INTERNATIONAL STANDARD

ISO/IEC 23093-3

First edition
2019-11

Information technology — Internet of media things —

Part 3: Media data formats and APIs

Technologies de l'information — Internet des objets media —

Partie 3: API et formats des données

Reference number
ISO/IEC 23093-3:2019(E)

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Fax: +41 22 749 09 47
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Contents

Page

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents) or the IEC list of patent declarations received (see http://patents.iec.ch).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see www.iso.org/iso/foreword.html.

This document was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 29, *Coding of audio, picture, multimedia and hypermedia information*.

A list of all parts in the ISO/IEC 23093 series can be found on the ISO website.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html.

Introduction

The ISO/IEC 23093 series provides an architecture and specifies APIs and compressed representation of data flowing between media things.

The APIs for the media things facilitate discovering other media things in the network, connecting and efficiently exchanging data between media things. The APIs also provide means for supporting transaction tokens in order to access valuable functionalities, resources, and data from media things.

Media things related information consists of characteristics and discovery data, setup information from a system designer, raw and processed sensed data, and actuation information. The ISO/IEC 23093 series specifies data formats of input and output for media sensors, media actuators, media storages, media analyzers, etc. Sensed data from media sensors can be processed by media analyzers to produce analysed data, and the media analyzers can be cascaded in order to extract semantic information.

This document contains the tools to describe data exchanged between media things (e.g. media sensors, media actuators, media analyzers, media storages) and their APIs. It addresses the normative aspects of the data and APIs for media things and also illustrates non-normative examples.

The International Organization for Standardization (ISO) and the International Electrotechnical Commission (IEC) draw attention to the fact that it is claimed that compliance with this document may involve the use of patents.

ISO and the IEC take no position concerning the evidence, validity and scope of these patent rights.

The holders of these patent rights have assured the ISO and IEC that they are willing to negotiate licences under reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statements of the holders of these patents right are registered with ISO and IEC. Information may be obtained from the patent database available at www.iso.org/patents.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights other than those identified in the patent database. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

Information technology — Internet of media things —

Part 3:
Media data formats and APIs

1 Scope

This document specifies syntax and semantics of description schemes to represent data exchanged by media things (e.g. media sensors, media actuators, media analyzers, media storages). Moreover, it specifies the APIs to exchange these data between media things.

This document does not specify how the process of sensing and analyzing is carried out but specifies the interfaces between the media things.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 15938-5:2003, *Information technology — Multimedia content description interface — Part 5: Multimedia description schemes*

ISO/IEC 23005-2, *Information technology — Media context and control — Part 2: Control information*

ISO/IEC 23005-5, *Information technology — Media context and control — Part 5: Data formats for interaction devices*

ISO/IEC 23093-1, *Information technology — Internet of media things — Part 1: Architecture*

ISO/IEC 23093-2, *Information technology — Internet of media things — Part 2: Discovery and communication API*

3 Terms, definitions, and abbreviated terms

3.1 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 23093-1 and 23093-2 and the following apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at https://www.iso.org/obp

— IEC Electropedia: available at http://www.electropedia.org/

3.1.1
media actuator
MActuator
MThing that can actuate

3.1.2
media aggregator
MAggregator
MThing that contains multiple MThings

3.1.3
media analyzer
MAnalyzer
MThing that can analyze media or metadata, and produce interpreted media, metadata, or commands

3.1.4
media manager
MManager
MThing that can register a list of MThings or be facilitated to search other MThings

3.1.5
media sensor
MSensor
MThing that can sense and produce media data

3.1.6
media storage
MStorage
MThing that can save media or metadata

3.2 Abbreviated terms

API application programming interface

MACV media actuator command vocabulary

MAOV media analyzer output vocabulary

MSOV media sensor output vocabulary

MTDL media thing description language

SCDV sensor capability description vocabulary

XML extensible mark-up language

XSI XML streaming instructions

3.3 Schema documents

In the main text of this document, the syntax of description schemes and descriptors is provided whenever possible as a single schema document.

In some cases though, and in particular for Clauses 6, 7 and 8, the syntax of description schemes and descriptors is provided as a collection of schema snippets imbricated with other text. In order to form a valid schema document, these schema components are intended to be gathered in the same document

with the schema wrapper provided at the head of the clause. For better readability, the relevant schema documents are provided in Annex B.

In all cases, each schema document has a `version` attribute, the value of which is "ISO/IEC 23093-3". Furthermore, an informative identifier is given as the value of the `id` attribute of the `schema` component. This identifier is non-normative and used as a convention in this document to reference another schema document. In particular, it is used for the `schemaLocation` attribute of the `include` and `import` schema components.

3.4 Use of prefixes

For clarity, throughout this document, consistent namespace prefixes are used.

"`xsi:`" prefix is not normative. It is a naming convention in this document to refer to an element of the `http://www.w3.org/2001/XMLSchema-instance` namespace.

"`xml:`" and "`xmlns:`" are normative prefixes defined in Reference [1]. The prefix "`xml:`" is by definition bound to "`http://www.w3.org/XML/1998/namespace`". The prefix "`xmlns:`" is used only for namespace bindings and is not itself bound to any namespace name.

All other prefixes used in either the text or examples of this document are not normative, e.g. "`mtdl:`", "`msov:`", "`macv:`", "`maov:`", "`mpeg7:`", "`scdv:`".

In particular, most of the informative examples in this document are provided as XML fragments without the normally required XML document declaration and, thus, miss a correct namespace binding context declaration. In these descriptions fragments, the different prefixes are bound to the `namespaces` as given in Table 1.

Table 1 — Mapping of prefixes to namespaces in examples and text

Prefix	Corresponding namespace
scdv	urn:mpeg:mpeg-v:2018:01-SCDV-NS
mpeg7	urn:mpeg:mpeg7:schema:2004
mtdl	urn:mpeg:mpeg-IoMT:2018:01-MTDL-NS
msov	urn:mpeg:mpeg-IoMT:2018:01-MSOV-NS
macv	urn:mpeg:mpeg-IoMT:2018:01-MACV-NS
maov	urn:mpeg:mpeg-IoMT:2018:01-MAOV-NS
xsi	http://www.w3.org/2001/XMLSchema-instance
xsd	http://www.w3.org/2001/XMLSchema

Unlike the informative descriptions examples, the normative specification of the syntax of tools in XML schema follows the namespace binding context defined in the relevant schema declaration such as the one defined in 6.2.

4 APIs

4.1 General

This subclause specifies APIs and their descriptions to operate MThings and/or exchange structured data between MThings. Figure 1 shows an example of "GET" and "SET" functions invoked between

MThings. For example, an MSensor should have “GET” functions to evoke and provide its sensed data. An MStorage should have “SET” functions to save sensed data obtained by an MSensor or to save analyzed data provided by an MAnalyzer. An MAnalyzer should provide “GET” functions to produce metadata by analyzing sensed data from MSensors or to generate metadata by analyzing data fed by other MAnalyzers. Finally, an MActuator should provide “SET” functions to control its functionalities. If there is no structured data exchanged between MThings, each MThing can have simple “SET” functions to be controlled by other MThings.

Figure 2 demonstrates an example of a function call sequence diagram between MThings. A speech recognizer (AZ1) requests an audio data to a microphone (S1) by invoking the function `getAudioURL()`. The microphone (S1) sends back the corresponding URL to the speech recognizer (AZ1). In this case, the return type of the URL is a simple string. If, however, an MSensor returns data with standard structures, the data can be delivered by the return type class either “MPEGVSensedDataType” or “IoMT SensedDataType”, which can be described by XML, JSON, or Binary representation.

A text-to-speech converter (AZ2) requests speech texts extracted from the audio data (i.e. sensed data from S1) to the speech recognizer (AZ1) by invoking the function `getSpeechText()`. The speech recognizer (AZ1) sends back the recognized speech texts with the standard structure to the text-to-speech converter (AZ2). The data provided by an MAnalyzer can be delivered by either a simple string like a URL or the return type class called “IoMT AnalyzedDataType”, which can be described by XML, JSON, or Binary representation.

Finally, the text-to-speech converter (AZ2) invokes the function `setAudioURL()` and the function `setPlay()` to actuate (i.e. generate the audible speech sound) the speaker (AC1). Again, the actuation data feeding to an MActuator can be delivered by either a simple string like a URL or the return type class of “MPEGVCommandType” or “IoMT ActuationDataType”, which can be described by XML, JSON, or Binary representation.

The function calls trigger MThings either to generate and exchange data or to control MThings.

The function definitions (APIs) are defined for MSensor, MActuator, MAnalyzer, MStorage, MManager, MAggregator, and their return type classes in the following subclauses.

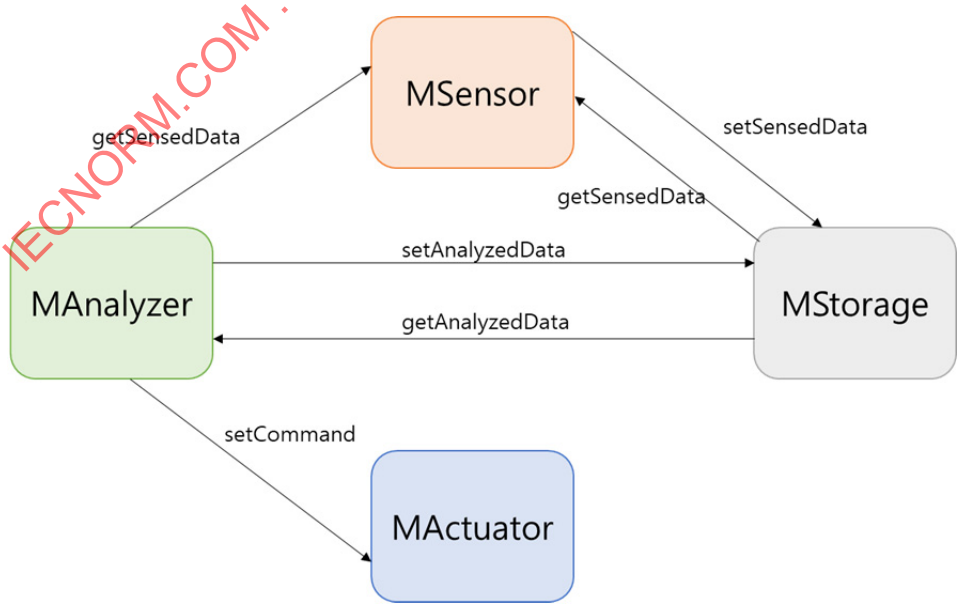


Figure 1 — Function invocation between MThings

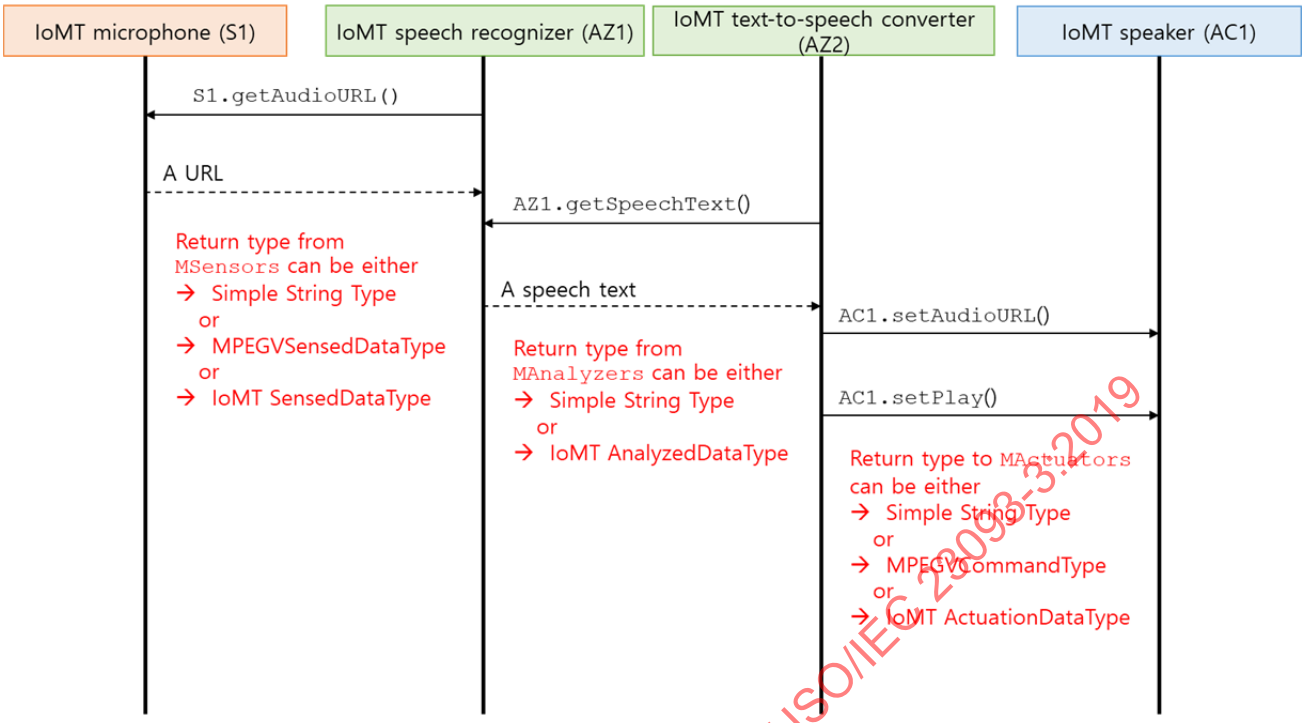


Figure 2 — Sequence diagram of function calls between MThings

4.2 APIs for IoMT sensors

4.2.1 General

This subclause defines API classes of IoMT sensors.

4.2.2 MSensor class

General

This subclause defines an MSensor class, which shall inherit the features of MThing class defined in ISO/IEC 23093-2.

APIs

Table 2 presents basic APIs of MSensor.

Table 2 — MSensor APIs

Nested Classes	
Modifier and Type	Method and Description
Constructor	
Constructor and Description	
MSensor()	
Default constructor.	
MSensor(String id)	
MSensor(String id, String serverIPAddress, integer serverPort)	
Fields	
Modifier and Type	Field and Description
Methods	
Modifier and Type	Method and Description
MPEGVCapabilityType	getMPEGVCapability()
	This function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and sensor capabilities from ISO/IEC 23005-2 (MPEG-V Part 2).
MPEGVSensedDataType	getMPEGVSendsedData()
	This function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and sensed data from ISO/IEC 23005-5 (MPEG-V Part 5).
CapabilityListType	getSensorCapabilityList();
	This function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and a capability list specified in this document.

CapabilityListType	getAvailableSensorCapabilityList()
	This function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and an available capability list specified in this document.
CapabilityListType	getAppliedSensorCapabilityList()
	This function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and an applied capability list specified in this document.
SensedDataType	getCapturedTime()
	This function returns a captured time of sensed data.
SensedDataType	getCapturedTime(String tid)
	This function returns a captured time of sensed data. The `tid` *is the transaction ID of a payment for using this function.*
float	getCapturedTime_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `getCapturedTime()`*. If* `tokenType` *is 0, it denotes "cryptocurrency", if* tokenType *is 1, it denotes "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) getCapturedTime_Cost(0, "BTC") or getCapturedTime_Cost(0, "00000001") Ex) getCapturedTime_Cost(1, "USD") or getCapturedTime_Cost(1, "10010100")

4.2.3 API for IoMT microphone

General

This subclause defines a class of an IoMT microphone which shall inherit the features of `MSensor` class.

APIs

Table 3 presents APIs of an IoMT microphone.

Table 3 — IoMT microphone API

Nested Classes	
Modifier and Type	Method and Description
Constructor	
Constructor and Description	
MMicrophone()	
Default constructor.	
MMicrophone(String id)	
MMicrophone(String id, String serverIPAddress, integer serverPort)	
Fields	
Modifier and Type	Field and Description
Methods	
Modifier and Type	Method and Description
String	getAudioURL()
	This function returns a URL of an audio source.
String	getAudioURL(String tid)
	This function returns a URL of an audio source. The `tid` *is the transaction ID of a payment for using this function.*

float	getAudioURL_CostPerMinute(int tokenType, String tokenName)
	This function returns the amount of tokens per minute to use `getAudioURL()`*. If* `tokenType` *is 0, it denotes "cryptocurrency", if* `tokenType` *is 1, it denotes "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) getAudioURL_CostPerMinute(0, "BTC") or getAudioURL_CostPerMinute(0, "00000001") Ex) getAudioURL_CostPerMinute(1, "USD") or getAudioURL_CostPerMinute(1, "10010100")
int	getAudioSamplingRate()
	This function returns a sampling rate of an audio source.
int	getAudioSamplingRate(String tid)
	This function returns a sampling rate of an audio source. The `tid` *is the transaction ID of a payment for using this function.*
float	getAudioSamplingRate_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `getAudioSamplingRate()`*. If* `tokenType` *is 0, it denotes "cryptocurrency", if* `tokenType` *is 1, it denotes "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) getAudioSamplingRate_Cost(0, "BTC") or getAudioSamplingRate_Cost(0, "00000001") Ex) getAudioSamplingRate_Cost(1, "USD") or getAudioSamplingRate_Cost(1, "10010100")

4.2.4 API for IoMT camera

General

This subclause defines a class of an IoMT camera which shall inherit the features of `MSensor` class.

APIs

Table 4 presents APIs of an IoMT camera.

Table 4 — IoMT camera API

Nested Classes	
Modifier and Type	Method and Description
Constructor	
Constructor and Description	
MCamera()	
Default constructor.	
MCamera(String id)	
MCamera(String id, String serverIPAddress, integer serverPort)	
Fields	
Modifier and Type	Field and Description
Methods	
Modifier and Type	Method and Description
String	getVideoURL()
	This function returns a URL of a video source.
String	getVideoURL(String tid)
	This function returns a URL of a video source. The `tid` *is the transaction ID of a payment for using this function.*

float	getVideoURL_CostPerMinute(int tokenType, String tokenName)
	This function returns the amount of tokens per minute to use `getVideoURL()`*. If* `tokenType` *is 0, it denotes "cryptocurrency", if* `tokenType` *is 1, it denotes "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) getVideoURL_CostPerMinute(0, "BTC") or getVideoURL_CostPerMinute(0, "00000001") Ex) getVideoURL_CostPerMinute(1, "USD") or getVideoURL_CostPerMinute(1, "10010100")
String	getImageURL()
	This function returns a URL of an image source.
String	getImageURL(String tid)
	This function returns a URL of an image source. The `tid` *is a token transaction ID of the corresponding service*
float	getImageURL_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `getImageURL()`*. If* `tokenType` *is 0, it denotes "cryptocurrency", if* `tokenType` *is 1, it denotes "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) getImageURL_Cost(0, "BTC") or getImageURL_Cost(0, "00000001") Ex) getImageURL_Cost(1, "USD") or getImageURL_Cost(1, "10010100")

4.2.5 API for IoMT RFID reader

General

This subclause defines a class of an IoMT RFID reader which shall inherit the features of `MSensor` class.

APIs

Table 5 presents APIs of an IoMT RFID reader.

Table 5 — IoMT RFID reader API

Nested Classes	
Modifier and Type	Method and Description
Constructor	
Constructor and Description	
MRFIDSensor()	
Default constructor.	
MRFIDSensor(String id)	
MRFIDSensor(String id, String serverIPAddress, integer serverPort)	
Fields	
Modifier and Type	Field and Description
Methods	
Modifier and Type	Method and Description
String	getRFIDtagInfo()
	This fucntion returns a RFID tag information.
String	getRFIDtagInfo(String tid)
	This fucntion returns a RFID tag information. The `tid` *is the transaction ID of a payment for using this function.*

float	getRFIDtagInfo_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `getRFIDtagInfo()`. *If* `tokenType` *is 0, it means "cryptocurrency", if* `tokenType` *is 1, it means "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) getRFIDtagInfo_Cost(0, "BTC") or getRFIDtagInfo_Cost(0, "00000001") Ex) getRFIDtagInfo_Cost(1, "USD") or getRFIDtagInfo_Cost(1, "10010100")

4.2.6 API for IoMT compass sensor

General

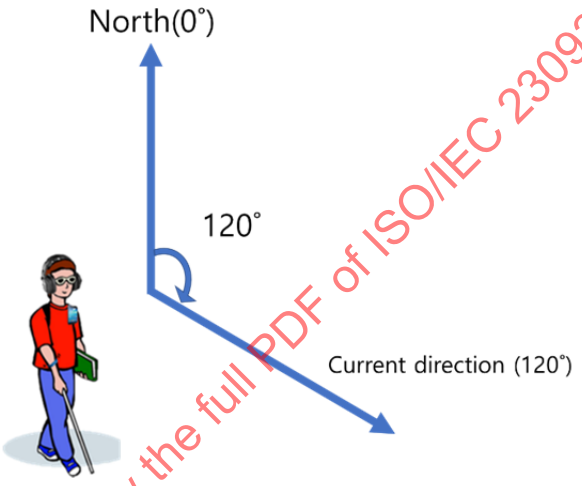
This subclause defines a class of an IoMT compass sensor which shall inherit the features of `MSensor` class.

APIs

Table 6 presents APIs of an IoMT compass sensor.

Table 6 — IoMT compass sensor API

Nested Classes	
Modifier and Type	Method and Description
Constructor	
Constructor and Description	
MCompass()	
Default constructor.	
MCompass(String id)	
MCompass(String id, String serverIPAddress, integer serverPort)	

Fields	
Modifier and Type	Field and Description
Methods	
Modifier and Type	Method and Description
float	getAzimuthAngle()
	The function returns the current clockwise azimuth angle from the north (Figure 3).  **Figure 3 — Azimuth angle**
float	getAzimuthAngle(String tid)
	The function returns the current clockwise azimuth angle from the north. The `tid` *is the transaction ID of a payment for using this function.*
float	getAzimuthAngle_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `getAzimuthAngle()`*. If* `tokenType` *is 0, it means "cryptocurrency", if* `tokenType` *is 1, it means "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) getAzimuthAngle_Cost(0, "BTC") or getAzimuthAngle_Cost(0, "00000001") Ex) getAzimuthAngle_Cost(1, "USD") or getAzimuthAngle_Cost(1, "10010100")

4.2.7 API for IoMT orientation sensor

General

This subclause defines a class of an IoMT orientation sensor which shall inherit the features of `MSensor` class.

APIs

Table 7 presents APIs of an IoMT orientation sensor.

Table 7 — IoMT orientation sensor API

Nested Classes	
Modifier and Type	Method and Description
Constructor	
Constructor and Description	
MOrientationSensor()	
Default constructor.	
MOrientationSensor(String id)	
MOrientationSensor(String id, String serverIPAddress, integer serverPort)	
Fields	
Modifier and Type	Field and Description
Methods	
Modifier and Type	Method and Description
MPEGVSensedDataType	getOrientation()
	The function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and sensed data defined by `OrientationSensorType` *from ISO/IEC 23005-5 (MPEG-V Part 5).*

MPEGVSensedDataType	getOrientation(String tid)
	The function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and sensed data defined by `OrientationSensorType` *from ISO/IEC 23005-5 (MPEG-V Part 5). The* `tid` *is the transaction ID of a payment for using this function.*
float	getOrientation_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `getOrientation()`*. If* `tokenType` *is 0, it means "cryptocurrency", if* `tokenType` *is 1, it means "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) getOrientation_Cost(0, "BTC") or getOrientation_Cost(0, "00000001") Ex) getOrientation_Cost(1, "USD") or getOrientation_Cost(1, "10010100")

4.2.8 API for IoMT position sensor

General

This subclause defines a class of an IoMT position sensor which shall inherit the features of `MSensor` class.

APIs

Table 8 presents APIs of an IoMT position sensor.

Table 8 — IoMT position sensor API

Nested Classes	
Modifier and Type	Method and Description
Constructor	
Constructor and Description	
MPositionSensor()	
Default constructor.	

MPositionSensor(String id)	
MPositionSensor(String id, String serverIPAddress, integer serverPort)	
Fields	
Modifier and Type	Field and Description
Methods	
Modifier and Type	Method and Description
MPEGVSensedDataType	getPosition()
	The function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and sensed data defined by `PositionSensorType` *from ISO/IEC 23005-5 (MPEG-V Part 5).*
MPEGVSensedDataType	getPosition(String tid)
	The function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and sensed data defined by `PositionSensorType` *from ISO/IEC 23005-5 (MPEG-V Part 5). The* `tid` *is the transaction ID of a payment for using this function.*
float	getPosition_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `getPosition()`*. If* `tokenType` *is 0, it means "cryptocurrency", if* `tokenType` *is 1, it means "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) getPosition_Cost(0, "BTC") or getPosition_Cost(0, "00000001") Ex) getPosition_Cost(1, "USD") or getPosition_Cost(1, "10010100")

4.2.9 API for IoMT global positioning sensor

General

This subclause defines a class of an IoMT global positioning sensor which shall inherit the features of `MSensor` class.

APIs

Table 9 presents APIs of an IoMT global positioning sensor.

Table 9 — IoMT global positioning sensor API

Nested Classes	
Modifier and Type	Method and Description
Constructor	
Constructor and Description	
MGlobalPositioningSensor()	
Default constructor.	
MGlobalPositioningSensor(String id)	
MGlobalPositioningSensor(String id, String serverIPAddress, integer serverPort)	
Fields	
Modifier and Type	Field and Description
Methods	
Modifier and Type	Method and Description
MPEGVSensedDataType	getGlobalPosition()
	This method returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and sensed data defined by `GlobalPositionSensorType` *from ISO/IEC 23005-5 (MPEG-V Part 5).*
MPEGVSensedDataType	getGlobalPosition(String tid)
	This method returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and sensed data defined by `GlobalPositionSensorType` *from ISO/IEC 23005-5 (MPEG-V Part 5). The* `tid` *is the transaction ID of a payment for using this function.*

float	getGlobalPosition_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `getGlobalPosition()`. *If* `tokenType` *is 0, it means "cryptocurrency", if* `tokenType` *is 1, it means "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) getGlobalPosition_Cost(0, "BTC") or getGlobalPosition_Cost(0, "00000001") Ex) getGlobalPosition_Cost(1, "USD") or getGlobalPosition_Cost(1, "10010100")
float	getLatitude()
	This function returns the value of latitude.
float	getLatitude(String tid)
	This function returns the value of latitude. The `tid` *is the transaction ID of a payment for using this function.*
float	getLatitude_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `getLatitude()`. *If* `tokenType` *is 0, it means "cryptocurrency", if* `tokenType` *is 1, it means "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) getLatitude_Cost(0, "BTC") or getLatitude_Cost(0, "00000001") Ex) getLatitude_Cost(1, "USD") or getLatitude_Cost(1, "10010100")
float	getLongitude()
	This function returns the value of longitude.
float	getLongitude(String tid)
	This function returns the value of longitude. The `tid` *is the transaction ID of a payment for using this function.*

float	getLongitude_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `getLongitude()`. *If* `tokenType` *is 0, it means "cryptocurrency", if* `tokenType` *is 1, it means "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) getLongitude_Cost(0, "BTC") or getLongitude_Cost(0, "00000001") Ex) getLongitude_Cost(1, "USD") or getLongitude_Cost(1, "10010100")

4.2.10 API for IoMT distance sensor

General

This subclause defines a class of an IoMT distance sensor which shall inherit the features of `MSensor` class.

APIs

Table 10 presents APIs of an IoMT distance sensor.

Table 10 — IoMT distance sensor API

Nested Classes	
Modifier and Type	Method and Description
Constructor	
Constructor and Description	
MDistanceSensor() *Default constructor.*	
MDistanceSensor(String id)	
MDistanceSensor(String id, String serverIPAddress, integer serverPort)	

Fields	
Modifier and Type	Field and Description
Methods	
Modifier and Type	Method and Description
MPEGVSensedDataType	getDistance()
	The function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and sensed data defined by `DistanceSensorType` *from ISO/IEC 23005-5 (MPEG-V Part5).*
MPEGVSensedDataType	getDistance(String tid)
	The function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and sensed data defined by `DistanceSensorType` *from ISO/IEC 23005-5 (MPEG-V Part 5). The* `tid` *is the transaction ID of a payment for using this function.*
float	getDistance_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `getDistance()`*. If* `tokenType` *is 0, it means "cryptocurrency", if* `tokenType` *is 1, it means "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) getDistance_Cost(0, "BTC") or getDistance_Cost(0, "00000001") Ex) getDistance_Cost(1, "USD") or getDistance_Cost(1, "10010100")

4.3 APIs for IoMT actuators

4.3.1 General

This subclause defines API classes of IoMT actuators.

4.3.2 MActuator class

General

This subclause defines an MActuator classthat shall inherit the features of `MThing` class defined in ISO/IEC 23093-2.

APIs

Table 11 presents basic APIs of MActuator.

Table 11 — MActuator API

Nested Classes	
Modifier and Type	Method and Description
Constructor	
Constructor and Description	
MActuator()	
Default constructor.	
MActuator(String id)	
MActuator(String id, String serverIPAddress, integer serverPort)	
Fields	
Modifier and Type	Field and Description
Methods	
Modifier and Type	Method and Description
MPEGVCapabilityType	getMPEGVCapability()
	The function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and actuator capabilities from ISO/IEC 23005-2 (MPEG-V Part 2).
int	setMPEGVCommand(MPEGVCommandType mpegvCommand)
	Set a command to an actuator following the specifications of actuator commands from ISO/IEC 23005-5 (MPEG-V Part 5). The function returns 1, if the task succeeds, returns 0, otherwise.

CapabilityListType	getCapabilityList()
	This function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and a capability list using data formats defined in this document.
CapabilityListType	getAvailableActuatorCapabilityList()
	This function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and an available capability list using data formats defined in this document.
CapabilityListType	getAppliedActuatorCapabilityList()
	This function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and an applied capability list using data formats defined in this document.

4.3.3 API for IoMT speaker

General

This subclause defines a class of an IoMT speaker which shall inherit the features of `MActuator` class.

APIs

Table 12 presents APIs of an IoMT speaker.

Table 12 — IoMT speaker API

Nested Classes	
Modifier and Type	Method and Description
Constructor	
Constructor and Description	
MSpeaker()	
Default constructor.	

MSpeaker(String id)	
MSpeaker(String id, String serverIPAddress, integer serverPort)	
Fields	
Modifier and Type	Field and Description
Methods	
Modifier and Type	Method and Description
int	setAudioURL(String url)
	This function receives a URL of an audio source sent by other MThings. The function returns 1, if the task succeeds, returns 0, otherwise.
int	setAudioURL(String tid, String url)
	This function receives a URL of an audio source sent by other MThings. The function returns 1, if the task succeeds, returns 0, otherwise. The `tid` *is the transaction ID of a payment for using this function.*
float	setAudioURL_CostPerMinute(int tokenType, String tokenName)
	This function returns the amount of tokens per minute to use `setAudioURL()`*. If* `tokenType` *is 0, it denotes "cryptocurrency", if* `tokenType` *is 1, it denotes "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) setAudioURL_CostPerMinute(0, "BTC") or setAudioURL_CostPerMinute(0, "00000001") Ex) setAudioURL_CostPerMinute(1, "USD") or setAudioURL_CostPerMinute(1, "10010100")
int	setPlay(ActuationDataType play)
	This function controls the type of play (e.g. stop, play, pause). The function returns 1, if the task succeeds, returns 0, otherwise.

int	setPlay(String tid, ActuationDataType play)
	This function controls the type of play (e.g. stop, play, pause). The function returns 1, if the task succeeds, returns 0, otherwise. The `tid` *is the transaction ID of a payment for using this function.*
float	setPlay_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `setPlay()`*. If* `tokenType` *is 0, it denotes "cryptocurrency", if* `tokenType` *is 1, it denotes "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) setPlay_Cost(0, "BTC") or setPlay_Cost(0, "00000001") Ex) setPlay_Cost(1, "USD") or setPlay_Cost(1, "10010100")
int	setVolume(ActuationDataType volume)
	This function adjusts the volume of the sound. The function returns 1, if the task succeeds, returns 0, otherwise.
int	setVolume(String tid, ActuationDataType volume)
	This function adjusts the volume of the sound. The function returns 1, if the task succeeds, returns 0, otherwise. The `tid` *is the transaction ID of a payment for using this function.*
float	setVolume_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `setVolume()`*. If* `tokenType` *is 0, it denotes "cryptocurrency", if* `tokenType` *is 1, it denotes "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) setVolume_Cost(0, "BTC") or setVolume_Cost(0, "00000001") Ex) setVolume_Cost(1, "USD") or setVolume_Cost(1, "10010100")

4.3.4 API for IoMT display

General

This subclause defines a class of an IoMT display which shall inherit the features of `MActuator` class.

APIs

Table 13 presents APIs of an IoMT display.

Table 13 — IoMT display API

Nested Classes	
Modifier and Type	Method and Description
Constructor	
Constructor and Description	
MDisplay()	
Default constructor.	
MDisplay(String id)	
MDisplay(String id, String serverIPAddress, integer serverPort)	
Fields	
Modifier and Type	Field and Description
Methods	
Modifier and Type	Method and Description
int	setVideoURL(String videoURL)
	This function receives a URL of a video source sent by other MThings. The function returns 1, if the task succeeds, returns 0, otherwise.

int	setVideoURL(String tid, String videoURL)
	This function receives a URL of a video source sent by other MThings. The function returns 1, if the task succeeds, returns 0, otherwise. The `tid` *is the transaction ID of a payment for using this function.*
float	setVideoURL_CostPerMinute(int tokenType, String tokenName)
	This function returns the amount of tokens per minute to use `setVideoURL()`*. If* `tokenType` *is 0, it denotes "cryptocurrency", if* `tokenType` *is 1, it denotes "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) setVideoURL_CostPerMinute(0, "BTC") or setVideoURL_CostPerMinute(0, "00000001") Ex) setVideoURL_CostPerMinute(1, "USD") or setVideoURL_CostPerMinute(1, "10010100")
int	setPlay(ActuationDataType playType)
	This function controls the type of play (e.g. stop, play, pause). This function returns 1, if the task succeeds, returns 0, otherwise.
int	setPlay(String tid, ActuationDataType playType)
	This function controls the type of play (e.g. stop, play, pause). This function returns 1, if the task succeeds, returns 0, otherwise. The `tid` *is the transaction ID of a payment for using this function.*
float	setPlay_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `setPlay()`*. If* `tokenType` *is 0, it denotes "cryptocurrency", if* `tokenType` *is 1, it denotes "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) setPlay_Cost(0, "BTC") or setPlay_Cost(0, "00000001") Ex) setPlay_Cost(1, "USD") or setPlay_Cost(1, "10010100")

int	setDisplayResolution(ActuationDataType resolution)
	This function controls the resolution of video display. The function returns 1, if the task succeeds, returns 0, otherwise.
int	setDisplayResolution(String tid, ActuationDataType resolution)
	This function controls the resolution of video display. The function returns 1, if the task succeeds, returns 0, otherwise. The `tid` *is the transaction ID of a payment for using this function.*
float	setDisplayResolution_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `setDisplayResolution()`*. If* `tokenType` *is 0, it denotes "cryptocurrency", if* `tokenType` *is 1, it denotes "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) setDisplayResolution_Cost(0, "BTC") or setDisplayResolution_Cost(0, "00000001") Ex) setDisplayResolution_Cost(1, "USD") or setDisplayResolution_Cost(1, "10010100")
int	setBrightness(ActuationDataType brightness)
	This function controls the brightness of a display. The function returns 1, if the task succeeds, returns 0, otherwise.
int	setBrightness(String tid, ActuationDataType brightness)
	This function controls the brightness of a display. The function returns 1, if the task succeeds, returns 0, otherwise. The `tid` *is the transaction ID of a payment for using this function.*

float	setBrightness_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `setBrightness()`*.If* `tokenType` *is 0, it denotes "cryptocurrency", if* `tokenType` *is 1, it denotes "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) setBrightness_Cost(0, "BTC") or setBrightness_Cost(0, "00000001") Ex) setBrightness_Cost(1, "USD") or setBrightness_Cost(1, "10010100")
Int	setDisplayMode(int displayType)
	This function sets the display type to either "opaque" or "transparent". If `displayType` *is 0, the display type sets to "opaque". If* `displayType` *is 1, the display mode sets to "transparent". The function returns 1, if the task succeeds, returns 0, otherwise.*
int	setDisplayMode(String tid, int displayType)
	This function sets the display type to either "opaque" or "transparent". If `displayType` *is 0, the display type sets to "opaque". If* `displayType` *is 1, the display mode sets to "transparent". The function returns 1, if the task succeeds, returns 0, otherwise. The* `tid` *is the transaction ID of a payment for using this function.*
float	setDisplayMode_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `setDisplayMode()`*.If* `tokenType` *is 0, it denotes "cryptocurrency", if* `tokenType` *is 1, it denotes "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) setDisplayMode_Cost(0, "BTC") or setDisplayMode_Cost(0, "00000001") Ex) setDisplayMode_Cost(1, "USD") or setDisplayMode_Cost(1, "10010100")
int	getDisplayMode()
	This function returns the current display type. The function returns 0, if the display is "opaque", returns 1, if the display is transparent.

int	getDisplayMode(String tid)
	This function returns the current display type. The function returns 0, if the display is "opaque", returns 1, if the display is transparent. The `tid` *is the transaction ID of a payment for using this function.*
float	getDisplayMode_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `getDisplayMode()`*.If* `tokenType` *is 0, it denotes "cryptocurrency", if* `tokenType` *is 1, it denotes "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) getDisplayMode_Cost(0, "BTC") or getDisplayMode_Cost(0, "00000001") Ex) getDisplayMode_Cost(1, "USD") or getDisplayMode_Cost(1, "10010100")

4.3.5 API for IoMT camera actuator

General

This subclause defines a class of an IoMT camera actuator which shall inherit the features of `MActuator` class.

APIs

Table 14 presents APIs of an IoMT camera actuator.

Table 14 — IoMT camera actuator API

Nested Classes	
Modifier and Type	Method and Description
Constructor	
Constructor and Description	
MCameraActuator()	
Default constructor.	
MCameraActuator(String id)	

MCameraActuator(String id, String serverIPAddress, integer serverPort)	
Fields	
Modifier and Type	Field and Description
Methods	
Modifier and Type	Method and Description
int	setCameraOrientation(ActuationDataType orientation)
	This function controls a camera orientation. The function returns 1, if the task succeeds, returns 0, otherwise.
int	setCameraOrientation(String tid, ActuationDataType orientation)
	This function receives the orientation sent by other MThings. This function returns 1, if the task secceeds, returns 0, otherwise. The `tid` *is the transaction ID of a payment for using this function.*
float	setCameraOrientation_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `setCameraOrientation()`*. If* `tokenType` *is 0, it means "cryptocurrency", if* `tokenType` *is 1, it means "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) setCameraOrientation_Cost(0, "BTC") or setCameraOrientation_Cost(0, "00000001") Ex) setCameraOrientation_Cost(1, "USD") or setCameraOrientation_Cost(1, "10010100")
int	setCameraZoom(ActuationDataType zoom)
	This function controls a camera zoom. The function returns 1, if the task succeeds, returns 0, otherwise.

int	setCameraZoom(String tid, ActuationDataType zoom)
	This function receives the zoom sent by other MHings. This function returns 1, if the task secceeds, returns 0, otherwise. The `tid` *is the transaction ID of a payment for using this function.*
float	setCameraZoom_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `setCameraZoom()`*. If* `tokenType` *is 0, it means "cryptocurrency", if* `tokenType` *is 1, it means "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) setCameraZoom_Cost(0, "BTC") or setCameraZoom_Cost(0, "00000001") Ex) setCameraZoom_Cost(1, "USD") or setCameraZoom_Cost(1, "10010100")
int	setCameraResolution(ActuationDataType resolution)
	This function controls the capturing resolution of a video source. The function returns 1, if the task succeeds, returns 0, otherwise.
int	setCameraResolution(String tid, ActuationDataType resolution)
	This function receives the resolution sent by other MThings. This function returns 1, if the task secceeds, returns 0, otherwise. The `tid` *is the transaction ID of a payment for using this function.*
float	setCameraResolution_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `setCameraResolution()`*. If* `tokenType` *is 0, it means "cryptocurrency", if* `tokenType` *is 1, it means "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) setCameraResolution_Cost(0, "BTC") or setCameraResolution_Cost(0, "00000001") Ex) setCameraResolution_Cost(1, "USD") or setCameraResolution_Cost(1, "10010100")

4.3.6 API for IoMT hand gesture actuator

General

This subclause defines a class of an IoMT hand gesture actuator which shall inherit the features of `MActuator` class.

APIs

Table 15 presents APIs of an IoMT hand gesture actuator.

Table 15 — IoMT hand gesture actuator API

Nested Classes	
Modifier and Type	Method and Description
Constructor	
Constructor and Description	
MHandGestureActuator()	
MHandGestureActuator(String id)	
MHandGestureActuator(String id, String serverIPAddress, integer serverPort)	
Fields	
Modifier and Type	Field and Description
Methods	
Modifier and Type	Method and Description
int	setHandGestureCommand(ActuationDataType GestureCommand)
	Set a command to an IoMT hand gesture actuator following the specification of the `GestureCommandCS` *in accordance with* A.4.3. *The function returns 1, if the task succeeds, returns 0, otherwise.*

int	setHandGestureCommand(String tid, ActuationDataType GestureCommand)
	Set a command to an IoMT hand gesture actuator following the specification of the `GestureCommandCS` (A.4.3). *The function returns 1, if the task succeeds, returns 0, otherwise. The* `tid` *is the transaction ID of a payment for using this function.*
float	setHandGestureCommand_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `setHandGestureCommand()`. *If* `tokenType` *is 0, it denotes "cryptocurrency", if* `tokenType` *is 1, it denotes "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) setHandGestureCommand_Cost(0, "BTC") or setHandGestureCommand_Cost(0, "00000001") Ex) setHandGestureCommand_Cost(1, "USD") or setHandGestureCommand_Cost(1, "10010100")

4.3.7 API for IoMT vibrator

General

This subclause defines a class of an IoMT vibator which shall inherit the features of `MActuator` class.

APIs

Table 16 presents APIs of an IoMT vibrator.

Table 16 — IoMT vibrator API

Nested Classes	
Modifier and Type	Method and Description
Constructor	
Constructor and Description	
MVibrator()	
Default constructor.	

MVibrator(String id)	
MVibrator(String id, String serverIPAddress, integer serverPort)	
Fields	
Modifier and Type	Field and Description
Methods	
Modifier and Type	Method and Description
int	setVibration(MPEGVCommandType vibration)
	The function sets a command with a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and a device command defined by `VibrationType` *from ISO/IEC 23005-5 (MPEG-V Part 5). The function returns 1, if the task succeeds, returns 0, otherwise.*
int	setVibration(String tid, MPEGVCommandType vibration)
	The function sets a command with a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and a device command defined by `VibrationType` *from ISO/IEC 23005-5 (MPEG-V Part 5). The function returns 1, if the task succeeds, returns 0, otherwise. The* `tid` *is the transaction ID of a payment for using this function.*
float	setVibration_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `setVibration()`*. If* `tokenType` *is 0, it means "cryptocurrency", if* `tokenType` *is 1, it means "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) setVibration_Cost(0, "BTC") or setVibration_Cost(0, "00000001") Ex) setVibration_Cost(1, "USD") or setVibration_Cost(1, "10010100")

4.3.8 API for IoMT sprayer

General

This subclause defines a class of an IoMT sprayer which shall inherit the features of `MActuator` class.

APIs

Table 17 presents APIs of an IoMT sprayer.

Table 17 — IoMT sprayer API

Nested Classes	
Modifier and Type	Method and Description
Constructor	
Constructor and Description	
MSprayer()	
Default constructor.	
MSprayer(String id)	
MSprayer(String id, String serverIPAddress, integer serverPort)	
Fields	
Modifier and Type	Field and Description
Methods	
Modifier and Type	Method and Description
int	setSprayWater(MPEGVCommandType waterSpray)
	This function sets a command to a water sprayer with a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and sensed data defined by `SprayerType` *from ISO/IEC 23005-5 (MPEG-V Part 5). This function returns 1, if the task succeeds, returns 0, otherwise.*

int	setSprayWater(String tid, MPEGVCommandType waterSpray)
	This function sets a command to a water sprayer with a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and sensed data defined by `SprayerType` *from ISO/IEC 23005-5 (MPEG-V Part 5). This function returns 1, if the task succeeds, returns 0, otherwise. The* `tid` *is the transaction ID of a payment for using this function.*
float	setSprayWater_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `setSprayWater()`*. If* `tokenType` *is 0, it means "cryptocurrency", if* `tokenType` *is 1, it means "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) setSprayWater_Cost(0, "BTC") or setSprayWater_Cost(0, "00000001") Ex) setSprayWater_Cost(1, "USD") or setSprayWater_Cost(1, "10010100")
int	setSprayScent(MPEGVCommandType scentSpray)
	This function sets a command to a scent sprayer with a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and sensed data defined by `ScentType` *from ISO/IEC 23005-5 (MPEG-V Part 5). This function returns 1, if the task succeeds, returns 0 otherwise.*
int	setSprayScent(String tid, MPEGVCommandType scentSpray)
	This function sets a command to a scent sprayer with a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and sensed data defined by `ScentType` *from ISO/IEC 23005-5 (MPEG-V Part 5). This function returns 1, if the task succeeds, returns 0, otherwise. The* `tid` *is the transaction ID of a payment for using this function.*
float	setSprayScent_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `setSprayScent()`*. If* `tokenType` *is 0, it means "cryptocurrency", if* `tokenType` *is 1, it means "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) setSprayScent_Cost(0, "BTC") or setSprayScent_Cost(0, "00000001") Ex) setSprayScent_Cost(1, "USD") or setSprayScent_Cost(1, "10010100")

int	setSprayFog(MPEGVCommandType fogSpray)
	This function sets a command to a fog sprayer with a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and sensed data defined by `FogType` *from ISO/IEC 23005-5 (MPEG-V Part 5). This function returns 1, if the task succeeds, returns 0, otherwise.*
int	setSprayFog(String tid, MPEGVCommandType fogSpray)
	This function sets a command to a fog sprayer with a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and sensed data defined by `FogType` *from ISO/IEC 23005-5 (MPEG-V Part 5). This function returns 1, if the task succeeds, returns 0, otherwise. The* `tid` *is the transaction ID of a payment for using this function.*
float	setSprayFog_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `setSprayFog()`*. If* `tokenType` *is 0, it means "cryptocurrency", if* `tokenType` *is 1, it means "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) setSprayFog_Cost(0, "BTC") or setSprayFog_Cost(0, "00000001") Ex) setSprayFog_Cost(1, "USD") or setSprayFog_Cost(1, "10010100")
int	setSprayBubble(MPEGVCommandType bubbleSpray)
	This function sets a command to a bubble sprayer with a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and sensed data defined by `BubbleType` *from ISO/IEC 23005-5 (MPEG-V Part 5). This function returns 1, if the task succeeds, returns 0, otherwise.*
int	setSprayBubble(String tid, MPEGVCommandType bubbleSpray)
	This function sets a command to a bubble sprayer with a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and sensed data defined by `BubbleType` *from ISO/IEC 23005-5 (MPEG-V Part 5). This function returns 1, if the task succeeds, returns 0, otherwise. The* `tid` *is the transaction ID of a payment for using this function.*

float	setSprayBubble_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `setSprayBubble()`. *If* `tokenType` *is 0, it means "cryptocurrency", if* `tokenType` *is 1, it means "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) setSprayBubble_Cost(0, "BTC") or setSprayBubble_Cost(0, "00000001") Ex) setSprayBubble_Cost(1, "USD") or setSprayBubble_Cost(1, "10010100")

4.3.9 API for IoMT light

General

This subclause defines a class of an IoMT light which shall inherit the features of `MActuator` class.

APIs

Table 18 presents APIs of an IoMT light.

Table 18 — IoMT light API

Nested Classes	
Modifier and Type	Method and Description
Constructor	
Constructor and Description	
MLight() *Default constructor.*	
MLight(String id)	
MLight(String id, String serverIPAddress, integer serverPort)	

Fields	
Modifier and Type	Field and Description
Methods	
Modifier and Type	Method and Description
int	setLight(MPEGVCommandType light)
	This function sets a command to a light with a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and sensed data defined by `LightType` *or* `FlashType` *from ISO/IEC 23005-5 (MPEG-V Part 5). This function returns 1, if the task succeeds, returns 0, otherwise.*
int	setLight(String tid, MPEGVCommandType light)
	This function sets a command to a light with a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and sensed data defined by `LightType` *or* `FlashType` *from ISO/IEC 23005-5 (MPEG-V Part 5). This function returns 1, if the task succeeds, returns 0, otherwise. The* `tid` *is the transaction ID of a payment for using this function.*
float	setLight_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `setLight()`*. If* `tokenType` *is 0, it means "cryptocurrency", if* `tokenType` *is 1, it means "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) setLight_Cost(0, "BTC") or setLight_Cost(0, "00000001") Ex) setLight_Cost(1, "USD") or setLight_Cost(1, "10010100")
int	setArrayedLight(MPEGVCommandType arrayedLight)
	This function sets a command to an arrayed light with a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and sensed data defined by `ArrayedLightType` *from ISO/IEC 23005-5 (MPEG-V Part 5). This function returns 1, if the task succeeds, returns 0, otherwise.*

int	setArrayedLight(String tid, MPEGVCommandType arrayedLight)
	This function sets a command to an arrayed light with a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and sensed data defined by `ArrayedLightType` *from ISO/IEC 23005-5 (MPEG-V Part 5). This function returns 1, if the task succeeds, returns 0, otherwise. The* `tid` *is the transaction ID of a payment for using this function.*
float	setArrayedLight_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `setArrayedLight()`*. If* `tokenType` *is 0, it means "cryptocurrency", if* `tokenType` *is 1, it means "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) setArrayedLight_Cost(0, "BTC") or setArrayedLight_Cost(0, "00000001") Ex) setArrayedLight_Cost(1, "USD") or setArrayedLight_Cost(1, "10010100")
int	setBrightness(ActuationDataType brightness)
	This function controls the brightness of the light. This function returns 1, if the task succeeds, returns 0, otherwise.
int	setBrightness(String tid, ActuationDataType brightness)
	This function controls the brightness of the light. This function returns 1, if the task succeeds, return 0, otherwise. `tid` *is a token transaction ID of the corresponding service.*
float	setBrightness_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `setBrightness()`*. If* `tokenType` *is 0, it means "cryptocurrency", if* `tokenType` *is 1, it means "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) setBrightness_Cost(0, "BTC") or setBrightness_Cost(0, "00000001") Ex) setBrightness_Cost(1, "USD") or setBrightness_Cost(1, "10010100")
int	setHue(ActuationDataType hue)
	This function controls the hue value of the light. This function returns 1, if the task succeeds, returns 0, otherwise.

int	setHue(String tid, ActuationDataType hue)
	This function controls the hue value of the light. This function returns 1, if the task succeeds, returns 0, otherwise. `tid` *is a token transaction ID of the corresponding service.*
float	setHue_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `setHue()`*. If* `tokenType` *is 0, it means "cryptocurrency", if* `tokenType` *is 1, it means "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) setHue_Cost(0, "BTC") or setHue_Cost(0, "00000001") Ex) setHue_Cost(1, "USD") or setHue_Cost(1, "10010100")
int	setSaturation(ActuationDataType saturation)
	This function controls the saturation value of the light. This function returns 1, if the task succeeds, returns 0, otherwise.
int	setSaturation(String tid, ActuationDataType saturation)
	This function controls the saturation value of the light. This function returns 1, if the task succeeds, returns 0, otherwise. `tid` *is a token transaction ID of the corresponding service.*
float	setSaturation_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `setSaturation()`*. If* `tokenType` *is 0, it means "cryptocurrency", if* `tokenType` *is 1, it means "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) setSaturation_Cost(0, "BTC") or setSaturation_Cost(0, "00000001") Ex) setSaturation_Cost(1, "USD") or setSaturation_Cost(1, "10010100")

4.4 APIs for IoMT analyzers

4.4.1 General

This subclause defines API classes of IoMT analyzers.

4.4.2 MAnalyzer class

General

This subclause defines an MAnalyzer class, which shall inherit the features of `MThing` class defined in ISO/IEC 23093-2.

APIs

Table 19 presents basic APIs of MAnalyzer.

Table 19 — MAnalyzer API

Nested Classes	
Modifier and Type	Method and Description
Constructor	
Constructor and Description	
MAnalyzer()	
Default constructor.	
MAnalyzer(String id)	
MAnalyzer(String id, String serverIPAddress, integer serverPort)	
Fields	
Modifier and Type	Field and Description
Methods	
Modifier and Type	Method and Description
CapabilityListType	getAnalyzerCapabilityList();
	This function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and a capability list using data formats defined in this document.

CapabilityListType	getAvailableAnalyzerCapabilityList()
	This function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and an available capability list using data formats defined in this document.
CapabilityListType	getAppliedAnalyzerCapabilityList()
	This function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and an applied capability list using data formats defined in this document.
AnalyzedDataType	getAnalyzedData()
	This function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and analyzed data following the specification in this document.

4.4.3 API for IoMT time synchronizer

General

This subclause defines a class of an IoMT time synchronizer which shall inherit the features of `MAnalyzer` class.

APIs

Table 20 presents APIs of an IoMT time synchronizer.

Table 20 — IoMT time synchronizer API

Nested Classes	
Modifier and Type	Method and Description
Constructor	
Constructor and Description	
MTimeSynchronizer()	
Default constructor.	

MTimeSynchronizer(String id)	
MTimeSynchronizer(String id, String serverIPAddress, integer serverPort)	
Fields	
Modifier and Type	Field and Description
Methods	
Modifier and Type	Method and Description
AnalyzedDataType	getSyncedVideo()
	This function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and video sources and audio sources with a time offset following the specification in this document.
AnalyzedDataType	getSyncedVideo(String tid)
	This function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and video sources and audio sources with a time offset following the specification in this document. The `tid` *is the transaction ID of a payment for using this function.*
float	getSyncedVideo_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `getSyncedVideo()`*. If* `tokenType` *is 0, it denotes "cryptocurrency", if* `tokenType` *is 1, it denotes "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) getSyncedVideo_Cost(0, "BTC") or getSyncedVideo_Cost(0, "00000001") Ex) getSyncedVideo_Cost(1, "USD") or getSyncedVideo_Cost(1, "10010100")

4.4.4 API for IoMT social event detector

General

This subclause defines a class of an IoMT social event detector which shall inherit the features of `MAnalyzer` class.

APIs

Table 21 presents APIs of an IoMT social event detector.

Table 21 — IoMT social event detector API

Nested Classes	
Modifier and Type	Method and Description
Constructor	
Constructor and Description	
MSocialEventDetector()	
Default constructor.	
MSocialEventDetector(String id)	
MSocialEventDetector(String id, String serverIPAddress, integer serverPort)	
Fields	
Modifier and Type	Field and Description
Methods	
Modifier and Type	Method and Description
AnalyzedDataType	getSocialEventName()
	This function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and a name of detected social event following the specification in this document.

AnalyzedDataType	getSocialEventName(String tid)
	This function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and a name of detected social event following the specification in this document. The `tid` *is the transaction ID of a payment for using this function.*
float	getSocialEventName_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `getSocialEventName()`*. If* `tokenType` *is 0, it denotes "cryptocurrency", if* `tokenType` *is 1, it denotes "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) getSocialEventName_Cost(0, "BTC") or getSocialEventName_Cost(0, "00000001") Ex) getSocialEventName _Cost(1, "USD") or getSocialEventName_Cost(1, "10010100")

4.4.5 API for IoMT hand gesture detector

General

This subclause defines a class of an IoMT hand gesture detector which shall inherit the features of `MAnalyzer` class.

APIs

Table 22 presents APIs of an IoMT hand gesture detector.

Table 22 — IoMT hand gesture detector API

Nested Classes	
Modifier and Type	Method and Description
Constructor	
Constructor and Description	
MHandGestureDetector()	

MHandGestureDetector(String id)	
MHandGestureDetector(String id, String serverIPAddress, integer serverPort)	
Fields	
Modifier and Type	Field and Description
Methods	
Modifier and Type	Method and Description
AnalyzedDataType	getDetectedHandContour()
	This function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and Bezier curves specified in this document.
AnalyzedDataType	getDetectedHandContour(String tid)
	This function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and Bezier curves specified in this document. The `tid` *is the transaction ID of a payment for using this function.*
float	getDetectedHandContour_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `getDetectedHandContour()`*. If* `tokenType` *is 0, it denotes "cryptocurrency", if* `tokenType` *is 1, it denotes "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) getDetectedHandContour_Cost(0, "BTC") or getDetectedHandContour_Cost(0, "00000001") Ex) getDetectedHandContour_Cost(1, "USD") or getDetectedHandContour_Cost(1, "10010100")

4.4.6 API for IoMT hand gesture recognizer

General

This subclause defines a class of an IoMT hand gesture recognizer which shall inherit the features of `MAnalyzer` class.

APIs

Table 23 presents APIs of an IoMT hand gesture recognizer.

Table 23 — IoMT hand gesture recognizer API

Nested Classes	
Modifier and Type	Method and Description
Constructor	
Constructor and Description	
MHandGestureRocognizer()	
MHandGestureRocognizer(String id)	
MHandGestureRocognizer(String id, String serverIPAddress, integer serverPort)	
Fields	
Modifier and Type	Field and Description
Methods	
Modifier and Type	Method and Description
AnalyzedDataType	getRecognizedHandPosture()
	The function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and hand postures specified in this document.

AnalyzedDataType	getRecognizedHandPosture(String tid)
	The function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and hand postures specified in this document. The `tid` *is the transaction ID of a payment for using this function.*
float	getRecognizedHandPosture_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `getRecognizedHandPosture()`*. If* `tokenType` *is 0, it denotes "cryptocurrency", if* `tokenType` *is 1, it denotes "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) getRecognizedHandPosture_Cost(0, "BTC") or getRecognizedHandPosture_Cost(0, "00000001") Ex) getRecognizedHandPosture_Cost(1, "USD") or getRecognizedHandPosture_Cost(1, "10010100")

4.4.7 API for IoMT healthcare information generator

General

This subclause defines a class of an IoMT healthcare information generator which shall inherit the features of `MAnalyzer` class.

APIs

Table 24 presents APIs of an IoMT healthcare information generator.

Table 24 — IoMT healthcare information generator API

Nested Classes	
Modifier and Type	Method and Description
Constructor	
Constructor and Description	
HealthcareInfoGenerator()	

HealthcareInfoGenerator(String id)	
HealthcareInfoGenerator(String id, String serverIPAddress, integer serverPort)	
Fields	
Modifier and Type	Field and Description
Methods	
Modifier and Type	Method and Description
AnalyzedDataType	HealthcareInfoGenerator()
	The function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and Healthcare information data type using data formats defined in this document.
AnalyzedDataType	HealthcareInfoGenerator(String tid)
	The function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and Healthcare information data type using data formats defined in this document. The `tid` *is the transaction ID of a payment for using this function.*
float	getHealthcareInfoGenerator_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `getHealthcareInfoGenerator_Cost()`*. If* `tokenType` *is 0, it denotes "cryptocurrency", if* `tokenType` *is 1, it denotes "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) getHealthcareInfoGenerator_Cost(0, "BTC") or getHealthcareInfoGenerator_Cost(0, "00000001") Ex) getHealthcareInfoGenerator_Cost(1, "USD") or getHealthcareInfoGenerator_Cost(1, "10010100")

4.4.8 API for IoMT speech recognizer

General

This subclause defines a class of an IoMT speech recognizer which shall inherit the features of `MAnalyzer` class.

APIs

Table 25 presents APIs of an IoMT speech recognizer.

Table 25 — IoMT speech recognizer API

Nested Classes	
Modifier and Type	Method and Description
Constructor	
Constructor and Description	
MSpeechRecognizer()	
Default constructor.	
MSpeechRecognizer(String id)	
MSpeechRecognizer(String id, String serverIPAddress, integer serverPort)	
Fields	
Modifier and Type	Field and Description
Methods	
Modifier and Type	Method and Description
String	getSpeechText()
	This function returns extracted speech text which is String type as the result of speech recognition.

String	getSpeechText(String tid)
	This function returns extracted speech text which is String type as the result of speech recognition. The `tid` *is the transaction ID of a payment for using this function.*
float	getSpeechText_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `getSpeechText()`*. If* `tokenType` *is 0, it means "cryptocurrency", if* `tokenType` *is 1, it means "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) getSpeechText _Cost(0, "BTC") or getSpeechText_Cost(0, "00000001") Ex) getSpeechText _Cost(1, "USD") or getSpeechText _Cost(1, "10010100")

4.4.9 API for IoMT text to speech converter

General

This subclause defines a class of an IoMT text to speech converter which shall inherit the features of `MAnalyzer` class.

APIs

Table 26 presents APIs of an IoMT text to speech converter.

Table 26 — IoMT text to speech converter API

Nested Classes	
Modifier and Type	Method and Description
Constructor	
Constructor and Description	
MTextToSpeech()	
Default constructor.	

MTextToSpeech(String id)	
MTextToSpeech(String id, String serverIPAddress, integer serverPort)	
Fields	
Modifier and Type	Field and Description
Methods.	
Modifier and Type	Method and Description
String	getSpeechURL()
	This function returns a URL of the speech audio source in String type which is the result of text to speech converter.
String	getSpeechURL(String tid)
	This function returns a URL of the speech audio source in String type which is the result of text to speech converter. The `tid` *is the transaction ID of a payment for using this function.*
float	getSpeechURL_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `getSpeechURL()`*. If* `tokenType` *is 0, it means "cryptocurrency", if* `tokenType` *is 1, it means "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) getSpeechURL _Cost(0, "BTC") or getSpeechURL_Cost(0, "00000001") Ex) getSpeechURL _Cost(1, "USD") or getSpeechURL_Cost(1, "10010100")

4.4.10 API for IoMT question analyzer

General

This subclause defines a class of an IoMT question analyzer which shall inherit the features of `MAnalyzer` class.

APIs

Table 27 presents APIs of an IoMT guestion analyzer.

Table 27 — IoMT question analyzer API

Nested Classes	
Modifier and Type	Method and Description
Constructor	
Constructor and Description	
QuestionAnalyzer()	
Default constructor.	
QuestionAnalyzer(String id)	
QuestionAnalyzer(String id, String serverIPAddress, integer serverPort)	
Fields	
Modifier and Type	Field and Description
Methods	
Modifier and Type	Method and Description
AnalyzedDataType	getUserQuestion()
	This function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and user question following the specification in this document.

AnalyzedDataType	getUserQuestion(String tid)
	This function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and user question following the specification in this document. The `tid` *is the transaction ID of a payment for using this function.*
float	getUserQuestion_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `getUserQuestion()`*. If* `tokenType` *is 0, it denotes "cryptocurrency", if* `tokenType` *is 1, it denotes "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) getUserQuestion_Cost(0, "BTC") or getUserQuestion_Cost(0, "00000001") Ex) getUserQuestion_Cost(1, "USD") or getUserQuestion_Cost(1, "10010100")

4.4.11 API for IoMT odor image to scent converter

General

This subclause defines a class of an IoMT odor image to scent converter which shall inherit the features of the `MAnalyzer` class.

APIs

Table 28 presents APIs of an IoMT odor image to scent converter.

Table 28 — IoMT odor image to scent converter API

Nested Classes	
Modifier and Type	Method and Description
Constructor	
Constructor and Description	
MOdorImageToScentConverter()	
Default constructor.	

MOdorImageToScentConverter(String id)	
MOdorImageToScentConverter(String id, String serverIPAddress, integer serverPort)	
Fields	
Modifier and Type	Field and Description
Methods	
Modifier and Type	Method and Description
AnalyzedDataType	getOdorImageToScentConverterOutput()
	This function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and recognized odor images following the specification in this document.
AnalyzerDataType	getOdorImageToScentConverterOutput(String tid)
	This function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and recognized odor images following the specification in this document. The `tid` *is the transaction ID of a payment for using this function.*
float	getOdorImageToScentConverterOutput_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `getOdorImageToScentConverterOutput()`*. If* `tokenType` *is 0, it denotes "cryptocurrency", if* `tokenType` *is 1, it denotes "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) getOdortImageToScentConverterOutput_Cost(0, "BTC") or getOdorImageToScentConverterOutput _Cost(0, "00000001") Ex) getOdorImageToScentConverterOutput_Cost(1, "USD") or getOdorImageToScentConverterOutput_Cost(1, "10010100")

4.4.12 API for IoMT direction guider

General

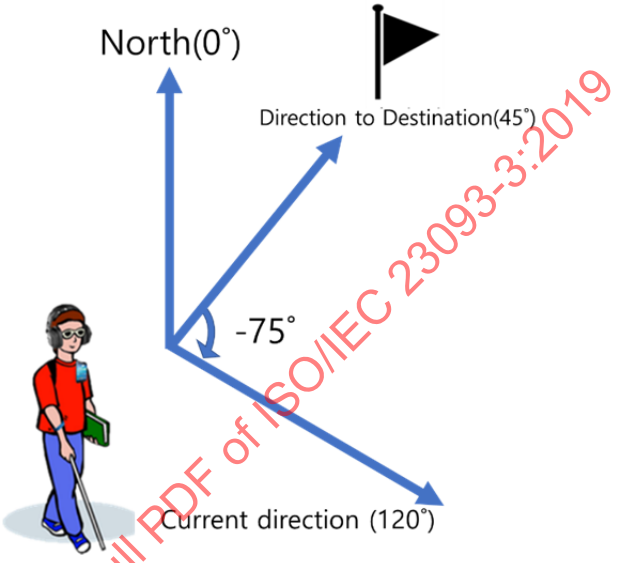
This subclause defines a class of an IoMT direction guider which shall inherit the features of `MAnalyzer` class.

APIs

Table 29 presents APIs of an IoMT direction guider.

Table 29 — IoMT direction guider API

Nested Classes	
Modifier and Type	Method and Description
Constructor	
Constructor and Description	
MDirectionGuider()	
Default constructor.	
MDirectionGuider(String id)	
MDirectionGuider(String id, String serverIPAddress, integer serverPort)	
Fields	
Modifier and Type	Field and Description

Methods	
Modifier and Type	Method and Description
float	getRelativeAzimuth()
	The function returns a relative clockwise azimuth angle between the current direction and the direction to destination. (In the case below, the method returns –75.) (Figure 4)  **Figure 4 — Relative azimuth angle**
float	getRelativeAzimuth(String tid)
	The function returns a relative clockwise azimuth angle between the current direction and the direction to destination.The `tid` *is the transaction ID of a payment for using this function.*
float	getRelativeAzimuth_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `getRelativeAzimuth()`*. If* `tokenType` *is 0, it means "cryptocurrency", if* `tokenType` *is 1, it means "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) getRelativeAzimuth_Cost(0, "BTC") or getRelativeAzimuth_Cost(0, "00000001") Ex) getRelativeAzimuth_Cost(1, "USD") or getRelativeAzimuth_Cost(1, "10010100")

String	getDirectionGuidingText()
	The function returns a human-readable description of the direction to the destination(e.g. "Turn 75 degrees to the left").
String	getDirectionGuidingText(String tid)
	This function returns a human-readable description of the direction to the destination(e.g. "Turn 75 degrees to the left"). The `tid` *is the transaction ID of a payment for using this function.*
float	getDirectionGuidingText_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `getDirectionGuidingText()`*. If* `tokenType` *is 0, it means "cryptocurrency", if* `tokenType` *is 1, it means "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) getDirectionGuidingText_Cost(0, "BTC") or getDirectionGuidingText_Cost(0, "00000001") Ex) getDirectionGuidingText_Cost(1, "USD") or getDirectionGuidingText_Cost(1, "10010100")

4.4.13 API for IoMT collision coordinator

General

This subclause defines a class of an IoMT collision coordinator which shall inherit the features of `MAnalyzer` class.

APIs

Table 30 presents APIs of an IoMT collision coordinator.

Table 30 — IoMT collision coordinator API

Nested Classes	
Modifier and Type	Method and Description
Constructor	
Constructor and Description	
MCollisionCoordinator()	
Default constructor.	
MCollisionCoordinator(String id)	
MCollisionCoordinator(String id, String serverIPAddress, integer serverPort)	
Fields	
Modifier and Type	Field and Description
Methods	
Modifier and Type	Method and Description
boolean	getDetectObstacle()
	The function returns whether there is an obstacle in front or not.
boolean	getDetectObstacle(String tid)
	The function returns whether there is an obstacle in front or not. The `tid` *is the transaction ID of a payment for using this function.*

float	getDetectObstacle_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `getDetectObstacle()`. *If* `tokenType` *is 0, it means "cryptocurrency", if* `tokenType` *is 1, it means "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) getDetectObstacle_Cost(0, "BTC") or getDetectObstacle_Cost(0, "00000001") Ex) getDetectObstacle_Cost(1, "USD") or getDetectObstacle_Cost(1, "10010100")
String	getTextOfDetectedObstacle()
	The function returns a human-readable statement about the obstacle detection.
String	getTextOfDetectedObstacle(String tid)
	The function returns a human-readable statement about the obstacle detection. The `tid` *is the transaction ID of a payment for using this function.*
float	getTextOfDetectedObstacle_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `getDetectObstacleText()`. *If* `tokenType` *is 0, it means "cryptocurrency", if* `tokenType` *is 1, it means "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) getTextOfDetectedObstacle_Cost(0, "BTC") or getTextOfDetectedObstacle_Cost(0, "00000001") Ex) getTextOfDetectedObstacle_Cost(1, "USD") or getTextOfDetectedObstacle_Cost(1, "10010100")

4.4.14 API for IoMT people counter

General

This subclause defines a class of an IoMT people counter, which shall inherit the features of `MAnalyzer` class.

APIs

Table 31 presents APIs of an IoMT people counter.

Table 31 — IoMT people counter API

Nested Classes	
Modifier and Type	Method and Description
Constructor	
Constructor and Description	
MPeopleCounter()	
Default constructor.	
MPeopleCounter(String id)	
MPeopleCounter(String id, String serverIPAddress, integer serverPort)	
Fields	
Modifier and Type	Field and Description
Methods	
Modifier and Type	Method and Description
int	getNumberofPeople()
	This function returns how many people appear in a video.

int	getNumberofPeople(String tid)
	This function returns how many people appear in a video. The `tid` *is a payment transaction ID of the corresponding service.*
int	getNumberofPeopleInPeriod(long start_time, long end_time)
	This function returns how many people appear in a video in the period. The `start_time` *denotes a start time of the period, and the* `end_time` *denotes an end time of the period. These are expressed in Unix Time in seconds.*
int	getNumberofPeopleInPeriod(String tid, long start_time, long end_time)
	This function returns how many people appear in a video in the period. The `start_time` *denotes a start time of the period, and the* `end_time` *denotes an end time of the period. These are expressed in Unix Time in seconds. The* `tid` *is the transaction ID of a payment for using this function.*
int	getActivityLevel()
	This function returns a number between 1 and 3 indicating how crowded the (filmed) place is. (1=low, 2=middle, 3=high).
int	getActivityLevel(string tid)
	This function returns a number between 1 and 3 indicating how crowded the (filmed) place is. (1=low, 2=middle, 3=high). The `tid` *is the transaction ID of a payment for using this function.*
float	getNumberofPeople_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `getNumberofPeople()`*. If* `tokenType` *is 0, it denotes "cryptocurrency", if* `tokenType` *is 1, it denotes "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) getNumberofPeople_Cost(0, "BTC") or getNumberofPeople_Cost(0, "00000001") Ex) getNumberofPeople_Cost(1, "USD") or getNumberofPeople_Cost(1, "10010100")

float	getNumberofPeopleInPeriod_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `getNumberofPeopleInPeriod()`*. If* `tokenType` *is 0, it denotes "cryptocurrency", if* `tokenType` *is 1, it denotes "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) getNumberofPeopleInPeriod_Cost(0, "BTC") or getNumberofPeopleInPeriod_Cost(0, "00000001") Ex) getNumberofPeopleInPeriod_Cost(1, "USD") or getNumberofPeopleInPeriod_Cost(1, "10010100")
float	getActivityLevel_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `getActivityLevel()`*. If* `tokenType` *is 0, it denotes "cryptocurrency", if* `tokenType` *is 1, it denotes "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) getActivityLevel_Cost(0, "BTC") or getActivityLevel_Cost(0, "00000001") Ex) getActivityLevel_Cost(1, "USD") or getActivityLevel_Cost(1, "10010100")

4.4.15 API for IoMT music frequency analyzer

General

This subclause defines a class of an IoMT music frequency analyzer, which shall inherit the features of `MAnalyzer` class.

APIs

Table 32 presents APIs of an IoMT music frequency analyzer.

Table 32 — IoMT music frequency analyzer API

Nested Classes	
Modifier and Type	Method and Description
Constructor	
Constructor and Description	
MMusicFrequencyAnalyzer()	
Default constructor.	
MMusicFrequencyAnalyzer(String id)	
MMusicFrequencyAnalyzer(String id, String serverIPAddress, integer serverPort)	
Fields	
Modifier and Type	Field and Description
Methods	
Modifier and Type	Method and Description
String	getAnalyzedMusicFreqURL()
	This function returns a URL of a generated music frequency.
String	getAnalyzedMusicFreqURL(String tid)
	This function returns a URL of a generated music frequency. `tid` *is a token transaction ID of the corresponding service.*

float	getAnalyzedMusicFreqURL_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `getAnalyzedMusicFreqURL()`*. If* `tokenType` *is 0, it means "cryptocurrency", if* `tokenType` *is 1, it means "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) getAnalyzedMusicFreqURL_Cost(0, "BTC") or getAnalyzedMusicFreqURL_Cost(0, "00000001") Ex) getAnalyzedMusicFreqURL_Cost(1, "USD") or getAnalyzedMusicFreqURL_Cost(1, "10010100")

4.4.16 API for IoMT light color converter

General

This subclause defines a class of an IoMT light color converter which shall inherit the features of `MAnalyzer` class.

APIs

Table 33 presents APIs of an IoMT light color converter.

Table 33 – IoMT light color converter API

Nested Classes	
Modifier and Type	Method and Description
Constructor	
Constructor and Description	
MLightColorConverter()	
Default constructor.	
MLightColorConverter(String id)	
MLightColorConverter(String id, String serverIPAddress, integer serverPort)	

Fields	
Modifier and Type	Field and Description
Methods	
Modifier and Type	Method and Description
AnalyzedDataType	getAnalyzedColorLightInfo()
	This function returns a class (i.e. Java or C++) or a structure(i.e. C) that shall include a returning type and a analyzed color light information according to music characteristics.
AnalyzedDataType	getAnalyzedColorLightInfo(String tid)
	This function returns a class (i.e. Java or C++) or a structure(i.e. C) that shall include a returning type and a analyzed color light information according to music characteristics. `tid` *is a token transaction ID of the corresponding service.*
float	getAnalyzedColorLightInfo_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `getAnalyzedColorLightInfo()`*. If* `tokenType` *is 0, it means "cryptocurrency", if* `tokenType` *is 1, it means "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) getAnalyzedColorLightInfo_Cost(0, "BTC") or getAnalyzedColorLightInfo_Cost(0, "00000001") Ex) getAnalyzedColorLightInfo_Cost(1, "USD") or getAnalyzedColorLightInfo_Cost(1, "10010100")

4.4.17 API for IoMT video content class generator

General

This subclause defines a class of the an IoMT video content class generator which shall inherit the features of `MAnalyzer` class.

APIs

Table 34 presents APIs of an IoMT video content class generator.

Table 34 — IoMT video content class generator API

Nested Classes	
Modifier and Type	Method and Description
Constructor	
Constructor and Description	
MVideoClassGenerator()	
Default constructor.	
MVideoClassGenerator(String id)	
MVideoClassGenerator(String id, String serverIPAddress, integer serverPort)	
Fields	
Modifier and Type	Field and Description
Methods	
Modifier and Type	Method and Description
AnalyzedDataType	getClassofVideoContent()
	This function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type and generated video content class.
AnalyzedDataType	getClassofVideoContent(String tid)
	This function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type and generated video content class. `tid` *is a token transaction ID of the corresponding service.*

float	getClassofVideoContent_Cost(int tokenType, String tokenName)
	This function returns the amount of tokens to use `getClassofVideoContent()`. *If* `tokenType` *is 0, it means "cryptocurrency", if* `tokenType` *is 1, it means "legal tender". The* `tokenName` *is described in string (e.g. term ID or binary representation) from* `TokenTypeCS` *specified in A.5. If the requested token is not supported, returns –1.* Ex) getClassofVideoContent_Cost(0, "BTC") or getClassofVideoContent_Cost(0, "00000001") Ex) getClassofVideoContent_Cost(1, "USD") or getClassofVideoContent_Cost(1, "10010100")

4.5 APIs for IoMT storages

4.5.1 General

This subclause defines API classes of IoMT storages.

4.5.2 MStorage class

General

This subclause defines an MStorage classthat shall inherit the features of `MThing` class defined in ISO/IEC 23093-2.

APIs

Table 35 presents basic APIs of MStorage.

Table 35 — MStorage API

Nested Classes	
Modifier and Type	Method and Description
Constructor	
Constructor and Description	
MStorage()	
Default constructor.	

MStorage(String id)	
MStorage(String id, String serverIPAddress, integer serverPort)	
Fields	
Modifier and Type	Field and Description
Methods	
Modifier and Type	Method and Description
String	saveVideo(String videoURL)
	This function saves a received video source with a URL. The function returns a unique file ID.
String	saveImage(String imageURL)
	This function saves a received image source with a URL. The function returns a unique file ID.
String	saveAudio(String audioURL)
	This function saves a received audio source with a URL. The function returns a unique file ID.
String	saveAnalyzedDataURL(String analyzedDataURL)
	This function saves a received analyzed data source with a URL. The function returns a unique file ID.
String	saveAnalyzedData(AnalyzedDataType analyzedData)
	This function saves a received analyzed data source. The function returns a unique file ID.
String	getVideo(String fileID)
	This function returns a video source with its URL which matches with the specified file ID.

String	getImage(String fileID)
	This function returns an image source with its URL which matches with the specified file ID.
String	getAudio(String fileID)
	This function returns an audio source with its URL which matches with the specified file ID.
String	getAnalyzedDataURL(String fileID)
	This function returns a URL of analyzed data which matches with the specified file ID.
AnalyzedDataType	getAnalyzedData(String fileID)
	This function returns analyzed data following the data formats defined in this document which matches with the specified file ID.
int	updateVideo(String videoURL, String fileID)
	This function updates an existing video source with the specified file ID to the video source with its URL. The function returns 1, if the task succeeds, returns 0, otherwise.
int	updateImage(String imageURL, String fileID)
	This function updates an existing image source with the specified file ID to the image source with its URL. The function returns 1, if the task succeeds, returns 0, otherwise.
int	updateAudio(String audioURL, String fileID)
	This function updates an existing audio source with the specified file ID to the audio source with its URL. The function returns 1, if the task succeeds, returns 0, otherwise.
int	updateAnalyzedDataURL(String analyzedDataURL, String fileID)
	This function updates an existing analyzed data with the specified file ID to the analyzed data with its URL. The function returns 1, if the task succeeds, returns 0, otherwise.

int	updateAnalyzedData(AnalyzedDataType analyzedData, String fileID)
	This function updates an existing analyzed data with the specified file ID to the analyzed data following the data formats defined in this document. The function returns 1, if the task succeeds, returns 0, otherwise.
int	deleteFile(String fileID)
	This function deletes an existing file with the specified file ID. The function returns 1, if the task succeeds, returns 0, otherwise.

4.6 APIs for IoMT managers

4.6.1 General

This subclause defines API classes of IoMT managers.

4.6.2 MManager class

General

This subclause defines an MMnager class, which shall inherit the features of `MThing` class defined in ISO/IEC 23093-2.

APIs

Table 36 presents basic APIs of MMnager.

Table 36 — MMnager API

Nested Classes	
Modifier and Type	Method and Description
Constructor	
Constructor and Description	
MManager()	
Default constructor.	
MManager(String id)	

MManager(String id, String serverIPAddress, integer serverPort)	
Fields	
Modifier and Type	Field and Description
Methods	
Modifier and Type	Method and Description
List<MThingInfoType>	getMThingListByCapability(String MThingType, String capability)
	This function returns a list of MThings among the registered MThings in the MManager. The `MThingType` *and* `capability` *(i.e. formal parameters) can be described in string (e.g. term ID or binary representation) from the* `MThingTypeCS` *and* `capabilityCS` *defined in A.1 and A.2, respectively.*
CapabilityListType	getManagerCapabilityList();
	This function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type(e.g. XML, Binary) and a capability list specified in this document.
CapabilityListType	getAvailableManagerCapabilityList()
	This function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and an available capability list specified in this document.
CapabilityListType	getAppliedManagerCapabilityList()
	This function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and an applied capability list specified in this document.
int	registerMThing(MThing MThingInfo)
	This function registers an MThing to the MManager. The function returns 1, if the task succeeds, returns 0, otherwise.

int	unregisterMThing(MThing MThingInfo)
	This function unregisters an MThing from the MManager. The function returns 1, if the task succeeds, returns 0, otherwise.
List<MThingInfoType>	getRegisteredMThingList()
	This function returns a list of MThings registered in the MManager.

4.7 APIs for IoMT aggregators

4.7.1 General

This subclause defines API classes of IoMT aggregators.

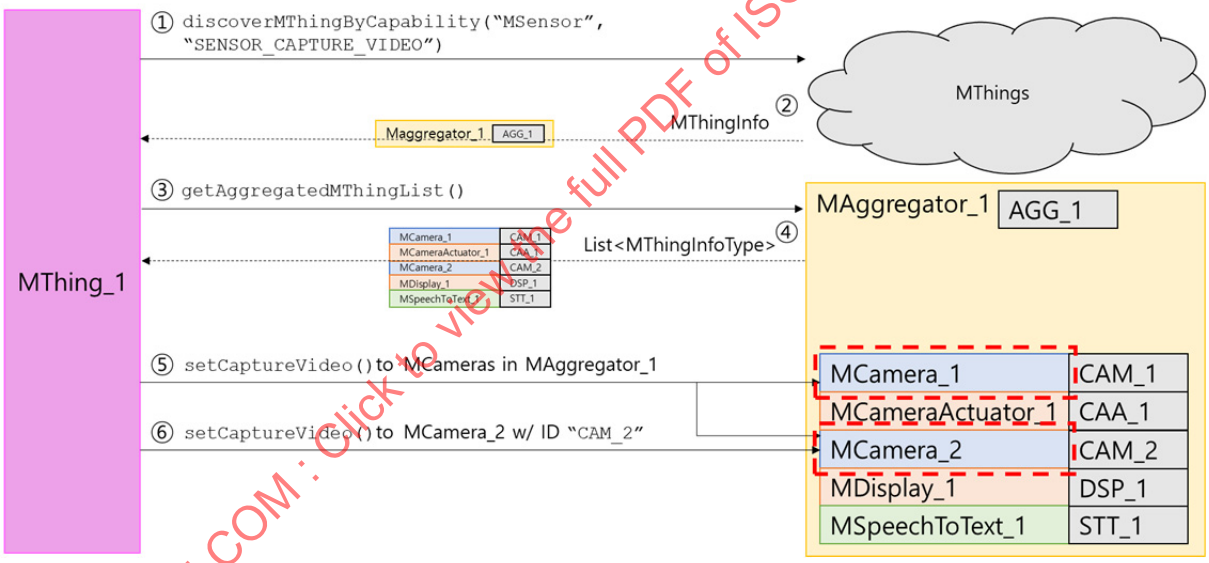


Figure 5 – Process of activating `setCaptureVideo()` to cameras in an MAggregator

Figure 5 shows a process of activating `setCaptureVideo()` to cameras in an MAggregator. An MThing (i.e. MThing_1) can search for MSensors with a specific capability (e.g. SENSOR_CAPTURE_VIDEO) with the function `discoverMThingByCapability()` (Figure 5, item 1). Suppose that the IoT network or server returns an MThing information of MAggregator_1 with its ID (Figure 5, item 2) as the search result. It can be possible because the MAggregator_1 possesses MSensors with the video capturing capability. The MThing_1 can ask a list of MThings that constitute the MAggregator_1 with the function `getAggregatedMThingList()` (Figure 5, item 3). The MAggregator_1 returns a list of aggregated MThing information (Figure 5, item 4) and the MThing_1 can send a command `setCaptureVideo()` to all cameras using the information received (Figure 5, item 5). The MThing_1 can also send a command `setCaptureVideo()` to a specific camera using its ID information (Figure 5, item 6).

Figure 6 and Figure 7 show the example codes of how to access all cameras in an MAggregator (Figure 5, item 5) and how to access a specific camera with ID in an MAggregator (Figure 5, item 6), respectively.

```
public static void main(String[] args) {
    // TODO Auto-generated method stub

        MAggregator mag01 = new MAggregator(connectedMAggregator);
        MCamera tempCamera = new MCamera();

        List<MThing> aggList = parseDataToList(mag01.getAggregatedMThingList());

        //command setCaptureVideo to MCameras
        for(int i=0; i<aggList.size(); i++){
            if(aggList.get(i) instanceof MCamera){
                tempCamera = (MCamera) aggList.get(i);
                tempCamera.setCaptureVideo();
            }
        }
}
```

Figure 6 — An example code of accessing all cameras in an MAggregator

```
public static void main(String[] args) {
    // TODO Auto-generated method stub

        MAggregator mag01 = new MAggregator(connectedMAggregator);
        MCamera tempCamera = new MCamera();

        List<MThing> aggList = parseDataToList(mag01.getAggregatedMThingList());

        //command setCaptureVideo to an MCamera with ID "CAM_2"
        for(int i=0; i<aggList.size(); i++){
            if(aggList.get(i).getIdRef().equals("CAM_2")){
                tempCamera = (MCamera) aggList.get(i);
                tempCamera.setCaptureVideo();
                break;
            }
        }
}
```

Figure 7 — An example code of accessing a specific camera in an MAggregator

4.7.2 MAggregator class

General

This subclause defines an MAggregator class, which shall inherit the features of `MThing` class defined in ISO/IEC 23093-2.

APIs

Table 37 presents basic APIs of MAggregator.

Table 37 — MAggregator API

Nested Classes	
Modifier and Type	Method and Description
Constructor	
Constructor and Description	
MAggregator()	
Default constructor.	
MAggregator(String id)	
MAggregator(String id, String serverIPAddress, integer serverPort)	
Fields	
Modifier and Type	Field and Description
Methods	
Modifier and Type	Method and Description
List<MThingInfoType>	getAggregatedMThingList()
	This function returns a list of MThings that consists of an MAggregator.
void	setAggregatedMThingList(List<MThingInfoType> aggregatedMThingList)
	This function sets an MAggregator with the designated list of MThings.
CapabilityListType	getAggregatorCapabilityList()
	This function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and a capability list specified in this document.

CapabilityListType	getAvailableAggregatorCapabilityList()
	This function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and an available capability list specified in this document.
CapabilityListType	getAppliedAggregatorCapabilityList()
	This function returns a class (i.e. Java or C++) or a structure (i.e. C) that shall include a returning type (e.g. XML, Binary) and an applied capability list specified in this document.

4.8 Return type class

4.8.1 General

This subclause specifies return class types of data exchanged between MThings. For an easy industrial adoption, the return type classes are provided in JAVA, C++, and C.

4.8.2 MPEGVCapabilityType

General

This subclause contains the capability description class types of MPEG-V sensors and actuators following ISO/IEC 23005-2 (MPEG-V Part 2) capability specification. Figure 8 shows how the one MThing (i.e. "MThing2") requests MPEG-V capability decription of another MThing (i.e. "MThing1") by invoking the `getMPEGVCapbility()` (Figure 8, items 1 and 2). The "MThing1" then sends back its MPEG-V capability description using the return class type (i.e. `MPEGVCapabilityType`) to the "MThing2" (Figure 8, item 7). Only steps 1, 2 and 7 are visible between MThings. Other steps are hidden processes performed inside the MThings.

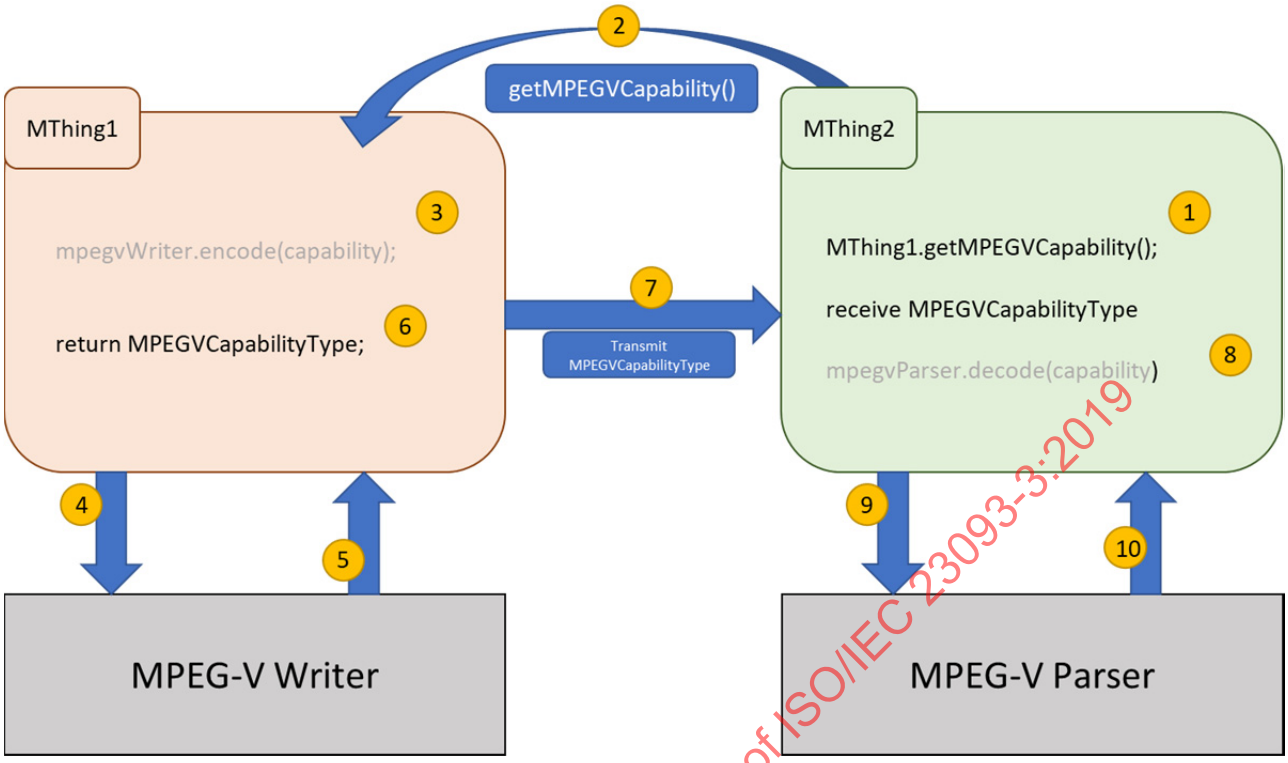


Figure 8 — Requesting a capability description of MThing described following ISO/IEC 23005-2 (MPEG-V Part 2) specification

JAVA class

Table 38 presents return class types of MPEG-V capablilty information in JAVA.

Table 38 — Return class types of MPEG-V capablilty information in JAVA

Constructor
Constructor and Description
MPEGVCapabilityType()
Default constructor.
MPEGVCapabilityType(int type, String mpegvData)

Fields	
Modifier and Type	Field and Description
int	type
	A returning type of capability metadata. *0 = None, 1 = XML, 2 = Binary, 3 = JSON*
String	mpegvData
	Capability informationthat shall follow the specifications of capabilities from ISO/IEC 23005-2 (MPEG-V Part 2) (i.e. sensor capability, device capability).
Methods	
Modifier and Type	Method and Description
int	getType()
void	setType(int type)
String	getMPEGVData()
void	setMPEGVData(String mpegvData)

C++ class

Table 39 presents return class types of MPEG-V capablilty information in C++.

Table 39 — Return class types of MPEG-V capablilty information in C++

Constructor and Destructor
Constructor and Destructor Description
MPEGVCapabilityType()
Default constructor.

MPEGVCapabilityType(int type, String mpegvData)	
~MPEGVCapabilityType()	
Default destructor.	
Member Data	
Modifier and Type	Member Data and Description
int	type
	A returning type of capability data. *0 = None, 1 = XML, 2 = Binary, 3 = JSON*
String	mpegvData
	Capability informationthat shall follow the specifications of capabilities from ISO/IEC 23005-2 (MPEG-V Part 2) (i.e. sensor capability, device capability).
Public Member Functions	
Modifier and Type	Public Member Functions and Description
int	getType()
void	setType(int type)
String	getMPEGVData()
void	setMPEGVData(String mpegvData)

C structure

Table 40 presents return class types of MPEG-V capablilty information in C.

Table 40 — Return class types of MPEG-V capablilty information in C

Data Field	
Modifier and Type	Data Field and Description
int	type
	A returning type of capability data. *0 = None, 1 = XML, 2 = Binary, 3 = JSON*
char*	mpegvData
	Capability informationthat shall follow the specifications of capabilities from ISO/IEC 23005-2 (MPEG-V Part 2) (i.e. sensor capability, device capability).

4.8.3 MPEGVSensedDataType

General

This subclause contains the sensed data description class types of MPEG-V sensors following ISO/IEC 23005-5 (MPEG-V Part 5) sensed data specification.

JAVA class

Table 41 presents return class types of MPEG-V sensed data in JAVA.

Table 41 — Return class types of MPEG-V sensed data in JAVA

Constructor
Constructor and Description
MPEGVSensedDataType()
Default constructor.
MPEGVSensedDataType(int type, String mpegvData)

Fields	
Modifier and Type	Field and Description
int	type
	A returning type of sensed data. *0 = None, 1 = XML, 2 = Binary, 3 = JSON*
String	mpegvData
	Sensed data of an IoMT sensor that shall follow the specifications of sensed data from ISO/IEC 23005-5 (MPEG-V Part 5).
Methods	
Modifier and Type	Method and Description
int	getType()
void	setType(int type)
String	getMPEGVData()
void	setMPEGVData(String mpegvData)

C++ class

Table 42 presents return class types of MPEG-V sensed data in C++.

Table 42 — Return class types of MPEG-V sensed data in C++

Constructor and Destructor
Constructor and Destructor Description
MPEGVSensedDataType()
Default constructor.

MPEGVSensedDataType(int type, String mpegvData)	
~MPEGVSensedDataType()	
Default destructor.	
Member Data	
Modifier and Type	Member Data and Description
int	type
	A returning type of sensed data. *0 = None, 1 = XML, 2 = Binary, 3 = JSON*
String	mpegvData
	Sensed data of an IoMT sensor that shall follow the specifications of sensed data from ISO/IEC 23005-5 (MPEG-V Part 5).
Public Member Functions	
Modifier and Type	Public Member Functions and Description
int	getType()
void	setType(int type)
String	getMPEGVData()
void	setMPEGVData(String mpegvData)

C structure

Table 43 presents return class types of MPEG-V sensed data in C.

Table 43 — Return class types of MPEG-V sensed data in C

Data Field	
Modifier and Type	Data Field and Description
int	type
	A returning type of sensed data. *0 = None, 1 = XML, 2 = Binary, 3 = JSON*
char*	mpegvData
	Sensed data of an IoMT sensor that shall follow the specifications of sensed data from ISO/IEC 23005-5 (MPEG-V Part 5).

4.8.4 MPEGVCommandType

General

This subclause contains the command description class types for MPEG-V actuators following ISO/IEC 23005-5 (MPEG-V Part 5) command specification.

JAVA class

Table 44 presents return class types of MPEG-V command in JAVA.

Table 44 — Return class types of MPEG-V command in JAVA

Constructor
Constructor and Description
MPEGVCommandType()
Default constructor.
MPEGVCommandType(int type, String mpegvData)

Fields	
Modifier and Type	Field and Description
int	type
	A returning type of sensed data. *0 = None, 1 = XML, 2 = Binary, 3 = JSON*
String	mpegvData
	Commandthat shall follow the specifications of device commands from ISO/IEC 23005-5 (MPEG-V Part 5).
Methods	
Modifier and Type	Method and Description
int	getType()
void	setType(int type)
String	getMPEGVData()
void	setMPEGVData(String mpegvData)

C++ class

Table 45 presents return class types of MPEG-V command in C++.

Table 45 — Return class types of MPEG-V command in C++

Constructor and Destructor
Constructor and Destructor Description
MPEGVCommandType()
Default constructor.

MPEGVCommandType(int type, String mpegvData)	
~MPEGVCommandType ()	
Default destructor.	
Member Data	
Modifier and Type	Member Data and Description
int	type
	A returning type of sensed data. *0 = None, 1 = XML, 2 = Binary, 3 = JSON*
String	mpegvData
	Commandthat shall follow the specifications of device commands from ISO/IEC 23005-5 (MPEG-V Part 5).
Public Member Functions	
Modifier and Type	Public Member Functions and Description
int	getType()
void	setType(int type)
String	getMPEGVData()
void	setMPEGVData(String mpegvData)

C structure

Table 46 presents return class types of MPEG-V command in C.

Table 46 — Return class types of MPEG-V command in C

Data Field	
Modifier and Type	Data Field and Description
int	type
	A returning type of sensed data. *0 = None, 1 = XML, 2 = Binary, 3 = JSON*
char*	mpegvData
	Commandthat shall follow the specifications of device commands from ISO/IEC 23005-5 (MPEG-V Part 5).

4.8.5 IoMT SensedDataType

General

This subclause contains the sensed data description class types of IoMT sensors following the data format specification in this document.

JAVA class

Table 47 presents return class types of IoMT sensed data in JAVA.

Table 47 — Return class types of IoMT sensed data in JAVA

Constructor
Constructor and Description
SensedDataType()
Default constructor.
SensedDataType(int type, String IoMTData)

Fields	
Modifier and Type	Field and Description
int	type
	A returning type of sensed data. *0 = None, 1 = XML, 2 = Binary, 3 = JSON*
String	IoMTData
	Sensed data of an IoMT sensor that shall follow the specifications of `MTDL` *and* `MSOV` *defined in this document.*
Methods	
Modifier and Type	Method and Description
int	getType()
void	setType(int type)
String	getIoMTData()
void	setIoMTData(String IoMTData)

C++ class

Table 48 presents return class types of IoMT sensed data in C++.

Table 48 — Return class types of IoMT sensed data in C++

Constructor and Destructor
Constructor and Destructor Description
SensedDataType()
Default constructor.

SensedDataType(int type, String IoMTData)	
~SensedDataType()	
Default destructor.	
Member Data	
Modifier and Type	Member Data and Description
int	type
	A returning type of sensed data.. *0 = None, 1 = XML, 2 = Binary, 3 = JSON*
String	IoMTData
	Sensed data of an IoMT sensor that shall follow the specifications of `MTDL` *and* `MSOV` *defined in this document.*
Public Member Functions	
Modifier and Type	Public Member Functions and Description
int	getType()
void	setType(int type)
String	getIoMTData()
void	setIoMTData(String IoMTData)

C structure

Table 49 presents return class types of IoMT sensed data in C.

Table 49 — Return class types of IoMT sensed data in C

Data Field	
Modifier and Type	Data Field and Description
int	type
	A returning type of sensed data. *0 = None, 1 = XML, 2 = Binary, 3 = JSON*
Char*	IoMTData
	Sensed data of an IoMT sensor that shall follow the specifications of `MTDL` *and* `MSOV` *defined in this document.*

4.8.6 IoMT ActuationDataType

General

This subclause contains the actuation data description class types of IoMT actuators following the data format specification in this document.

JAVA class

Table 50 presents return class types of IoMT actuation data in JAVA.

Table 50 — Return class types of IoMT actuation data in JAVA

Constructor
Constructor and Description
ActuationDataType()
Default constructor.
ActuationDataType(int type, String IoMTData)

Fields	
Modifier and Type	Field and Description
int	type
	A returning type of actuation data. *0 = None, 1 = XML, 2 = Binary, 3 = JSON*
String	IoMTData
	Actuation data from another MThings that shall follow the specifications of `MTDL` *and* `MACV` *defined in this document.*
Methods	
Modifier and Type	Method and Description
int	getType()
void	setType(int type)
String	getIoMTData()
void	setIoMTData(String IoMTData)

C++ class

Table 51 presents return class types of IoMT actuation data in C++.

Table 51 — Return class types of IoMT actuation data in C++

Constructor and Destructor
Constructor and Destructor Description
ActuationDataType()
Default constructor.

ActuationDataType(int type, String IoMTData)	
~ActuationDataType()	
Default destructor.	
Member Data	
Modifier and Type	Member Data and Description
int	type
	A returning type of actuation data. *0 = None, 1 = XML, 2 = Binary, 3 = JSON*
String	IoMTData
	Actuation data from another MThings that shall follow the specifications of `MTDL` *and* `MACV` *defined in this document.*
Public Member Functions	
Modifier and Type	Public Member Functions and Description
int	getType()
void	setType(int type)
String	getIoMTData()
void	setIoMTData(String IoMTData)

C structure

Table 52 presents return class types of IoMT actuation data in C.

Table 52 — Return class types of IoMT actuation data in C

Data Field	
Modifier and Type	Data Field and Description
int	type
	A returning type of actuation data. *0 = None, 1 = XML, 2 = Binary, 3 = JSON*
Char*	IoMTData
	Actuation data from another MThings that shall follow the specifications of `MTDL` *and* `MACV` *defined in this document.*

4.8.7 IoMT AnalyzedDataType

General

This subclause contains the analyzed data description class types of IoMT analyzers following the data format specification in this document.

JAVA class

Table 53 presents return class types of IoMT analyzed data in JAVA.

Table 53 — Return class types of IoMT analyzed data in JAVA

Constructor
Constructor and Description
AnalyzedDataType()
Default constructor.
AnalyzedDataType(int type, String IoMTData)

Fields	
Modifier and Type	Field and Description
int	type
	A returning type of analyzed data. *0 = None, 1 = XML, 2 = Binary, 3 = JSON*
String	IoMTData
	Analyzed data of an IoMT analyzer that shall follow the specifications of `MTDL` *and* `MAOV` *defined in this document.*
Methods	
Modifier and Type	Method and Description
int	getType()
void	setType(int type)
String	getIoMTData()
void	setIoMTData(String IoMTData)

C++ class

Table 54 presents return class types of IoMT analyzed data in C++.

Table 54 — Return class types of IoMT analyzed data in C++

Constructor and Destructor
Constructor and Destructor Description
AnalyzedDataType()
Default constructor.

AnalyzedDataType(int type, String IoMTData)	
~AnalyzedDataType()	
Default destructor.	
Member Data	
Modifier and Type	Member Data and Description
int	type
	A returning type of analyzed data. *0 = None, 1 = XML, 2 = Binary, 3 = JSON*
String	IoMTData
	Analyzed data of an IoMT analyzer that shall follow the specifications of `MTDL` *and* `MAOV` *defined in this document.*
Public Member Functions	
Modifier and Type	Public Member Functions and Description
int	getType()
void	setType(int type)
String	getIoMTData()
void	setIoMTData(String IoMTData)

C structure

Table 55 presents return class types of IoMT analyzed data in C.

Table 55 — Return class types of IoMT analyzed data in C

Data Field	
Modifier and Type	Data Field and Description
int	type
	A returning type of analyzed data. *0 = None, 1 = XML, 2 = Binary, 3 = JSON*
Char*	IoMTData
	Analyzed data of an IoMT analyzer that shall follow the specifications of `MTDL` *and* `MAOV` *defined in this document.*

4.8.8 IoMT CapabilityListType

General

This subclause contains the capability data description class types of MThings following the data format specification in this document.

JAVA class

Table 56 presents return class types of IoMT capability list data in JAVA.

Table 56 — Return class types of IoMT capability list data in JAVA

Constructor
Constructor and Description
CapabilityListType()
Default constructor.
CapabilityListType(int type, String IoMTData)

Fields	
Modifier and Type	Field and Description
int	type
	A returning type of MThing capability list data. *0 = None, 1 = XML, 2 = Binary, 3 = JSON*
String	IoMTData
	A capability list of an MThing.
Methods	
Modifier and Type	Method and Description
int	getType()
void	setType(int type)
String	getIoMTData()
void	setIoMTData(String IoMTData)

C++ class

Table 57 presents return class types of IoMT capability list data in C++.

Table 57 — Return class types of IoMT capability list data in C++

Constructor and Destructor
Constructor and Destructor Description
CapabilityListType()
Default constructor.

CapabilityListType(int type, String IoMTData)	
~CapabilityListType()	
Default destructor.	
Member Data	
Modifier and Type	Member Data and Description
int	type
	A returning type of MThing capability list data. *0 = None, 1 = XML, 2 = Binary, 3 = JSON*
String	IoMTData
	A capability list of an MThing.
Public Member Functions	
Modifier and Type	Public Member Functions and Description
int	getType()
Void	setType(int type)
String	getIoMTData()
Void	setIoMTData(String IoMTData)

C structure

Table 58 presents return class types of IoMT capability list data in C.

Table 58 — Return class types of IoMT capability list data in C

Data Field	
Modifier and Type	Data Field and Description
int	type
	A returning type of MThing capability list data. *0 = None, 1 = XML, 2 = Binary, 3 = JSON*
char*	IoMTData
	A capability list of an MThing.

4.8.9 IoMT MThingInfoType

General

This subclause contains the MThing information description class types of MThings following the data format specification in this document.

JAVA class

Table 59 presents return class types of MThing information in JAVA.

Table 59 — Return class types of MThing information in JAVA

Constructor
Constructor and Description
MThingInfoType()
Default constructor.
MThingInfoType(int type, String MThingInfo)

Fields	
Modifier and Type	Field and Description
int	type
	A returning type of MThing information. *0 = None, 1 = XML, 2 = Binary, 3 = JSON*
String	MThingInfo
	MThing information that shall be described by the data format of `MThingInfo` *specified in this document.*
Methods	
Modifier and Type	Method and Description
int	getType()
void	setType(int type)
String	getMThingInfo()
void	setMThingInfo(String MThingInfo)

C++ class

Table 60 presents return class types of MThing information in C++.

Table 60 — Return class types of MThing information in C++

Constructor and Destructor
Constructor and Destructor Description
MThingInfoType()
Default constructor.

MThingInfoType(int type, String MThingInfo)	
~ MThingInfoType()	
Default destructor.	
Member Data	
Modifier and Type	Member Data and Description
int	type
	A returning type of MThing information. *0 = None, 1 = XML, 2 = Binary, 3 = JSON*
String	MThingInfo
	MThing information that shall be described by the data format of `MThingInfo` *specified in this document.*
Public Member Functions	
Modifier and Type	Public Member Functions and Description
int	getType()
void	setType(int type)
String	getMThingInfo()
void	setMThingInfo(String MThingInfo)

C structure

Table 61 presents return class types of MThing information in C.

Table 61 — Return class types of MThing information in C

Data Field	
Modifier and Type	Data Field and Description
int	type
	A returning type of an MThing information. *0 = None, 1 = XML, 2 = Binary, 3 = JSON*
char*	MThingInfo
	MThing information that shall be described by the data format of `MThingInfo` *specified in this document.*

5 Media thing description language

5.1 General

This subclause describes a basic structure of the tools in this document in the form of media thing description language (MTDL) including the schema wrapper conventions, basic data types, root element, and top-level elements.

5.2 Schema wrapper

The syntax of description tools specified in this subclause is provided as a collection of schema components, consisting notably of type definitions and element declarations. In order to form a valid schema document, these schema components should be gathered in the same document with the following declaration defining in particular the target namespace and the namespaces prefixes.

```
<?xml version="1.0"?>
<schema
  xmlns="http://www.w3.org/2001/XMLSchema"
  xmlns:mtdl="urn:mpeg:mpeg-IoMT:2018:01-MTDL-NS"
  xmlns:mpeg7="urn:mpeg:mpeg7:schema:2004"
  xmlns:ct="urn:mpeg:mpeg-v:2018:01-CT-NS"
  targetNamespace="urn:mpeg:mpeg-IoMT:2018:01-MTDL-NS"
  elementFormDefault="qualified" attributeFormDefault="unqualified"
  version="ISO/IEC 23093-3" id="MPEG-IOMT-MTDL.xsd">

  <import namespace="urn:mpeg:mpeg-IoMT:2018:01-MTDV-NS"
schemalocation="http://standards.iso.org/ittf/PubliclyAvailableStandards/MPEG-
IOMT_schema_files/MPEG-IOMT-MTDV.xsd"/>
```

```
  <import namespace="urn:mpeg:mpeg7:schema:2004"
schemaLocation="http://standards.iso.org/ittf/PubliclyAvailableStandards/MPEG-
7_schema_files/mpeg7-v2.xsd"/>
  <import namespace="urn:mpeg:mpeg-v:2018:01-CT-NS"
schemaLocation="http://standards.iso.org/ittf/PubliclyAvailableStandards/MPEG-
V_schema_files/MPEG-V-CT.xsd"/>
```

Additionally, the following line should be appended to the resulting schema document in order to obtain a well-formed XML document.

```
</schema>
```

5.3 Base datatypes and elements

5.3.1 Syntax

```
<attributeGroup name="MThingBaseAttributes">
   <attribute name="id" type="string" use="optional"/>
   <attribute name="idRef" type="string" use="required"/>
   <attribute name="ipAddress" type="string" use="optional"/>
   <attribute name="url" type="anyURI" use="optional"/>
</attributeGroup>

<element name="MSensor" type="mtdl:MSensorType"/>
<element name="MActuator" type="mtdl:MActuatorType"/>
<element name="MAnalyzer" type="mtdl:MAnalyzerType"/>
<element name="MStorage" type="mtdl:MStorageType"/>
<element name="MManager" type="mtdl:MManagerType"/>
<element name="MAggregator" type="mtdl:MAggregatorType"/>

<complexType name="SupportedTokenListType">
      <sequence minOccurs="1" maxOccurs="1">
               <element name="cryptocurrencyList"
type="mtdl:cryptocurrencyListType" minOccurs="0" maxOccurs="1"/>
               <element name="legalTenderList" type="mtdl:LegalTenderListType"
minOccurs="0" maxOccurs="1"/>
      </sequence>
</complexType>

<complexType name="cryptocurrencyListType">
   <sequence>
      <element name="cryptocurrency" type="mtdl:cryptocurrencyType"
minOccurs="0" maxOccurs="unbounded"/>
   </sequence>
</complexType>

<complexType name="cryptocurrencyType">
   <attribute name="cryptocurrencyName" type="mpeg7:termReferenceType"
use="required"/>
   <attribute name="cryptocurrencyWalletAddress" type="string" use="required"/>
</complexType>

<complexType name="LegalTenderListType">
   <sequence>
      <element name="legalTender" type="mtdl:LegalTenderType" minOccurs="0"
maxOccurs="unbounded"/>
```

```
    </sequence>
</complexType>

<complexType name="LegalTenderType">
    <attribute name="legalTenderName" type="mpeg7:termReferenceType"
use="required"/>
    <attribute name="legalTenderWalletAddress" type="string" use="required"/>
</complexType>
```

5.3.2 Semantics

Table 62 — Semantics of the `MThingBaseAttributes`

Name	*Definition*
`idFlag`	This field, which is only present in the binary representation, indicates the presence of `id` attribute. If it is set to "1", the `id` attribute is following.
`id`	Describes the unique identifier of a data instance.
`idRef`	Describes the unique identifier of an MThing.
`ipAddressFlag`	This field, which is only present in the binary representation, indicates the presence of `ipAddress` attribute. If it is set to "1", the `ipAddress` attribute is following.
`ipAddress`	Describes the IP address of an MThing.
`urlFlag`	This field, which is only present in the binary representation, indicates the presence of `url` attribute. If it is set to "1", the `url` attribute is following.
`url`	Describes a URL address of an MThing.

Table 63 — Semantics of the MThing base elements

Name	*Definition*
`MSensor`	Describes an MSensor.
`MActuator`	Describes an MActuator.
`MAnalyzer`	Describes an MAnalyzer.
`MStorage`	Describes an MStorage.
`MManager`	Describes an MManager.
`MAggregator`	Describes an MAggregator.

Table 64 — Semantics of the MThingBaseTypes

Name	*Definition*
SupportedTokenListType	Tool for describing a media token (MToken) list which is supported by an MThing.
cryptocurrencyList	Describes a list of cryptocurrencies that an MThing supports.
legalTenderList	Describes a list of legal tenders that an MThing supports.
cryptocurrencyListType	Tool for describing a cryptocurrency list which is supported by an MThing.
cryptocurrency	Describes the cryptocurrency which is used for a transaction.
cryptocurrencyType	Tool for describing a cryptocurrency.
cryptocurrencyName	Describes the alphabetic code of the cryptocurrency.
cryptocurrencyWalletAddress	Describes a wallet address of the cryptocurrency.
LegalTenderListType	Tool for describing a list of legal tenders.
legalTender	Describes a legal tender which is used for a transaction.
LegalTenderType	Tool for describing a legal tender.
legalTenderName	Describes the alphabetic code of the legal tender.
legalTenderWalletAddress	Describes a wallet address of the legal tender.

5.4 Root element

5.4.1 Syntax

```
<element name="MThingInfo">
   <complexType>
      <sequence>
         <element name="TimeStamp" type="mpegvct:AbsoluteTimeType"
minOccurs="0"/>
         <element name="supportedTokenList" type="mtdl:SupportedTokenListType"
minOccurs="0" maxOccurs="1"/>
         <choice>
            <element ref="mtdl:MSensor"/>
            <element ref="mtdl:MActuator"/>
            <element ref="mtdl:MAnalyzer"/>
            <element ref="mtdl:MStorage"/>
            <element ref="mtdl:MManager"/>
            <element ref="mtdl:MAggregator"/>
         </choice>
      </sequence>
      <attributeGroup ref="mtdl:MThingBaseAttributes"/>
   </complexType>
</element>
```

5.4.2 Semantics

Table 65 — Semantics of the `MThingInfo` type

Name	*Definition*
`MThingInfo`	Serves as the root element for MThing metadata.
`timeStamp`	Provides the absolute time information of MThing metadata as defined in ISO/IEC 23005-6.
`supportedTokenList`	Describes a list of tokens that an MThing supports.

5.5 Media sensor description language

5.5.1 General

This subclause specifies tools for describing media sensors. The following subclause defines a complex type of `MSensorType`, which contains the MSensor capability description along with an abstract type of `SensedDataBaseType`, which the sensed data of an individual MSensor should inherit.

5.5.2 Syntax

```
<!-- ################################################ -->
<!--  MSensor Base Type                               -->
<!-- ################################################ -->
<complexType name="MSensorType">
    <sequence>
        <element name="sensorCapabilityList" type="mtdl:SensorCapabilityListType"
minOccurs="0"/>
        <element name="sensedData" type="mtdl:SensedDataBaseType" minOccurs="0"/>
    </sequence>
</complexType>

<complexType name="SensorCapabilityListType">
    <sequence>
        <element name="sensorCapability" type="mtdl:SensorCapabilityType"
maxOccurs="unbounded"/>
    </sequence>
    <attribute name="listType" type="mtdl:sensorCapabilityListEnumType"/>
</complexType>

<simpleType name="sensorCapabilityListEnumType">
    <restriction base="string">
        <enumeration value="allSensorCapabilityList"/>
        <enumeration value="availableSensorCapabilityList"/>
        <enumeration value="appliedSensorCapabilityList"/>
    </restriction>
</simpleType>

<complexType name="SensorCapabilityType">
    <sequence>
        <element name="sensorCapabilityParameter"
type="mtdl:SensorCapabilityParameterType" minOccurs="0" maxOccurs="unbounded"/>
    </sequence>
```

```
    <attribute name="sensorCapabilityName" type="mpeg7:termReferenceType"
use="required"/>
</complexType>

<complexType name="SensorCapabilityParameterType">
    <attribute name="sensorCapabilityParameterName"
type="mpeg7:termReferenceType" use="required"/>
</complexType>

<complexType name="SensedDataBaseType" abstract="true"/>
```

5.5.3 Semantics

Table 66 — Semantics of the `MSensor` tbase type

Name	*Definition*
`sensorCapabilityList`	List of the MSensor's capabilities.
`sensedData`	Sensed Data by an MSensor.
`SensorCapabilityListType`	Tool for describing sensor capability.
`sensorCapability`	List of the MSensor's entire capabilities.
`listType`	Tool for specify the type of sensor capabilities among `allSensorCapabilityList`, `availableSensorCapabilityList`, and `appliedSensorCapabilityList`.
`sensorCapabilityListEnumType`	Enumeration list which can be one of the capability type.
`SensorCapabilityType`	Tool for describing capabilities.
`sensorCapabilityParameter`	List of the MSensor's capability parameters.
`sensorCapabilityName`	Name of a capability of MSensor. The type of the capability shall be described using the `mpeg7:termReferenceType` defined in ISO/IEC 15938-5:2003, 7.6. A classification scheme that may be used for this purpose is the `SensorCapabilityCS` defined in A.2.1.
`SensorCapabilityParameterType`	Tool for describing capability parameters.
`sensorCapabilityParameterName`	Name of a capability parameter. The type of the capability parameter shall be described using the `mpeg7:termReferenceType` defined in ISO/IEC 15938-5:2003, 7.6. A classification scheme that may be used for this purpose is the `SensorCapabilityParameterCS` defined in A.3.1.
`SensedDataBaseType`	Tool for describing a sensed data.

5.5.4 Example

This example shows a MSensor with a data instance ID of "MS001-0012" and a device ID of "MS001". This MSensor has capabilities of "SENSOR_CAPTURE_VIDEO" and "SENSOR_CAPTURE_AUDIO".

```
<mtdl:MThingInfo id="MS001-0012" idRef="MS001">
    <mtdl:MSensor>
        <mtdl:sensorCapabilityList listType="allSensorCapabilityList">
            <mtdl:sensorCapability sensorCapabilityName="SENSOR_CAPTURE_VIDEO"/>
            <mtdl:sensorCapability sensorCapabilityName="SENSOR_CAPTURE_AUDIO"/>
        </mtdl:sensorCapabilityList>
    </mtdl:MSensor>
</mtdl:MThingInfo>
```

5.6 Media actuator description language

5.6.1 General

This subclause specifies tools for describing media actuators. The following subclause defines a complex type of MActuatorType, which contains the MActuator capability description along with an abstract type of ActuationDataBaseType, which the actuation commands of an individual MActuator should inherit.

5.6.2 Syntax

```
<!-- ################################################ -->
<!--  MActuator Base Type                             -->
<!-- ################################################ -->
<complexType name="MActuatorType">
    <sequence>
        <element name="actuatorCapabilityList"
type="IoMT:ActuatorCapabilityListType" minOccurs="0"/>
        <element name="actuationData" type="mtdl:ActuationDataBaseType"
minOccurs="0"/>
    </sequence>
</complexType>

<complexType name="ActuatorCapabilityListType">
    <sequence>
        <element name="actuatorCapability" type="mtdl:ActuatorCapabilityType"
maxOccurs="unbounded"/>
    </sequence>
    <attribute name="listType" type="mtdl:ActuatorCapabilityListEnumType"/>
</complexType>

<simpleType name="ActuatorCapabilityListEnumType">
    <restriction base="string">
        <enumeration value="allActuatorCapabilityList"/>
        <enumeration value="availableActuatorCapabilityList"/>
        <enumeration value="appliedActuatorCapabilityList"/>
    </restriction>
</simpleType>

<complexType name="ActuatorCapabilityType">
    <sequence>
        <element name="actuatorCapabilityParameter"
type="mtdl:ActuatorCapabilityParameterType" minOccurs="0" maxOccurs="unbounded"/>
```

```
    </sequence>
    <attribute name="actuatorCapabilityName" type="mpeg7:termReferenceType"
use="required"/>
</complexType>

<complexType name="ActuatorCapabilityParameterType">
    <attribute name="actuatorCapabilityParameterName"
type="mpeg7:termReferenceType" use="required"/>
</complexType>

<complexType name="ActuationDataBaseType" abstract="true"/>
```

5.6.3 Semantics

Table 67 — Semantics of the `MActuator` base type

Name	*Definition*
`actuatorCapabilityList`	List of the MActuator's capabilities.
`actuationData`	Actuation data from other MThings.
`ActuatorCapabilityListType`	Tool for describing actuator capability.
`actuatorCapability`	List of the MActuator's entire capabilities.
`listType`	Tool for specify the type of actuator capabilities among `allActuatorCapabilityList`, `availableActuatorCapabilityList`, and `appliedActuatorCapabilityList`.
`actuatorCapabilityListEnumType`	Enumeration list which can be one of the capability type.
`ActuatorCapabilityType`	Tool for describing capabilities.
`actuatorCapabilityParameter`	List of the MActuator's capability parameters.
`actuatorCapabilityName`	Name of a capability of MActuator. The type of the capability shall be described using the `mpeg7:termReferenceType` defined in ISO/IEC 15938-5:2003, 7.6. A classification scheme that may be used for this purpose is the `ActuatorCapabilityCS` defined in A.2.2.
`ActuatorCapabilityParameterType`	Tool for describing capability parameters.
`actuatorCapabilityParameterName`	Name of a capability parameter. The type of the capability parameter shall be described using the `mpeg7:termReferenceType` defined in ISO/IEC 15938-5:2003, 7.6. A classification scheme that may be used for this purpose is the `ActuatorCapabilityParameterCS` defined in A.3.2.
`ActuationDataBaseType`	Tool for describing a actuation data.

5.6.4 Example

This example shows a MActuator with a data instance ID of "MAC003-0001" and a device ID of "MAC003". This MActuator has capabilities of "ACTUATOR_SPRAY_WATER", "ACTUATOR_SPRAY_FOG", and "ACTUATOR_SPRAY_BUBBLE".

```
<mtdl:MThingInfo id="MAC003-0001" idRef="MAC003">
    <mtdl:MActuator>
        <mtdl:actuatorCapabilityList listType="allActuatorCapabilityList">
            <mtdl:actuatorCapability
actuatorCapabilityName="ACTUATOR_SPRAY_WATER"/>
            <mtdl:actuatorCapability actuatorCapabilityName="ACTUATOR_SPRAY_FOG"/>
            <mtdl:actuatorCapability
actuatorCapabilityName="ACTUATOR_SPRAY_BUBBLE"/>
        </mtdl:actuatorCapabilityList>
    </mtdl:MActuator>
</mtdl:MThingInfo>
```

5.7 Media analyzer description language

5.7.1 General

This subclause specifies tools for describing media analyzer. The following subclause defines a complex type of MAnalyzerType, which contains the MAnalyzer capability description along with an abstract type of AnalyzedDataBaseType, which the analyzed data of an individual MAnalyzer should inherit.

5.7.2 Syntax

```
<!-- ################################################ -->
<!--  MAnalyzer Base Type                             -->
<!-- ################################################ -->
<complexType name="MAnalyzerType">
    <sequence>
        <element name="analyzerCapabilityList"
type="IoMT:AnalyzerCapabilityListType" minOccurs="0"/>
        <element name="analyzedData" type="mtdl:AnalyzedDataBaseType"
minOccurs="0"/>
    </sequence>
</complexType>

<complexType name="AnalyzerCapabilityListType">
    <sequence>
        <element name="analyzerCapability" type="mtdl:AnalyzerCapabilityType"
maxOccurs="unbounded"/>
    </sequence>
    <attribute name="listType" type="mtdl:AnalyzerCapabilityListEnumType"/>
</complexType>

<simpleType name="AnalyzerCapabilityListEnumType">
    <restriction base="string">
        <enumeration value="allAnalyzerCapabilityList"/>
        <enumeration value="availableAnalyzerCapabilityList"/>
        <enumeration value="appliedAnalyzerCapabilityList"/>
    </restriction>
</simpleType>
```

```
<complexType name="AnalyzerCapabilityType">
   <sequence>
      <element name="analyzerCapabilityParameter"
type="mtdl:AnalyzerCapabilityParameterType" minOccurs="0" maxOccurs="unbounded"/>
   </sequence>
   <attribute name="analyzerCapabilityName" type="mpeg7:termReferenceType"
use="required"/>
</complexType>

<complexType name="AnalyzerCapabilityParameterType">
   <attribute name="analyzerCapabilityParameterName"
type="mpeg7:termReferenceType" use="required"/>
</complexType>

<complexType name="AnalyzedDataBaseType" abstract="true"/>
```

5.7.3 Semantics

Table 68 — Semantics of the `MAnalyzer` base type

Name	*Definition*
`analyzerCapabilityList`	List of the MAnalyzer's capabilities.
`analyzedData`	Sensed Data by an MAnalyzer.
`AnalyzerCapabilityListType`	Tool for describing analyzer capability.
`analyzerCapability`	List of the MAnalyzer's entire capabilities.
`listType`	Tool for specify the type of analyzer capabilities among `allAnalyzerCapabilityList`, `availableAnalyzerCapabilityList`, and `appliedAnalyzerCapabilityList`.
`analyzerCapabilityListEnumType`	Enumeration list which can be one of the capability type.
`AnalyzerCapabilityType`	Tool for describing capabilities.
`analyzerCapabilityParameter`	List of the MAnalyzer's capability parameters.
`analyzerCapabilityName`	Name of a capability of MAnalyzer. The type of the capability shall be described using the `mpeg7:termReferenceType` defined in ISO/IEC 15938-5:2003, 7.6. A classification scheme that may be used for this purpose is the `AnalyzerCapabilityCS` defined in A.2.3.
`AnalyzerCapabilityParameterType`	Tool for describing capability parameters.

Name	*Definition*
analyzerCapabilityParameterName	Name of a capability parameter. The type of the capability parameter shall be described using the mpeg7:termReferenceType defined in ISO/IEC 15938-5:2003, 7.6. A classification scheme that may be used for this purpose is the AnalyzerCapabilityParameterCS defined in A.3.3.
AnalyzedDataBaseType	Tool for describing a analyzed data.

5.7.4 Example

This example shows a MAnalyzer with a data instance ID of "MAZ005-0001" and a device ID of "MAZ005". This MAnalyzer has a capability of "ANALYZER_GUIDE_DIRECTION".

```
<mtdl:MThingInfo id="MAZ005-0001" idRef="MAZ005">
    <mtdl:MAnalyzer>
        <mtdl:analyzerCapabilityList listType="allAnalyzerCapabilityList">
            <mtdl:analyzerCapability
analyzerCapabilityName="ANALYZER_GUIDE_DIRECTION"/>
        </mtdl:analyzerCapabilityList>
    </mtdl:MAnalyzer>
</mtdl:MThingInfo>
```

5.8 Media storage description language

5.8.1 General

This subclause specifies tools for describing media storages. The following subclause defines a complex type of MStorageType, which contains the MStorage capability description along with an abstract type of StorageDataBaseType, which the storage commands of an individual MStorage should inherit.

5.8.2 Syntax

```
<!-- ################################################ -->
<!--  MStorage Base Type                              -->
<!-- ################################################ -->
<complexType name="MStorageType">
    <sequence>
        <element name="storageCapabilityList"
type="IoMT:StorageCapabilityListType" minOccurs="0"/>
        <element name="storageCommand" type="mtdl:StorageCommandBaseType"
minOccurs="0"/>
    </sequence>
</complexType>

<complexType name="StorageCapabilityListType">
    <sequence>
        <element name="storageCapability" type="mtdl:StorageCapabilityType"
maxOccurs="unbounded"/>
    </sequence>
    <attribute name="listType" type="mtdl:StorageCapabilityListEnumType"/>
</complexType>
```

```
<simpleType name="StorageCapabilityListEnumType">
    <restriction base="string">
        <enumeration value="allStorageCapabilityList"/>
        <enumeration value="availableStorageCapabilityList"/>
        <enumeration value="appliedStorageCapabilityList"/>
    </restriction>
</simpleType>

<complexType name="StorageCapabilityType">
    <sequence>
        <element name="storageCapabilityParameter"
type="mtdl:StorageCapabilityParameterType" minOccurs="0" maxOccurs="unbounded"/>
    </sequence>
    <attribute name="storageCapabilityName" type="mpeg7:termReferenceType"
use="required"/>
</complexType>

<complexType name="StorageCapabilityParameterType">
    <attribute name="storageCapabilityParameterName"
type="mpeg7:termReferenceType" use="required"/>
</complexType>

<complexType name="StorageCommandBaseType" abstract="true"/>
```

5.8.3 Semantics

Table 69 — Semantics of the `MStorage` base type

Name	*Definition*
`storageCapabilityList`	List of the MStorage's capabilities.
`StorageCapabilityListType`	Tool for describing storage capability.
`storageCommand`	Command to control the storage.
`storageCapability`	List of the MStorage's entire capabilities.
`listType`	Tool for specify the type of storage capabilities among `allStorageCapabilityList`, `availableStorageCapabilityList`, and `appliedStorageCapabilityList`.
`storageCapabilityListEnumType`	Enumeration list which can be one of the capability type.
`StorageCapabilityType`	Tool for describing capabilities.
`storageCapabilityParameter`	List of the MStorage's capability parameters.
`storageCapabilityName`	Name of a capability of MStorage. The type of the capability shall be described using the `mpeg7:termReferenceType` defined in ISO/IEC 15938-5:2003, 7.6. A classification scheme that may be used for this purpose is the `StorageCapabilityCS` defined in A.2.4.

Name	*Definition*
StorageCapabilityParameterType	Tool for describing storage capability parameters.
storageCapabilityParameterName	Name of a capability parameter. The type of the capability parameter shall be described using the mpeg7:termReferenceType defined in ISO/IEC 15938-5:2003, 7.6. A classification scheme that may be used for this purpose is the StorageCapabilityParameterCS defined in A.3.3.
StorageCommandBaseType	Tool for describing storage control commands.

5.8.4 Example

This example shows an MStorage with a data instance ID of "MST006-0001" and a device ID of "MST006". This MStorage has capabilities of "STORAGE_SAVE", "STORAGE_READ", "STORAGE_DELETE", and "STORAGE_UPDATE".

```
<mtdl:MThingInfo id="MST006-0001" idRef="MST006">
   <mtdl:MStorage>
      <mtdl:storageCapabilityList listType="allStorageCapabilityList">
         <mtdl:storageCapability storageCapabilityName="STORAGE_SAVE"/>
         <mtdl:storageCapability storageCapabilityName="STORAGE_READ"/>
         <mtdl:storageCapability storageCapabilityName="STORAGE_DELETE"/>
         <mtdl:storageCapability storageCapabilityName="STORAGE_UPDATE"/>
      </mtdl:storageCapabilityList>
   </mtdl:MStorage>
</mtdl:MThingInfo>
```

5.9 Media manager description language

5.9.1 General

This subclause specifies tools for describing media thing managers, which can act as an IoMT server. The following subclause defines a complex type of MManagerType, which contains the MManager capability description along with an abstract type of ManagerDataBaseType, which the server processing data of an individual MManager should inherit.

5.9.2 Syntax

```
<!-- ################################################ -->
<!--  MManager Base Type                              -->
<!-- ################################################ -->
<complexType name="MManagerType">
   <sequence>
      <element name="managerCapabilityList"
type="IoMT:ManagerCapabilityListType" minOccurs="0"/>
      <element name="managementData" type="mtdl:ManagerDataBaseType"
minOccurs="0"/>
   </sequence>
</complexType>
```

```
<complexType name="ManagerCapabilityListType">
    <sequence>
        <element name="managerCapability" type="mtdl:ManagerCapabilityType"
maxOccurs="unbounded"/>
    </sequence>
    <attribute name="listType" type="mtdl:ManagerCapabilityListEnumType"/>
</complexType>

<simpleType name="ManagerCapabilityListEnumType">
    <restriction base="string">
        <enumeration value="allManagerCapabilityList"/>
        <enumeration value="availableManagerCapabilityList"/>
        <enumeration value="appliedManagerCapabilityList"/>
    </restriction>
</simpleType>

<complexType name="ManagerCapabilityType">
    <sequence>
        <element name="managerCapabilityParameter"
type="mtdl:ManagerCapabilityParameterType" minOccurs="0" maxOccurs="unbounded"/>
    </sequence>
    <attribute name="managerCapabilityName" type="mpeg7:termReferenceType"
use="required"/>
</complexType>

<complexType name="ManagerCapabilityParameterType">
    <attribute name="managerCapabilityParameterName"
type="mpeg7:termReferenceType" use="required"/>
</complexType>

<complexType name="ManagerDataBaseType" abstract="true"/>
```

5.9.3 Semantics

Table 70 — Semantics of the `MManager` base type

Name	*Definition*
`managerCapabilityList`	List of the MManager's capabilities.
`ManagerCapabilityListType`	Tool for describing manager capability.
`managementData`	Management Data by an MManager.
`managerCapability`	List of the MManager's entire capabilities.
`listType`	Tool for specify the type of manager capabilities among `allManagerCapabilityList`, `availableManagerCapabilityList`, and `appliedManagerCapabilityList`.
`managerCapabilityListEnumType`	Enumeration list which can be one of the capability type.
`ManagerCapabilityType`	Tool for describing capabilities.

Name	*Definition*
managerrCapabilityParameter	List of the MManager's capability parameters.
managerCapabilityName	Name of a capability of MManager. The type of the capability shall be described using the mpeg7:termReferenceType defined in ISO/IEC 15938-5:2003, 7.6. A classification scheme that may be used for this purpose is the ManagerCapabilityCS defined in A.2.4.
ManagerCapabilityParameterType	Tool for describing capability parameters.
managerCapabilityParameterName	Name of a capability parameter. The type of the capability parameter shall be described using the mpeg7:termReferenceType defined in ISO/IEC 15938-5:2003, 7.6. A classification scheme that may be used for this purpose is the ManagerCapabilityParameterCS defined in A.3.5.
ManagementDataBaseType	Tool for describing a management data.

5.9.4 Example

This example shows an MManager with a data instance ID of "MMN006-0001" and a device ID of "MMN006". This MManager has capabilities of "MANAGER_RETRIVE_MTHINGS_BY_CAPABILITY", "MANAGER_REGISTER_MTHING", and "MANAGER_UNREGISTER_MTHING".

```
<mtdl:MThingInfo id="MMN006-0001" idRef="MMN006">
   <mtdl:MManager>
      <mtdl:managerCapabilityList listType="allManagerCapabilityList">
         <mtdl:managerCapability
managerCapabilityName="MANAGER_RETRIVE_MTHINGS_BY_CAPABILITY"/>
         <mtdl:managerCapability
managerCapabilityName="MANAGER_REGISTER_MTHING"/>
         <mtdl:managerCapability
managerCapabilityName="MANAGER_UNREGISTER_MTHING"/>
      </mtdl:managerCapabilityList>
   </mtdl:MManager>
</mtdl:MThingInfo>
```

5.10 Media aggregator description language

5.10.1 General

This subclause specifies tools for describing a media thing aggregator, which is composed of multiple indivisual MThings. The following subclause defines a complex type of MAggregatorType, which contains the MAggregator capability description along with the its group of MThings inside.

5.10.2 Syntax

```
<!-- ################################################ -->
<!--  MAggregator Base Type                           -->
<!-- ################################################ -->
<complexType name="MAggregatorType">
   <sequence>
      <element name="aggregatorCapabilityList"
type="IoMT:AggregatorCapabilityListType" minOccurs="0"/>
      <element name="aggregatedMThingList" type="mtdl:AggregatedMThingListType"
minOccurs="0"/>
   </sequence>
</complexType>

<complexType name="AggregatorCapabilityListType">
   <sequence>
      <element name="aggregatorCapability" type="mtdl:AggregatorCapabilityType"
maxOccurs="unbounded"/>
   </sequence>
   <attribute name="listType" type="mtdl:AggregatorCapabilityListEnumType"/>
</complexType>

<simpleType name="AggregatorCapabilityListEnumType">
   <restriction base="string">
      <enumeration value="allAggregatorCapabilityList"/>
      <enumeration value="availableAggregatorCapabilityList"/>
      <enumeration value="appliedAggregatorCapabilityList"/>
   </restriction>
</simpleType>

<complexType name="AggregatorCapabilityType">
   <sequence>
      <element name="aggregatorCapabilityParameter"
type="mtdl:AggregatorCapabilityParameterType" minOccurs="0"
maxOccurs="unbounded"/>
   </sequence>
   <attribute name="aggregatorCapabilityName" type="mpeg7:termReferenceType"
use="required"/>
</complexType>

<complexType name="AggregatorCapabilityParameterType">
   <attribute name="aggregatorCapabilityParameterName"
type="mpeg7:termReferenceType" use="required"/>
</complexType>

<complexType name="AggregatedMThingListType">
   <sequence>
      <element name="MSensor" type="mtdl:MSensorType" minOccurs="0"
maxOccurs="unbounded"/>
      <element name="MActuator" type="mtdl:MActuatorType" minOccurs="0"
maxOccurs="unbounded"/>
      <element name="MAnalyzer" type="mtdl:MAnalyzerType" minOccurs="0"
maxOccurs="unbounded"/>
      <element name="MStorage" type="mtdl:MStorageType" minOccurs="0"
maxOccurs="unbounded"/>
      <element name="MManager" type="mtdl:MManagerType" minOccurs="0"
maxOccurs="unbounded"/>
   </sequence>
</complexType>
```

5.10.3 Semantics

Table 71 — Semantics of the `MAggregator` base type

Name	*Definition*
`aggregatorCapabilityList`	List of the MAggregator's capabilities.
`aggregatedMThingList`	List of aggregated MThing list.
`AggregatorCapabilityListType`	Tool for describing aggregator capability.
`aggregatorCapability`	List of the MAggregator's entire capabilities.
`listType`	Tool for specify the type of aggregator capabilities among `allAggregatorCapabilityList`, `availableAggregatorCapabilityList`, and `appliedAggregatorCapabilityList`.
`aggregatorCapabilityListEnumType`	Enumeration list which can be one of the capability type.
`AggregatorCapabilityType`	Tool for describing capabilities.
`aggregatorCapabilityParameter`	List of the MAggregator's capability parameters.
`aggregatorCapabilityName`	Name of a capability of MManager. The type of the capability shall be described using the `mpeg7:termReferenceType` defined in ISO/IEC 15938-5:2003, 7.6. A classification scheme that may be used for this purpose is the `AggregatorCapabilityCS` defined in A.2.6.
`AggregatorCapabilityParameterType`	Tool for describing capability parameters.
`aggregatorCapabilityParameterName`	Name of a capability parameter. The type of the capability parameter shall be described using the `mpeg7:termReferenceType` defined in ISO/IEC 15938-5:2003, 7.6. A classification scheme that may be used for this purpose is the `AggregatorCapabilityParameterCS` defined in A.3.6.
`AggregatedMThingListType`	Tool for describing an aggregated MThing list.

5.10.4 Example

This example shows a `MAggregator` with a data instance ID of "MAG007-0001" and a device ID of "MAG007". This `MAggregator` has a `MCamera` and a `MCameraActuator` in it. The `MCamera` has a capability of "`SENSOR_CAPTURE_VIDEO`" representing that the corresponding `MCamera` is able to capture a video. The `MCameraActuator` has a capability of "`ACTUATOR_CHANGE_RESOLUTION`" representing that the corresponding `MCameraActuator` is able to change the camera resolution.

```
<mtdl:MThingInfo id="MAG007-0001" idRef="MAG007">
   <mtdl:MAggregator>
      <mtdl:aggregatedMThingList>
```

```
          <mtdl:MSensor>
             <mtdl:sensorCapabilityList>
                <mtdl:sensorCapability
sensorCapabilityName="SENSOR_CAPTURE_VIDEO"/>
             </mtdl:sensorCapabilityList>
          </mtdl:MSensor>
          <mtdl:MActuator>
             <mtdl:actuatorCapabilityList>
                <mtdl:actuatorCapability
actuatorCapabilityName="ACTUATOR_CHANGE_RESOLUTION"/>
             </mtdl:actuatorCapabilityList>
          </mtdl:MActuator>
       </mtdl:aggregatedMThingList>
    </mtdl:MAggregator>
</mtdl:MThingInfo>
```

Another example below shows the capabilities of smart glasses that are a typical example of an MAggregator. It is assumed that the smart glasses in the example contain several sensors, actuators, analyzers, and storages. As shown in the example, the available capabilitites of MSensors contained in the smart glasses are `SENSOR_CAPTURE_VIDEO`, `SENSOR_CAPTURE_AUDIO`, `SENSOR_STREAM_VIDEO`, `SENSOR_CAPTURE_STEREO_VIDEO`, and `SENSOR_STREAM_STEREO_VIDEO`. Using these capability lists, other MThings in the network can discover the sensor functionalities of the smart glasses. After collecting the sensor functionalities of the smart glasses, a `sensorCapabilityParameter` depending on each capability is shown. The `sensorCapabilityParameter` presents the supportable video/audio compression methods and protocols such as `VIDEO_CODEC_MP4_AVC`, `AUDIO_CODEC_PCM`, and `VIDEO_STREAMING_PROTOCOL_HTTP`. For example, in this case, a sensor with the `SENSOR_CAPTURE_VIDEO` capability can encode a video using `VIDEO_CODEC_MP4_AVC`.

The MActuators of the smart glasses have capabilities of `ACTUATOR_DISPLAY_TRANSPARENT`, `ACTUATOR_VIBRATE`, `ACTUATOR_PLAY_AUDIO`, and `ACTUATOR_PLAY_VIDEO`.

In addition, capabilities of MAnalyzers contained in the smart glasses are `ANALYZER_DETECT_HAND_GESTURE`, `ANALYZER_RECOGNIZE_HAND_GESTURE`, `ANALYZER_MAP_HAND_GESTURE_COMMAND`, `ANALYZER_RECOGNIZE_SPEECH`, and `ANALYZER_ANALYZE_QUESTION`. The `allAnalyzerCapabilityList` presents all capabilities of MAnalyzers in the smart glasses supporting other MThings. And the storage of the smartglasses has a capability of "STORAGE_SAVE", which can store "MP4" files.

```
<mtdl:MThingInfo ID="SMGLASS-20180001" idRef="SMGLASS">
  <mtdl:MAggregator>
    <mtdl:aggregatorCapabilityList listType="availableAggregatorCapabilityList">
      <mtdl:aggregatorCapability
aggregatorCapabilityName="AGGREGATOR_SHOW_AGGREGATED_MTHINGS"/>
    </mtdl:aggregatorCapabilityList>
    <mtdl:aggregatedMThingList>
      <!--MSensor-->
      <mtdl:MSensor>
        <mtdl:sensorCapabilityList listType="availableSensorCapabilityList">
          <mtdl:sensorCapability sensorCapabilityName="SENSOR_CAPTURE_VIDEO">
            <mtdl:sensorCapabilityParameter
sensorCapabilityParameterName="VIDEO_CODEC_MP4_AVC"/>
          </mtdl:sensorCapability>
          <mtdl:sensorCapability sensorCapabilityName="SENSOR_CAPTURE_AUDIO">
            <mtdl:sensorCapabilityParameter
sensorCapabilityParameterName="AUDIO_CODEC_PCM"/>
```

```
          </mtdl:sensorCapability>
          <mtdl:sensorCapability sensorCapabilityName="SENSOR_STREAM_VIDEO">
            <mtdl:sensorCapabilityParameter
sensorCapabilityParameterName="VIDEO_STREAMING_PROTOCOL_HTTP"/>
          </mtdl:sensorCapability>
          <mtdl:sensorCapability
sensorCapabilityName="SENSOR_CAPTURE_STEREO_VIDEO">
            <mtdl:sensorCapabilityParameter
sensorCapabilityParameterName="VIDEO_CODEC_MP4_AVC"/>
          </mtdl:sensorCapability>
          <mtdl:sensorCapability
sensorCapabilityName="SENSOR_STREAM_STEREO_VIDEO">
            <mtdl:sensorCapabilityParameter
sensorCapabilityParameterName="VIDEO_STREAMING_PROTOCOL_HTTP"/>
          </mtdl:sensorCapability>
        </mtdl:sensorCapabilityList>
      </mtdl:MSensor>
      <!--/MSensor-->
      <!--MActuator-->
      <mtdl:MActuator>
        <mtdl:actuatorCapabilityList listType="allActuatorCapabilityList">
          <mtdl:actuatorCapability
actuatorCapabilityName="ACTUATOR_DISPLAY_TRANSPARENT"/>
          <mtdl:actuatorCapability actuatorCapabilityName="ACTUATOR_VIBRATE"/>
          <mtdl:actuatorCapability actuatorCapabilityName="ACTUATOR_PLAY_AUDIO"/>
          <mtdl:actuatorCapability actuatorCapabilityName="ACTUATOR_PLAY_VIDEO"/>
        </mtdl:actuatorCapabilityList>
      </mtdl:MActuator>
      <!--/MActuator-->
      <!--MAnalyzer-->
      <mtdl:MAnalyzer>
        <mtdl:analyzerCapabilityList listType="allAnalyzerCapabilityList">
          <mtdl:analyzerCapability
analyzerCapabilityName="ANALYZER_DETECT_HAND_GESTURE"/>
          <mtdl:analyzerCapability
analyzerCapabilityName="ANALYZER_RECOGNIZE_HAND_GESTURE"/>
          <mtdl:analyzerCapability
analyzerCapabilityName="ANALYZER_MAP_HAND_GESTURE_COMMAND"/>
          <mtdl:analyzerCapability
analyzerCapabilityName="ANALYZER_RECOGNIZE_SPEECH"/>
          <mtdl:analyzerCapability
analyzerCapabilityName="ANALYZER_ANALYZE_QUESTION"/>
        </mtdl:analyzerCapabilityList>
      </mtdl:MAnalyzer>
      <!--/MAnalyzer-->
      <!--MStorage-->
      <mtdl:MStorage>
        <mtdl:storageCapabilityList listType="availableStorageCapabilityList">
          <mtdl:storageCapability storageCapabilityName="STORAGE_SAVE">
            <mtdl:storageCapabilityParameter
storageCapabilityParameterName="MP4"/>
          </mtdl:storageCapability>
        </mtdl:storageCapabilityList>
      </mtdl:MStorage>
      <!--/MStorage-->
    </mtdl:aggregatedMThingList>
  </mtdl:MAggregator>
</mtdl:MThingInfo>
```

6 Media sensor output vocabulary

6.1 General

This subclause specifies syntax and semantics of the media sensor output vocabulary which comprises the following media sensors:

— IoMT sensed data captured time.

NOTE MSOV has been designed in an extensible way and additional media sensors can be added easily.

EXAMPLE Additional media sensors can be added as extensions to `mtdl:SensedDataBaseType` and conformance to MTDL.

6.2 Schema wrapper

The syntax of description tools specified in this subclause is provided as a collection of schema components, consisting notably in type definitions and element declarations. In order to form a valid schema document, these schema components should be gathered in the same document with the following declaration defining in particular the target namespace and the namespaces prefixes.

```
<?xml version="1.0"?>
<schema
  xmlns="http://www.w3.org/2001/XMLSchema"
  xmlns:mtdl="urn:mpeg:mpeg-IoMT:2018:01-MTDL-NS"
  xmlns:msov="urn:mpeg:mpeg-IoMT:2018:01-MSOV-NS"
  xmlns:mpeg7="urn:mpeg:mpeg7:schema:2004"
  targetNamespace="urn:mpeg:mpeg-IoMT:2018:01-MSOV-NS"
  elementFormDefault="qualified" attributeFormDefault="unqualified"
  version="ISO/IEC 23093-3" id="MPEG-IOMT-MSOV.xsd">

  <import namespace="urn:mpeg:mpeg7:schema:2004"
schemaLocation="http://standards.iso.org/ittf/PubliclyAvailableStandards/MPEG-
7_schema_files/mpeg7-v2.xsd"/>
  <import namespace="urn:mpeg:mpeg-IoMT:2018:01-MTDL-NS"
schemaLocation="http://standards.iso.org/ittf/PubliclyAvailableStandards/MPEG-
IOMT_schema_files/MPEG-IOMT-MTDL.xsd"/>
```

Additionally, the following line should be appended to the resulting schema document in order to obtain a well-formed XML document.

```
</schema>
```

6.3 IoMT sensed data captured time

6.3.1 General

This subclause specifies data formats to describe the sensed data captured time that can be produced by IoMT sensors.

6.3.2 Syntax

```
<!-- ################################################### -->
<!-- Data formats for IoMT sensed data caputured time    -->
<!-- ################################################### -->
<complexType name="CapturedTimeType">
    <complexContent>
        <extension base="mtdl:SensedDataBaseType">
            <sequence>
                <element name="CapturedTime" type="mpeg7:TimePointType" />
            </sequence>
        </extension>
    </complexContent>
</complexType>
```

6.3.3 Semantics

Table 72 — Semantics of the `CapturedTimeType`

Name	*Definition*
`CapturedTimeType`	Tool for specifying the time point at which capturing of sensed data started based on Gregorian date time and time zone refering to ISO 8601.
`CapturedTime`	Describes the captured time of sensed data.

6.3.4 Example

This example shows how the captured time of sensed data from an MSesnor can be described.

```
<mtdl:MThingInfo>
    <mtdl:MSensor>
        <mtdl:sensedData xsi:type="msov:CapturedTimeType">
            <msov:CapturedTime>2018-09-07T10:47+09:00</msov:CapturedTime>
        </mtdl:sensedData>
    </mtdl:MSensor>
</mtdl:MThingInfo>
```

7 Media actuator command vocabulary

7.1 General

This subclause specifies syntax and semantics of the media actuator command vocabulary which comprises the following media actuators:

— IoMT speaker;

— IoMT display;

— IoMT camera actuator;

— IoMT camera light.

NOTE MACV has been designed in an extensible way and additional media actuators can be added easily.

EXAMPLE Additional media actuators can be added as extensions to `mtdl:ActuationDataBaseType` and conformance to MTDL.

7.2 Schema wrapper

The syntax of description tools specified in this subclause is provided as a collection of schema components, consisting notably in type definitions and element declarations. In order to form a valid schema document, these schema components should be gathered in the same document with the following declaration defining in particular the target namespace and the namespaces prefixes.

```
<?xml version="1.0"?>
<schema
  xmlns="http://www.w3.org/2001/XMLSchema"
  xmlns:mtdl="urn:mpeg:mpeg-IoMT:2018:01-MTDL-NS"
  xmlns:macv="urn:mpeg:mpeg-IoMT:2018:01-MACV-NS"
  xmlns:scdv="urn:mpeg:mpeg-v:2018:01-SCDV-NS"
  targetNamespace="urn:mpeg:mpeg-IoMT:2018:01-MACV-NS"
  elementFormDefault="qualified" attributeFormDefault="unqualified"
  version="ISO/IEC 23093-3" id="MPEG-IOMT-MACV.xsd">

  <import namespace="urn:mpeg:mpeg-IoMT:2018:01-MTDL-NS"
schemaLocation="http://standards.iso.org/ittf/PubliclyAvailableStandards/MPEG-
IOMT_schema_files/MPEG-IOMT-MTDL.xsd"/>
  <import namespace="urn:mpeg:mpeg-v:2018:01-SCDV-NS"
schemalocation="http://standards.iso.org/ittf/PubliclyAvailableStandards/MPEG-
V_schema_files/MPEG-V-SCDV.xsd"/>
```

Additionally, the following line should be appended to the resulting schema document in order to obtain a well-formed XML document.

```
</schema>
```

7.3 IoMT speaker

7.3.1 General

This subclause specifies data formats to describe the commands that can control the IoMT speaker.

7.3.2 Syntax

```
<!-- ################################################ -->
<!-- Data formats for IoMT speaker commands           -->
<!-- ################################################ -->
<complexType name="AudioPlayType">
   <complexContent>
      <extension base="mtdl:ActuationDataBaseType">
         <sequence>
            <element name="playType" type="macv:AudioActuationType"/>
         </sequence>
      </extension>
   </complexContent>
</complexType>
```



```
<simpleType name="AudioActuationType">
    <restriction base="string">
        <enumeration value="play"/>
        <enumeration value="stop"/>
        <enumeration value="pause"/>
        <enumeration value="rewind2x"/>
        <enumeration value="rewind4x"/>
        <enumeration value="rewind8x"/>
        <enumeration value="rewind16x"/>
        <enumeration value="rewind32x"/>
        <enumeration value="fastForward2x"/>
        <enumeration value="fastForward4x"/>
        <enumeration value="fastForward8x"/>
        <enumeration value="fastForward16x"/>
        <enumeration value="fastForward32x"/>
    </restriction>
</simpleType>

<complexType name="setVolumeType">
    <complexContent>
        <extension base="mtdl:ActuationDataBaseType">
            <sequence>
                <element name="volume">
                    <simpleType>
                        <restriction base="integer">
                            <minInclusive value="0"/>
                            <maxInclusive value="100"/>
                        </restriction>
                    </simpleType>
                </element>
            </sequence>
        </extension>
    </complexContent>
</complexType>
```


7.3.3 Semantics

Table 73 — Semantics of the IoMT `speaker command` types

Name	*Definition*
`AudioPlayType`	Tool for describing a command for a speaker device to follow.
`playType`	Describes the audio play command.
`AudioActuationType`	Describes the commands which are play, stop, rewind 2x to 32x, and fast forward 2x to 32x.
`setVolumeType`	Tool for setting the volume of the speaker.
`Volume`	Describes the volume to speaker in percent between 0 and 100.

7.3.4 Example

This example shows how to set fast forward at 2x speed of an IoMT speaker.

```
<mtdl:actuationData xsi:type="macv:AudioPlayType">
    <macv:playType>fastForward2x</macv:playType>
</mtdl:actuationData>
```

This example shows how to set the volume 80 % of an IoMT speaker.

```
<mtdl:actuationData xsi:type="macv:setVolumeType">
    <macv:volume>80</macv:volume>
</mtdl:actuationData>
```

7.4 IoMT display

7.4.1 General

This subclause specifies data formats to describe the commands that control the IoMT display.

7.4.2 Syntax

```
<!-- ################################################ -->
<!-- Data formats for IoMT display commands           -->
<!-- ################################################ -->
<complexType name="VideoPlayType">
    <complexContent>
        <extension base="mtdl:ActuationDataBaseType">
            <sequence>
                <element name="playType" type="macv:VideoActuationType"/>
            </sequence>
        </extension>
    </complexContent>
</complexType>

<simpleType name="VideoActuationType">
    <restriction base="string">
        <enumeration value="stop"/>
        <enumeration value="play"/>
        <enumeration value="pause"/>
        <enumeration value="rewind2x"/>
        <enumeration value="rewind4x"/>
        <enumeration value="rewind8x"/>
        <enumeration value="rewind16x"/>
        <enumeration value="rewind32x"/>
        <enumeration value="fastForward2x"/>
        <enumeration value="fastForward4x"/>
        <enumeration value="fastForward8x"/>
        <enumeration value="fastForward16x"/>
        <enumeration value="fastForward32x"/>
    </restriction>
</simpleType>

<complexType name="setBrightnessType">
    <complexContent>
        <extension base="mtdl:ActuationDataBaseType">
            <sequence>
                <element name="brightness">
                    <simpleType>
                        <restriction base="integer">
```

```
                        <minInclusive value="0"/>
                        <maxInclusive value="100"/>
                    </restriction>
                </simpleType>
            </element>
        </sequence>
    </extension>
  </complexContent>
</complexType>
```

7.4.3 Semantics

Table 74 — Semantics of the IoMT `display command` types

Name	*Definition*
`VideoPlayType`	Tool for describing a command for a display device to follow.
`playType`	Describes the video play command.
`VideoActuationType`	Describes the commands which are play, stop, rewind 2x to 32x, and fast forward 2x to 32x.
`setBrightnessType`	Tool for setting the brightness of the display.
`Brightness`	Describes the brightness of display.

7.4.4 Example

This example shows how to set "stop playing" of an IoMT display.

```
<mtdl:actuationData xsi:type="macv:VideoPlayType">
   <macv:playType>stop</macv:playType>
</mtdl:actuationData>
```

This example shows how to set brightness to 100 % of an IoMT display.

```
<mtdl:actuationData xsi:type="macv:setBrightnessType">
   <macv:brightness>100</macv:brightness>
</mtdl:actuationData>
```

7.5 IoMT camera actuator

7.5.1 General

This subclause specifies data formats to describe the commands that can control the IoMT camera actuator.

7.5.2 Syntax

```
<!-- ################################################ -->
<!-- Definition of IoMT camera actuation commands     -->
<!-- ################################################ -->
<complexType name="setCameraOrientationType">
   <complexContent>
      <extension base="mtdl:ActuationDataBaseType">
         <sequence>
            <element name="yaw">
               <simpleType>
                  <restriction base="integer">
                     <minInclusive value="-180"/>
                     <maxInclusive value="180"/>
                  </restriction>
               </simpleType>
            </element>
            <element name="pitch">
               <simpleType>
                  <restriction base="integer">
                     <minInclusive value="-180"/>
                     <maxInclusive value="180"/>
                  </restriction>
               </simpleType>
            </element>
            <element name="roll">
               <simpleType>
                  <restriction base="integer">
                     <minInclusive value="-180"/>
                     <maxInclusive value="180"/>
                  </restriction>
               </simpleType>
            </element>
         </sequence>
      </extension>
   </complexContent>
</complexType>
<complexType name="setCameraZoomType">
   <complexContent>
      <extension base="mtdl:ActuationDataBaseType">
         <sequence>
            <element name="zoom" type="nonNegativeFloat"/>
         </sequence>
      </extension>
   </complexContent>
</complexType>
<complexType name="setCameraResolutionType">
   <complexContent>
      <extension base="mtdl:ActuationDataBaseType">
         <sequence>
            <element name="resolution" type="scdv:ResolutionType"/>
         </sequence>
      </extension>
   </complexContent>
</complexType>
```

7.5.3 Semantics

Table 75 — Semantics of the IoMT `Camera actuator command types`

Name	*Definition*
`setCameraOrientationType`	Tool for describing a command to set a camera orientation.
`Yaw`	Describes the angle to rotate in z-axis, Ψ(yaw) in degrees between −180 and 180.
`Pitch`	Describes the angle to rotate in y-axis, Θ(pitch) in degrees between −180 and 180.
`Roll`	Describes the angle to rotate in x-axis, φ (roll), in degrees between −180 and 180.
`setCameraZoomType`	Tool for a command to set the camera zoom.
`Zoom`	Describes the magnitude of zooming. The number less than 1 indicates the zoom-out, and the number greater than 1 indicates the zoom-in.
`setCameraResolutionType`	Tool for setting the camera resolution.
`resolution`	Describes the capturing resolution, which is composed of a width and a height.

7.5.4 Example

This example shows how to set the orientation of an IoMT camera to its the yaw angle 90°, the pitch angle 45°, and the roll angle 60°.

```
<mtdl:actuationData xsi:type="macv:setCameraOrientationType">
    <macv:yaw>90</macv:yaw>
    <macv:pitch>45</macv:pitch>
    <macv:roll>60</macv:roll>
</mtdl:actuationData>
```

This example shows how to set the zoom-in to 4 times.

```
<mtdl:actuationData xsi:type="macv:setCameraZoomType">
    <macv:zoom>4</macv:zoom>
</mtdl:actuationData>
```

This example shows how to set the camera resolution to 1920x1080.

```
<mtdl:actuationData xsi:type="macv:setCameraResolutionType">
    <macv:resolution>
        <macv:Width>1920</macv:Width>
        <macv:Height>1080</macv:Height>
    </macv:resolution>
</mtdl:actuationData>
```

7.6 IoMT light

7.6.1 General

This subclause specifies data formats to describe the commands that can control the IoMT light.

7.6.2 Syntax

```
<!-- ################################################ -->
<!-- Definition of IoMT Light actuation commands      -->
<!-- ################################################ -->
<complexType name="setColorLightType">
   <complexContent>
      <extension base="mtdl:ActuationDataBaseType">
         <sequence>
            <element name="brightness" minOccurs="0">
               <simpleType>
                  <restriction base="float">
                     <minInclusive value="0.0"/>
                     <maxInclusive value="1"/>
                  </restriction>
               </simpleType>
            </element>
            <element name="saturation" minOccurs="0">
               <simpleType>
                  <restriction base="float">
                     <minInclusive value="0.0"/>
                     <maxInclusive value="1"/>
                  </restriction>
               </simpleType>
            </element>
            <element name="hue" minOccurs="0">
               <simpleType>
                  <restriction base="integer">
                     <minInclusive value="0"/>
                     <maxInclusive value="360"/>
                  </restriction>
               </simpleType>
            </element>
         </sequence>
      </extension>
   </complexContent>
</complexType>
```


7.6.3 Semantics

Table 76 — Semantics of the IoMT `light command` types

Name	*Definition*
`setColorLightType`	Tool for describing a command to set color of lights.
`brightness`	Describes the value of brightness in the HSV color model.
`saturation`	Describes the value of saturation in the HSV color model.

Name	*Definition*
`hue`	Describes the value of hue in the HSV color model.

7.6.4 Example

This example shows a command to set color of a light with the brightness 0.7, the saturation 0.8, and the hue 240.

```
<mtdl:actuationData xsi:type="macv:setColorLightType">
    <macv:brightness>0.7</macv:brightness>
    <macv:saturation>0.8</macv:saturation>
    <macv:hue>240</macv:hue>
</mtdl:actuationData>
```

This example shows a command to set color of a light with only the brightness 0.8.

```
<mtdl:actuationData xsi:type="macv:setColorLightType">
    <macv:brightness>0.8</macv:brightness>
</mtdl:actuationData>
```

This example shows a command to set color of a light with the saturation 0.9 and the hue 270.

```
<mtdl:actuationData xsi:type="macv:setColorLightType">
    <macv:saturation>0.9</macv:saturation>
    <macv:hue>270</macv:hue>
</mtdl:actuationData>
```

8 Media analyzer output vocabulary

8.1 General

This subclause specifies syntax and semantics of the media analyzer output vocabulary which comprises the following media analyzers:

— IoMT time synchronizer;

— IoMT social event detector;

— IoMT hand gesture detector;

— IoMT hand gesture recognizer;

— IoMT hand gesture command generator;

— IoMT healthcare information generator;

— IoMT odor-image scent recognizer;

— IoMT question analyzer;

— IoMT music frequency analyzer;

— IoMT video content class generator.

NOTE MAOV has been designed in an extensible way and additional media analyzers can be added easily.

EXAMPLE Additional media analyzers can be added as extensions to `mtdl:AnalyzedDataBaseType` and conformance to MTDL.

8.2 Schema wrapper

The syntax of description tools specified in this subclause is provided as a collection of schema components, consisting notably of type definitions and element declarations. In order to form a valid schema document, these schema components should be gathered in the same document with the following declaration defining in particular the target namespace and the namespaces prefixes.

```
<?xml version="1.0"?>
  <schema
  xmlns="http://www.w3.org/2001/XMLSchema"
  xmlns:mtdl="urn:mpeg:mpeg-IoMT:2018:01-MTDL-NS"
  xmlns:maov="urn:mpeg:mpeg-IoMT:2018:01-MAOV-NS"
  xmlns:scdv="urn:mpeg:mpeg-v:2018:01-SCDV-NS"
  xmlns:mpeg7="urn:mpeg:mpeg7:schema:2004"
  targetNamespace="urn:mpeg:mpeg-IoMT:2018:01-MAOV-NS"
  elementFormDefault="qualified" attributeFormDefault="unqualified"
  version="ISO/IEC 23093-3" id="MPEG-IOMT-MAOV.xsd">

   <import namespace="urn:mpeg:mpeg-IoMT:2018:01-MTDL-NS"
schemaLocation="http://standards.iso.org/ittf/PubliclyAvailableStandards/MPEG-
IOMT_schema_files/MPEG-IOMT-MTDL.xsd"/>
  <import namespace="urn:mpeg:mpeg-v:2018:01-SCDV-NS"
schemalocation="http://standards.iso.org/ittf/PubliclyAvailableStandards/MPEG-
V_schema_files/MPEG-V-SCDV.xsd"/>
  <import namespace="urn:mpeg:mpeg7:schema:2004"
schemaLocation="http://standards.iso.org/ittf/PubliclyAvailableStandards/MPEG-
7_schema_files/mpeg7-v2.xsd"/>
```

Additionally, the following line should be appended to the resulting schema document in order to obtain a well-formed XML document.

```
</schema>
```


8.3 IoMT time synchronizer

8.3.1 General

This subclause specifies data formats to describe the outputs that can be produced by the `MTimeSynchronizer`. The `MTimeSynchronizer` obtains video sources and captured times from two different `MCameras`, and calculates the time offset between two different videos.

8.3.2 Syntax

```
<!-- ################################-->
<!-- Definition of Synced Video Type    -->
<!-- ################################-->

<complexType name="SyncedVideoType">
   <complexContent>
      <extension base="mtdl:AnalyzedDataBaseType">
         <sequence>
            <choice>
               <sequence>
                  <element name="ReferenceVideoURL" type ="anyURI"/>
                  <element name="TargetVideoURL" type="anyURI"/>
               </sequence>
               <sequence>
                  <element name="ReferenceAudioURL" type = "anyURI" />
                  <element name="TargetAudioURL" type="anyURI"/>
               </sequence>
            </choice>
            <element name="TimeOffset" type="mpeg7:MediaTimeType"/>
         </sequence>
      </extension>
   </complexContent>
</complexType>
```

8.3.3 Semantics

Table 77 — Semantics of the IoMT `time synchronizer` types

Name	*Definition*
`SyncedVideoType`	Tool for describing information of video with time information.
`AnalyzedDataBaseType`	Tool for describing an AnalyzedDataBaseType.
`ReferenceVideoURL`	Describes a URL that identifies the reference video source.
`TargetVideoURL`	Describes a URL that identifies the target video source.
`ReferenceAudioURL`	Describes a URL that identifies the reference audio source.
`TargetAudioURL`	Describes a URL that identifies the target audio source.
`TimeOffset`	Describes time offset of media for time synchronization using video, audio and media time information. As defined in ISO/IEC 15938-5.

8.3.4 Example

This example shows the time offset between *con1.avi* and *con2.avi*. The *con2.avi* is 15 frames faster than the *con1.avi*. The time offset represents a difference of 15 frames on a 1/30 frame basis.

```
<mtdl:analyzedData xsi:type="maov:SyncedVideoType">
   <maov:ReferenceVideoURL>http://www.moeg.com/con1.avi</maov:ReferenceVideoURL>
   <maov:TargetVideoURL>http://www.moeg.com/con2.avi</maov:TargetVideoURL>
```

```
    <maov:TimeOffset>
       <mpeg7:MediaRelIncrTimePoint
mediaTimeUnit="PT1N30F">15</mpeg7:MediaRelIncrTimePoint>
    </maov:TimeOffset>
</mtdl:analyzedData>
```

8.4 IoMT social event detector

8.4.1 General

This subclause specifies data format for a social event analyzer output. The `SocialEventType` is used to represent information of analyzed social event.

8.4.2 Syntax

```
<!-- ##############################################-->
<!-- Definition of Social Event Detector Type     -->
<!-- ##############################################-->

<complexType name="SocialEventType">
   <complexContent>
      <extension base="mtdl:AnalyzedDataBaseType">
         <sequence>
            <element name="SocialEventName" type="mpeg7:termReferenceType"/>
         </sequence>
      </extension>
   </complexContent>
</complexType>
```

8.4.3 Semantics

Table 78 — Semantics of the IoMT `social event detector` types

Name	*Definition*
`SocialEventType`	Tool for describing the analyzed social event.
`SocialEventName`	Describes the analysed social event type of a video source. The name of the social event shall be described using the `mpeg7:termReferenceType` defined in ISO/IEC 15938-5:2003, 7.6. A classification scheme that may be used for this purpose is the `SocialEventCS` defined in A.4.2.

8.4.4 Example

This example shows that the detected social event is the birthday.

```
<mtdl:analyzedData xsi:type="maov:SocialEventType">
   <maov:SocialEventName>BIRTHDAY_SOCIAL_EVENT</maov:SocialEventName>
</mtdl:analyzedData>
```

8.5 IoMT hand gesture detector

8.5.1 General

This subclause specifies data format for a hand gesture detector output. The `HandGestureType` is used to represent information of detected hand gesture contours.

8.5.2 Syntax

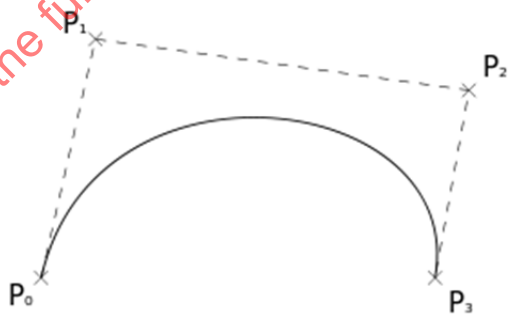
```
<!-- ################################################ -->
<!-- Definition of Hand Gesture Type                  -->
<!-- ################################################ -->
<complexType name="HandGestureType">
    <complexContent>
        <extension base="mtdl:AnalyzedDataBaseType">
            <choice>
                <element name="HandContour" type="maov:HandContourType"
minOccurs="0" maxOccurs="unbounded"/>
                <element name="HandTrajectory" type="maov:HandTrajectoryType"
minOccurs="0" maxOccurs="unbounded"/>
            </choice>
            <attribute name="lefthand" type="boolean" default="false"/>
            <attribute name="threeDspace" type="boolean" default="false"/>
            <attribute name="frameRate" type="decimal" use="optional"
default="30"/>
        </extension>
    </complexContent>
</complexType>

<complexType name="HandContourType">
    <sequence>
        <choice>
            <element name="coordinate" type="mpeg7:integerVector" minOccurs="0"/>
            <element name="GroupBezierCurve" type="maov:GroupBezierCurveType"
minOccurs="0"/>
        </choice>
        <element name="CenterPosition" type="mpeg7:integerVector" minOccurs="0"/>
    </sequence>
    <attribute name="useLastContour" type="boolean" default="false"/>
</complexType>

<complexType name="GroupBezierCurveType">
    <sequence maxOccurs="unbounded">
        <element name="InitialStartPoint" type="mpeg7:integerVector"/>
        <sequence maxOccurs="unbounded">
            <element name="BezierCurve" type="maov:BezierCurveType"/>
        </sequence>
    </sequence>
    <attribute name="OrderOfBezierCurve" type="mpeg7:unsigned8" default="3"/>
    <attribute name="FittingError" type="double" use="optional"/>
</complexType>

<complexType name="BezierCurveType">
    <sequence>
    <element name="StartEndPoint" type="mpeg7:integerVector"/>
        <sequence>
            <element name="ControlPoint" type="mpeg7:integerVector"
maxOccurs="unbounded"/>
        </sequence>
    </sequence>
```

```
</complexType>

<complexType name="HandTrajectoryType">
    <choice>
        <element name="GroupBezierCurve" type="maov:GroupBezierCurveType"
minOccurs="0"/>
        <sequence minOccurs="0" maxOccurs="unbounded">
            <element name="CenterPosition" type="mpeg7:integerVector"
minOccurs="0"/>
        </sequence>
    </choice>
    <attribute name="trajectoryType" type="boolean" default="false"/>
</complexType>
```

8.5.3 Semantics

Table 79 — Semantics of the IoMT `hand gesture detector` types

Name	*Definition*
`HandGestureType`	Tool for describing a hand gesture. The hand gesture description is used for representation of a hand contour or hand trajectory.
`HandContour`	Describes a hand contour extracted from the incoming image sequences by the processing of hand gesture detection. The hand contour description can be used to generate hand gesture based commands for user interaction.
`HandTrajectory`	Describes a hand trajectory extracted from the incoming image sequences by the processing of hand trajectory extraction. The hand trajectory description can be used to generate hand gesture based commands for user interaction.
`ObjectShape`	Describes an object shape extracted from the incoming image sequences by the processing of object detection. The object shape description can be used to exploit object shape in diverse IoMT applications.
`lefthand`	Describes the indication whether the left hand is used for the generating hand gesture. If this value is "false", then the right hand is used for generating a hand gesture command, otherwise, the left hand is used. The default is "false".
`threeDspace`	Describes the indication whether the gesture is represented in 3-D space or 2-D space. For example, a hand trajectory can be represented as a 3-D trajectory or 2-D trajectory that is the projection of the 3-D trajectory into the 2-D space.
`frameRate`	Describes the frame rate of the incoming image sequence. `framerate` is used to calculate the time interval between consecutive hand contours or the speed of hand motion, etc.
`HandContourType`	Tool for describing a hand contour. A hand contour description can be used to represent a general hand contour, from which gesture-based interaction commands can be generated as a result of the gesture recognition.

Name	*Definition*
`Coordinate`	Describes a hand contour in a form of a set of coordinates in 2-D or 3-D space.
`GroupBezierCurve`	Describes a hand contour or a hand trajectory in a form of a set of curves in 2-D space. A curve can be represented in a form of a set of 2-D coordinates.
`CenterPosition`	Describes a center position of the detected hand contour in a form of the 2-D coordinates. A hand trajectory can be generated from a set of center positions of hands later.
`useLastContour`	Describes the indication whether the current contour description is available or the last available contour is used. When there is no need to generate a hand contour description of the detected hand for each frame or the valid hand contour is not detected for a certain frame, the last available contour description can be used instead as the current frame's contour description.
`GroupBezierCurveType`	Tool for describing a set of Bezier curves description. A group of Bezier curve description can be used to represent a hand gesture or a trajectory. Bezier curve is composed of two control points (P_1, P_2), one is a start point (P_0), and the other is an end point (P_3). Two control points are used for curvature of contour (Figure 9).  **Figure 9 — Bezier curve**
`InitialStartPoint`	Describes the initial point for the first Bezier curve of a set of Bezier curves representing the whole hand contour. By describing the initial point, the overlapping point between the consecutive curves can be described once instead of describing the same point redundantly. The start and end point of hand contour which is called InitialStartPoint is the same due to the closed hand contour. InitialStartPoint is the same with the last Bezier curve's end point when an hand contour is represented by a set of consecutive Bezer curves.
`BezierCurve`	Describes a part of the hand contour in a form of Bezier curve. For each Bezier curve, (the order of Bezier curve n + 1) points are used to fitting the given curve by a polynomial. By using the initial point of the first Bezier curve, n-points are used to represent each curve of consecutive curves representing hand contour instead of (n + 1) points. Each point is described as a form of coordinate in 2-D space.

Name	*Definition*
`OrderOfBezierCurve`	Describes the order of Bezier curve, which is the order of polynomial fitting the given curve. For n-th Bezier curve, (n + 1) points in 2-D space, which consists of the starting point, end point, and (n-1) control points, are used to represent the given curve.
`FittingError`	Describes the fitting error of the Bezier curve. The fitting error is used to represent the accuracy of Bezier curve based description of the given hand gesture in optional.
`BezierCurveType`	Tool for describing a Bezier curve. A Bezier curve uses (the order of Bezier curve n + 1) points to fit the given curve by a polynomial. By using the initial point of the first Bezier curve, n-points are used to represent each of the consecutive curves representing hand contour instead of (n + 1) points. Each point is described as a form of coordinate in 2-D space.
`ControlPoint`	Describes a control point for Bezier curve in a form of 2-D coordinate.
`StartEndPoint`	Describes a start and an end point of a Bezier curve in a form of a 2-D coordinate. The start and end point are the overlapping points between consecutive curves representing the given hand contour. Because one hand gesture consists of a set of Bezier curves, the start point and the end point of each Bezier curves are overlapped. To avoid duplication of the start point and the end point, `StartEndPoint` is used to both points.
`HandTrajectoryType`	Tool for describing a hand trajectory. A hand trajectory description can be used to represent the general hand trajectory, from which gesture-based interaction commands can be generated as a result of the gesture recognition. It can be represented by a group of Bezier curves or a sequence of center positions by hands.
`trajectoryType`	Describes the indication whether the trajectory is defined and generated in the unit of predefined time interval or in the moving duration. The type of motion interval (`motionInterval`) means the trajectory is generated during the hand moving duration that is identified by a motion detection. The type of time interval (`timeInterval`) indicates the trajectory is generated in the unit of predefined time interval. true: time interval / false: motion interval

8.5.4 Example

In this instance, the right hand is used for generating a hand gesture command because the `lefthand` value is “false”. The set of Bezier curves constituting the hand contour is represented by a Bezier points list. The hand contour generated by this example is shown in Figure 10.

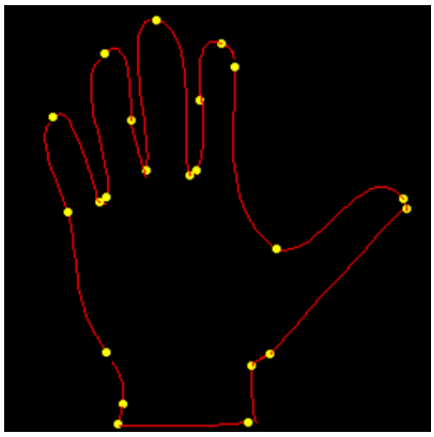


Figure 10 — Hand contour generated from an IoMT hand gesture detector

```
<mtdl:analyzedData        xsi:type="maov:HandGestureType"        lefthand="false"
threeDspace="false">
  <maov:HandContour>
    <maov:GroupBezierCurve>
      <maov:InitialStartPoint>
        80 22</ maov:InitialStartPoint>
        <maov:BezierCurve>
          <maov:StartEndPoint>87 9</maov:StartEndPoint>
          <maov:ControlPoint>70 17</maov:ControlPoint>
          <maov:ControlPoint>90 82</maov:ControlPoint>
        </maov:BezierCurve>
        <maov:BezierCurve>
          <maov:StartEndPoint>85 103</maov:StartEndPoint>
          <maov:ControlPoint>84 103</maov:ControlPoint>
          <maov:ControlPoint>76 74</maov:ControlPoint>
        </maov:BezierCurve>
        <maov:BezierCurve>
          <maov:StartEndPoint>76 69</maov:StartEndPoint>
          <maov:ControlPoint>76 70</maov:ControlPoint>
          <maov:ControlPoint>78 6</maov:ControlPoint>
        </maov:BezierCurve>
        <maov:BezierCurve>
          <maov:StartEndPoint>57 33</maov:StartEndPoint>
          <maov:ControlPoint>40 52</maov:ControlPoint>
          <maov:ControlPoint>69 98</maov:ControlPoint>
        </maov:BezierCurve>
        <maov:BezierCurve>
          <maov:StartEndPoint>61 119</maov:StartEndPoint>
          <maov:ControlPoint>60 120</maov:ControlPoint>
          <maov:ControlPoint>57 118</maov:ControlPoint>
        </maov:BezierCurve>
        <maov:BezierCurve>
          <maov:StartEndPoint>27 71</maov:StartEndPoint>
          <maov:ControlPoint>18 85</maov:ControlPoint>
          <maov:ControlPoint>35 111</maov:ControlPoint>
        </maov:BezierCurve>
        <maov:BezierCurve>
          <maov:StartEndPoint>39 127</maov:StartEndPoint>
          <maov:ControlPoint>48 172</maov:ControlPoint>
          <maov:ControlPoint>38 174</maov:ControlPoint>
        </maov:BezierCurve>
        <maov:BezierCurve>
          <maov:StartEndPoint>65 214</maov:StartEndPoint>
```

```
          <maov:ControlPoint>70 222</maov:ControlPoint>
          <maov:ControlPoint>53 230</maov:ControlPoint>
        </maov:BezierCurve>
        <maov:BezierCurve>
          <maov:StartEndPoint>71 241</maov:StartEndPoint>
          <maov:ControlPoint>70 241</maov:ControlPoint>
          <maov:ControlPoint>68 251</maov:ControlPoint>
        </maov:BezierCurve>
        <maov:BezierCurve>
          <maov:StartEndPoint>69 252</maov:StartEndPoint>
          <maov:ControlPoint>96 253</maov:ControlPoint>
          <maov:ControlPoint>123 253</maov:ControlPoint>
        </maov:BezierCurve>
        <maov:BezierCurve>
          <maov:StartEndPoint>151 250</maov:StartEndPoint>
          <maov:ControlPoint>149 250</maov:ControlPoint>
          <maov:ControlPoint>147 220</maov:ControlPoint>
        </maov:BezierCurve>
        <maov:BezierCurve>
          <maov:StartEndPoint>149 216</maov:StartEndPoint>
          <maov:ControlPoint>149 213</maov:ControlPoint>
          <maov:ControlPoint>158 210</maov:ControlPoint>
        </maov:BezierCurve>
        <maov:BezierCurve>
          <maov:StartEndPoint>160 209</maov:StartEndPoint>
          <maov:ControlPoint>162 260</maov:ControlPoint>
          <maov:ControlPoint>245 112</maov:ControlPoint>
        </maov:BezierCurve>
        <maov:BezierCurve>
          <maov:StartEndPoint>241 123</maov:StartEndPoint>
          <maov:ControlPoint>241 120</maov:ControlPoint>
          <maov:ControlPoint>240 117</maov:ControlPoint>
        </maov:BezierCurve>
        <maov:BezierCurve>
          <maov:StartEndPoint>239 116</maov:StartEndPoint>
          <maov:ControlPoint>215 87</maov:ControlPoint>
          <maov:ControlPoint>184 163</maov:ControlPoint>
        </maov:BezierCurve>
        <maov:BezierCurve>
          <maov:StartEndPoint>160 144</maov:StartEndPoint>
          <maov:ControlPoint>127 118</maov:ControlPoint>
          <maov:ControlPoint>140 68</maov:ControlPoint>
        </maov:BezierCurve>
        <maov:BezierCurve>
          <maov:StartEndPoint>138 32</maov:StartEndPoint>
          <maov:ControlPoint>137 27</maov:ControlPoint>
          <maov:ControlPoint>133 24</maov:ControlPoint>
        </maov:BezierCurve>
        <maov:BezierCurve>
          <maov:StartEndPoint>130 23</maov:StartEndPoint>
          <maov:ControlPoint>112 16</maov:ControlPoint>
          <maov:ControlPoint>118 54</maov:ControlPoint>
        </maov:BezierCurve>
        <maov:BezierCurve>
          <maov:StartEndPoint>118 58</maov:StartEndPoint>
          <maov:ControlPoint>118 41</maov:ControlPoint>
          <maov:ControlPoint>123 93</maov:ControlPoint>
        </maov:BezierCurve>
        <maov:BezierCurve>
          <maov:StartEndPoint>114 101</maov:StartEndPoint>
```

```
          <maov:ControlPoint>113 101</maov:ControlPoint>
          <maov:ControlPoint>111 103</maov:ControlPoint>
        </maov:BezierCurve>
        <maov:BezierCurve>
          <maov:StartEndPoint>111 102</maov:StartEndPoint>
          <maov:ControlPoint>105 77</maov:ControlPoint>
          <maov:ControlPoint>118 9</maov:ControlPoint>
        </maov:BezierCurve>
      </maov:GroupBezierCurve>
  </maov:HandContour>
</mtdl:analyzedData>
```

In this instance, a user uses their right hand to generate a hand trajectory because the `lefthand` value is "false". A list of Bezier points consisting of one trajectory representing the movement of hand are provided. Center point list are sampled by the type of time interval so `trajectoryType` is "true". A hand trajectory generated from an IoMT hand gesture detector is shown in Figure 11.

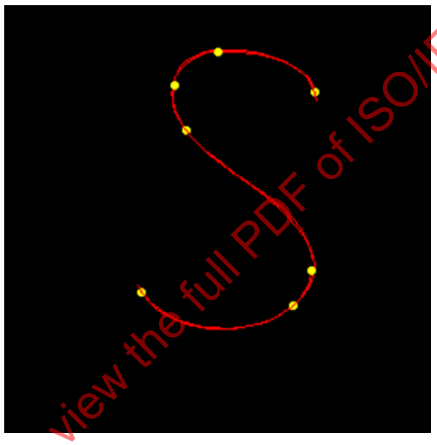


Figure 11 — Hand trajectory generated from an IoMT hand gesture detector

```
<mtdl:analyzedData      xsi:type="maov:HandGestureType"        lefthand="false"
threeDspace="false">
  <maov:HandTrajectory trajectoryType="true">
    <maov:GroupBezierCurve>
      <maov:InitialStartPoint>126 28</maov:InitialStartPoint>
      <maov:BezierCurve>
        <maov:StartEndPoint>126 28</maov:StartEndPoint>
        <maov:ControlPoint>99 32</maov:ControlPoint>
        <maov:ControlPoint>94 59</maov:ControlPoint>
      </maov:BezierCurve>
      <maov:BezierCurve>
        <maov:StartEndPoint>112 79</maov:StartEndPoint>
        <maov:ControlPoint>135 104</maov:ControlPoint>
        <maov:ControlPoint>226 148</maov:ControlPoint>
      </maov:BezierCurve>
      <maov:BezierCurve>
        <maov:StartEndPoint>165 186</maov:StartEndPoint>
        <maov:ControlPoint>149 195</maov:ControlPoint>
        <maov:ControlPoint>80 198</maov:ControlPoint>
      </maov:BezierCurve>
      <maov:BezierCurve>
        <maov:StartEndPoint>80 167</maov:StartEndPoint>
        <maov:ControlPoint>80 200</maov:ControlPoint>
        <maov:ControlPoint>188 207</maov:ControlPoint>
```

```
      </maov:BezierCurve>
      <maov:BezierCurve>
        <maov:StartEndPoint>185 153</maov:StartEndPoint>
        <maov:ControlPoint>182 104</maov:ControlPoint>
        <maov:ControlPoint>85 97</maov:ControlPoint>
      </maov:BezierCurve>
      <maov:BezierCurve>
        <maov:StartEndPoint>104 41</maov:StartEndPoint>
        <maov:ControlPoint>112 16</maov:ControlPoint>
        <maov:ControlPoint>187 27</maov:ControlPoint>
      </maov:BezierCurve>
      <maov:BezierCurve>
        <maov:StartEndPoint>187 57</maov:StartEndPoint>
        <maov:ControlPoint>187 30</maov:ControlPoint>
        <maov:ControlPoint>142 28</maov:ControlPoint>
      </maov:BezierCurve>
    </maov:GroupBezierCurve>
  </maov:HandTrajectory>
</mtdl:analyzedData>
```

8.6 IoMT hand gesture recognizer

8.6.1 General

This subclause specifies data format for a hand gesture recognizer output. The `HandPostureType` is used to represent information of a recognized hand posture.

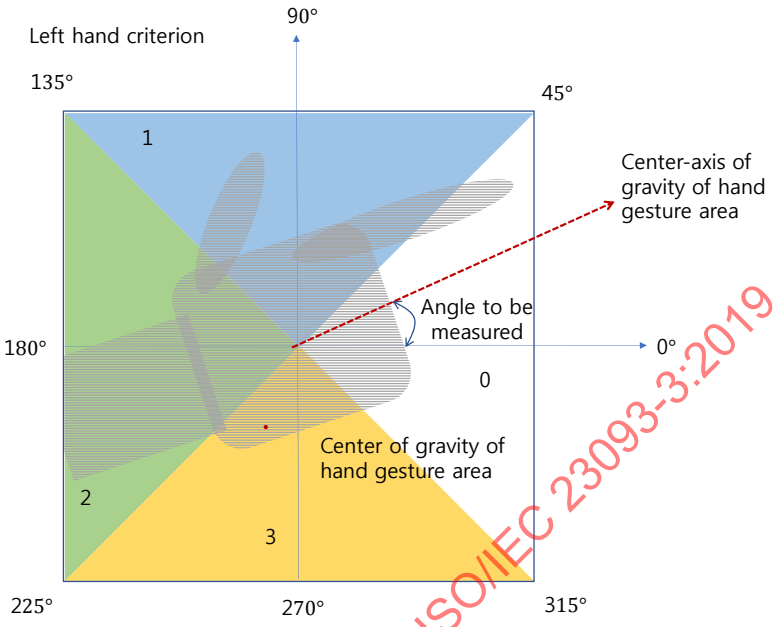
8.6.2 Syntax

```
<!-- ############################################### -->
<!-- Definition of Hand Posture Type                 -->
<!-- ############################################### -->
<complexType name="HandPostureType">
   <complexContent>
      <extension base="mtdl:AnalyzedDataBaseType">
         <attribute name="Thumb" type="boolean" use="required"/>
         <attribute name="Index" type="boolean" use="required"/>
         <attribute name="Middle" type="boolean" use="required"/>
         <attribute name="Ring" type="boolean" use="required"/>
         <attribute name="Little" type="boolean" use="required"/>
         <attribute name="Angle" default="0">
            <simpleType>
               <restriction base="integer">
                  <minInclusive value="0"/>
                  <maxInclusive value="3"/>
               </restriction>
            </simpleType>
         </attribute>
         <attribute name="InnerContour" type="boolean" use="required"/>
         <attribute name="HandSide" type="boolean" use="required"/>
      </extension>
   </complexContent>
</complexType>
```

8.6.3 Semantics

Table 80 — Semantics of the IoMT `hand gesture recognizer` types

Name	*Definition*
`HandPostureType`	Provides an abstract of Hand Posture Type.
`Thumb`	Describes a thumb finger status. If the finger is opened, then the value is true. Otherwise, the value is false.
`Index`	Describes an index finger status. If the finger is opened, then the value is true. Otherwise, the value is false.
`Middle`	Describes a middle finger status. If the finger is opened, then the value is true. Otherwise, the value is false.
`Ring`	Describes a ring finger status. If the finger is opened, then the value is true. Otherwise, the value is false.
`Little`	Describes a little finger status. If the finger is opened, then the value is true. Otherwise, the value is false.
`Angle`	Describes an angle of a hand gesture representing the counterclockwise rotation in the unit of 90°. The center of gravity of the hand area is placed at the original of the 2D plan, and the angle between the center axis of gravity of the hand gesture area and the positive horizontal line (zero degree) is measured as shown in Figure 12. The start point and end point of the center axis of gravity is the center of gravity and the figure side of hand, respectively. In this way, the zero degree is when the center axis of gravity is placed in the positive horizontal line and the degree is increased counterclockwise. The measured angle is quantized into one of the four regions as shown in Figure 12, then the value of the element of angle represents the corresponding region below. 0: 315 °~45° (white) / 1: 45°~135° (blue) / 2: 135°~215° (green) / 3: 215°~315° (yellow)

Name	*Definition*
	
	Figure 12 — An angle description of a hand gesture
`InnerContour`	Describes wether an inner contour exists or not in the given hand contour. If the value of `Innercontour` is "true", the given hand contour indicates the gesture of "okay".
`HandSide`	Describes the hand side in the user's point of view. If "true" then palm-side of hand, otherwise, back-side of hand.

8.6.4 Example

The thumb and index fingers, in which values are "true", are open and the remaining fingers, in which values are "false", are closed. The `Angle` 0 means the hand posture where the palm side is located between 315°~45°. And there is no inner contour because the `InnerContour value` is "false". Because the `HandSide` value is "true", the palm side is in the user's point of view.

```
<mtdl:analyzedData xsi:type="maov:HandPostureType" Thumb="true" Index="true"
Middle="false" Ring="false" Little="false" Angle="0" InnerContour="false"
HandSide="true"/>
```

8.7 IoMT hand gesture command generator

8.7.1 General

This subclause specifies a data format for a hand gesture command generator output. The `HandGestureCommandType` is used to represent information of a hand gesture command generated.

8.7.2 Syntax

```
<!-- ################################################ -->
<!-- Definition of Hand Gesture Command Generator Type                -->
<!-- ################################################ -->
<complexType name="HandGestureCommandType">
   <complexContent>
      <extension base="mtdl:AnalyzedDataBaseType">
         <sequence>
            <element name="HandGestureCommand" type="mpeg7:termReferenceType"/>
         </sequence>
      </extension>
   </complexContent>
</complexType>
```

8.7.3 Semantics

Table 81 — Semantics of the IoMT `hand gesture command generator` types

Name	*Definition*
`HandGestureCommandType`	Tool for describing an output of a hand gesture command generator.
`HandGestureCommand`	Describes the capability parameters of Hand gesture command generator detector. The type of the capability shall be described using the `mpeg7:termReferenceType` defined in ISO/IEC 15938-5:2003, 7.6. A classification scheme that may be used for this purpose is the `GestureCommandCS` defined in A.4.3.

8.7.4 Example

In this instance, the recognized hand posture in the hand gesture recognizer is mapped into corresponding specific command. In this example, the mapped command is "PLAY" something.

```
<mtdl:analyzedData xsi:type="maov:HandGestureCommandType">
   <maov:HandGestureCommandName>play</maov:HandGestureCommandName>
</mtdl:analyzedData>
```

8.8 IoMT healthcare information generator

8.8.1 General

This subclause specifies data format for a healthcare information generator output. The `HealthcareInfoType` is used to represent information of a healthcare information generated.

8.8.2 Syntax

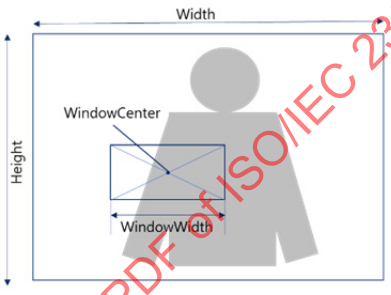
```
<!-- ################################################ -->
<!-- Definition of Healthcare Information Type        -->
<!-- ################################################ -->
<complexType name="HealthcareInfoType">
<complexContent>
```

```
    <extension base="mtdl:AnalyzedDataBaseType">
        <sequence>
            <element name="Patient" type="maov:PatientType"/>
            <element name="CommonMediaInfo" type="maov:CommonMediaInfoType"/>
            <choice>
                <element name="ImageInfo" type="maov:ImageInfoType"/>
                <element name="VideoInfo" type="maov:VideoInfoType"/>
            </choice>
        </sequence>
    </extension>
</complexContent>
</complexType>

<complexType name="PatientType">
    <all>
        <element name="PatientName" type="maov:PatientNameType" minOccurs="0"/>
        <element name="PatientAge" type="string" minOccurs="0"/>
        <element name="PatientSex" type="string" minOccurs="0"/>
    </all>
</complexType>

<complexType name="PatientNameType">
    <all>
        <element name="FamilyName" type="string" minOccurs="0"/>
        <element name="GivenName" type="string" minOccurs="0"/>
        <element name="MiddleName" type="string" minOccurs="0"/>
    </all>
</complexType>

<complexType name="CommonMediaInfoType">
    <all>
        <element name="BitDepth" type="decimal" minOccurs="0"/>
        <element name="Width" type="decimal" minOccurs="0"/>
        <element name="Height" type="decimal" minOccurs="0"/>
        <element name="SamplesPerPixel" type="decimal" minOccurs="0"/>
        <element name="PixelAspectRatioX" type="decimal" minOccurs="0"/>
        <element name="PixelAspectRatioY" type="decimal" minOccurs="0"/>
    </all>
</complexType>

<complexType name="ImageInfoType">
    <all>
        <element name="WindowWidth" type="decimal" minOccurs="0"/>
        <element name="WindowCenter" type="mpeg7:integerVector" minOccurs="0"/>
        <element name="NumberOfFrames" type="decimal" minOccurs="0"/>
    </all>
</complexType>

<complexType name="VideoInfoType">
    <all>
        <element name="VideoFormat" type="string" minOccurs="0"/>
        <element name="FrameRate" type="decimal" minOccurs="0"/>
    </all>
</complexType>
```

8.8.3 Semantics

Table 82 — Semantics of the IoMT `healthcare information generator` types

Name	*Definition*
`HealthcareInfoType`	Tool for describing the healthcare information related to a patient and image/video characteristics.
`Patient`	Describes the patient related information which is obtained from a speech recognizer.
`CommonMediaInfo`	Describes the common media related information on videos and images, which is obtained from image/video characteristics and/or is extracted from the incoming image/video.
`ImageInfo`	Describes the image information which is obtained from a hand gesture recognizer and/or is extracted from the incoming image.
`VideoInfo`	Describes the video information which is obtained from video characteristics and/or is extracted from the incoming video.
`PatientName`	Describes the full name of a patient including family name, given name, and middle name.
`PatientAge`	A patient's age.
`PatientSex`	A patient's sex.
`PatientNameType`	Provides the information of patient name.
`FamilyName`	A patient's family name.
`GivenName`	A patient's given name.
`MiddleName`	A patient's middle name.
`CommonMediaInfoType`	Tool for describing the common information on image/video characteristics.
`BitDepth`	Describes the number of bits allocated to each sample of pixel. Each sample shall have the same number of bits to represent its value.
`Width`	Describes the width of image/video in the unit of pixel.
`Height`	Describes the height of image/video in the unit of pixel.
`SamplesPerPixel`	Describes the number of color component per pixel. For monochrome image/video, the number of component is 1. For RGB and other three-color models such as YUV and YCbCr, the value of this element is 3.
`PixelAspectRatioX`	Describes the ratio of the horizontal size of the pixels in the image specified by integer value.

Name	*Definition*
PixelAspectRatioY	Describes the ratio of the vertical size of the pixels in the image specified by integer value.
ImageInfoType	Provides an abstract of information on image characteristics.
WindowWidth	Window width specifies the width of image to be displayed in the unit of pixel.
WindowCenter	Window center specifies the center position of image to be displayed as a 2D coordinate (Figure 13). The position of (0, 0) is the top-left pixel position in an image. The first value represents X coordination and the last value represents Y coordination.  **Figure 13 — A window center**
NumberOfFrames	Number of frames in a multi-frame image which is a series of images with a common image header. The value of this element is generated by a speech recognizer.
VideoInfoType	Provides an abstract of information on video characteristics.
VideoFormat	Describes the compression format of a video.
FrameRate	Describes the number of frames per second.

8.8.4 Examples

In this instance, a patient's family name is `Chun` and his given name is `Sungmoon`. The gender of the patient is `male`. He is `40` years old. This example presents the information of image. It needs `8-bit` depth in which the pixel valu is available in the range of 0 ~ 255. The image resolution with the width by height is `1920x1080`. Pixel aspect ratio is `1:1` which means a square pixel. The window `width` and the `center` mean a capturing region of which the width is `1000` pixel and the center position is `(500, 250)`. And only one picture is ready to be transmitted.

Example A: image

```
<mtdl:analyzedData xsi:type="maov:HealthcareInfoType">
  <maov:Patient>
    <maov:PatientName>
      <maov:FamilyName>Chun</maov:FamilyName>
      <maov:GivenName>Sungmoon</maov:GivenName>
```

```
    </maov:PatientName>
      <maov:PatientAge>40</maov:PatientAge>
      <maov:PatientSex>Male</maov:PatientSex>
  </maov:Patient>
  <maov:CommonMediaInfo>
    <maov:BitDepth>8</maov:BitDepth>
    <maov:Width>1920</maov:Width>
    <maov:Height>1080</maov:Height>
    <maov:SamplesPerPixel>3</maov:SamplesPerPixel>
    <maov:PixelAspectRatioX>1</maov:PixelAspectRatioX>
    <maov:PixelAspectRatioY>1</maov:PixelAspectRatioY>
  </maov:CommonMediaInfo>
  <maov:ImageInfo>
    <maov:WindowWidth>1000</maov:WindowWidth>
    <maov:WindowCenter>500 250</maov:WindowCenter>
    <maov:NumberOfFrames>1</maov:NumberOfFrames>
  </maov:ImageInfo>
</mtdl:analyzedData>
```

In this instance, a patient's family name is `Chun` and his given name is `Sungmoon`. The gender of the patient is a `male`. He is `40` years old. This example presentsthe information of `image`. It needs `8-bit` depth in which the pixel valus is available in the range of 0 ~ 255. The image resolution with width by height is `1920x1080`. Pixel aspect ratio is `1:1` which means a square pixel. This video is compressed by using `MPEG-4 AVC`. The frame rate is `60Hz`.

Example B: video

```
<mtdl:analyzedData xsi:type="maov:HealthcareInfoType">
  <maov:Patient>
    <maov:PatientName>
      <maov:FamilyName>Chun</maov:FamilyName>
      <maov:GivenName>Sungmoon</maov:GivenName>
    </maov:PatientName>
    <maov:PatientAge>40</maov:PatientAge>
    <maov:PatientSex>Male</maov:PatientSex>
  </maov:Patient>
  <maov:CommonMediaInfo>
    <maov:BitDepth>8</maov:BitDepth>
    <maov:Width>1920</maov:Width>
    <maov:Height>1080</maov:Height>
    <maov:SamplesPerPixel>3</maov:SamplesPerPixel>
    <maov:PixelAspectRatioX>1</maov:PixelAspectRatioX>
    <maov:PixelAspectRatioY>1</maov:PixelAspectRatioY>
  </maov:CommonMediaInfo>
  <maov:VideoInfo>
    <maov:VideoFormat>MPEG-4 AVC</maov:VideoFormat>
    <maov:FrameRate>60</maov:FrameRate>
  </maov:VideoInfo>
</mtdl:analyzedData>
```

8.9 IoMT odor image to scent converter

8.9.1 General

This subclause specifies data format for an odor image to scent converter output. The `odorImageType` is used to represent the recognized odor image.

8.9.2 Syntax

```
<!-- ################################################ -->
<!-- Definition of Odor Image to Scent Converter      -->
<!-- ################################################ -->
<complexType name="odorImageType">
   <complexContent>
      <extension base="mtdl:AnalyzedDataBaseType">
         <sequence>
            <element name="odorImageOutputList"
type="maov:odorImageOutputListType"/>
         </sequence>
      </extension>
   </complexContent>
</complexType>

<complexType name="odorImageOutputListType">
   <sequence>
      <element name="odorImageOutput" type="maov:odorImageOutputType"
maxOccurs="unbounded"/>
   </sequence>
</complexType>

<complexType name="odorImageOutputType">
   <attribute name="odorImageLabel" type="mpeg7:termReferenceType"
use="optional"/>
   <attribute name="odorImageCharacteristics" type="mpeg7:termReferenceType"
use="optional"/>
   <attribute name="confidenceLevel" type="unsignedInt" use="optional"/>
   <attribute name="confidenceLevelofCharacterlistics" type="unsignedInt"
use="optional"/>
</complexType>
```


8.9.3 Semantics

Table 83 — Semantics of the IoMT `odor image to scent converter` types

Name	*Definition*
`odorImageType`	Tool for describing the output of the "IoMT odor image to scent converter".
`odorImageOutputList`	Describes the output list of the "IoMT odor image to scent converter".
`odorImageOutputListType`	Tool for describing the output list of the "IoMT odor image to scent converter".

Name	*Definition*
odorImageOutput	Describes the output of the "IoMT odor image to scent converter".
odorImageOutputType	Tool for describing the scent effect in the odor image recognized by the "IoMT odor image to scent converter".
odorImageLabel	Describes the recognized scent effect's label in the odor image. The type shall be described using the mpeg7:termReferenceType defined in ISO/IEC 15938-5:2003, 7.6. A classification scheme that may be used for this purpose is the ScentCS defined in ISO/IEC 23005-6.
odorImageCharacteristics	Describes the recognized scent effect's characteristics in the odor image. The type shall be described using the mpeg7:termReferenceType defined in ISO/IEC 15938-5:2003, 7.6. A classification scheme that may be used for this purpose is the OdorImageCharacteristicsCS defined in A.4.1.
confidenceLevel	Describes the confidence level of the scent effect's label recognized by the "IoMT odor image to scent converter".
confidenceLevelofCharacterlistics	Describes the confidence level of the scent effect's characteristics recognized by the "IoMT odor image to scent converter"

8.9.4 Example

This example shows the analysis result of the "IoMT odor image to scent converter". The result consists of two outputs with four attributes: odorImageLabel, odorImageCharacteritics, confidenceLevel and confidenceLevelCharacteristics. The first output of the result is odorImageLabel "coffee_cream", odorImageCharacteristics "DRY", confidenceLevel "60" and confidenceLevelCharacteristics "50". The second output of the result is odorImageLabel "orange" and confidenceLevel "0".

```
<mtdl:analyzedData xsi:type="maov:odorImageType">
    <maov:odorImageOutputList>
        <maov:odorImageOutput>
            <maov:odorImageLabel>coffee_cream</maov:odorImageLabel>
            <maov:odorImageCharacterlistics>DRY</maov:odorImageCharacterlistics>
            <maov:confidenceLevel>60</maov:confidenceLevel>
            <maov:confidenceLevelofCharacterlistics>50</maov:confidenceLevelofCharacterlist
ics>
        </maov:odorImageOutput>
        <maov:odorImageOutput>
            <maov:odorImageLabel>orange</maov:odorImageLabel>
            <maov:confidenceLevel>0</maov:confidenceLevel>
        </maov:odorImageOutput>
    </maov:odorImageOutputList>
</mtdl:analyzedData>
```

8.10 IoMT question analyzer

8.10.1 General

This subclause specifies data formats to describe the outputs that can be produced by the IoMT QuestionAnalyzer. The QuestionAnalyzer takes a user's question in text as an input and produces a

question analysis result. The result of question analysis can be used as an input for generating user interaction commands as well as input to the Question Answering (QA) Server to provide answers/information to the user. The UserQuestionType is used to represent the analyzed user's question, for which answers can be provided by QA service. If it is analyzed as a control command, it is sent to the actuator.

8.10.2 Syntax

```
<!-- ################################################ -->
<!-- Definition of Question Analysis Type             -->
<!-- ################################################ -->
<complexType name="QuestionAnalysisType">
    <complexContent>
        <extension base="mtdl:AnalyzedDataBaseType">
            <sequence>
                <element name="analyzedQuestion"
type="maov:UserQuestionType" minOccurs="0"/>
                <element name="language" type="language" minOccurs="0"/>
            </sequence>
        </extension>
    </complexContent>
</complexType>

<complexType name="UserQuestionType">
    <sequence>
        <element name="qtopic" type="string" minOccurs="0"/>
        <element name="qfocus" type="mpeg7:termReferenceType" minOccurs="0"/>
        <element name="qCsemantic" type="mpeg7:termReferenceType" minOccurs="0"/>
    </sequence>
    <attribute name="qdomain" type="string"/>
</complexType>
```

8.10.3 Semantics

Table 84 — Semantics of the IoMT `question analyzer` types

Name	*Definition*
`QuestionAnalysisType`	Provides an abstract of description of question analysis, which can be done in a processing unit.
`anlyzedQuestion`	Describes analyzed question resulting from the question analysis.
`language`	Indicates the language of the input question. If present, the `Language` element should take precedence over other language indications present within the input question.
`UserQuestionType`	Provides abstracts of User Question description. Describes user's utterance that is the output of speech recognition process. User Question is sent to QA server for providing answers to the user. If it is a control command, it is sent to the actuator.

Name	*Definition*
qtopic	Describes topic of the question. Question topic is the object or event that the question is about. Ex. Qtopic is King Lear in "Who is the author of King Lear?".
qfocus	Describes the focus of the question, which is one of 5W1H. The type of the focus shall be described using the mpeg7:termReferenceType defined in ISO/IEC 15938-5:2003, 7.6. A classification scheme that may be used for this purpose is the QfocusCS defined in A.4.4. Ex. What, where, who, what policy.
qCsemantic	Describes the question classification based on the meaning/purpose of the question. The type of the question classification shall be described using the mpeg7:termReferenceType defined in ISO/IEC 15938-5:2003, 7.6. A classification scheme that may be used for this purpose is the QCsemanticCS defined in A.4.5. Ex. What does MPEG stand for? (Request for terminology). Could you please turn on the TV? (Request for command)
qdomain	Describes the domain of the question such as "science", "weather", "history". Ex. Who is the third king of Yi dynasty in Korea? (qdomain: history)

8.10.4 Examples

This example shows the question analysis result of the user's question, "Who is the author of King Lear?" The result consists of analyzedQuestion and language, "en-us". The question analysis result in the example shows that the domain of the question is "Literature", the topic of the question is "King Lear", the focus of the question is "WHO_QUESTION" and the purpose of the question is to "REQUEST_FOR_INFORMATION"

```
<mtdl:analyzedData xsi:type="maov:QuestionAnalysisType">
   <maov:analyzedQuestion qdomain="Literature">
      <maov:qtopic>King Lear</maov:qtopic>
      <maov:qfocus>WHO_QUESTION</maov:qfocus>
      <maov:qCsemantic>REQUEST_FOR_INFORMATION</maov:qCsemantic>
   </maov:analyzedQuestion>
   <maov:language>en-us</maov:language>
</mtdl:analyzedData>
```

This example shows the result of the analyzed question of "How do you make Kimchi?" The question analysis result in the example shows that the domain of the question is "Cooking", the topic of the question is "Kimchi", the focus of the question is "HOW_QUESTION" and the purpose of the question is to "REQUEST_FOR_METHOD".

```
<mtdl:analyzedData xsi:type="maov:QuestionAnalysisType">
   <maov:analyzedQuestion qdomain="Cooking">
      <maov:qtopic>Kimchi</maov:qtopic>
      <maov:qfocus>HOW_QUESTION</maov:qfocus>
```

```
        <maov:qCsemantic>REQUEST_FOR_METHOD</maov:qCsemantic>
    </maov:analyzedQuestion>
    <maov:language>en-us</maov:language>
</mtdl:analyzedData>
```

8.11 IoMT music frequency analyzer

8.11.1 General

This subclause specifies data format for a MusicFrequencyAnalyzer analyzer output. The `MusicCharacterType` is used to represent information of analysed music frequency.

8.11.2 Syntax

```
<!-- ############################################-->
<!-- Definition of Analyzed Music Frequency Type    -->
<!-- ############################################-->

<complexType name="AnalyzedMusicFrequencyType">
<complexContent>
    <extension base="mtdl:AnalyzedDataBaseType">
        <sequence>
            <element name="AudioSamplingRate" type="nonNegativeInteger"/>
            <element name="NumberOfPoints" type="nonNegativeInteger"/>
            <element name="Spectrogram" type="maov:SpectrogramType" maxOccurs="1"
minOccurs="1"/>
        </sequence>
    </extension>
</complexContent>
</complexType>

<complexType name="SpectrogramType">
    <sequence>
        <element name="SpectroPoint" type="maov:SpectroPointType"
maxOccurs="unbounded"/>
    </sequence>
</complexType>

<complexType name="SpectroPointType">
    <sequence>
        <element name="FreqAmplitude" type="nonNegativefloat"/>
        <element name="PointNumber" type="integer"/>
    </sequence>
</complexType>
```

8.11.3 Semantics

Table 85 — Semantics of the IoMT `music frequency analyzer` types

Name	*Definition*
`AnalyzedMusicFrequencyType`	Tool for describing analyzed music frequency.
`AudioSamplingRate`	Describes the audio sampling rate of the audio source.
`NumberOfPoints`	Describes the number of samples to be analysed at one time.
`SpectrogramType`	Tool for describing the result of the Fourier analysis of the signal or audio.
`SpectroPoint`	Describes the amplitude value and the count identifying a particular frequency which is corresponding to the amplitude.
`SpectroPointType`	Tool for describing the result of Fourier analysis result of the signal or audio.
`FreqAmplitude`	Describes the amplitude of a particular frequency of each point acquired by the Fourier analysis.
`PointNumber`	Describes the number identifying a particular frequency for which the `FreqAmplitude` is given.

8.11.4 Examples

This example shows that sampling rate of an audio source is 44100 Hz and the bandwidth will be 22050Hz by the Nyquist theorem. This given bandwidth will be divided into 4 bands and the range of each band will be 5512.5Hz. The `FreqAmplitude` means the frequency amplitude of each band, so the `FreqAmplitude` of the first band is 337.5, the `FreqAmplitude` of the second band is 912.5, the `FreqAmplitude` of the third band is 33, and the `FreqAmplitude` of the last(highest) band is 12.5.

```
<mtdl:analyzedData xsi:type="maov:AnalyzedMusicFrequencyType">
    <maov:AudioSamplingRate>44100</maov:AudioSamplingRate>
    <maov:NumberOfPoints>8</maov:NumberOfPoints>
    <maov:Spectrogram>
        <maov:SpectroPoint>
            <maov:FreqAmplitude>337.5</maov:FreqAmplitude>
            <maov:PointNumber>0</maov:PointNumber>
        </maov:SpectroPoint>
        <maov:SpectroPoint>
            <maov:FreqAmplitude>912.5</maov:FreqAmplitude>
            <maov:PointNumber>1</maov:PointNumber>
        </maov:SpectroPoint>
        <maov:SpectroPoint>
            <maov:FreqAmplitude>33</maov:FreqAmplitude>
            <maov:PointNumber>2</maov:PointNumber>
        </maov:SpectroPoint>
        <maov:SpectroPoint>
            <maov:FreqAmplitude>12.5</maov:FreqAmplitude>
```

```
            <maov:PointNumber>3</maov:PointNumber>
        </maov:SpectroPoint>
    </maov:Spectrogram>
</mtdl:analyzedData>
```

8.12 IoMT video content class generator

8.12.1 General

This subclause specifies data formats for a video content class generator output. The `VideoContentClassType` is used to represent information of analyzed video content class.

8.12.2 Syntax

```
<!-- ################################################ -->
<!-- Definition of Video Content Class Generator Type -->
<!-- ################################################ -->

<complexType name="VideoContentClassType">
    <complexContent>
        <extension base="mtdl:AnalyzedDataBaseType">
            <sequence>
                <element name="VideoContentClass" type="mpeg7:termReferenceType"/>
            </sequence>
        </extension>
    </complexContent>
</complexType>
```

8.12.3 Semantics

Table 86 — Semantics of the IoMT `video content class generator types`

Name	*Definition*
`VideoContentClassType`	Tool for describing generated video content class.
`VideoContentClass`	Describes the analyzed content class or genre of a video source. The name of the specific video class or genre shall be described using the `mpeg7:termReferenceType` defined in ISO/IEC 15938-5:2003, 7.6. A classification scheme that may be used for this purpose is the `VideoContentClassCS` defined in A.4.6.

8.12.4 Examples

This example shows that generated video content class is sports which is analyzed by VideoContentClassGenerator.

```
<mtdl:analyzedData xsi:type="maov:VideoContentClassType">
    <maov:VideoContentClass>Sports</maov:VideoContentClass>
</mtdl:analyzedData>
```

Annex A
(normative)

Classification scheme

A.1 MThingTypeCS

```
<Term termId="MSensor">
    <Name xml:lang="en">Media Sensor</Name>
    <Definition xml:lang="en">
       An MThing capable of sensing media
    </Definition>
</Term>

<Term termId="MActuator">
    <Name xml:lang="en">Media Actuator</Name>
    <Definition xml:lang="en">
       An MThing capable of actuating, rendering, and/or displaying media and
related metadata
    </Definition>
</Term>

<Term termId="MAnalyzer">
    <Name xml:lang="en">Media Analyzer</Name>
    <Definition xml:lang="en">
       An MThing capable of analyzing media, and producing related media and/or
metadata
    </Definition>
</Term>

<Term termId="MStorage">
    <Name xml:lang="en">Media Storage</Name>
    <Definition xml:lang="en">
       An MThing capable of storing media or metadata
    </Definition>
</Term>

<Term termId="MManager">
    <Name xml:lang="en">MThing Manager</Name>
    <Definition xml:lang="en">
       An MThing capable of managing a list of MThings
    </Definition>
</Term>

<Term termId="MAggregator">
    <Name xml:lang="en">MThing Aggregator</Name>
    <Definition xml:lang="en">
       An MThing composed of two or more sub-MThings
    </Definition>
</Term>
```

Table A.1 — Binary representation of `MThingTypeCS`

Binary representation	Term ID of MThingTypeCS
00000000	MSensor
00000001	MActuator
00000010	MAnalyzer
00000011	MStorage
00000100	MManager
00000101	MAggregator
00000110 – 11111111	Reserved

A.2 CapabilityCS

A.2.1 SensorCapabilityCS

```
<Term termId="SENSOR_CAPTURE_VIDEO">
<Name xml:lang="en">CAPTURE_VIDEO</Name>
<Definition xml:lang="en">
   An MSensor is capable of captureing a video
</Definition>
</Term>

<Term termId="SENSOR_CAPTURE_AUDIO">
    <Name xml:lang="en">CAPTURE_AUDIO</Name>
    <Definition xml:lang="en">
       An MSensor is capable of Capturing an audio
    </Definition>
</Term>

<Term termId="SENSOR_STREAM_VIDEO">
<Name xml:lang="en">STREAM_VIDEO</Name>
    <Definition xml:lang="en">
       An MSensor is capable of streaming a video
    </Definition>
</Term>

<Term termId="SENSOR_STREAM_STEREO_VIDEO">
<Name xml:lang="en">STREAM_STEREO_VIDEO</Name>
    <Definition xml:lang="en">
       An MSensor is capable of streaming a stereo video
    </Definition>
</Term>

<Term termId="SENSOR_STREAM_AUDIO">
    <Name xml:lang="en">STREAM_AUDIO</Name>
    <Definition xml:lang="en">
       An MSensor is capable of streaming an audio
    </Definition>
</Term>
```

```
<Term termId="SENSOR_MEASURE_DISTANCE">
    <Name xml:lang="en">MEASURE_DISTANCE</Name>
    <Definition xml:lang="en">
       An MSensor is capable of measuring a distance
    </Definition>
</Term>

  <Term termId="SENSOR_MEASURE_LATITUDE">
    <Name xml:lang="en">MEASURE_LATITUDE</Name>
    <Definition xml:lang="en">
       An MSensor is capable of measuring a latitude
    </Definition>
</Term>

  <Term termId="SENSOR_MEASURE_LONGITUDE">
    <Name xml:lang="en">MEASURE_LONGITUDE</Name>
    <Definition xml:lang="en">
       An MSensor is capable of measuring a longitude
    </Definition>
</Term>

<Term termId="SENSOR_MEASURE_AZIMUTH">
<Name xml:lang="en">MEASURE_AZIMUTH</Name>
   <Definition xml:lang="en">
       An MSensor is capable of measuring an azimuth
    </Definition>
</Term>

<Term termId="SENSOR_MEASURE_YAW_ANGLE">
    <Name xml:lang="en">MEASURE_YAW_ANGLE</Name>
    <Definition xml:lang="en">
       An MSensor is capable of measuring a pitch angle
    </Definition>
</Term>

<Term termId="SENSOR_MEASURE_PITCH_ANGLE">
    <Name xml:lang="en">MEASURE_PITCH_ANGLE</Name>
    <Definition xml:lang="en">
       An MSensor is capable of measuring a pitch angle
    </Definition>
</Term>

<Term termId="SENSOR_MEASURE_ROLL_ANGLE">
    <Name xml:lang="en">MEASURE_ROLL_ANGLE</Name>
    <Definition xml:lang="en">
       An MSensor is capable of measuring a roll angle
    </Definition>
</Term>

<Term termId="SENSOR_CAPTURE_STEREO_VIDEO">
    <Name xml:lang="en">CAPTURE_STEREO_VIDEO</Name>
    <Definition xml:lang="en">
       An MSensor is capable of capturing a stereo video
</Term>

<Term termId="SENSOR_CAPTURE_IMAGE">
    <Name xml:lang="en">CAPTURE_IMAGE</Name>
    <Definition xml:lang="en">
       An MSensor is capable of capture an image
    </Definition>
```

```
</Term>

<Term termId="SENSOR_MEASURE_GLOBAL_POSITION">
    <Name xml:lang="en">MEASURE_GLOBAL_POSITION</Name>
    <Definition xml:lang="en">
      An Msensor is capable of measuring the global position
    </Definition>
</Term>
```

Table A.2 — Binary representation of `SensorCapabilityCS`

Binary representation	Term ID of SensorCapabilityCS
00000000	SENSOR_CAPTURE_VIDEO
00000001	SENSOR_CAPTURE_AUDIO
00000010	SENSOR_STREAM_VIDEO
00000011	SENSOR_STREAM_STEREO_VIDEO
00000100	SENSOR_STREAM_AUDIO
00000101	SENSOR_MEASURE_DISTANCE
00000110	SENSOR_MEASURE_LATITUDE
00000111	SENSOR_MEASURE_LONGITUDE
00001000	SENSOR_MEASURE_AZIMUTH
00001001	SENSOR_MEASURE_YAW_ANGLE
00001010	SENSOR_MEASURE_PITCH_ANGLE
00001011	SENSOR_MEASURE_ROLL_ANGLE
00001100	SENSOR_CAPTURE_STEREO_VIDEO
00001101	SENSOR_CAPTURE_IMAGE
00001110	SENSOR_MEASURE_GLOBAL_POSITION
00001111 - 11111111	Reserved

A.2.2 ActuatorCapabilityCS

```
<Term termId="ACTUATOR_VIBRATE">
    <Name xml:lang="en">VIBRATE</Name>
    <Definition xml:lang="en">
       An MActuator is capable of vibrating
    </Definition>
</Term>
```

```
<Term termId="ACTUATOR_PLAY_AUDIO">
    <Name xml:lang="en">PLAY_AUDIO</Name>
    <Definition xml:lang="en">
       An MActuator is capable of play an audio
    </Definition>
</Term>

<Term termId="ACTUATOR_CHANGE_COLOR_TEMPERATURE">
    <Name xml:lang="en">CHANGE_COLOR_TEMPERATURE</Name>
    <Definition xml:lang="en">
       An MActuator is capable of changing the color temperature
    </Definition>
</Term>

<Term termId="ACTUATOR_CHANGE_SATURATION">
    <Name xml:lang="en">CHANGE_SATURATION</Name>
    <Definition xml:lang="en">
       An MActuator is capable of changing the saturation
    </Definition>
</Term>

<Term termId="ACTUATOR_CHANGE_BRIGHTNESS">
    <Name xml:lang="en">CHANGE_BRIGHTNESS</Name>
    <Definition xml:lang="en">
       An MActuator is capable of changing the brightness
    </Definition>
</Term>

<Term termId="ACTUATOR_CHANGE_HUE">
    <Name xml:lang="en">CHANGE_HUE</Name>
    <Definition xml:lang="en">
       An MActuator is capable of changing the hue
    </Definition>
</Term>

<Term termId="ACTUATOR_CHANGE_VOLUME">
    <Name xml:lang="en">CHANGE_VOLUME</Name>
    <Definition xml:lang="en">
       An MActuator is capable of changing the volume
    </Definition>
</Term>

<Term termId="ACTUATOR_PLAY_VIDEO">
    <Name xml:lang="en">PLAY_VIDEO</Name>
    <Definition xml:lang="en">
       An MActuator is capable of playing a video
    </Definition>
</Term>

<Term termId="ACTUATOR_CHANGE_ZOOM">
    <Name xml:lang="en">CHANGE_ZOOM</Name>
    <Definition xml:lang="en">
       An MActuator is capable of zooming
    </Definition>
</Term>

<Term termId="ACTUATOR_CHANGE_RESOLUTION">
    <Name xml:lang="en">CHANGE_RESOLUTION</Name>
    <Definition xml:lang="en">
       An MActuator is capable of changing resolutions
    </Definition>
```

```
</Term>

<Term termId="ACTUATOR_CHANGE_ORIENTATION">
   <Name xml:lang="en">CHANGE_ORIENTATION</Name>
    <Definition xml:lang="en">
       An MActuator is capable of changing the orientation
    </Definition>
</Term>

<Term termId="ACTUATOR_REWIND">
    <Name xml:lang="en">REWIND</Name>
    <Definition xml:lang="en">
       An MActuator is capable of rewinding a video or an audio
    </Definition>
</Term>

<Term termId="ACTUATOR_FASTFORWARD">
    <Name xml:lang="en">FASTFORWARD</Name>
    <Definition xml:lang="en">
       An MActuator is capable of fastforwarding a video or an audio
    </Definition>
</Term>
<Term termId="ACTUATOR_LIGHTING">
    <Name xml:lang="en">LIGHTING</Name>
    <Definition xml:lang="en">
       An MActuator is capable of turning on/off a light
   </Definition>
</Term>

<Term termId="ACTUATOR_ARRAYED_LIGHTING">
    <Name xml:lang="en">ARRAYED_LIGHTING</Name>
    <Definition xml:lang="en">
       An MActuator is capable of turning on/off an arrayed light
    </Definition>
</Term>

<Term termId="ACTUATOR_SPRAY_WATER">
    <Name xml:lang="en">SPRAY_WATER</Name>
    <Definition xml:lang="en">
       An MActuator is capable of spraying water
    </Definition>
</Term>

<Term termId="ACTUATOR_SPRAY_SCENT">
    <Name xml:lang="en">SPRAY_SCENT</Name>
    <Definition xml:lang="en">
       An MActuator is capable of spraying scent
    </Definition>
</Term>

<Term termId="ACTUATOR_SPRAY_FOG">
    <Name xml:lang="en">SPRAY_FOG</Name>
    <Definition xml:lang="en">
       An MActuator is capable of spraying fog
    </Definition>
</Term>

<Term termId="ACTUATOR_SPRAY_BUBBLE">
    <Name xml:lang="en">SPRAY_BUBBLE</Name>
    <Definition xml:lang="en">
       An MActuator is capable of spraying bubble
```

```
    </Definition>
</Term>

<Term termId="ACTUATOR_DISPLAY_TRANSPARENT">
    <Name xml:lang="en">DISPLAY_TRANSPARENT</Name>
    <Definition xml:lang="en">
       A transparent display which can see-through
    </Definition>
</Term>

<Term termId="ACTUATOR_DISPLAY_OPAQUE">
    <Name xml:lang="en">DISPLAY_OPAQUE</Name>
    <Definition xml:lang="en">
       An opaque display which can't support the see-through
    </Definition>
</Term>
```

Table A.3 — Binary representation of `ActuatorCapabilityCS`

Binary representation	Term ID of ActuatorCapabilityCS
00000000	ACTUATOR_VIBRATE
00000001	ACTUATOR_PLAY_AUDIO
00000010	ACTUATOR_CHANGE_COLOR_TEMPERATURE
00000011	ACTUATOR_CHANGE_SATURATION
00000100	ACTUATOR_CHANGE_BRIGHTNESS
00000101	ACTUATOR_CHANGE_HUE
00000110	ACTUATOR_CHANGE_VOLUME
00000111	ACTUATOR_PLAY_VIDEO
00001000	ACTUATOR_CHANGE_ZOOM
00001001	ACTUATOR_CHANGE_RESOLUTION
00001010	ACTUATOR_CHANGE_ORIENTATION
00001011	ACTUATOR_REWIND
00001100	ACTUATOR_FASTFORWARD
00001101	ACTUATOR_LIGHTING
00001110	ACTUATOR_ARRAYED_LIGHTING
00001111	ACTUATOR_SPRAY_WATER
00010000	ACTUATOR_SPRAY_SCENT
00010001	ACTUATOR_SPARY_FOG

Binary representation	Term ID of ActuatorCapabilityCS
00010010	ACTUATOR_SPRAY_BUBBLE
00010011	ACTUATOR _DISPLAY_TRANSPARENT
00010100	ACTUATOR _DISPLAY_OPAQUE
00010101 - 11111111	Reserved

A.2.3 AnalyzerCapabilityCS

```
<Term termId="ANALYZER_DETECT_SOCIAL_EVENT">
    <Name xml:lang="en">DETECT_SOCIAL_EVENT</Name>
    <Definition xml:lang="en">
       An MAnalyzer is capable of detecting a social event in a video
    </Definition>
</Term>

<Term termId="ANALYZER_DETECT_HAND_GESTURE">
    <Name xml:lang="en">DETECT_HAND_GESTURE</Name>
    <Definition xml:lang="en">
       An MAnalyzer is capable of detecting hand gestures
    </Definition>
</Term>

<Term termId="ANALYZER_RECOGNIZE_HAND_GESTURE">
    <Name xml:lang="en">RECOGNIZE_HAND_GESTURE</Name>
    <Definition xml:lang="en">
       An MAnalyzer is capable of recognizing hand gestures
    </Definition>
</Term>

<Term termId="ANALYZER_MAP_HAND_GESTURE_COMMAND">
    <Name xml:lang="en">MAP_HAND_GESTURE_COMMAND</Name>
    <Definition xml:lang="en">
       An MAnalyzer is capable of mapping a hand gesture to the corresponding
command
    </Definition>
</Term>

<Term termId="ANALYZER_RECOGNIZE_SPEECH">
    <Name xml:lang="en">RECOGNIZE_SPEECH</Name>
    <Definition xml:lang="en">
       An MAnalyzer is capable of recognizing a speech
    </Definition>
</Term>

<Term termId="ANALYZER_ANALYZE_QUESTION">
    <Name xml:lang="en">ANALYZE_QUESTION</Name>
    <Definition xml:lang="en">
       An MAnalyzer is capable of analyse questions
    </Definition>
</Term>

<Term termId="ANALYZER_RECOGNIZE_SCENT_IMAGE">
    <Name xml:lang="en">RECOGNIZE_SCENT_IMAGE</Name>
```

```
    <Definition xml:lang="en">
       An MAnalyzer is capable of recognizing scent objects or scenes from an
image
    </Definition>
</Term>

<Term termId="ANALYZER_CONVERT_TEXT_TO_SPEECH">
    <Name xml:lang="en">CONVERT_TEXT_TO_SPEECH</Name>
    <Definition xml:lang="en">
       An MAnalyzer is capable of converting text to speech
    </Definition>
</Term>

<Term termId="ANALYZER_DETECT_COLLISION">
    <Name xml:lang="en">DETECT_COLLISION</Name>
    <Definition xml:lang="en">
       An MAnalyzer is capable of detecting collisions
    </Definition>
</Term>

<Term termId="ANALYZER_GUIDE_DIRECTION">
    <Name xml:lang="en">GUIDE_DIRECTION</Name>
    <Definition xml:lang="en">
       An MAnalyzer is capable of guiding directions
    </Definition>
</Term>

<Term termId="ANALYZER_GENERATE_VIDEO_CONTENT_CLASS">
    <Name xml:lang="en">GENERATE_VIDEO_CONTENT_CLASS</Name>
    <Definition xml:lang="en">
       An MAnalyzer is capable of generating a video contrent class from a video
    </Definition>
</Term>

<Term termId="ANALYZER_GENERATE_HEALTHCARE_INFO">
    <Name xml:lang="en">GENERATE_HEALTHCARE_INFO</Name>
    <Definition xml:lang="en">
       An MAnalyzer is capable of generating healthcare information
    </Definition>
</Term>
```


Table A.4 — Binary representation of `AnalyzerCapabilityCS`

Binary representation	Term ID of AnalyzerCapabilityCS
00000000	ANALYZER_DETECT_SOCIAL_EVENT
00000001	ANALYZER_DETECT_HAND_GESTURE
00000010	ANALYZER_RECOGNIZE_HAND_GESTURE
00000011	ANALYZER_MAP_HAND_GESTURE_COMMAND
00000100	ANALYZER_RECOGNIZE_SPEECH
00000101	ANALYZER_ANALYZE_QUESTION

Binary representation	Term ID of AnalyzerCapabilityCS
00000110	ANALYZER_RECOGNIZE_SCENT_IMAGE
00000111	ANALYZER_CONVERT_TEXT_TO_SPEECH
00001000	ANALYZER_DETECT_COLLISION
00001001	ANALYZER_GUIDE_DIRECTION
00001010	ANALYZER_GENERATE_VIDEO_CONTENT_CLASS
00001011	ANALYZER_GENERATE_HEALTHCARE_INFO
00001100 - 11111111	Reserved

A.2.4 StorageCapabilityCS

```
<Term termId="STORAGE_SAVE">
    <Name xml:lang="en">STORAGE_SAVE</Name>
    <Definition xml:lang="en">
       An MStorage is capable of saving data
    </Definition>
</Term>

<Term termId="STORAGE_READ">
    <Name xml:lang="en">STORAGE_READ</Name>
    <Definition xml:lang="en">
       An MStorage is capable of reading data
    </Definition>
</Term>

<Term termId="STORAGE_DELETE">
    <Name xml:lang="en">STORAGE_DELETE</Name>
    <Definition xml:lang="en">
       An MStorage is capable of deleting data
    </Definition>
</Term>

<Term termId="STORAGE_UPDATE">
    <Name xml:lang="en">STORAGE_UPDATE</Name>
    <Definition xml:lang="en">
       An MStorage is capable of updating data
    </Definition>
</Term>
```

Table A.5 — Binary representation of `StorageCapabilityCS`

Binary representation	Term ID of StorageCapabilityCS
00000000	STORAGE_SAVE
00000001	STORAGE_READ
00000010	STORAGE_DELETE
00000011	STORAGE_UPDATE
00000100 - 11111111	Reserved

A.2.5 ManagerCapabilityCS

```
<Term termId="MANAGER_RETRIVE_MTHINGS_BY_CAPABILITY">
    <Name xml:lang="en">RETRIVE_MTHINGS_BY_CAPABILITY</Name>
    <Definition xml:lang="en">
       An MManager is capable of searching MThings by capability
    </Definition>
</Term>

<Term termId="MANAGER_RETRIVE_MTHINGS_BY_TOKEN">
    <Name xml:lang="en">RETRIVE_MTHINGS_BY_TOKEN</Name>
    <Definition xml:lang="en">
       An MManager is capable of searching MThings by token
    </Definition>
</Term>

<Term termId="MANAGER_REGISTER_MTHING">
    <Name xml:lang="en">REGISTER_MTHING</Name>
    <Definition xml:lang="en">
       An MManager is capable of registering an MThing
    </Definition>
</Term>

<Term termId="MANAGER_UNREGISTER_MTHING">
    <Name xml:lang="en">UNREGISTER_MTHING</Name>
    <Definition xml:lang="en">
       An MManager is capable of unregistering an MThing
    </Definition>
</Term>

<Term termId="MANAGER_SHOW_REGISTERED_MTHINGS">
    <Name xml:lang="en">SHOW_REGISTERED_MTHINGS</Name>
    <Definition xml:lang="en">
       An MManager is capable of showing registered MThings
    </Definition>
</Term>
```

Table A.6 — Binary representation of `ManagerCapabilityCS`

Binary representation	Term ID of ManagerCapabilityCS
00000000	MANAGER_RETRIVE_MTHINGS_BY_CAPABILITY
00000001	MANAGER_RETRIVE_MTHINGS_BY_TOKEN
00000010	MANAGER_REGISTER_MTHING
00000011	MANAGER_UNREGISTER_MTHING
00000100	MANAGER_SHOW_REGISTERED_MTHINGS
00000101 - 11111111	Reserved

A.2.6 AggregatorCapabilityCS

```
<Term termId="AGGREGATOR_INCLUDE_MTHING">
    <Name xml:lang="en">AGGREGATOR_INCLUDE_MTHING</Name>
    <Definition xml:lang="en">
       An MAggregator is capable of including an MThing
    </Definition>
</Term>

<Term termId="AGGREGATOR_EXCLUDE_MTHING">
    <Name xml:lang="en">AGGREGATOR_EXCLUDE_MTHING</Name>
    <Definition xml:lang="en">
       An MAggregator is capable of excluding an MThing
    </Definition>
</Term>

<Term termId="AGGREGATOR_SHOW_AGGREGATED_MTHINGS">
    <Name xml:lang="en">AGGREGATOR_SHOW_AGGREGATED_MTHINGS</Name>
    <Definition xml:lang="en">
       An MAggregator is capable of showing a list of MThings aggregated
    </Definition>
</Term>
```

Table A.7 — Binary representation of `AggregatorCapabilityCS`

Binary representation	Term ID of AggregatorCapabilityCS
00000000	AGGREGATOR_INCLUDE_MTHING
00000001	AGGREGATOR_EXCLUDE_MTHING
00000010	AGGREGATOR_SHOW_AGGREGATED_MTHINGS
00000011 - 11111111	Reserved

A.3 CapabilityParameterCS

A.3.1 SensorCapabilityParameterCS

```
<Term termId="AUDIO_CODEC_PCM">
    <Name xml:lang="en">PCM Codec</Name>
    <Definition xml:lang="en">
       Audio is streamed in PCM codec
    </Definition>
</Term>

<Term termId="AUDIO_CODEC_AC3">
    <Name xml:lang="en">AC3 Codec</Name>
    <Definition xml:lang="en">
       Audio is streamed in AC3 codec
    </Definition>
</Term>

<Term termId="AUDIO_STREAMING_PROTOCOL_HTTP">
    <Name xml:lang="en">Audio Streaming HTTP Protocol</Name>
    <Definition xml:lang="en">
       Audio is streamed in HTTP protocol
    </Definition>
</Term>

<Term termId="AUDIO_STREAMING_PROTOCOL_RTP">
    <Name xml:lang="en">Audio Streaming RTP Protocol</Name>
    <Definition xml:lang="en">
       Audio is streamed in RTP Protocol
    </Definition>
</Term>

<Term termId="AUDIO_CODEC_MP3">
    <Name xml:lang="en">MP3 Codec</Name>
    <Definition xml:lang="en">
       Audio is streamed in MP3 codec
    </Definition>
</Term>

<Term termId="AUDIO_CODEC_AAC">
    <Name xml:lang="en">AAC Codec</Name>
    <Definition xml:lang="en">
       Audio is streamed in AAC codec
    </Definition>
</Term>

<Term termId="AUDIO_CODEC_APE">
    <Name xml:lang="en">APE Codec</Name>
    <Definition xml:lang="en">
       Audio is streamed in APE codec
    </Definition>
</Term>

<Term termId="AUDIO_CODEC_WMA">
    <Name xml:lang="en">WMA Codec</Name>
    <Definition xml:lang="en">
       Audio is streamed in WMA codec
    </Definition>
</Term>
```

```
<Term termId="AUDIO_CODEC_MPC">
    <Name xml:lang="en">MPC Codec</Name>
    <Definition xml:lang="en">
       Audio is streamed in MPC codec
    </Definition>
</Term>

<Term termId="AUDIO_CODEC_FLAC">
    <Name xml:lang="en">FLAC Codec</Name>
    <Definition xml:lang="en">
       Audio is streamed in FLAC codec
    </Definition>
</Term>

<Term termId="VIDEO_CODEC_MP4_AVC">
    <Name xml:lang="en">mpeg-4 AVC Codec</Name>
    <Definition xml:lang="en">
       Video is streamed in mpeg-4 AVC codec
    </Definition>
</Term>

<Term termId="VIDEO_CODEC_MPEG1">
    <Name xml:lang="en">MPEG1 Codec</Name>
    <Definition xml:lang="en">
       Video is streamed in MPEG1 codec
    </Definition>
</Term>

<Term termId="VIDEO_CODEC_MPEG2">
    <Name xml:lang="en">MPEG2 Codec</Name>
    <Definition xml:lang="en">
       Video is streamed in MPEG2 codec
    </Definition>
</Term>

<Term termId="VIDEO_CODEC_MPEG4_ASP">
    <Name xml:lang="en">MPEG4 ASP Codec</Name>
    <Definition xml:lang="en">
       Video is streamed in MPEG4 ASP codec
    </Definition>
</Term>

<Term termId="VIDEO_STREAMING_PROTOCOL_HTTP">
    <Name xml:lang="en">VIDEO Streaming HTTP Protocol</Name>
    <Definition xml:lang="en">
       VIDEO is streamed in HTTP protocol.
    </Definition>
</Term>

<Term termId="VIDEO_STREAMING_PROTOCOL_RTP">
    <Name xml:lang="en">VIDEO Streaming RTP Protocol</Name>
    <Definition xml:lang="en">
       VIDEO is streamed in RTP Protocol.
    </Definition>
</Term>
```

Table A.8 — Binary representation of `SensorCapabilityParameterCS`

Binary representation	Term ID of SensorCapabilityParameterCS
00000000	AUDIO_CODEC_PCM
00000001	AUDIO_CODEC_AC3
00000010	AUDIO_STREAMING_PROTOCOL_HTTP
00000011	AUDIO_STREAMING_PROTOCOL_RTP
00000100	AUDIO_CODEC_MP3
00000101	AUDIO_CODEC_AAC
00000110	AUDIO_CODEC_APE
00000111	AUDIO_CODEC_WMA
00001000	AUDIO_CODEC_MPC
00001001	AUDIO_CODEC_FLAC
00001010	VIDEO_CODEC_MP4_AVC
00001011	VIDEO_CODEC_MPEG1
00001100	VIDEO_CODEC_MPEG2
00001101	VIDEO_CODEC_MPEG4_ASP
00001110	VIDEO_STREAMING_PROTOCOL_HTTP
00001111	VIDEO_STREAMING_PROTOCOL_RTP
00010000 - 11111111	Reserved

A.3.2 ActuatorCapabilityParameterCS

```
<Term termId="AUDIO_CODEC_PCM">
    <Name xml:lang="en">PCM Codec</Name>
    <Definition xml:lang="en">
       Audio is streamed in PCM codec
    </Definition>
</Term>

<Term termId="AUDIO_CODEC_AC3">
    <Name xml:lang="en">AC3 Codec</Name>
    <Definition xml:lang="en">
       Audio is streamed in AC3 codec
    </Definition>
</Term>

<Term termId="AUDIO_STREAMING_PROTOCOL_HTTP">
    <Name xml:lang="en">Audio Streaming HTTP Protocol</Name>
```

```
    <Definition xml:lang="en">
       Audio is streamed in HTTP protocol
    </Definition>
</Term>

<Term termId="AUDIO_STREAMING_PROTOCOL_RTP">
    <Name xml:lang="en">Audio Streaming RTP Protocol</Name>
    <Definition xml:lang="en">
       Audio is streamed in RTP Protocol
    </Definition>
</Term>

<Term termId="AUDIO_CODEC_MP3">
    <Name xml:lang="en">MP3 Codec</Name>
    <Definition xml:lang="en">
       Audio is streamed in MP3 codec
    </Definition>
</Term>

<Term termId="AUDIO_CODEC_AAC">
    <Name xml:lang="en">AAC Codec</Name>
    <Definition xml:lang="en">
       Audio is streamed in AAC codec
    </Definition>
</Term>

<Term termId="AUDIO_CODEC_APE">
    <Name xml:lang="en">APE Codec</Name>
    <Definition xml:lang="en">
       Audio is streamed in APE codec
    </Definition>
</Term>

<Term termId="AUDIO_CODEC_WMA">
    <Name xml:lang="en">WMA Codec</Name>
    <Definition xml:lang="en">
       Audio is streamed in WMA codec
    </Definition>
</Term>

<Term termId="AUDIO_CODEC_MPC">
    <Name xml:lang="en">MPC Codec</Name>
    <Definition xml:lang="en">
       Audio is streamed in MPC codec
    </Definition>
</Term>

<Term termId="AUDIO_CODEC_FLAC">
    <Name xml:lang="en">FLAC Codec</Name>
    <Definition xml:lang="en">
       Audio is streamed in FLAC codec
    </Definition>
</Term>

<Term termId="VIDEO_CODEC_H264">
    <Name xml:lang="en">H264 Codec</Name>
    <Definition xml:lang="en">
       Video is streamed in H.264 codec
    </Definition>
</Term>
```

```
<Term termId="VIDEO_CODEC_MPEG1">
    <Name xml:lang="en">MPEG1 Codec</Name>
    <Definition xml:lang="en">
       Video is streamed in MPEG1 codec
    </Definition>
</Term>

<Term termId="VIDEO_CODEC_MPEG2">
    <Name xml:lang="en">MPEG2 Codec</Name>
    <Definition xml:lang="en">
       Video is streamed in MPEG2 codec
    </Definition>
</Term>

<Term termId="VIDEO_CODEC_MPEG4_ASP">
    <Name xml:lang="en">MPEG4 ASP Codec</Name>
    <Definition xml:lang="en">
       Video is streamed in MPEG4 ASP codec
    </Definition>
</Term>
```

Table A.9 — Binary representation of `ActuatorCapabilityParameterCS`

Binary representation	Term ID of ActuatorCapabilityParameterCS
00000000	AUDIO_CODEC_PCM
00000001	AUDIO_CODEC_AC3
00000010	AUDIO_STREAMING_PROTOCOL_HTTP
00000011	AUDIO_STREAMING_PROTOCOL_RTP
00000100	AUDIO_CODEC_MP3
00000101	AUDIO_CODEC_AAC
00000110	AUDIO_CODEC_APE
00000111	AUDIO_CODEC_WMA
00001000	AUDIO_CODEC_MPC
00001001	AUDIO_CODEC_FLAC
00001010	VIDEO_CODEC_H264
00001011	VIDEO_CODEC_MPEG1
00001100	VIDEO_CODEC_MPEG2
00001101	VIDEO_CODEC_MPEG4_ASP
00001110 - 11111111	Reserved

A.3.3 AnalyzerCapabilityParameterCS

```
<Term termId="VIDEO">
    <Name xml:lang="en">VIDEO</Name>
    <Definition xml:lang="en">
       Video is an input to be analyzed
    </Definition>
</Term>

<Term termId="AUDIO">
    <Name xml:lang="en">AUDIO</Name>
    <Definition xml:lang="en">
       Audio is an input to be analyzed
    </Definition>
</Term>

<Term termId="IMAGE">
    <Name xml:lang="en">IMAGE</Name>
    <Definition xml:lang="en">
       Image is an input to be analyzed
    </Definition>
</Term>

<Term termId="TEXT">
    <Name xml:lang="en">TEXT</Name>
    <Definition xml:lang="en">
       Text is an input to be analyzed
    </Definition>
</Term>

<Term termId="3D_GRAPHICS">
    <Name xml:lang="en">3D_GRAPHICS</Name>
    <Definition xml:lang="en">
       3D graphics are an input to be analyzed
    </Definition>
</Term>

<Term termId="SENSED_DATA_EXCEPT_MEDIA">
    <Name xml:lang="en">SENSED_DATA_EXCEPT_MEDIA</Name>
    <Definition xml:lang="en">
       Sensed data except media data are an input to be analyzed
    </Definition>
</Term>
```


Table A.10 — Binary representation of `AnalyzerCapabilityParameterCS`

Binary representation	Term ID of AnalyzerCapabilityParameterCS
00000000	VIDEO
00000001	AUDIO
00000010	IMAGE
00000011	TEXT
00000100	3D_GRAPHICS

Binary representation	Term ID of AnalyzerCapabilityParameterCS
00000101	SENSED_DATA_EXCEPT_MEDIA
00000110 - 11111111	Reserved

A.3.4 StorageCapabilityParameterCS

```
<Term termId="DOC">
    <Name xml:lang="en">DOC</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Microsoft Word Document
    </Definition>
</Term>

<Term termId="DOCX">
    <Name xml:lang="en">DOCX</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Microsoft Word Open XML Document
    </Definition>
</Term>

<Term termId="LOG">
    <Name xml:lang="en">LOG</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Log File
    </Definition>
</Term>

<Term termId="MSG">
    <Name xml:lang="en">MSG</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Outlook Mail Message
    </Definition>
</Term>

<Term termId="ODT">
    <Name xml:lang="en">ODT</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing OpenDocument Text Document
    </Definition>
</Term>

<Term termId="PAGES">
    <Name xml:lang="en">PAGES</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Pages Document
    </Definition>
</Term>

<Term termId="RTF">
    <Name xml:lang="en">RTF</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Rich Text Format File
    </Definition>
</Term>
```

```
<Term termId="TEX">
    <Name xml:lang="en">TEX</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing LaTeX Source Document
    </Definition>
</Term>

<Term termId="TXT">
    <Name xml:lang="en">TXT</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Plain Text File
    </Definition>
</Term>

<Term termId="WPD">
    <Name xml:lang="en">WPD</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing WordPerfect Document
    </Definition>
</Term>

<Term termId="WPS">
    <Name xml:lang="en">WPS</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Microsoft Works Word Processor Document
    </Definition>
</Term>

<Term termId="CSV">
    <Name xml:lang="en">CSV</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Comma Separated Values File
    </Definition>
</Term>

<Term termId="DAT">
    <Name xml:lang="en">DAT</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Data File
    </Definition>
</Term>

<Term termId="GED">
    <Name xml:lang="en">GED</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing GEDCOM Genealogy Data File
    </Definition>
</Term>

<Term termId="KEY">
    <Name xml:lang="en">KEY</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Keynote Presentation
    </Definition>
</Term>

<Term termId="KEYCHAIN">
    <Name xml:lang="en">KEYCHAIN</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Mac OS X Keychain File
    </Definition>
```

```
</Term>

<Term termId="PPS">
    <Name xml:lang="en">PPS</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing PowerPoint Slide Show
    </Definition>
</Term>

<Term termId="PPT">
    <Name xml:lang="en">PPT</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing PowerPoint Presentation
    </Definition>
</Term>

<Term termId="PPTX">
    <Name xml:lang="en">PPTX</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing PowerPoint Open XML Presentation
    </Definition>
</Term>

<Term termId="SDF">
    <Name xml:lang="en">SDF</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Standard Data File
    </Definition>
</Term>

<Term termId="TAR">
    <Name xml:lang="en">TAR</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Consolidated Unix File Archive
    </Definition>
</Term>

<Term termId="TAX2016">
    <Name xml:lang="en">TAX2016</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing TurboTax 2016 Tax Return
    </Definition>
</Term>

<Term termId="TAX2017">
    <Name xml:lang="en">TAX2017</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing TurboTax 2017 Tax Return
    </Definition>
</Term>

<Term termId="VCF">
    <Name xml:lang="en">VCF</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing vCard File
    </Definition>
</Term>

<Term termId="XML">
    <Name xml:lang="en">XML</Name>
    <Definition xml:lang="en">
```

```
        An MStorage is capable of storing XML File
    </Definition>
</Term>

<Term termId="AIF">
    <Name xml:lang="en">AIF</Name>
    <Definition xml:lang="en">
        An MStorage is capable of storing Audio Interchange File Format
    </Definition>
</Term>

<Term termId="IFF">
    <Name xml:lang="en">IFF</Name>
    <Definition xml:lang="en">
        An MStorage is capable of storing Interchange File Format
    </Definition>
</Term>

<Term termId="M3U">
    <Name xml:lang="en">M3U</Name>
    <Definition xml:lang="en">
        An MStorage is capable of storing Media Playlist File
    </Definition>
</Term>

<Term termId="M4A">
    <Name xml:lang="en">M4A</Name>
    <Definition xml:lang="en">
        An MStorage is capable of storing MPEG-4 Audio File
    </Definition>
</Term>

<Term termId="MID">
    <Name xml:lang="en">MID</Name>
    <Definition xml:lang="en">
        An MStorage is capable of storing MIDI File
    </Definition>
</Term>

<Term termId="MP3">
    <Name xml:lang="en">MP3</Name>
    <Definition xml:lang="en">
        An MStorage is capable of storing MP3 Audio File
    </Definition>
</Term>

<Term termId="MPA">
    <Name xml:lang="en">MPA</Name>
    <Definition xml:lang="en">
        An MStorage is capable of storing MPEG-2 Audio File
    </Definition>
</Term>

<Term termId="WAV">
    <Name xml:lang="en">WAV</Name>
    <Definition xml:lang="en">
        An MStorage is capable of storing WAVE Audio File
    </Definition>
</Term>

<Term termId="WMA">
```

```
    <Name xml:lang="en">WMA</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Windows Media Audio File
    </Definition>
</Term>

<Term termId="3G2">
    <Name xml:lang="en">3G2</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing 3GPP2 Multimedia File
    </Definition>
</Term>

<Term termId="3GP">
    <Name xml:lang="en">3GP</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing 3GPP Multimedia File
    </Definition>
</Term>

<Term termId="ASF">
    <Name xml:lang="en">ASF</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Advanced Systems Format File
    </Definition>
</Term>

<Term termId="AVI">
    <Name xml:lang="en">AVI</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Audio Video Interleave File
    </Definition>
</Term>

<Term termId="FLV">
    <Name xml:lang="en">FLV</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Animate Video File
    </Definition>
</Term>

<Term termId="M4V">
    <Name xml:lang="en">M4V</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing iTunes Video File
    </Definition>
</Term>

<Term termId="MOV">
    <Name xml:lang="en">MOV</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Apple QuickTime Movie
    </Definition>
</Term>

<Term termId="MP4">
    <Name xml:lang="en">MP4</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing MPEG-4 Video File
    </Definition>
</Term>
```

```
<Term termId="MPG">
    <Name xml:lang="en">MPG</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing MPEG Video File
    </Definition>
</Term>

<Term termId="RM">
    <Name xml:lang="en">RM</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing RealMedia File
    </Definition>
</Term>

<Term termId="SRT">
    <Name xml:lang="en">SRT</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing SubRip Subtitle File
    </Definition>
</Term>

<Term termId="SWF">
    <Name xml:lang="en">SWF</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Shockwave Flash Movie
    </Definition>
</Term>

<Term termId="VOB">
    <Name xml:lang="en">VOB</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing DVD Video Object File
    </Definition>
</Term>

<Term termId="WMV">
    <Name xml:lang="en">WMV</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Windows Media Video File
    </Definition>
</Term>

<Term termId="3DM">
    <Name xml:lang="en">3DM</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Rhino 3D Model
    </Definition>
</Term>

<Term termId="3DS">
    <Name xml:lang="en">3DS</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing 3D Studio Scene
    </Definition>
</Term>

<Term termId="MAX">
    <Name xml:lang="en">MAX</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing 3ds Max Scene File
```

```
    </Definition>
</Term>

<Term termId="OBJ">
    <Name xml:lang="en">OBJ</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Wavefront 3D Object File
    </Definition>
</Term>

<Term termId="BMP">
    <Name xml:lang="en">BMP</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Bitmap Image File
    </Definition>
</Term>

<Term termId="DDS">
    <Name xml:lang="en">DDS</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing DirectDraw Surface
    </Definition>
</Term>

<Term termId="GIF">
    <Name xml:lang="en">GIF</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Graphical Interchange Format File
    </Definition>
</Term>

<Term termId="JPG">
    <Name xml:lang="en">JPG</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing JPEG Image
    </Definition>
</Term>

<Term termId="PNG">
    <Name xml:lang="en">PNG</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Portable Network Graphic
    </Definition>
</Term>

<Term termId="PSD">
    <Name xml:lang="en">PSD</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Adobe Photoshop Document
    </Definition>
</Term>

<Term termId="PSPIMAGE">
    <Name xml:lang="en">PSPIMAGE</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing PaintShop Pro Image
    </Definition>
</Term>

<Term termId="TGA">
    <Name xml:lang="en">TGA</Name>
```

```
    <Definition xml:lang="en">
       An MStorage is capable of storing Targa Graphic
    </Definition>
</Term>

<Term termId="THM">
    <Name xml:lang="en">THM</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Thumbnail Image File
    </Definition>
</Term>

<Term termId="TIF">
    <Name xml:lang="en">TIF</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Tagged Image File
    </Definition>
</Term>

<Term termId="TIFF">
    <Name xml:lang="en">TIFF</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Tagged Image File Format
    </Definition>
</Term>

<Term termId="YUV">
    <Name xml:lang="en">YUV</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing YUV Encoded Image File
    </Definition>
</Term>

<Term termId="AI">
    <Name xml:lang="en">AI</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Adobe Illustrator File
    </Definition>
</Term>

<Term termId="EPS">
    <Name xml:lang="en">EPS</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Encapsulated PostScript File
    </Definition>
</Term>

<Term termId="PS">
    <Name xml:lang="en">PS</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing PostScript File
    </Definition>
</Term>

<Term termId="SVG">
    <Name xml:lang="en">SVG</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Scalable Vector Graphics File
    </Definition>
</Term>
```

```
<Term termId="INDD">
    <Name xml:lang="en">INDD</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Adobe InDesign Document
    </Definition>
</Term>

<Term termId="PCT">
    <Name xml:lang="en">PCT</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Picture File
    </Definition>
</Term>

<Term termId="PDF">
    <Name xml:lang="en">PDF</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Portable Document Format File
    </Definition>
</Term>

<Term termId="XLR">
    <Name xml:lang="en">XLR</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Works Spreadsheet
    </Definition>
</Term>

<Term termId="XLS">
    <Name xml:lang="en">XLS</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Excel Spreadsheet
    </Definition>
</Term>

<Term termId="XLSX">
    <Name xml:lang="en">XLSX</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Microsoft Excel Open XML Spreadsheet
    </Definition>
</Term>

<Term termId="ACCDB">
    <Name xml:lang="en">ACCDB</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Access 2007 Database File
    </Definition>
</Term>

<Term termId="DB">
    <Name xml:lang="en">DB</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Database File
    </Definition>
</Term>

<Term termId="DBF">
    <Name xml:lang="en">DBF</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Database File
    </Definition>
```

```
</Term>

<Term termId="MDB">
    <Name xml:lang="en">MDB</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Microsoft Access Database
    </Definition>
</Term>

<Term termId="PDB">
    <Name xml:lang="en">PDB</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Program Database
    </Definition>
</Term>

<Term termId="SQL">
    <Name xml:lang="en">SQL</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Structured Query Language Data File
    </Definition>
</Term>

<Term termId="APK">
    <Name xml:lang="en">APK</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Android Package File
    </Definition>
</Term>

<Term termId="APP">
    <Name xml:lang="en">APP</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Mac OS X Application
    </Definition>
</Term>

<Term termId="BAT">
    <Name xml:lang="en">BAT</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing DOS Batch File
    </Definition>
</Term>

<Term termId="CGI">
    <Name xml:lang="en">CGI</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Common Gateway Interface Script
    </Definition>
</Term>

<Term termId="COM">
    <Name xml:lang="en">COM</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing DOS Command File
    </Definition>
</Term>

<Term termId="EXE">
    <Name xml:lang="en">EXE</Name>
    <Definition xml:lang="en">
```

```
        An MStorage is capable of storing Windows Executable File
    </Definition>
</Term>

<Term termId="GADGET">
    <Name xml:lang="en">GADGET</Name>
    <Definition xml:lang="en">
        An MStorage is capable of storing Windows Gadget
    </Definition>
</Term>

<Term termId="JAR">
    <Name xml:lang="en">JAR</Name>
    <Definition xml:lang="en">
        An MStorage is capable of storing Java Archive File
    </Definition>
</Term>

<Term termId="WSF">
    <Name xml:lang="en">WSF</Name>
    <Definition xml:lang="en">
        An MStorage is capable of storing Windows Script File
    </Definition>
</Term>

<Term termId="B">
    <Name xml:lang="en">B</Name>
    <Definition xml:lang="en">
        An MStorage is capable of storing Grand Theft Auto 3 Saved Game File
    </Definition>
</Term>

<Term termId="DEM">
    <Name xml:lang="en">DEM</Name>
    <Definition xml:lang="en">
        An MStorage is capable of storing Video Game Demo File
    </Definition>
</Term>

<Term termId="GAM">
    <Name xml:lang="en">GAM</Name>
    <Definition xml:lang="en">
        An MStorage is capable of storing Saved Game File
    </Definition>
</Term>

<Term termId="NES">
    <Name xml:lang="en">NES</Name>
    <Definition xml:lang="en">
        An MStorage is capable of storing Nintendo (NES) ROM File
    </Definition>
</Term>

<Term termId="ROM">
    <Name xml:lang="en">ROM</Name>
    <Definition xml:lang="en">
        An MStorage is capable of storing N64 Game ROM File
    </Definition>
</Term>

<Term termId="SAV">
```

```
    <Name xml:lang="en">SAV</Name>
    <Definition xml:lang="en">
      An MStorage is capable of storing Saved Game
    </Definition>
</Term>

<Term termId="DWG">
    <Name xml:lang="en">DWG</Name>
    <Definition xml:lang="en">
      An MStorage is capable of storing AutoCAD Drawing Database File
    </Definition>
</Term>

<Term termId="DXF">
    <Name xml:lang="en">DXF</Name>
    <Definition xml:lang="en">
      An MStorage is capable of storing Drawing Exchange Format File
    </Definition>
</Term>

<Term termId="GPX">
    <Name xml:lang="en">GPX</Name>
    <Definition xml:lang="en">
      An MStorage is capable of storing GPS Exchange File
    </Definition>
</Term>

<Term termId="KML">
    <Name xml:lang="en">KML</Name>
    <Definition xml:lang="en">
      An MStorage is capable of storing Keyhole Markup Language File
    </Definition>
</Term>

<Term termId="KMZ">
    <Name xml:lang="en">KMZ</Name>
    <Definition xml:lang="en">
      An MStorage is capable of storing Google Earth Placemark File
    </Definition>
</Term>

<Term termId="ASP">
    <Name xml:lang="en">ASP</Name>
    <Definition xml:lang="en">
      An MStorage is capable of storing Active Server Page
    </Definition>
</Term>

<Term termId="ASPX">
    <Name xml:lang="en">ASPX</Name>
    <Definition xml:lang="en">
      An MStorage is capable of storing Active Server Page Extended File
    </Definition>
</Term>

<Term termId="CER">
    <Name xml:lang="en">CER</Name>
    <Definition xml:lang="en">
      An MStorage is capable of storing Internet Security Certificate
    </Definition>
</Term>
```

```
<Term termId="CFM">
    <Name xml:lang="en">CFM</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing ColdFusion Markup File
    </Definition>
</Term>

<Term termId="CSR">
    <Name xml:lang="en">CSR</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Certificate Signing Request File
    </Definition>
</Term>

<Term termId="CSS">
    <Name xml:lang="en">CSS</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Cascading Style Sheet
    </Definition>
</Term>

<Term termId="DCR">
    <Name xml:lang="en">DCR</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Shockwave Media File
    </Definition>
</Term>

<Term termId="HTM">
    <Name xml:lang="en">HTM</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Hypertext Markup Language File
    </Definition>
</Term>

<Term termId="HTML">
    <Name xml:lang="en">HTML</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Hypertext Markup Language File
    </Definition>
</Term>

<Term termId="JS">
    <Name xml:lang="en">JS</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing JavaScript File
    </Definition>
</Term>

<Term termId="JSP">
    <Name xml:lang="en">JSP</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Java Server Page
    </Definition>
</Term>

<Term termId="PHP">
    <Name xml:lang="en">PHP</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing PHP Source Code File
```

```
    </Definition>
</Term>

<Term termId="RSS">
    <Name xml:lang="en">RSS</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Rich Site Summary
    </Definition>
</Term>

<Term termId="XHTML">
    <Name xml:lang="en">XHTML</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Extensible Hypertext Markup Language
File
    </Definition>
</Term>

<Term termId="CRX">
    <Name xml:lang="en">CRX</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Chrome Extension
    </Definition>
</Term>

<Term termId="PLUGIN">
    <Name xml:lang="en">PLUGIN</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Mac OS X Plugin
    </Definition>
</Term>

<Term termId="FNT">
    <Name xml:lang="en">FNT</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Windows Font File
    </Definition>
</Term>

<Term termId="FON">
    <Name xml:lang="en">FON</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Generic Font File
    </Definition>
</Term>

<Term termId="OTF">
    <Name xml:lang="en">OTF</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing OpenType Font
    </Definition>
</Term>

<Term termId="KMZ">
    <Name xml:lang="en">KMZ</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Google Earth Placemark File
    </Definition>
</Term>

<Term termId="ASP">
```

```
    <Name xml:lang="en">ASP</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Active Server Page
    </Definition>
</Term>

<Term termId="ASPX">
    <Name xml:lang="en">ASPX</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Active Server Page Extended File
    </Definition>
</Term>

<Term termId="CER">
    <Name xml:lang="en">CER</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Internet Security Certificate
    </Definition>
</Term>

<Term termId="CFM">
    <Name xml:lang="en">CFM</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing ColdFusion Markup File
    </Definition>
</Term>

<Term termId="CSR">
    <Name xml:lang="en">CSR</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Certificate Signing Request File
    </Definition>
</Term>

<Term termId="CSS">
    <Name xml:lang="en">CSS</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Cascading Style Sheet
    </Definition>
</Term>

<Term termId="DCR">
    <Name xml:lang="en">DCR</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Shockwave Media File
    </Definition>
</Term>

<Term termId="HTM">
    <Name xml:lang="en">HTM</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Hypertext Markup Language File
    </Definition>
</Term>

<Term termId="HTML">
    <Name xml:lang="en">HTML</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Hypertext Markup Language File
    </Definition>
</Term>
```

```
<Term termId="JS">
    <Name xml:lang="en">JS</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing JavaScript File
    </Definition>
</Term>

<Term termId="JSP">
    <Name xml:lang="en">JSP</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Java Server Page
    </Definition>
</Term>

<Term termId="PHP">
    <Name xml:lang="en">PHP</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing PHP Source Code File
    </Definition>
</Term>

<Term termId="RSS">
    <Name xml:lang="en">RSS</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Rich Site Summary
    </Definition>
</Term>

<Term termId="XHTML">
    <Name xml:lang="en">XHTML</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Extensible Hypertext Markup Language
File
    </Definition>
</Term>

<Term termId="CRX">
    <Name xml:lang="en">CRX</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Chrome Extension
    </Definition>
</Term>

<Term termId="PLUGIN">
    <Name xml:lang="en">PLUGIN</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Mac OS X Plugin
    </Definition>
</Term>

<Term termId="FNT">
    <Name xml:lang="en">FNT</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Windows Font File
    </Definition>
</Term>

<Term termId="FON">
    <Name xml:lang="en">FON</Name>
    <Definition xml:lang="en">
```

```
        An MStorage is capable of storing Generic Font File
    </Definition>
</Term>

<Term termId="OTF">
    <Name xml:lang="en">OTF</Name>
    <Definition xml:lang="en">
        An MStorage is capable of storing OpenType Font
    </Definition>
</Term>

<Term termId="GZ">
    <Name xml:lang="en">GZ</Name>
    <Definition xml:lang="en">
        An MStorage is capable of storing Gnu Zipped Archive
    </Definition>
</Term>

<Term termId="PKG">
    <Name xml:lang="en">PKG</Name>
    <Definition xml:lang="en">
        An MStorage is capable of storing Mac OS X Installer Package
    </Definition>
</Term>

<Term termId="RAR">
    <Name xml:lang="en">RAR</Name>
    <Definition xml:lang="en">
        An MStorage is capable of storing WinRAR Compressed Archive
    </Definition>
</Term>

<Term termId="RPM">
    <Name xml:lang="en">RPM</Name>
    <Definition xml:lang="en">
        An MStorage is capable of storing Red Hat Package Manager File
    </Definition>
</Term>

<Term termId="SITX">
    <Name xml:lang="en">SITX</Name>
    <Definition xml:lang="en">
        An MStorage is capable of storing StuffIt X Archive
    </Definition>
</Term>

<Term termId="TARGZ">
    <Name xml:lang="en">TARGZ</Name>
    <Definition xml:lang="en">
        An MStorage is capable of storing Compressed Tarball File
    </Definition>
</Term>

<Term termId="ZIP">
    <Name xml:lang="en">ZIP</Name>
    <Definition xml:lang="en">
        An MStorage is capable of storing Zipped File
    </Definition>
</Term>

<Term termId="ZIPX">
```

```
    <Name xml:lang="en">ZIPX</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Extended Zip File
    </Definition>
</Term>

<Term termId="BIN">
    <Name xml:lang="en">BIN</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Binary Disc Image
    </Definition>
</Term>

<Term termId="CUE">
    <Name xml:lang="en">CUE</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Cue Sheet File
    </Definition>
</Term>

<Term termId="DMG">
    <Name xml:lang="en">DMG</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Mac OS X Disk Image
    </Definition>
</Term>

<Term termId="ISO">
    <Name xml:lang="en">ISO</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Disc Image File
    </Definition>
</Term>

<Term termId="MDF">
    <Name xml:lang="en">MDF</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Media Disc Image File
    </Definition>
</Term>

<Term termId="TOAST">
    <Name xml:lang="en">TOAST</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Toast Disc Image
    </Definition>
</Term>

<Term termId="VCD">
    <Name xml:lang="en">VCD</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Virtual CD
    </Definition>
</Term>

<Term termId="C">
    <Name xml:lang="en">C</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing C/C++ Source Code File
    </Definition>
</Term>
```

```
<Term termId="CLASS">
    <Name xml:lang="en">CLASS</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Java Class File
    </Definition>
</Term>

<Term termId="CPP">
    <Name xml:lang="en">CPP</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing C++ Source Code File
    </Definition>
</Term>

<Term termId="CS">
    <Name xml:lang="en">CS</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing C# Source Code File
    </Definition>
</Term>

<Term termId="DTD">
    <Name xml:lang="en">DTD</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Document Type Definition File
    </Definition>
</Term>

<Term termId="FLA">
    <Name xml:lang="en">FLA</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Adobe Animate Animation
    </Definition>
</Term>

<Term termId="H">
    <Name xml:lang="en">H</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing C/C++/Objective-C Header File
    </Definition>
</Term>

<Term termId="JAVA">
    <Name xml:lang="en">JAVA</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Java Source Code File
    </Definition>
</Term>

<Term termId="LUA">
    <Name xml:lang="en">LUA</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Lua Source File
    </Definition>
</Term>

<Term termId="M">
    <Name xml:lang="en">M</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Objective-C Implementation File
```

```
    </Definition>
</Term>

<Term termId="PL">
    <Name xml:lang="en">PL</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Perl Script
    </Definition>
</Term>

<Term termId="PY">
    <Name xml:lang="en">PY</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Python Script
    </Definition>
</Term>

<Term termId="SH">
    <Name xml:lang="en">SH</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Bash Shell Script
    </Definition>
</Term>

<Term termId="SLN">
    <Name xml:lang="en">SLN</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Visual Studio Solution File
    </Definition>
</Term>

<Term termId="SWIFT">
    <Name xml:lang="en">SWIFT</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Swift Source Code File
    </Definition>
</Term>

<Term termId="VB">
    <Name xml:lang="en">VB</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Visual Basic Project Item File
    </Definition>
</Term>

<Term termId="VCXPROJ">
    <Name xml:lang="en">VCXPROJ</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Visual C++ Project
    </Definition>
</Term>

<Term termId="XCODEPROJ">
    <Name xml:lang="en">XCODEPROJ</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Xcode Project
    </Definition>
</Term>

<Term termId="BAK">
    <Name xml:lang="en">BAK</Name>
```

```
    <Definition xml:lang="en">
       An MStorage is capable of storing Backup File
    </Definition>
</Term>

<Term termId="TMP">
    <Name xml:lang="en">TMP</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Temporary File
    </Definition>
</Term>

<Term termId="CRDOWNLOAD">
    <Name xml:lang="en">CRDOWNLOAD</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Chrome Partially Downloaded File
    </Definition>
</Term>

<Term termId="ICS">
    <Name xml:lang="en">ICS</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Calendar File
    </Definition>
</Term>

<Term termId="MSI">
    <Name xml:lang="en">MSI</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Windows Installer Package
    </Definition>
</Term>

<Term termId="PART">
    <Name xml:lang="en">PART</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Partially Downloaded File
    </Definition>
</Term>

<Term termId="TORRENT">
    <Name xml:lang="en">TORRENT</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing BitTorrent File
    </Definition>
</Term>

<Term termId="HWP">
    <Name xml:lang="en">HWP</Name>
    <Definition xml:lang="en">
       An MStorage is capable of storing Hangul Word Processor File
    </Definition>
</Term>
```

Table A.11 — Binary representation of `StorageCapabilityParameterCS`

Binary representation	Term ID of StorageCapabilityParameterCS
00000000	DOC
00000001	DOCX
00000010	LOG
00000011	MSG
00000100	ODT
00000101	PAGES
00000110	RTF
00000111	TEX
00001000	TXT
00001001	WPD
00001010	WPS
00001011	CSV
00001100	DAT
00001101	GED
00001110	KEY
00001111	KEYCHAIN
00010000	PPS
00010001	PPT
00010010	PPTX
00010011	SDF
00010100	TAR
00010101	TAX2016
00010110	TAX2017
00010111	VCF
00011000	XML
00011001	AIF

Binary representation	Term ID of StorageCapabilityParameterCS
00011010	IFF
00011011	M3U
00011100	M4A
00011101	MID
00011110	MP3
00011111	MPA
00100000	WAV
00100001	WMA
00100010	3G2
00100011	3GP
00100100	ASF
00100101	AVI
00100110	FLV
00100111	M4V
00101000	MOV
00101001	MP4
00101010	MPG
00101011	RM
00101100	SRT
00101101	SWF
00101110	VOB
00101111	WMV
00110000	3DM
00110001	3DS
00110010	MAX
00110011	OBJ
00110100	BMP

Binary representation	Term ID of StorageCapabilityParameterCS
00110101	DDS
00110110	GIF
00110111	JPG
00111000	PNG
00111001	PSD
00111010	PSPIMAGE
00111011	TGA
00111100	THM
00111101	TIF
00111110	TIFF
00111111	YUV
01000000	AI
01000001	EPS
01000010	PS
01000011	SVG
01000100	INDD
01000101	PCT
01000110	PDF
01000111	XLR
01001000	XLS
01001001	XLSX
01001010	ACCDB
01001011	DB
01001100	DBF
01001101	MDB
01001110	PDB
01001111	SQL

Binary representation	Term ID of StorageCapabilityParameterCS
01010000	APK
01010001	APP
01010010	BAT
01010011	CGI
01010100	COM
01010101	EXE
01010110	GADGET
01010111	JAR
01011000	WSF
01011001	B
01011010	DEM
01011011	GAM
01011100	NES
01011101	ROM
01011110	SAV
01011111	DWG
01100000	DXF
01100001	GPX
01100010	KML
01100011	KMZ
01100100	ASP
01100101	ASPX
01100110	CER
01100111	CFM
01101000	CSR
01101001	CSS
01101010	DCR

Binary representation	Term ID of StorageCapabilityParameterCS
01101011	HTM
01101100	HTML
01101101	JS
01101110	JSP
01101111	PHP
01110000	RSS
01110001	XHTML
01110010	CRX
01110011	PLUGIN
01110100	FNT
01110101	FON
01110110	OTF
01110111	TTF
01111000	CAB
01111001	CPL
01111010	CUR
01111011	DESKTHEMEPACK
01111100	DLL
01111101	DMP
01111110	DRV
01111111	ICNS
10000000	ICO
10000001	LNK
10000010	SYS
10000011	CFG
10000100	INI
10000101	PRF

Binary representation	Term ID of StorageCapabilityParameterCS
10000110	HQX
10000111	MIM
10001000	UUE
10001001	7Z
10001010	CBR
10001011	DEB
10001100	GZ
10001101	PKG
10001110	RAR
10001111	RPM
10010000	SITX
10010001	TARGZ
10010010	ZIP
10010011	ZIPX
10010100	BIN
10010101	CUE
10010110	DMG
10010111	ISO
10011000	MDF
10011001	TOAST
10011010	VCD
10011011	C
10011100	CLASS
10011101	CPP
10011110	CS
10011111	DTD
10100000	FLA

Binary representation	Term ID of StorageCapabilityParameterCS
10100001	H
10100010	JAVA
10100011	LUA
10100100	M
10100101	PL
10100110	PY
10100111	SH
10101000	SLN
10101001	SWIFT
10101010	VB
10101011	VCXPROJ
10101100	XCODEPROJ
10101101	BAK
10101110	TMP
10101111	CRDOWNLOAD
10110000	ICS
10110001	MSI
10110010	PART
10110011	TORRENT
10110100	HWP
10110101 - 11111111	Reserved

A.3.5 ManagerCapabilityParameterCS

```
<Term termId="SENSOR">
    <Name xml:lang="en">SENSOR</Name>
    <Definition xml:lang="en">
       A sensor is managed
    </Definition>
</Term>
```

```
<Term termId="ACTUATOR">
    <Name xml:lang="en">ACTUATOR</Name>
    <Definition xml:lang="en">
       An actuator is managed
    </Definition>
</Term>

<Term termId="ANALYZER">
    <Name xml:lang="en">ANALYZER</Name>
    <Definition xml:lang="en">
       An analyzer is managed
    </Definition>
</Term>

<Term termId="STORAGE">
    <Name xml:lang="en">STORAGE</Name>
    <Definition xml:lang="en">
       A storage is managed
    </Definition>
</Term>

<Term termId="MANAGER">
    <Name xml:lang="en">MANAGER</Name>
    <Definition xml:lang="en">
       A manager is managed
    </Definition>
</Term>

<Term termId="AGGREGATOR">
    <Name xml:lang="en">AGGREGATOR</Name>
    <Definition xml:lang="en">
       An aggregator is managed
    </Definition>
</Term>
```


Table A.12 — Binary representation of `ManagerCapabilityParameterCS`

Binary representation	Term ID of ManagerCapabilityParameterCS
00000000	SENSOR
00000001	ACTUATOR
00000010	ANALYZER
00000011	STORAGE
00000100	MANAGER
00000101	AGGREGATOR
00000110 - 11111111	Reserved

A.3.6 AggregatorCapabilityParameterCS

```
<Term termId="SENSOR">
    <Name xml:lang="en">SENSOR</Name>
    <Definition xml:lang="en">
       A sensor is aggregated
    </Definition>
</Term>

<Term termId="ACTUATOR">
    <Name xml:lang="en">ACTUATOR</Name>
    <Definition xml:lang="en">
       An actuator is aggregated
    </Definition>
</Term>

<Term termId="ANALYZER">
    <Name xml:lang="en">ANALYZER</Name>
    <Definition xml:lang="en">
       An analyzer is aggregated
    </Definition>
</Term>

<Term termId="STORAGE">
    <Name xml:lang="en">STORAGE</Name>
    <Definition xml:lang="en">
       A storage is aggregated
    </Definition>
</Term>

<Term termId="MANAGER">
    <Name xml:lang="en">MANAGER</Name>
    <Definition xml:lang="en">
       A manager is aggregated
    </Definition>
</Term>

<Term termId="AGGREGATOR">
    <Name xml:lang="en">AGGREGATOR</Name>
    <Definition xml:lang="en">
       An aggregator is aggregated
    </Definition>
</Term>
```


Table A.13 — Binary representation of `AggregatorCapabilityParameterCS`

Binary representation	Term ID of AggregatorCapabilityParameterCS
00000000	SENSOR
00000001	ACTUATOR
00000010	ANALYZER
00000011	STORAGE
00000100	MANAGER

Binary representation	Term ID of AggregatorCapabilityParameterCS
00000101	AGGREGATOR
00000110 - 11111111	Reserved

A.4 AnalyzerCS

A.4.1 OdorImageCharacteristicsCS

```
<Term termId="BALANCED">
    <Name xml:lang="en">Balanced Scent</Name>
    <Definition xml:lang="en">
       Balanced Odor Characteristics
    </Definition>
</Term>

<Term termId="DIFFUSIVE">
    <Name xml:lang="en">Diffusive Scent</Name>
    <Definition xml:lang="en">
       Diffusive Odor Characteristics
    </Definition>
</Term>

<Term termId="DRY">
    <Name xml:lang="en">Dry Scent</Name>
    <Definition xml:lang="en">
       Dry Odor Characteristics
    </Definition>
</Term>

<Term termId="FLAT">
    <Name xml:lang="en">Flat Scent</Name>
    <Definition xml:lang="en">
       Flat Odor Characteristics
    </Definition>
</Term>

<Term termId="FRESH">
    <Name xml:lang="en">Fresh Scent</Name>
    <Definition xml:lang="en">
       Fresh Odor Characteristics
    </Definition>
</Term>

<Term termId="HARSH">
    <Name xml:lang="en">Harsh Scent</Name>
    <Definition xml:lang="en">
       Harsh Odor Characteristics
    </Definition>
</Term>

<Term termId="HEAVY">
    <Name xml:lang="en">Heavy Scent</Name>
    <Definition xml:lang="en">
       Heavy Odor Characteristics
```

```
    </Definition>
</Term>

<Term termId="MUSTY">
    <Name xml:lang="en">Musty Scent</Name>
    <Definition xml:lang="en">
       Musty Odor Characteristics
    </Definition>
</Term>

<Term termId="LIGHT">
    <Name xml:lang="en">Light Scent</Name>
    <Definition xml:lang="en">
       Light Odor Characteristics
    </Definition>
</Term>

<Term termId="RICH">
    <Name xml:lang="en">Rich Scent</Name>
    <Definition xml:lang="en">
       Rich Odor Characteristics
    </Definition>
</Term>

<Term termId="ROUND">
    <Name xml:lang="en">Round Scent</Name>
    <Definition xml:lang="en">
       Round Odor Characteristics
    </Definition>
</Term>

<Term termId="SHARP">
    <Name xml:lang="en">Sharp Scent</Name>
    <Definition xml:lang="en">
       Sharp Odor Characteristics
    </Definition>
</Term>

<Term termId="SWEET">
    <Name xml:lang="en">Sweet Scent</Name>
    <Definition xml:lang="en">
       Sweet Odor Characteristics
    </Definition>
</Term>

<Term termId="SMOOTH">
    <Name xml:lang="en">Smooth Scent</Name>
    <Definition xml:lang="en">
       Smooth Odor Characteristics
    </Definition>
</Term>
```

Table A.14 — Binary representation of `OdorImageCharacteristicsCS`

Binary representation	Term ID of OdorImageCharacteristicsCS
00000000	BALANCED
00000001	DIFFUSIVE
00000010	DRY
00000011	FLAT
00000100	FRESH
00000101	HARSH
00000110	HEAVY
00000111	MUSTY
00001000	LIGHT
00001001	RICH
00001010	ROUND
00001011	SHARP
00001100	SWEET
00001101	SMOOTH
00001110 - 11111111	Reserved

A.4.2 SocialEventCS

```
<Term termId="BIRTHDAY_SOCIAL_EVENT">
    <Name xml:lang="en">Birthday social event</Name>
    <Definition xml:lang="en">
       Detected social event type is birthday.
    </Definition>
</Term>

<Term termId="FUNERAL_SOCIAL_EVENT">
    <Name xml:lang="en">Funeral social event</Name>
    <Definition xml:lang="en">
       Detected social event type is funeral.
    </Definition>
</Term>

<Term termId="WEDDING_SOCIAL_EVENT">
    <Name xml:lang="en">Wedding social event</Name>
    <Definition xml:lang="en">
       Detected social event type is wedding.
    </Definition>
</Term>
```

```
<Term termId="GRADUATION_SOCIAL_EVENT">
    <Name xml:lang="en">Graduation social event</Name>
    <Definition xml:lang="en">
       Detected social event type is gradudation.
    </Definition>
</Term>

<Term termId="ENTRANCE_SOCIAL_EVENT">
    <Name xml:lang="en">Entrance Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is entrance.
    </Definition>
</Term>

<Term termId="ATHLETIC_COMPETITION_SOCIAL_EVENT">
    <Name xml:lang="en">Athletic Competition Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is athletic competition.
    </Definition>
</Term>

<Term termId="SPORT_PARTY_SOCIAL_EVENT">
    <Name xml:lang="en">Sport Party Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is sport party.
    </Definition>
</Term>

<Term termId="PROM_SOCIAL_EVENT">
    <Name xml:lang="en">Prom Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is prom.
    </Definition>
</Term>

<Term termId="BRIDAL_SHOWER_SOCIAL_EVENT">
    <Name xml:lang="en">Bridal Shower Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is bridal shower.
    </Definition>
</Term>

<Term termId="CHRISTMAS_SOCIAL_EVENT">
    <Name xml:lang="en">Christmas Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is Christmas.
    </Definition>
</Term>

<Term termId="CONFERENCE_SOCIAL_EVENT">
    <Name xml:lang="en">Conference Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is conference.
    </Definition>
</Term>

<Term termId="LANGUAGE_EXCHANGE_SOCIAL_EVENT">
    <Name xml:lang="en">Language Exchange Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is language exchange.
```

```
    </Definition>
</Term>

<Term termId="AWARDS_CEREMONIES_SOCIAL_EVENT">
    <Name xml:lang="en">Awards Ceremonies Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is awards ceremonies.
    </Definition>
</Term>

<Term termId="BABY_SHOWERS_SOCIAL_EVENT">
    <Name xml:lang="en">Baby Showers Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is baby showers.
    </Definition>
</Term>

<Term termId="BANQUETS_SOCIAL_EVENT">
    <Name xml:lang="en">Banquets Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is banquets.
    </Definition>
</Term>

<Term termId="BAR_BAT_MITZVAHS_SOCIAL_EVENT">
    <Name xml:lang="en">Bar Bat Mitzvahs Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is bar bat mitzvahs.
    </Definition>
</Term>

<Term termId="BLACK_TIE_EVENTS_SOCIAL_EVENT">
    <Name xml:lang="en">Black Tie Events Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is black tie events.
    </Definition>
</Term>

<Term termId="BOY_SCOUTS_SOCIAL_EVENT">
    <Name xml:lang="en">Boy Scouts Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is boy scouts.
    </Definition>
</Term>

<Term termId=" GIRL_SCOUTS_SOCIAL_EVENT ">
    <Name xml:lang="en"> Girl Scouts Social Event </Name>
    <Definition xml:lang="en">
       Detected social event type is girl scouts.
    </Definition>
</Term>

<Term termId="CLASS_REUNIONS_SOCIAL_EVENT">
    <Name xml:lang="en">Class Reunions Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is class reunions.
    </Definition>
</Term>

<Term termId="FAMILY_REUNIONS_SOCIAL_EVENT">
    <Name xml:lang="en">Family Reunions Social Event</Name>
```

```
    <Definition xml:lang="en">
       Detected social event type is family reunions.
    </Definition>
</Term>

<Term termId="CLUB_EVENTS_SOCIAL_EVENT">
    <Name xml:lang="en">Club Events Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is club events.
    </Definition>
</Term>

<Term termId="COSTUME_PARTIES_SOCIAL_EVENT">
    <Name xml:lang="en">Costume Parties Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is costume parties.
    </Definition>
</Term>

<Term termId="CULTURAL_EVENTS_SOCIAL_EVENT">
    <Name xml:lang="en">Cultural Events Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is cultural events.
    </Definition>
</Term>

<Term termId="ENGAGEMENT_PARTIES_SOCIAL_EVENT">
    <Name xml:lang="en">Engagement Parties Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is engagement parties.
    </Definition>
</Term>

<Term termId="FAREWELL_PARTIES_SOCIAL_EVENT">
    <Name xml:lang="en">Farewell Parties Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is farewell parties.
    </Definition>
</Term>

<Term termId="FUNDRAISING_EVENTS_SOCIAL_EVENT">
    <Name xml:lang="en">Fundraising Events Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is fundraising events.
    </Definition>
</Term>

<Term termId="GALAS_SOCIAL_EVENT">
    <Name xml:lang="en">Galas Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is galas.
    </Definition>
</Term>

<Term termId="GOLF_OUTINGS_SOCIAL_EVENT">
    <Name xml:lang="en">Golf Outings Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is golf outings.
    </Definition>
</Term>
```

```
<Term termId="HOLIDAY_EVENTS_SOCIAL_EVENT">
    <Name xml:lang="en">Holiday Events Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is holiday events.
    </Definition>
</Term>

<Term termId="HALLOWEEN_PARTIES_SOCIAL_EVENT">
    <Name xml:lang="en">Halloween Parties Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is Halloween parties.
    </Definition>
</Term>

<Term termId="MILESTONE_PARTIES_SOCIAL_EVENT">
    <Name xml:lang="en">Milestone Parties Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is milestone parties.
    </Definition>
</Term>

<Term termId="PICNICS_SOCIAL_EVENT">
    <Name xml:lang="en">Picnics Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is picnics.
    </Definition>
</Term>

<Term termId="QUINCEANERAS_SOCIAL_EVENT">
    <Name xml:lang="en">Quinceaneras Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is quinceaneras.
    </Definition>
</Term>

<Term termId="RECOGNITION_DINNERS_SOCIAL_EVENT">
    <Name xml:lang="en">Recognition Dinners Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is recognition dinners.
    </Definition>
</Term>

<Term termId="RELIGIOUS_FUNCTIONS_SOCIAL_EVENT">
    <Name xml:lang="en">Religious Functions Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is religious functions.
    </Definition>
</Term>

<Term termId="RETIREMENT_CELEBRATIONS_SOCIAL_EVENT">
    <Name xml:lang="en">Retirement Celebrations Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is retirement celebrations.
    </Definition>
</Term>

<Term termId="SCHOOL_DANCES_SOCIAL_EVENT">
    <Name xml:lang="en">School Dances Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is school dances.
    </Definition>
```

```
</Term>

<Term termId="SEASONAL_EVENTS_SOCIAL_EVENT">
    <Name xml:lang="en">Seasonal Events Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is seasonal events.
    </Definition>
</Term>

<Term termId="SWEET_SIXTEEN_PARTIES_SOCIAL_EVENT">
    <Name xml:lang="en">Sweet Sixteen Parties Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is sweet sixteen parties.
    </Definition>
</Term>

<Term termId="THEMED_PARTIES_SOCIAL_EVENT">
    <Name xml:lang="en">Themed Parties Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is themed parties.
    </Definition>
</Term>

<Term termId="CONVENTIONS_SOCIAL_EVENT">
    <Name xml:lang="en">Conventions Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is conventions.
    </Definition>
</Term>

<Term termId="EXPOS_SOCIAL_EVENT">
    <Name xml:lang="en">Expos Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is expos.
    </Definition>
</Term>

<Term termId="GRAND_OPENINGS_SOCIAL_EVENT">
    <Name xml:lang="en">Grand Openings Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is grand openings.
    </Definition>
</Term>

<Term termId="COMPANY_PICNICS_SOCIAL_EVENT">
    <Name xml:lang="en">Company Picnics Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is company picnics.
    </Definition>
</Term>

<Term termId="PROMOTIONS_MARKETING_SOCIAL_EVENT">
    <Name xml:lang="en">Promotions Marketing Social Event</Name>
    <Definition xml:lang="en">
       Detected social event type is promotions marketing.
    </Definition>
</Term>

<Term termId="TEAMBUILDING_EVENTS_SOCIAL_EVENT">
    <Name xml:lang="en">Teambuilding Events Social Event</Name>
    <Definition xml:lang="en">
```

```
      Detected social event type is teambuilding events.
    </Definition>
</Term>

<Term termId="TRADE_SHOWS_SOCIAL_EVENT">
    <Name xml:lang="en">Trade Shows Social Event</Name>
    <Definition xml:lang="en">
      Detected social event type is trade shows.
    </Definition>
</Term>

<Term termId="ORIENTATION_SOCIAL_EVENT">
    <Name xml:lang="en">Orientation Social Event</Name>
    <Definition xml:lang="en">
      Detected social event type is orientation.
    </Definition>
</Term>

<Term termId="BBQ_SOCIAL_EVENT">
    <Name xml:lang="en">BBQ Social Event</Name>
    <Definition xml:lang="en">
      Detected social event type is BBQ.
    </Definition>
</Term>
```

Table A.15 — Binary representation of `SocialEventCS`

Binary representation	Term ID of SocialEventCS
00000000	BIRTHDAY_SOCIAL_EVENT
00000001	FUNERAL_SOCIAL_EVENT
00000010	WEDDING_SOCIAL_EVENT
00000011	GRADUATION_SOCIAL_EVENT
00000100	ENTRANCE_SOCIAL_EVENT
00000101	ATHLETIC_COMPETITION_SOCIAL_EVENT
00000110	SPORT_PARTY_SOCIAL_EVENT
00000111	PROM_SOCIAL_EVENT
00001000	BRIDAL_SHOWER_SOCIAL_EVENT
00001001	CHRISTMAS_SOCIAL_EVENT
00001010	CONFERENCE_SOCIAL_EVENT
00001011	LANGUAGE_EXCHANGE_SOCIAL_EVENT
00001100	AWARDS_CEREMONIES_SOCIAL_EVENT

Binary representation	Term ID of SocialEventCS
00001101	BABY_SHOWERS_SOCIAL_EVENT
00001110	BANQUETS_SOCIAL_EVENT
00001111	BAR_BAT_MITZVAHS_SOCIAL_EVENT
00010000	BLACK_TIE_EVENTS_SOCIAL_EVENT
00010001	BOY_SCOUTS_SOCIAL_EVENT
00010010	GIRL_SCOUTS_SOCIAL_EVENT
00010011	CLASS_REUNIONS_SOCIAL_EVENT
00010100	FAMILY_REUNIONS_SOCIAL_EVENT
00010101	CLUB_EVENTS_SOCIAL_EVENT
00010110	COSTUME_PARTIES_SOCIAL_EVENT
00010111	CULTURAL_EVENTS_SOCIAL_EVENT
00011000	ENGAGEMENT_PARTIES_SOCIAL_EVENT
00011001	FAREWELL_PARTIES_SOCIAL_EVENT
00011010	FUNDRAISING_EVENTS_SOCIAL_EVENT
00011011	GALAS_SOCIAL_EVENT
00011100	GOLF_OUTINGS_SOCIAL_EVENT
00011101	HOLIDAY_EVENTS_SOCIAL_EVENT
00011110	HALLOWEEN_PARTIES_SOCIAL_EVENT
00011111	MILESTONE_PARTIES_SOCIAL_EVENT
00100000	PICNICS_SOCIAL_EVENT
00100001	QUINCEANERAS_SOCIAL_EVENT
00100010	RECOGNITION_DINNERS_SOCIAL_EVENT
00100011	RELIGIOUS_FUNCTIONS_SOCIAL_EVENT
00100100	RETIREMENT_CELEBRATIONS_SOCIAL_EVENT
00100101	SCHOOL_DANCES_SOCIAL_EVENT
00100110	SEASONAL_EVENTS_SOCIAL_EVENT
00100111	SWEET_SIXTEEN_PARTIES_SOCIAL_EVENT

Binary representation	Term ID of SocialEventCS
00101000	THEMED_PARTIES_SOCIAL_EVENT
00101001	CONVENTIONS_SOCIAL_EVENT
00101010	EXPOS_SOCIAL_EVENT
00101011	GRAND_OPENINGS_SOCIAL_EVENT
00101100	COMPANY_PICNICS_SOCIAL_EVENT
00101101	PROMOTIONS_MARKETING_SOCIAL_EVENT
00101110	TEAMBUILDING_EVENTS_SOCIAL_EVENT
00101111	TRADE_SHOWS_SOCIAL_EVENT
00110000	ORIENTATION_SOCIAL_EVENT
00110001	BBQ_SOCIAL_EVENT
00110010 - 11111111	Reserved

A.4.3 GestureCommandCS

```
<Term termId="PLAY">
   <Name xml:lang="en">play</Name>
   <Definition xml:lang="en">
       play
   </Definition>
</Term>

<Term termId="STOP">
   <Name xml:lang="en">Stop</Name>
   <Definition xml:lang="en">
       stop
   </Definition>
</Term>

<Term termId="PAUSE">
   <Name xml:lang="en">Pause</Name>
   <Definition xml:lang="en">
       Pause
   </Definition>
</Term>

<Term termId="REWIND2X">
   <Name xml:lang="en">Rewind2x</Name>
   <Definition xml:lang="en">
       Rewind2x
   </Definition>
</Term>

<Term termId="REWIND4X">
   <Name xml:lang="en">Rewind4x</Name>
```

```
    <Definition xml:lang="en">
        Rewind4x
    </Definition>
</Term>

<Term termId="FASTFORWARD2X">
    <Name xml:lang="en">Fastforward2x</Name>
    <Definition xml:lang="en">
        Fastforward2x
    </Definition>
</Term>

<Term termId="FASTFORWARD4X">
    <Name xml:lang="en">Fastforward4x</Name>
    <Definition xml:lang="en">
        Fastforward4x
    </Definition>
</Term>

<Term termId="ZOOMIN">
    <Name xml:lang="en">Zoomin</Name>
    <Definition xml:lang="en">
        Zoom in
    </Definition>
</Term>

<Term termId="ZOOMOUT">
    <Name xml:lang="en">Zoomout</Name>
    <Definition xml:lang="en">
        Zoom out
    </Definition>
</Term>

<Term termId="CHANGEVIEW">
    <Name xml:lang="en">Changeview</Name>
    <Definition xml:lang="en">
        Change view
    </Definition>
</Term>

<Term termId="ENTER">
    <Name xml:lang="en">Enter</Name>
    <Definition xml:lang="en">
        Enter
    </Definition>
</Term>

<Term termId="CANCLE">
    <Name xml:lang="en">Cancel</Name>
    <Definition xml:lang="en">
        Cancel
    </Definition>
</Term>

<Term termId="SELECT">
    <Name xml:lang="en">Select</Name>
    <Definition xml:lang="en">
        Select
    </Definition>
</Term>
```

```
<Term termId="GO_FORWARD">
   <Name xml:lang="en">Go_forward</Name>
   <Definition xml:lang="en">
       Go forward
   </Definition>
</Term>

<Term termId="GO_LEFT">
   <Name xml:lang="en">Go_left</Name>
   <Definition xml:lang="en">
       Go forward
   </Definition>
</Term>

<Term termId="GO_RIGHT">
   <Name xml:lang="en">Go_right</Name>
  <Definition xml:lang="en">
       Go forward
   </Definition>
</Term>

<Term termId="TURN_LEFT">
   <Name xml:lang="en">Turn_left</Name>
   <Definition xml:lang="en">
       Turn left
   </Definition>
</Term>

<Term termId="TURN_RIGHT">
   <Name xml:lang="en">Turn_right</Name>
   <Definition xml:lang="en">
       Turn right
   </Definition>
</Term>

<Term termId="SHOT">
   <Name xml:lang="en">Shot</Name>
   <Definition xml:lang="en">
       Shot
   </Definition>
</Term>
```


Table A.16 — Binary representation of `GestureCommandCS`

Binary representation	Term ID of GestureCommandCS
00000000	PLAY
00000001	STOP
00000010	PAUSE
00000011	REWIND2X
00000100	REWIND4X
00000101	FASTFORWARD2X

Binary representation	Term ID of GestureCommandCS
00000110	FASTFORWARD4X
00000111	ZOOMIN
00001000	ZOOMOUT
00001001	CHANGEVIEW
00001010	ENTER
00001011	CANCLE
00001100	SELECT
00001101	GO_FORWARD
00001110	GO_LEFT
00001111	GO_RIGHT
00010000	TURN_LEFT
00010001	TURN_RIGHT
00010010	SHOT
00010011 – 11111111	Reserved

A.4.4 QfocusCS

```
<Term termId="WHAT_QUESTION">
    <Name xml:lang="en">What_question</Name>
    <Definition xml:lang="en">
       Indicates that the question is "what" question to the QA server
    </Definition>
</Term>

<Term termId="WHERE_QUESTION">
    <Name xml:lang="en">Where_question</Name>
    <Definition xml:lang="en">
       Indicates that the question is "where" question to the QA server
    </Definition>
</Term>

<Term termId="WHEN_QUESTION">
    <Name xml:lang="en">When_question</Name>
    <Definition xml:lang="en">
       Indicates that the question is "when" question to the QA server
    </Definition>
</Term>

<Term termId="WHO_QUESTION">
    <Name xml:lang="en">Who_question</Name>
    <Definition xml:lang="en">
```

```
      Indicates that the question is "who" question to the QA server
    </Definition>
</Term>

<Term termId="WHY_QUESTION">
    <Name xml:lang="en">Why_question</Name>
    <Definition xml:lang="en">
      Indicates that the question is "why" question to the QA server
    </Definition>
</Term>

<Term termId="HOW_QUESTION">
    <Name xml:lang="en">How_question</Name>
    <Definition xml:lang="en">
      Indicates that the question is "how" question to the QA server
    </Definition>
</Term>
```

Table A.17 — Binary representation of `QfocusCS`

Binary representation	Term ID of QfocusCS
00000000	WHAT_QUESTION
00000001	WHERE_QUESTION
00000010	WHEN_QUESTION
00000011	WHO_QUESTION
00000100	WHY_QUESTION
00000101	HOW_QUESTION
00000110 - 11111111	Reserved

A.4.5 QCsemanticCS

```
<Term termId="REQUEST_FOR_COMMAND">
    <Name xml:lang="en">Request for command</Name>
    <Definition xml:lang="en">
      Request for command to the actuator
    </Definition>
</Term>

<Term termId="REQUEST_FOR_TERMINOLOGY">
    <Name xml:lang="en">Request for terminology</Name>
    <Definition xml:lang="en">
      Request for terminology to the QA server
    </Definition>
</Term>

<Term termId="REQUEST_FOR_MEANING">
    <Name xml:lang="en">Request for meaning</Name>
    <Definition xml:lang="en">
      Request for meaning to the QA server
```

```
    </Definition>
</Term>

<Term termId="REQUEST_FOR_INFORMATION">
    <Name xml:lang="en">Request for information</Name>
    <Definition xml:lang="en">
       Request for information to the QA server
    </Definition>
</Term>

<Term termId="REQUEST_FOR_METHOD">
    <Name xml:lang="en">Request for method</Name>
    <Definition xml:lang="en">
       Request for method to the QA server
    </Definition>
</Term>
```

Table A.18 — Binary representation of `QCsemanticCS`

Binary representation	Term ID of QCsemanticCS
00000000	REQUEST_FOR_COMMAND
00000001	REQUEST_FOR_TERMINOLOGY
00000010	REQUEST_FOR_MEANING
00000011	REQUEST_FOR_INFORMATION
00000100	REQUEST_FOR_METHOD
00000101 - 11111111	Reserved

A.4.6 VideoContentClassCS

```
<Term termId="POLITICS">
    <Name xml:lang="en">Politics</Name>
    <Definition xml:lang="en">
       Generate video content class is politics.
    </Definition>
</Term>

<Term termId="ECONOMY">
    <Name xml:lang="en">Economy</Name>
    <Definition xml:lang="en">
       Generate video content class is economy.
    </Definition>
</Term>

<Term termId="INTERNATIONAL">
    <Name xml:lang="en">International</Name>
    <Definition xml:lang="en">
       Generate video content class is International.
    </Definition>
</Term>
```

```
<Term termId="COMMENTARY">
    <Name xml:lang="en">Commentary</Name>
    <Definition xml:lang="en">
       Generate video content class is Commentary.
    </Definition>
</Term>

<Term termId="DEBATES">
    <Name xml:lang="en">Debates</Name>
    <Definition xml:lang="en">
       Generate video content class is Debates.
    </Definition>
</Term>

<Term termId="CONFERENCES">
    <Name xml:lang="en">Conferences</Name>
    <Definition xml:lang="en">
       Generate video content class is Conferences.
    </Definition>
</Term>

<Term termId="SOCIAL_ISSUES">
    <Name xml:lang="en">Social issues</Name>
    <Definition xml:lang="en">
       Generate video content class is Social issues.
    </Definition>
</Term>

<Term termId="DISCUSSION">
    <Name xml:lang="en">Discussion</Name>
    <Definition xml:lang="en">
       Generate video content class is Discussion.
    </Definition>
</Term>

<Term termId="INTERVIEWS">
    <Name xml:lang="en">Interviews</Name>
    <Definition xml:lang="en">
       Generate video content class is Interviews.
    </Definition>
</Term>
<Term termId="FASHION">
    <Name xml:lang="en">Fashion</Name>
    <Definition xml:lang="en">
       Generate video content class is Fashion.
    </Definition>
</Term>
<Term termId="LIFESTYLE">
    <Name xml:lang="en">Lifestyle</Name>
    <Definition xml:lang="en">
       Generate video content class is Lifestyle.
    </Definition>
</Term>

<Term termId="HEALTH_MEDICAL">
    <Name xml:lang="en">Health/medical</Name>
    <Definition xml:lang="en">
       Generate video content class is Health/medical.
    </Definition>
</Term>
```

```
<Term termId="SHOPPING">
    <Name xml:lang="en">Shopping</Name>
    <Definition xml:lang="en">
       Generate video content class is Shopping.
    </Definition>
</Term>

<Term termId="BEAUTY">
    <Name xml:lang="en">Beauty</Name>
    <Definition xml:lang="en">
       Generate video content class is Beauty.
    </Definition>
</Term>

<Term termId="GOURMET_EATING">
    <Name xml:lang="en">Gourmet eating</Name>
    <Definition xml:lang="en">
       Generate video content class is Gourmet eating.
    </Definition>
</Term>

<Term termId="MUKBANG">
    <Name xml:lang="en">Mukbang</Name>
    <Definition xml:lang="en">
       Generate video content class is Mukbang.
    </Definition>
</Term>

<Term termId="COOKING">
    <Name xml:lang="en">Cooking</Name>
    <Definition xml:lang="en">
       Generate video content class is Cooking.
    </Definition>
</Term>

<Term termId="SPECIAL_EVENTS">
    <Name xml:lang="en">Special events</Name>
    <Definition xml:lang="en">
       Generate video content class is Special events.
    </Definition>
</Term>

<Term termId="DAILY_NEWS">
    <Name xml:lang="en">Daily news</Name>
    <Definition xml:lang="en">
       Generate video content class is Daily news.
    </Definition>
</Term>

<Term termId="CURRENT_AFFAIRS">
    <Name xml:lang="en">Current affairs</Name>
    <Definition xml:lang="en">
       Generate video content class is Current affairs.
    </Definition>
</Term>

<Term termId="WEATHER_FORECAST">
    <Name xml:lang="en">Weather forecast</Name>
    <Definition xml:lang="en">
       Generate video content class is Weather forecast.
    </Definition>
```