SMPTE STANDARD

Media Dispatch Protocol (MDP) — MDP/XML/HTTP Mapping Specification

Table of Contents

Approved
December 14, 2007

Foreword

SMPTE (the Society of Motion Picture and Television Engineers) is an internationally-recognized standards developing organization. Headquartered and incorporated in the United States of America, SMPTE has members in over 80 countries on six continents. SMPTE's Engineering Documents, including Standards, Recommended Practices and Engineering Guidelines, are prepared by SMPTE's Technology Committees. Participation in these Committees is open to all with a bona fide interest in their work. SMPTE cooperates closely with other standards-developing organizations, including ISO, IEC and ITU.

SMPTE Engineering Documents are drafted in accordance with the rules given in Part XIII of its Administrative Practices.

SMPTE Standard 2032-2 was prepared by Technology Committee S22 on Committee on Television Systems Technology.

Introduction

This section is entirely informative and does not form an integral part of this document.

The Media Dispatch Protocol (MDP) is a means of orchestrating the delivery of media files over IP networks. It defines a standard mechanism for implementations to initiate a delivery, to negotiate the details of the delivery, to provide information about the progress of the delivery, and to provide a confirmation of the outcome of the delivery.

MDP allows organizations to transfer files at agreed times, using agreed transfer protocols, and using an agreed set of secure technologies (where appropriate). However, the protocol is not overly prescriptive in regards to which protocols or technologies shall be used; rather it provides a framework which allows organizations to choose those that best suit their own needs.

MDP is a multi-part standard:

Part 1, the MDP protocol specification, defines the logical messages, data structures and data dictionary of MDP.

Part 2, this part, defines a representation of the messages and structures defined in Part 1.

Part 3, the MDP profile specification, defines subsets of the protocol and rules on the use of these subsets.

Part 4, the MDP Engineering Guideline, introduces and describes MDP and how it can be used in particular scenarios.

1 Scope

This standard defines the mapping of the Media Dispatch Protocol message set and data structures onto HTTP request / response pairs and XML elements.

2 Conformance Notation

Normative text is text that describes elements of the design that are indispensable or contains the conformance language keywords: "shall", "should", or "may". Informative text is text that is potentially helpful to the user, but not indispensable, and can be removed, changed, or added editorially without affecting interoperability. Informative text does not contain any conformance keywords.

All text in this document is, by default, normative, except: the Introduction, any section explicitly labeled as "Informative" or individual paragraphs that start with "Note:"

The keywords "shall" and "shall not" indicate requirements strictly to be followed in order to conform to the document and from which no deviation is permitted.

The keywords, "should" and "should not" indicate that, among several possibilities, one is recommended as particularly suitable, without mentioning or excluding others; or that a certain course of action is preferred but not necessarily required; or that (in the negative form) a certain possibility or course of action is deprecated but not prohibited.

The keywords "may" and "need not" indicate courses of action permissible within the limits of the document.

The keyword “reserved” indicates a provision that is not defined at this time, shall not be used, and may be defined in the future. The keyword “forbidden” indicates “reserved” and in addition indicates that the provision will never be defined in the future.

A conformant implementation according to this document is one that includes all mandatory provisions ("shall") and, if implemented, all recommended provisions ("should") as described. A conformant implementation need not implement optional provisions ("may") and need not implement them as described.

3 Normative References

The following standards contain provisions which, through reference in this text, constitute provisions of this standard. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this standard are encouraged to investigate the possibility of applying the most recent edition of the standards indicated below.

SMPTE 2032-1-2007, Media Dispatch Protocol (MDP) — Protocol Specification

W3C Extensible Markup Language (XML) 1.1 (Second Edition)

W3C XML Schema 1.1 — Part 1: Structures

W3C XML Schema 1.1 — Part 2: Datatypes

IETF RFC 1867, Form-Based File Upload in HTML

IETF RFC 2279, UTF-8 — A Transformation Format of ISO 10646

IETF RFC 2616, Hypertext Transfer Protocol v1.1

IETF RFC 2818, HTTP Over TLS

IETF RFC 3023, XML Media Types

4 Acronyms and Definitions

The full glossary of acronyms and definitions used in the MDP specification is given in the MDP protocol specification (SMPTE 2032-1). It is not repeated here to avoid any divergence of meaning.

5 Background (Informative)

The MDP protocol specification (SMPTE 2032-1) defines the logical messages and data structures that MDP agents can use for orchestration of file-based media deliveries. However, it does not define a representation for these. Instead this is specified in a mapping specification; this approach provides more flexibility as it allows new mappings to be adopted as required without the need to modify the basic protocol specification.

MDP message exchanges take the form of request-response pairs, and so can be easily mapped to any protocol that provides such a facility. HTTP (RFC 2616) is one such protocol, and this document describes how the abstract MDP request-response pairs are implemented on top of HTTP. It also describes how MDP data structures are represented as XML elements within the HTTP payloads.

6 Mapping of Commands and Queries

A command or query shall be sent as an HTTP v1.1 POST request, as defined in RFC 2616.

The request shall have MIME media type multipart/form-data as defined in RFC 1867.

The POST request shall have the following HTTP parameters:

1. A `messagetype` parameter with value either `command` or `query`.

2. A `message` parameter with value set to the name of the message as defined in SMPTE 2032-1.

3. Zero or more parameters, corresponding to the parameters of the MDP message. The name of each parameter shall correspond to its name in SMPTE 2032-1. The value of the parameter shall have the type specified in SMPTE 2032-1. Where this type is specified as element, property or list, the value shall be represented in XML form as specified in section 8 of this standard.

HTTP Parameters may appear in any order within the POST request.

7 Mapping of Confirmations and Replies

A confirmation or reply shall be sent as the payload of an HTTP v1.1 200 OK response, as defined in RFC 2616.

This payload shall be as specified in the definition of the message in SMPTE 2032-1, and represented in XML form as specified in section 8 of this standard.

The response shall have XML media type `application/xml` as defined in RFC 3023.

8 Mapping of Data Structures

8.1 Elements

An element shall be represented as an XML element with start- and end-tags set to the element name:

```
<name> ... </name>
```

8.2 Properties

A property shall be represented as an XML element with start- and end-tags set to the property name and content set to the property value:

```
<name>value</name>
```

8.3 Lists

A list shall be represented as an XML element with start- and end-tags set to the name of the data structure contained with the list, appended by `_list`:

```
<name_list>

  <name> ... </name>

  <name> ... </name>

</name_list>
```

8.4 Character Encoding

All XML shall be encoded with UTF-8 as specified in RFC 2279.

8.5 Namespace

All elements, lists and properties shall be in the following namespace:

```
http://www.smpte-ra.org/schemas/2032-2/2007/MDP
```

8.6 Well-formedness

All XML shall be well-formed. The `BAD_PAYLOAD` reason property shall be used to indicate XML that is not well-formed.

8.7 Validity

All XML shall be valid according to the MDP/XML/HTTP schema specified on the SMPTE Registration Authority at the following URL:

```
http://www.smpte-ra.org/schemas/2032-2/2007/MDP/mdp.xsd
```

The schema is specified according to W3C XML Schema 1.1.

In the event of conflict between the schema and normative text in any part of the MDP specification, the normative text shall take priority.

The `INVALID_PAYLOAD` reason property shall be used to indicate XML that is not valid. If an implementation cannot distinguish between badly-formed and invalid XML, then it shall send `BAD_PAYLOAD`.

9 Security

Messages may be sent using either unsecured HTTP/1.1, or HTTP/1.1 over TLS. Agents supporting this mapping shall support both cases.

Note: This standard does not require agents to adopt any particular mechanism for authenticating each other. Nor does it require agents to use any particular key size, or encryption cipher, or any other aspect of TLS.

10 `protocol` Administrative Register

SMPTE Headquarters shall make the protocol Administrative Register available in XML format at the following URL:

`http://www.smpte-ra.org/registers/2032-1/2007/MDP/protocol.xml`

The XML in this document shall be in the following namespace:

`http://www.smpte-ra.org/schemas/2032-2/2007/MDP/registers`

The document shall be valid according to the schema specified on the SMPTE Registration Authority at the following URL:

`http://www.smpte-ra.org/schemas/2032-2/2007/MDP/registers.xsd`

The document shall have a root XML element `propertyregister`. This shall contain the following sequence (in the order specified):

1. A `name` element containing the string "`protocol`".
2. An `effectivedate` element of type `xs:dateTime` containing the Effective Date of the register.
3. An `entry_list` element containing zero or more `entry` elements.

Each `entry` element shall contain the following sequence (in the order specified):

1. A `value` element of type `xs:string` containing the value of the `protocol` property.
2. A `description` element of type `xs:string` containing the brief description of the protocol.
3. A `reference` element of type `xs:string` containing the reference to the full description or specification.

Note: A section of the XML register document is shown in A.5 for information.

Annex A (Informative)
Examples

A.1 Command

The following shows an example of an HTTP POST payload in which a `cmd_sendingmanifest` command is sent using the MDP/XML/HTTP mapping:

```
POST http://broadcaster.com/mdp/xml/http/ HTTP/1.1
Content-Type: multipart/form-data; boundary=xYzZY

 (other HTTP headers not shown)

--xYzZY
Content-Disposition: form-data; name="messagetype"

command
--xYzZY
Content-Disposition: form-data; name="message"

cmd_sendingmanifest
--xYzZY
Content-Disposition: form-data; name="agent"

http://facility.com/mdp/xml/http/
--xYzZY
Content-Disposition: form-data; name="txorg"

facility.com
--xYzZY
Content-Disposition: form-data; name="rxorg"

broadcaster.com
--xYzZY
Content-Disposition: form-data; name="manifest"

<manifest xmlns="http://www.smpte-ra.org/schemas/2032-2/2007/MDP"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.smpte-ra.org/schemas/2032-2/2007/MDP mdp.xsd">
      <transactionid>0123456789abcdef</transactionid>
      <initiator>facility.com</initiator>
      <target>broadcaster.com</target>
      <projectid>P0123456</projectid>
    ...
</manifest>
--xYzZY--
```

A.2 Query

The following shows an example of an HTTP POST payload in which a `qry_requestmanifest` query is sent using the MDP/XML/HTTP mapping:

```
POST http://broadcaster.com/mdp/xml/http/ HTTP/1.1
Content-Type: multipart/form-data; boundary=aBcDE
(other HTTP headers deleted)

--aBcDE
Content-Disposition: form-data; name="messagetype"

query
--aBcDE
Content-Disposition: form-data; name="message"
```

```
 qry_requestmanifest
 --aBcDE
 Content-Disposition: form-data; name="agent"

 http://facility.com/mdp/xml/http/
 --aBcDE
 Content-Disposition: form-data; name="txorg"

 facility.com
 --aBcDE
 Content-Disposition: form-data; name="rxorg"

 broadcaster.com
 --aBcDE
Content-Disposition: form-data; name="projectid"

P0123456
--aBcDE
Content-Disposition: form-data; name="transactionid"

0123456789abcdef
--aBcDE--
```

A.3 Confirmation

The following shows an example of an HTTP response payload in which a `cnf_ok` confirmation is sent using the MDP/XML/HTTP mapping:

```
HTTP/1.1 200 OK
Content-Type: application/xml; charset=utf-8

(other HTTP headers deleted)

<?xml version="1.0" encoding="UTF-8" ?>
<ok xmlns="http://www.smpte-ra.org/schemas/2032-2/2007/MDP"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" />
```

A.4 Reply

The following shows an example of an HTTP response payload in which a `rpl_manifest` reply is sent using the MDP/XML/HTTP mapping (in response to a `qry_requestmanifest`):

```
HTTP/1.1 200 OK
Content-Type: application/xml; charset=utf-8

(other HTTP headers deleted)

<?xml version="1.0" encoding="UTF-8" ?>
<manifest xmlns="http://www.smpte-ra.org/schemas/2032-2/2007/MDP"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://www.smpte-ra.org/schemas/2032-2/2007/MDP
 mdp.xsd">
       <transactionid>0123456789abcdef</transactionid>
       <initiator>facility.com</initiator>
       <target>broadcaster.com</target>
       <projectid>P0123456</projectid>
    ...
</manifest>
```

The following shows an example of an HTTP response payload in which a cnf_error confirmation is sent using the MDP/XML/HTTP mapping. An `rpl_error` reply is similar:

```
HTTP/1.0 200 OK
Content-Type: application/xml; charset=utf-8

(other HTTP headers deleted)

<?xml version="1.0" encoding="UTF-8" ?>
<error xmlns="http://www.smpte-ra.org/schemas/2032-2/2007/MDP"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.smpte-ra.org/schemas/2032-2/2007/MDP
       mdp.xsd ">
       <reason>INVALID_ID</reason>
       <comment>transactionid was invalid</comment>
</error>
```

A.5 `protocol` register

The following is a section from the Administrative Register for the `protocol` property, represented as defined in section 10.

```
<?xml version="1.0" encoding="UTF-8"?>
<propertyregister xmlns="http://www.smpte-ra.org/schemas/2032-2/2007/MDP/registers"
  xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
  xsi:schemaLocation="http://www.smpte-ra.org/schemas/2032-2/2007/MDP mdp.xsd">
   <name>protocol</name>
   <effectivedate>2007-07-06</effectivedate>
   <entry list>
      <entry>
         <value>HTTP</value>
         <description>HTTP/1.1</description>
         <reference>www.ietf.org/rfc/rfc2616.txt</reference>
      </entry>
      <entry>
         <value>HTTP/TLS</value>
         <description>HTTP/1.1 over TLS</description>
         <reference>www.ietf.org/rfc/rfc2818.txt</reference>
      </entry>
      <entry>
         <value>FTP</value>
         <description>File Transfer Protocol</description>
         <reference>www.ietf.org/rfc/rfc959.txt</reference>
      </entry>
      ...
   </entry list>
</propertyregister>
```