that is used within
the SyncML DTD. The DTD is to be identified by the URN string
"syncml:metinf". Single element types from this name space can be
referenced as follows:

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

Comments should be sent to syncml@syncml.org -->

<!-- Root Element -->

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

<!-- 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)>

<!ELEMENT MaxObjSize (#PCDATA)>

<!-- Experimental Meta Information extension -->

<!ELEMENT EMI (#PCDATA)>

<!-- Dynamic Memory -->

<!ELEMENT Mem (SharedMem?, FreeMem, FreeID)>
```

```
<!-- Free Memory in the number of identifiers -->

<!ELEMENT FreeID (#PCDATA)>

<!-- Free Memory in bytes -->

<!ELEMENT FreeMem (#PCDATA)>

<!-- Shared Memory -->

<!ELEMENT SharedMem EMPTY>

<!-- 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 mapped into the single SyncML WBXML code space.

DTD Name	WBXML PUBLICID Token (Hex Value)	Formal Public Identifier
SyncML	FD1	-//SYNCML//DTD SyncML 1.1//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 x01 (one) within the SyncML Code Space.

DTD Name	WBXML Code Page Token (Hex Value)	Formal Public Identifier
SyncML	00	-//SYNCML//DTD SyncML 1.1//EN
MetInf	01	-//SYNCML//DTD MetInf 1.1//EN

7.2 Token Definitions

The following WBXML token codes represent element types (i.e., tags) form code page x01 (one), Meta-Information DTD.

Element Type Name	WBXML Tag Token (Hex Value)
Anchor	05
EMI	06
Format	07
FreeID	08
FreeMem	09
Last	0A
Mark	0B
MaxMsgSize	0C
MaxObjSize	15
Mem	0D
MetInf	0E
Next	0F
NextNonce	10
SharedMem	11
Size	12
Type	13
Version	14

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".

An implementation which does not include a particular option MUST be prepared to interoperate with another implementation which does include the option, though perhaps with reduced functionality.

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
EMI	MAY	MAY	MAY	MAY
Format	MUST	MUST	MUST	MUST
FreeID	MAY	MUST	SHOULD	MAY
FreeMem	MAY	MUST	SHOULD	MAY
Last	MUST	MUST	MUST	MUST
Mark	MAY	MAY	MAY	MAY
MaxMsgSize	MAY	MUST	MAY	MUST
MaxObjSize	MAY	MUST	MAY	MUST
Mem	MAY	MUST	SHOULD	MAY
MetInf	MUST	MUST	MUST	MUST
Next	MUST	MUST	MUST	MUST
NextNonce	MUST	MUST	MAY	MUST
SharedMem	MAY	MUST	SHOULD	MAY
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 Protocol, SyncML.

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

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