Contents

1. SCOPE ........................................................................................................................................................................5

2. REFERENCES ...................................................................................................................................................................6
   2.1 NORMATIVE REFERENCES .................................................................................................................................6
   2.2 INFORMATIVE REFERENCES ...............................................................................................................................6

3. TERMINOLOGY AND CONVENTIONS .......................................................................................................................7
   3.1 CONVENTIONS......................................................................................................................................................7
   3.2 DEFINITIONS......................................................................................................................................................7
   3.3 ABBREVIATIONS .............................................................................................................................................7

4. INTRODUCTION ..........................................................................................................................................................8

5. XML USAGE .................................................................................................................................................................9
   5.1 INCLUDING EMAIL OBJECT IN OTHER XML DOCUMENTS ..................................................................................9
   5.2 XML NAMESPACE................................................................................................................................................9
   5.3 XML ATTRIBUTES .........................................................................................................................................9
   5.4 WBXML ...............................................................................................................................................................9

6. MIME USAGE .............................................................................................................................................................9

7. DATA TYPES ...........................................................................................................................................................10
   7.1 DATETIME .......................................................................................................................................................10
   7.2 BOOL ................................................................................................................................................................10
   7.3 TEXT .................................................................................................................................................................11
   7.4 INT ..................................................................................................................................................................11

8. MARK-UP LANGUAGE DESCRIPTION ....................................................................................................................13
   8.1 EMAIL ...............................................................................................................................................................13
   8.2 READ ...............................................................................................................................................................13
   8.3 FORWARDED ................................................................................................................................................13
   8.4 REPLIED .........................................................................................................................................................13
   8.5 RECEIVED ..................................................................................................................................................13
   8.6 CREATED.......................................................................................................................................................14
   8.7 MODIFIED ..................................................................................................................................................14
   8.8 DELETED ......................................................................................................................................................14
   8.9 FLAGGED ......................................................................................................................................................14
   8.10 EMAILITEM ................................................................................................................................................15
       8.10.1 enc.........................................................................................................................................................15
   8.11 EXTENSION FIELDS ..........................................................................................................................................16
       8.11.1 Unique naming......................................................................................................................................16
       8.11.2 Ext..........................................................................................................................................................16
       8.11.3 XNam..................................................................................................................................................16
       8.11.4 XVal..................................................................................................................................................17

9. DTD ...........................................................................................................................................................................18

10. SYNCML DATA SYNCHRONIZATION USAGE ......................................................................................................20
    10.1 CTCAP ..........................................................................................................................................................20
        10.1.1 enc .......................................................................................................................................................21
        10.1.2 texttype ...............................................................................................................................................21
        10.1.3 attachtype ............................................................................................................................................21
    10.2 DATA SYNC RECORD AND FIELD LEVEL FILTERING .....................................................................................22
        10.2.1 Email Media Object Filter ..................................................................................................................22
    10.3 EMAIL OBJECT REPLACE EXAMPLE ...........................................................................................................27

APPENDIX A. CHANGE HISTORY (INFORMATIVE) ....................................................................................................29
    A.1 APPROVED VERSION HISTORY ....................................................................................................................29
    A.2 DRAFT/CANDIDATE VERSION 1.2 HISTORY ................................................................................................29
Tables

Table 1 enc attribute values ....................................................................................................................................................15
1. Scope

The **email** data object is presented in this document. The content-specific aspects of synchronization (filtering keywords, etc…) are listed and clarified.
2. References

2.1 Normative References


2.2 Informative References
3. Terminology and Conventions

3.1 Conventions

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in [RFC2119].

All sections and appendixes, except “Scope” and “Introduction”, are normative, unless they are explicitly indicated to be informative.

3.2 Definitions

Data type  The schema used to represent a data object (e.g., text/calendar MIME content type for an iCalendar representation of calendar information or text/directory MIME content type for a vCard representation of contact information).

3.3 Abbreviations

OMA  Open Mobile Alliance
4. Introduction

The email data object can be used to represent an interpersonal electronic mail object.
5. XML Usage

The email objects are represented in a mark-up language defined by [XML]. The email is an XML application. The email DTD (Document Type Definition) defines the XML document type used to represent email object. The mail DTD can be found in Section 9, but it is not necessary to read the DTD in order to understand it.

Email objects are specified using well-formed XML. However, the email need not be valid XML. That is, the email objects do not need to specify the XML declaration or prolog. They only need to specify the body of the XML document. This restriction allows for the email objects to be specified with greater terseness than well-formed, valid XML documents.

5.1 Including email object in other XML documents

When using the email object as content of another XML document element the mark-up characters MUST be properly escaped or the CDATA sections MUST be used. See [XML] specification for details on character escaping and usage of CDATA sections. See section 10.3 of this document for an example.

5.2 XML Namespaces

Email objects to date have no elements that may include elements of the other namespaces.

5.3 XML Attributes

In order to simplify the implementation of the email on small devices, the email objects have been intentionally designed to use the XML elements only. Currently no XML attributes are being defined for email objects.

5.4 WBXML

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, email uses shortened element type. This provides a minor reduction in verbosity.

Additionally, the email objects can be encoded in a tokenized, binary format defined by [WBXML]. The use of [WBXML] format is external to specification of the email and should be transparent to any application. The combination of the use of shortened element type names and an alternative binary format makes email competitive, from a compressed format perspective, with alternative, but private, binary representations.

6. MIME Usage

The [RFC2045] Internet standard provides an industry-accepted mechanism for identifying different content types. The email object is identified by a MIME media type. The application/vnd.omads-email+xml MIME content type MUST be used to indicate the email object wherever such indication is required.
7. Data types

The following basic data type definitions are provided for referencing from other parts of this document.

7.1 datetime

Usage: This value type is used to identify values that specify a precise calendar date and time of day.

Description:

The datetime data type is used to identify values that contain a precise calendar date and time of day. The format is based on the [ISO 8601] complete representation, basic format for a calendar date and time of day. The text format is a concatenation of the "date", followed by the LATIN CAPITAL LETTER T character (US-ASCII decimal 84) time designator, followed by the "time" format.

The datetime data type expresses time values in two forms:

The form of date and time with UTC offset MUST NOT be used. For example, the following is not valid for a date-time value:

```
...<datefield>19980119T230000-0800</datefield> <!-- Invalid time format -->
...```

**FORM #1: DATE WITH LOCAL TIME**

The date with local time form is simply a date-time value that does not contain the UTC designator nor does it reference a time zone. For example, the following represents January 18, 1998, at 11 PM:

```
...<datefield>19980118T230000</datefield> <!-- January 18, 1998, 11 PM -->
...```

This notation of datetime type is to be used by devices that have no knowledge of the time zone in which they operate. In this case, the datetime value that is being transferred is usually the same as the value that is being stored and shown to the user in the application UI.

**FORM #2: DATE WITH UTC TIME**

The date with UTC time, or absolute time, is identified by a LATIN CAPITAL LETTER Z suffix character (US-ASCII decimal 90), the UTC designator, appended to the time value. For example, the following represents January 19, 1998, at 0700 UTC:

```
...<datefield>19980119T070000Z</datefield> <!-- January 19,1998,07:00 UTC -->
...```

7.2 bool

Usage: To be used for Boolean type fields
Restrictions: A text value that MUST be either “true” to indicate Boolean “true” or “false” to indicate “false”. If the field is not present its value is assumed to be “false”.

Example:

```xml
...<booleanfield>true</booleanfield> <!-- the field is set to “true” --> ...
```

7.3 text

Usage: To be used for textual fields

Restrictions: If the field is not present its value is assumed to be an empty string.

Example:

```xml
...<textfield>Hello World!</textfield> ...
```

7.4 int

Usage: To be used for integer numeric fields.

Restrictions: The format for the integer values is defined here in an ABNF notation [RFC2234].

```plaintext
nonzero-digit = "1"/ "2"/ "3"/ "4"/ "5"/ "6"/ "7"/ "8"/ "9"
ochal-digit = "0"/ "1"/ "2"/ "3"/ "4"/ "5"/ "6"/ "7"
hexadecimal-digit = "0"/ "1"/ "2"/ "3"/ "4"/ "5"/ "6"/ "7"/ "8"/ "9"
/"a"/ "b"/ "c"/ "d"/ "e"/ "f"
/"A"/ "B"/ "C"/ "D"/ "E"/ "F"
decimal-constant = nonzero-digit *("0" / nonzero-digit)
hexadecimal-constant = ("0x" / "0X") 1*hexadecimal-digit
```
octal-constant = "0" 1*octal-digit

integer-value = *1("+" / "-") (decimal-constant / hexadecimal-constant / octal-constant)

Example:

...<negativevalue>-1234</negativevalue>
<positivevalue>1234</positivevalue>
<anotherpositivevalue>+0xfffabc5</anotherpositivevalue>
<octal>010</octal> <!-- octal value equivalent to decimal 8 -->
...

Error! Reference source not found.
Error! Reference source not found.
8. Mark-up Language Description

8.1 Email

Usage: Indicates the beginning of the object

Parent elements: None

Content model:

```
Email (read?, forwarded?, replied?, received?, created?, modified?,
deleted?, flagged?, emailitem?, Ext*)
```

8.2 read

Usage: Specifies whether the email has been read

Parent elements: Email

Restrictions: bool type field as specified in section 7.2.

Content model:

```
read (#PCDATA)
```

8.3 forwarded

Usage: Specifies whether the email has been forwarded

Parent elements: Email

Restrictions: bool type field as specified in section 7.2.

Content model:

```
forwarded (#PCDATA)
```

8.4 replied

Usage: Specifies whether the email has been replied

Parent elements: Email

Restrictions: bool type field as specified in section 7.2.

Content model:

```
replied (#PCDATA)
```

8.5 received

Usage: Specifies the date and time when the email was received

Parent elements: Email

Restrictions: datet ime type field as specified in section 7.1.
Content model:

8.6 created

Usage: Specifies the date and time when the email was created

Parent elements: Email

Restrictions: datetime type field as specified in section 7.1.

Content model:

8.7 modified

Usage: Specifies the date and time when any of the email object fields was last changed. Initially (after the draft item is created or a new message received) the modified date is equal to created date. As the state of the object changes (e.g. draft message gets submitted or unread message gets read) the modified field gets updated to reflect the date and time of the last state change.

Parent elements: Email

Restrictions: Error! Reference source not found. type field as specified in section 7.1.

Content model:

8.8 deleted

Usage: Specifies whether the email has been scheduled for deletion

Parent elements: Email

Restrictions: bool type field as specified in section 7.2.

Content model:

8.9 flagged

Usage: Specifies whether the email has been flagged

Parent elements: Email

Restrictions: bool type field as specified in section 7.2.

Content model:
8.10 emailitem

Usage: contains the email header and body as specified in RFC822 / RFC2822

Parent elements: Email

Restrictions: The supporters MUST implement this property. If the field is not present within the object, the object is assumed to have an empty body.

Content model:

```
emailitem (#PCDATA)
```

8.10.1 enc

Usage: declares the mechanism used to encode the content of the element. This is used to avoid corrupting the XML content of the element with the presence of characters which do not belong to the valid ranges of characters as defined by the [XML]

Parent elements: emailitem

Restrictions: The following table lists standard enc values that MUST be understood by the conforming implementations.

<table>
<thead>
<tr>
<th>Enc</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;quoted-printable&quot;</td>
<td>The contents of the element is encoded using quoted-printable algorithm as specified by the section 6.7 of the [RFC2045]</td>
</tr>
<tr>
<td>&quot;base64&quot;</td>
<td>The contents of the element is encoded using quoted-printable algorithm as specified by the section 6.8 of the [RFC2045]</td>
</tr>
</tbody>
</table>

Table 1 enc attribute values

If the enc attribute is not present, the content is assumed to have no encoding.

The implementations SHOULD NOT use other enc attribute values than specified in the [Table 1]. In case of other enc values the usage of these encodings MUST conform to the rules defined by the[RFC2045] for Content-Transfer-Encoding.

Content model:

```
emailitem enc (CDATA #IMPLIED)
```
8.11 Extension fields

8.11.1 Unique naming

If an extension field is required, the following naming convention MUST be followed in order to prevent undesirable field name collisions.

\[
\begin{align*}
x-name & = "x-" \text{vendorid "-" 1*(ALPHA / DIGIT / ")")} ; \text{field name} \\
\text{vendorid} & = 3*(ALPHA / DIGIT) ; \text{Vendor identification} \\
\text{ALPHA} & = %x41-5A / %x61-7A ; \text{A-Z / a-z} \\
\text{DIGIT} & = %x30-39 ; \text{0-9}
\end{align*}
\]

8.11.2 Ext

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

Parent Elements: Email

Restrictions: The Ext 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 Ext element type multiple times. This element type is optional.

Content Model:

\[
\text{Ext (XNam, XVal*)}
\]

Attributes: None.

Example: The following example specifies a non-standard extension, named "CliVer" for a fictitious company, Foo, which takes values of "5.0", "5.01" or "5.02".

\[
<\text{Ext}>
  <\text{XNam} x-Foo-CliVer</XNam>
  <\text{XVal}>5.0</XVal>
  <\text{XVal}>5.01</XVal>
  <\text{XVal}>5.02</XVal>
</\text{Ext}>
\]

8.11.3 XNam

Usage: Specifies the name of one of the extension element types.

Parent Elements: Ext

Restrictions: The element type is required whenever an Ext element is present.

Content Model:

\[
\text{XNam (#PCDATA)}
\]

Attributes: None.
Example:

```xml
<Ext>
  <XNam>x-Foo-CliVer</XNam>
  <XVal>5.0</XVal>
  <XVal>5.01</XVal>
  <XVal>5.02</XVal>
</Ext>
```

8.11.4 XVal

Usage: Specifies one of the valid values for an extension element type.

Parent Elements: Ext

Restrictions:

Content Model:

```
XVal (#PCDATA)
```

Attributes: None.

Example:

```xml
<Ext>
  <XNam>x-Foo-CliVer</XNam>
  <XVal>5.0</XVal>
  <XVal>5.01</XVal>
  <XVal>5.02</XVal>
</Ext>
```
9. DTD

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!-- Root Element -->
<!ELEMENT Email (read?, forwarded?, replied?, received?, created?, modified?, deleted?, flagged?, emailitem?, Ext*)>
<!ELEMENT read (#PCDATA)>  
<!ELEMENT forwarded (#PCDATA)>  
<!ELEMENT replied (#PCDATA)>  
<!ELEMENT received (#PCDATA)>  
<!ELEMENT created (#PCDATA)>  
<!ELEMENT modified (#PCDATA)>  
<!ELEMENT deleted (#PCDATA)>  
<!ELEMENT flagged (#PCDATA)>  
<!ELEMENT emailitem (#PCDATA)>  
<!ATTLIST emailitem enc CDATA #IMPLIED>  
<!ELEMENT Ext (XNam, XVal*)>  
<!ELEMENT XNam (#PCDATA)>  
<!ELEMENT XVal (#PCDATA)>  
<!-- End of DTD Definition -->
```
10. SyncML Data Synchronization Usage

The following sections describe the content-specific recommendations for using the data synchronization [DSREPU] protocol with email data objects.

10.1 CTCap

Refer to [DSDEVDTD] for further details on the specification of the Device Information DTD.

```xml
<CTCap>
  <CTType>application/vnd.omads-email+xml</CTType>
  <Property>
    <PropName>read</PropName>
    <DataType>bool</DataType>
    <DisplayName>Read</DisplayName>
  </Property>
  <Property>
    <PropName>forwarded</PropName>
    <DataType>bool</DataType>
    <DisplayName>Forwarded</DisplayName>
  </Property>
  <Property>
    <PropName>replied</PropName>
    <DataType>bool</DataType>
    <DisplayName>Replied</DisplayName>
  </Property>
  <Property>
    <PropName>received</PropName>
    <DataType>datetime</DataType>
    <DisplayName>Date received</DisplayName>
  </Property>
  <Property>
    <PropName>created</PropName>
    <DataType>datetime</DataType>
    <DisplayName>Date created</DisplayName>
  </Property>
</CTCap>
```
10.1.1 enc

The PropParam element type with the value of enc MUST utilise ValEnum element type to indicate the supported encoding algorithms. The example above (10.1) illustrates a section of CTCap that lists the standard encoding algorithms as supported.

10.1.2 texttype

This PropParam specifies which content types are recognized for textual part of the message. If no enumeration values specified, means that all types are allowed. If texttype is not present within the CTCap, means that textual information is not required by the sender. This particular case is useful for scenarios in which sender chooses to sync only limited information for the messages (e.g. [RFC2822] headers only).

10.1.3 attachtype

This PropParam specifies which content types are recognized for attached objects. If no enumeration values specified, means that all types are allowed. If attachtype is not present within the CTCap, means that attachment objects are not
required by the sender. This particular case is useful for scenarios in which sender chooses to sync only limited information for the messages (e.g. [RFC2822] headers only).

10.2 Data Sync Record and Field Level Filtering

10.2.1 Email Media Object Filter

Filtering for email objects can be specified using both Record and Field elements.

In the case of Record elements, the set of recommended keywords to support are as follows:

\[ \text{ct-filter-keyword} = \text{email-field} \mid \text{search-keyword} \]

10.2.1.1 email-field

In addition to the actual email fields defined in this document, the following set of keywords is recommended to be supported so that they can be specified in a filtering query to limit the amount of items. The types of values to be used with these keywords are described in section 7.

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Interpretation</th>
<th>ct-filter-value Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>BCC</td>
<td>Contents of the [RFC2822] &quot;Bcc:&quot; destination address field</td>
<td>text</td>
</tr>
<tr>
<td>CC</td>
<td>Contents of the [RFC2822] &quot;Cc:&quot; destination address field</td>
<td>text</td>
</tr>
<tr>
<td>FROM</td>
<td>Contents of the [RFC2822] &quot;From:&quot; originator field</td>
<td>text</td>
</tr>
<tr>
<td>IMPORTANCE</td>
<td>The case-insensitive values &quot;low&quot;, &quot;normal&quot; and &quot;high&quot; are allowed as specified by the [RFC1327].</td>
<td>text</td>
</tr>
<tr>
<td>NOATTACH</td>
<td>Item does not contain attachments</td>
<td>bool</td>
</tr>
<tr>
<td>NOBODY</td>
<td>Item does not contain [RFC2822] body (i.e. item is an empty message with only [RFC2822] header fields defined)</td>
<td>bool</td>
</tr>
<tr>
<td>SIZE</td>
<td>The number of octets in the content of emailItem field</td>
<td>int</td>
</tr>
<tr>
<td>SUBJECT</td>
<td>Contents of the [RFC2822] &quot;Subject:&quot; informational field</td>
<td>text</td>
</tr>
</tbody>
</table>
### 10.2.1.3 Example 1: Record filtering

In this scenario, the client wishes to synchronize e-mail messages whose internal date is within or later than June 1, 2003 and importance is either “normal” or “high”

1. During the initial sync, the client and server exchange their device info.

2. The client analyses the server’s device info, and the client notes that the server supports receiving filters on the Email data store for queries using the “syncml:filtertype-cgi” grammar.
   
   a. The server includes in its device info the Filter-Rx and FilterCap elements that it supports. This includes the “IMPORTANCE” and “created” keywords.

   b. The client doesn’t require filtering on any additional fields, so it determines that this server supports the filter it wishes to send.

```xml
<Datastore>
  <SourceRef>./email/inbox</SourceRef>
  <DisplayName>Email Inbox</DisplayName>
  ...
  <Filter-Rx>
    <CTType>syncml:filtertype-cgi</CTType>
    <VerCT>1.0</VerCT>
  </Filter-Rx>
  <CTCap>
    ...
  </CTCap>
  <FilterCap>
    <CTType>syncml:filtertype-cgi</CTType>
    <VerCT>1.0</VerCT>
    <FilterKeyword>IMPORTANCE</FilterKeyword>
    <FilterKeyword>created</FilterKeyword>
  </FilterCap>
</Datastore>
```

3. The client sends an Alert for the Email data store with a filter.
a. It includes a Filter Record element with a Meta Type value of “syncml:filtertype-cgi” to indicate the grammar being used.

b. The filter query in the Item Data element contains a value of “created&GT;20030601T000000Z&AND;IMPORTANCE&NE;low” to constrain the items synchronized to those whose internal date is within or later than June 1, 2003 and importance is either “normal” or “high”.

```xml
<Alert>
  <Data>200</Data>
  <Item>
    <Target>
      <LocURI>./email/inbox</LocURI>
    </Target>
    <Filter>
      <Meta><Type>application/vnd.omads-email+xml</Type></Meta>
      <Record>
        <Item>
          <Meta><Type>syncml:filtertype-cgi</Type></Meta>
          <Data>
            created&GT;20030601T000000Z&AND;IMPORTANCE&NE;low
          </Data>
        </Item>
      </Record>
    </Filter>
  </Item>
</Alert>
```

4. The server receives the Alert with the Filter Record element.
   a. It determines that it supports the filter operation for the data store, content type, filter grammar, and properties.
   b. It replies with a status code of 200 for the Alert, indicating that it can satisfy the request to sync with filtering.

5. The synchronization process continues normally.
a. The client sends all of its changes to the server (the filter constraint is not imposed on it in this scenario).

b. The server sends changes only for items that satisfy the filter query.

### 10.2.1.4 Example 2: Content filtering

In this scenario, the client wishes to synchronize only plain text bodies with all attachments removed and truncate the resulting emailitem content to 2 kilobytes.

1. During the initial sync, the client and server exchange their device info.

2. The client sends an Alert for the Email data store with a filter. It includes a Filter Field element containing a Property element set to “emailitem” containing a MaxSize element set to 2048 (2K), specifying that only “text/plain” type parts of the text can be included.
3. The server receives the Alert with the Filter Record element.

   c. It determines that it supports the filter operation for the data store, content type, filter grammar, and properties.
d. It replies with a status code of 200 for the Alert, indicating that it can satisfy the request to sync with filtering.

4. The synchronization process continues normally.

e. The client sends all of its changes to the server (the filter constraint is not imposed on it in this scenario).

f. Server recalculates the emailitem fields for the client to include only plain text. For the items that contain more than 2048 bytes in resulting emailitem field, the server truncates the field content.

10.3 Email object replace example

```
...<Sync>
...

<Replace>
  <CmdID>6</CmdID>
  <Meta>
    <Type xmlns='syncml:metinf'>application/vnd.omad-email+xml</Type>
  </Meta>
  <Item>
    <Source>
      <LocURI>123</LocURI>
    </Source>
    ...
    <Data><![CDATA[
      <Email>
        <read>false</read>
        <created>20030807T231830Z</created>
        <emailitem>
          <!-- The content of the email as specified by the RFC 2822 -->
          <![CDATA[From: <sender@mail.com>
            To: <receiver@mail.com>
            ...]]]]&gt; <![CDATA[
          </emailitem>
        </Email>]]>
    </Data>
```
In this example the device that has previously indicated that it supports “created”, “read” and “emailitem” fields for the email object data type receives the update of the object.
Appendix A. Change History (Informative)

A.1 Approved Version History

<table>
<thead>
<tr>
<th>Reference</th>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>n/a</td>
<td>n/a</td>
<td>No prior version –or– No previous version within OMA</td>
</tr>
</tbody>
</table>

A.2 Draft/Candidate Version 1.2 History

<table>
<thead>
<tr>
<th>Document Identifier</th>
<th>Date</th>
<th>Sections</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Draft Versions OMA-SyncML-DataObjEmail-V1_2</td>
<td>19 Jan 04</td>
<td>Title page, p. 2</td>
<td>Changed dates and copyrights</td>
</tr>
<tr>
<td></td>
<td>26 Jan 04</td>
<td>Last page</td>
<td>Removed page with empty appendix B</td>
</tr>
<tr>
<td></td>
<td>01 Mar 04</td>
<td>TP, 2.1</td>
<td>Clerical changes from consistency review</td>
</tr>
<tr>
<td></td>
<td>14 Apr 04</td>
<td>8.1, 8.7, 9</td>
<td>Added “modified” element to the “Email” parent element, “Email” content model updated, DTD updated.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8.10, 9, 10.1</td>
<td>Added “enc” attribute for “emailitem” element, updated corresponding parts of CTCaps example and DTD; table numbers updated.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10.2.1.3</td>
<td>Updated example description to match the example code.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10.3</td>
<td>Fixed CDATA sections in email object replace example</td>
</tr>
<tr>
<td></td>
<td>05 May 04</td>
<td>Title page</td>
<td>Changed date</td>
</tr>
<tr>
<td>Candidate Version OMA-SyncML-DataObjEmail-V1_2</td>
<td>01 Jun 2004</td>
<td>n/a</td>
<td>Status changed to Candidate by TP TP ref # OMA-TP-2004-0178-DS-V1_2-for-candidate</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Filename and name of specification updated</td>
<td></td>
</tr>
</tbody>
</table>