Moving Picture, Audio and Data Coding
by Artificial Intelligence
www.mpai.community

MPAI Technical Specification

Human and Machine Communication
MPAI-HMC

V2.0

WARNING

Use of the technologies described in this Technical Specification may infringe patents, copyrights or intellectual property rights of MPAI Members or non-members.

MPAI and its Members accept no responsibility whatsoever for damages or liability, direct or consequential, which may result from the use of this Technical Specification.

Readers are invited to review Notices and Disclaimers.

Technical Specification
Human and Machine Communication (MPAI-HMC)
V2.0

1 Foreword 4
2 Introduction (Informative) 6
3 Scope 6
4 Definitions 7
5 References 7
5.1 Normative References 7
5.2 Informative References 7
6 AI Workflows 8
6.1 Technical Specification 8
6.2 Reference Software 8
6.3 Conformance Testing 8
6.4 Performance Assessment 9
7 AI Modules 9
7.1 General 9
7.1.1 Technical Specifications 9
7.1.2 Reference Software 10
7.1.3 Conformance Testing 10
7.1.4 Performance Assessment 11
7.2 AI modules from MPAI-CAE 11
7.2.1 Audio Analysis Transform 11
7.2.2 Audio Descriptors Multiplexing 13
7.2.3 Audio Object Identification 15
7.2.4 Audio Scene Description 16
7.2.5 Audio Separation and Enhancement 18
7.2.6 Audio Source Localisation 21
7.2.7 Audio Synthesis Transform 23
7.3 AI modules from MPAI-HMC 25
7.3.1 Scene Integration and Description 25
7.3.2 Entity and Context Understanding 26
7.4 AI modules from MPAI-MMC 30
7.4.1 Automatic Speech Recognition 30
7.4.2 Entity Dialogue Processing 33
7.4.3 Entity Speech Description 36
7.4.4 Entity Text Description 37
7.4.5 Natural Language Understanding 38
7.4.6 Personal Status Extraction 40
7.4.7 Personal Status Multiplexing 44
7.4.8 PS-Speech Interpretation 45
7.4.9 PS-Text Interpretation 46
7.4.10 Speaker Identity Recognition 47
7.4.11 Text and Speech Translation 50
7.4.12 Text-To-Speech 53
7.4.13 Text-to-Text Translation 55
7.5 AI modules from MPAI-OSD 57

7.5.1 Audio-Visual Alignment 57
7.5.2 Audio-Visual Event Description 58
7.5.3 Audio-Visual Scene Demultiplexing 60
7.5.4 Visual Direction Identification 62
7.5.5 Visual Instance Identification 63
7.5.6 Visual Object Extraction 65
7.5.7 Visual Object Identification 66
7.5.8 Visual Scene Description 68
7.6 AI modules from MPAI-PAF 70
7.6.1 Audio-Visual Scene Rendering 70
7.6.2 Face Identity Recognition 72
7.6.3 Entity Body Description 74
7.6.4 Face Identity Recognition 75
7.6.5 Portable Avatar Demultiplexing 77
7.6.6 PS-Face Interpretation 79
7.6.7 PS-Gesture Interpretation 80
7.6.8 Personal Status Display 81
7.6.9 Portable Avatar Multiplexing 84
8 Data Types 86
8.1 Data Types from MPAI-AIF 87
8.1.1 Machine Learning Model 87
8.2 Data Types from MPAI-CAE 88
8.2.1 Audio Basic Scene Descriptors 88
8.2.2 Audio Basic Scene Geometry 89
8.2.3 Audio Object 90
8.2.4 Audio Scene Descriptors 91
8.2.5 Audio Scene Geometry 92
8.3 Data Types from MPAI-MMC 93
8.3.1 Cognitive State 93
8.3.2 Emotion 96
8.3.3 Intention 100
8.3.4 Meaning 101
8.3.5 Personal Status 102
8.3.6 Social Attitude 104
8.3.7 Speech Descriptors 111
8.3.8 Speech Object 112
8.3.9 Text Descriptors 113
8.3.10 Text Object 114
8.4 Data Types from MPAI-OSD 115
8.4.1 Audio-Visual Basic Scene Descriptors 115
8.4.2 Audio-Visual Basic Scene Geometry 117
8.4.3 Audio-Visual Event Descriptors 118
8.4.4 Audio-Visual Scene Descriptors 120
8.4.5 Audio-Visual Scene Geometry 121
8.4.6 Instance Identifier 122
8.4.7 Point of View 123
8.4.8 Selector 124
8.4.9 Space Time 125
8.4.10 Spatial Attitude 126
8.4.11 Time 127

8.4.12 Visual Basic Scene Descriptors 127
8.4.13 Visual Basic Scene Geometry 128
8.4.14 Visual Object 129
8.4.15 Visual Scene Descriptors 131
8.4.16 Visual Scene Geometry 132
8.5 Data Types from MPAI-PAF 133
8.5.1 Avatar 133
8.5.2 Body Descriptors 134
8.5.3 Face Descriptors 134
8.5.4 Portable Avatar 137
9 Profiles 138

1 Foreword

The international, unaffiliated, non-profit ***Moving Picture, Audio, and Data Coding by Artificial Intelligence (MPAI)*** organisation was established in September 2020 in the context of:

1. **Increasing** use of Artificial Intelligence (AI) technologies applied to a broad range of domains affecting millions of people
2. **Marginal** reliance on standards in the development of those AI applications
3. **Unprecedented** impact exerted by standards on the digital media industry affecting billions of people

believing that AI-based data coding standards will have a similar positive impact on the Information and Communication Technology industry.

The design principles of the MPAI organisation as established by the MPAI Statutes are the development of AI-based Data Coding standards in pursuit of the following policies:

1. Publish upfront clear Intellectual Property Rights licensing frameworks.
2. Adhere to a rigorous standard development process.
3. Be friendly to the AI context but, to the extent possible, remain agnostic to the technology thus allowing developers freedom in the selection of the more appropriate – AI or Data Processing – technologies for their needs.
4. Be attractive to different industries, end users, and regulators.
5. Address five standardisation areas:
    1. *Data Type*, a particular type of Data, e.g., Audio, Visual, Object, Scenes, and Descriptors with as clear semantics as possible.
    2. *Qualifier*, specialised Metadata conveying information on Sub-Types, Formats, and Attributes of a Data Type.
    3. *AI Module* (AIM), processing elements with identified functions and input/output Data Types.
    4. *AI Workflow* (AIW), MPAI-specified configurations of AIMs with identified functions and input/output Data Types.
    5. *AI Framework* (AIF), an environment enabling dynamic configuration, initialisation, execution, and control of AIWs.
6. Provide appropriate Governance of the ecosystem created by MPAI Technical Specifications enabling users to:
    1. *Operate* Reference Software Implementations of MPAI Technical Specifications provided together with Reference Software Specifications
    2. *Test* the conformance of an implementation with a Technical Specification using the Conformance Testing Specification.
    3. *Assess* the performance of an implementation of a Technical Specification using the Performance Assessment Specification.

4. *Obtain* conforming implementations possibly with a performance assessment report from a trusted source through the MPAI Store.

Today, the MPAI organisation operated on four solid pillars:

1. The MPAI Patent Policy specifies the MPAI standard development process and the Framework Licence development guidelines.
2. *Technical Specification: Artificial Intelligence Framework (MPAI-AIF)* V2.1 specifies an environment enabling initialisation, dynamic configuration, and control of AIWs in the standard AI Framework environment depicted in Figure 1. An AI Framework can execute AI applications called AI Workflows (AIW) typically including interconnected AI Modules (AIM). MPAI-AIF supports small- and large-scale high-performance components and promotes solutions with improved explainability.

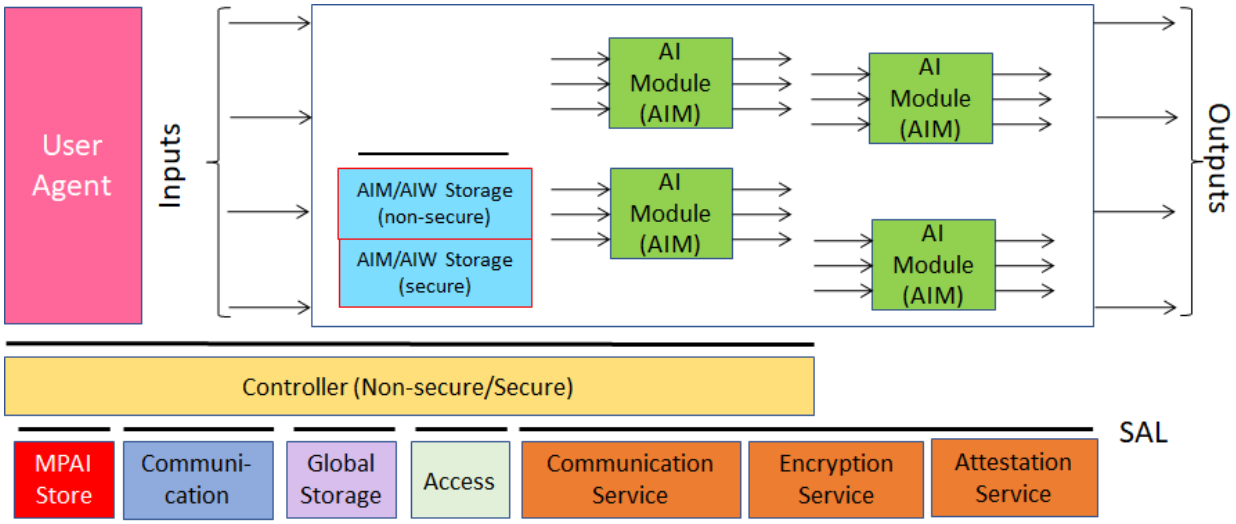

Figure 1 – The AI Framework (MPAI-AIF) V2 Reference Model

3. ***Technical Specification: Data Types, Formats, and Attributes (MPAI-TFA) V1.2*** specifies Qualifiers, a type of metadata supporting the operation of AIMs receiving data from other AIMs. Qualifiers convey information on Sub-Types (e.g., the type of colour), Formats (e.g., the type of compression and transport), and Attributes (e.g., semantic information in the Content). Although Qualifiers are human-readable, they are only intended to be used by AIMs. Therefore, Text, Speech, Audio, Visual, and other Data exchanged by AIWs and AIMs should be interpreted as being composed of Content (Text, Speech, Audio, and Visual as appropriate) and associated Qualifiers. Therefore a Text Object is composed of Text Data and Text Qualifier. The specification of most MPAI Data Types reflects this point.
4. ***Technical Specification: Governance of the MPAI Ecosystem (MPAI-GME) V1.1*** defines the following elements:
    1. Standards, i.e., the ensemble of Technical Specifications, Reference Software, Conformance Testing, and Performance Assessment.
    2. Developers of MPAI-specified AIMs and Integrators of MPAI-specified AIWS (Implementers).
    3. MPAI Store in charge of making AIMs and AIWs submitted by Implementers available to Integrators and End Users.
    4. Performance Assessors, independent entities assessing the performance of implementations in terms of Reliability, Replicability, Robustness, and Fairness.
    5. End Users.

The interaction between and among actors of the MPAI Ecosystem are depicted in Figure 2.

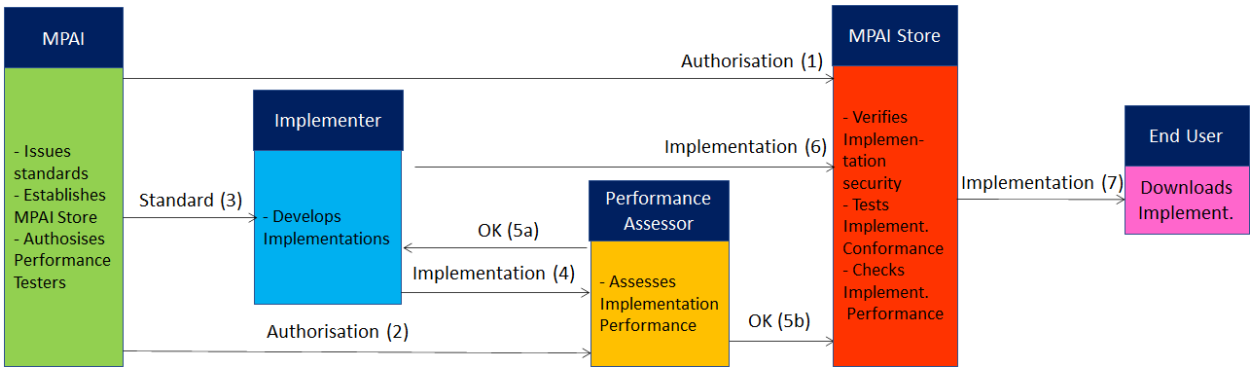

Figure 2 – The MPAI Ecosystem

2 Introduction (Informative)

Technical Specification: Human and Machine Communication (MPAI-HMC) V2.0 specifies two AI Modules and reuses AI Modules and Data Types from other MPAI Technical Specifications to enable advanced forms of communication between humans and machines and applies those AI Modules and Data Types to the Communicating Entities in Context (HMC-CEC) Use Case. The referenced Technical Specifications are:

1. Two foundational MPAI Technical Specifications:
    1. AI Framework (MPAI-AIF) V2.1.
    2. Data Types, Formats, and Qualifiers (MPAI-TFA) V1.2
2. Four application MPAI Technical Specifications:
    1. Context-based Audio Enhancement (MPAI-CAE) V2.3.
    2. Multimodal Conversation (MPAI-MMC) V2.3.
    3. Object and Scene Description (MPAI-OSD) V1.2.
    4. Portable Avatar Format (MPAI-PAF) V1.3.
3. One ancillary MPAI Technical Specification AI Module Profiles (MPAI-PRF) V1.0.

Capitalised terms are defined in Table 1 if they are MPAI-HMC-specific and in the Definitions web page which contains definitions of terms used in all current MPAI Technical Specifications. Non-capitalised terms have the commonly intended meaning.

The MPAI-HMC V2.0 Technical Specification is web-based. All referenced entities are accessible via web links. The document version retains the links but includes the specification of the directly referenced entities. Access to JSON is only provided via web links.

This Chapter is Informative. The Chapters and Sections of this Technical Specification are Normative unless they are explicitly identified as Informative.

3 Scope

Technical Specification: Human and Machine Communication (MPAI-HMC) V2.0 – referred to in the following as MPAI-HMC V2.0 or MPAI-HMC – specifies technologies that enable advanced forms of communication between humans in a real space or represented in a Virtual Space, and Machines represented as humanoids in a Virtual Space or rendered as humanoids in a real space.

The HMC communicating parties are generically called *Entities*. To the extent possible, MPAI-HMC strives to be neutral to the nature – real or virtual – of an Entity.

The word *Communication* is to be intended to mean that an Entity should have the capability to understand the semantics of the media involved – Text, Speech, Audio, Visual, and 3D Models. Different MPAI-HMC implementations may provide different levels of understanding and

responding. At this state, MPAI-HMC does not provide the means to assess an Entity's capability to understand and provide pertinent responses.

MPAI-HMC V2.0 applies the technologies from HMC (AI Modules) and other MPAI Technical Specifications (AI Modules and Data Types) to the *Communicating Entities in Context* Use Case as a first instance offering effective forms of human-Machine communication that can be implemented in an interoperable fashion.

MPAI-HMC has been developed by CAE-DC, MMC-DC, PAF-DC, and the MPAI-OSD group of the Requirements Standing Committee.

MPAI may develop future versions of this Technical Specification or new Technical Specifications related to the Human and Machine Communication area.

4 Definitions

Capitalised terms are defined in Table 1 if they are MPAI-HMC-specific and in Definitions which defines the terms used in all current MPAI Technical Specifications. Non-capitalised terms have the commonly intended meaning.
As HMC technologies are mostly specified by other Technical Specifications, Table 1 includes a limited number of definitions. All other terms are defined by Definitions.

Table 1 – General MPAI-HMC terms

Term	Definition
Communication Item	An element generated by a Machine communicating with an Entity expressed by a Portable Avatar.
Context	Information describing the attributes and the environment of an Entity, such as language, culture etc.
Culture	The collection of ideas, language, customs, and social behaviour governing the way a human, or a group of humans perceive, express and behave.

5 References

This page provides normative and information references. The full set of non-MPAI normative references can be accessed online.

5.1 Normative References

1. MPAI; Technical Specification: Governance of the MPAI Ecosystem (MPAI-GME) V1.1.
2. MPAI; Technical Specification: Artificial Intelligence Framework (MPAI-AIF) V2.1.
3. MPAI; Technical Specification: Context-based Audio Enhancement (MPAI-CAE) V2.3.
4. MPAI; Technical Specification: Multimodal Conversation (MPAI-MMC) V2.3.
5. MPAI; Technical Specification: Object and Scene Description (MPAI-OSD) V1.2.
6. MPAI; Technical Specification: Portable Avatar Format (MPAI-PAF) V1.3.
7. MPAI; Technical Specification: AI Module Profiles (MPAI-PRF) V1.0.
8. MPAI; Technical Specification: Data Types, Formats, and Attributes (MPAI-TFA) V1.2.

5.2 Informative References

9. MPAI; The MPAI Statutes.
10. MPAI; The MPAI Patent Policy.

11. Technical Specification: Compression and Understanding of Industrial Data (MPAI-CUI) V1.1.
12. MPAI; Technical Specification: Connected Autonomous Vehicle (MPAI-MMM) – Architecture V1.1.
13. MPAI; Technical Specification: Connected Autonomous Vehicle (MPAI-MMM) – Technologies V1.0.
14. MPAI; Technical Specification: MPAI Metaverse Model (MPAI-MMM) – Architecture V1.2.
15. MPAI; Technical Specification: MPAI Metaverse Model (MPAI-MMM) – Technologies V1.0.
16. MPAI; Technical Specification: Neural Network Watermarking (MPAI-NNW) V1.0.
17. MPAI; Technical Specification: Neural Network Traceability (MPAI-NNW) V1.0.

6 AI Workflows

6.1 Technical Specification

Technical Specification: Multimodal Conversation (MPAI-HMC) V2.0 assumes that workflows are based on *Technical Specification: AI Framework (MPAI-AIF) V2.1* specifying the standard AI Framework (AIF) that enables initialisation, dynamic configuration, execution, and control of AI Workflows (AIW) composed of interconnected AI Modules (AIM).
Table 1 provides the link to the currently specified AI Workflow. The provides the AIW function, reference model, Input/Output Data, Functions of AIMs used, Input/Output Data of AIMs, and links to the AIW-related AIMs, and JSON metadata.
All AI-Workflows specified by MPAI-HMC V2.0 supersede those specified by previous MPAI-MMC specifications which can still be used if their version is explicitly indicated.

Table 1 – AIWs of MPAI-HMC V2.0

Acronym	Title	JSON
MPAI-HMC	**Communicating Entities in Context**	**X**

6.2 Reference Software

As a rule, MPAI provides Reference Software implementing the Technical Specification released with the BSD-3-Clause licence and the following disclaimers:

1. The purpose of the Reference Software is to demonstrate a working Implementation of an AIW, not to provide a ready-to-use product.
2. MPAI disclaims the suitability of the Software for any other purposes that those of the MPAI-HMC Standard and does not guarantee that it is secure.
3. Users shall verify that they have the right to use any third-party software required by the Reference Software Implementation.
4. Users should note that the Reference Software Implementation may require the acceptance of licences from third-party repositories.

Note that, at this stage, the MPAI-HMC AIW is only partly implemented.

6.3 Conformance Testing

An implementation of an AI Workflow conforms with MPAI-HMC if it accepts as input and produces as output Data and/or Data Objects (Data and its Qualifier) conforming with those specified or referenced by MPAI-HMC.

The Conformance of an instance of a Data is to be expressed by a sentence like "Data validates against the Data Type Schema". This means that:

- Any Data Sub-Type is as indicated in the Qualifier.
- The Data Format is indicated by the Qualifier.
- Any File and/or Stream have the Formats indicated by the Qualifier.
- Any Attribute of the Data is of the type or validates against the Schema specified in the Qualifier.

The method to Test the Conformance of a Data or Data Object instance is specified in the *Data Types* chapter.

6.4 Performance Assessment

Performance is a multidimensional entity because it can have various connotations. Therefore, the Performance Assessment Specification provides methods to measure how well an AIW performs its function, using a metric that depends on the nature of the function, such as:

1. Quality: the Performance of a Communicating Entities in Context AIW can measure how well the AIW responds to a question.
2. Bias: Performance of a Communicating Entities in Context AIW can measure the quality of responses in dependence of the type of message received.
3. Legal compliance: the Performance of an AIW can measure the compliance of the AIW to a regulation, e.g., the European AI Act.
4. Ethical compliance: the Performance Assessment of an AIW can measure the compliance of an AIW to a target ethical standard.

Note that, at this stage, a limited number of MPAI-HMC AIMs include a Performance Assessment Specification.

7 AI Modules

7.1 General

7.1.1 Technical Specifications

Table 1 provides the links to the specifications and the JSON syntax of all AIMs specified by ***Technical Specification: Human and Machine Communication (MPAI-HMC) V2.0***. The MPAI-HMC V2.0 AI-Modules supersede those specified by earlier versions than V2.0. They may still be used if their version is explicitly signaled. The Composite AIMs are in **bold**.

Table 1 – Basic and Composite AI Modules

Acronym	Specification	JSON
HMC-ECU	**Entity and Context Understanding**	**X**
HMC-SID	**AV Scene Integration and Description**	**X**

Table 2 provides the full list of with web links to the AI Modules utilised by HMC-CEC organised according to the Technical Specifications specifying them.

Table 2 - AI Modules utilised by HMC-CEC organised by Technical Specifications

CAE	MMC	OSD	PAF
Audio Analysis Transform	Automatic Speech Recognition	Audio-Visual Alignment	Audio-Visual Scene Rendering

Audio Descriptors Multiplexing	Entity Dialogue Processing	Audio-Visual Event Description	Face Identity Recognition
Audio Object Identification	Entity Speech Description	Audio-Visual Scene Demultiplexing	Entity Body Description
Audio Scene Description	Entity Text Description	**Audio-Visual Scene Description**	Entity Face Description
Audio Separation and Enhancement	Natural Language Understanding	Visual Direction Identification	Portable Avatar Demultiplexing
Audio Source Localisation	**Personal Status Extraction**	Visual Instance Identification	PS-Face Interpretation
Audio Synthesis Transform	Personal Status Multiplexing	Visual Object Extraction	PS-Gesture Interpretation
HMC	PS-Speech Interpretation	**Visual Object Identification**	**Personal Status Display**
AV Scene Integration and Description	PS-Text Interpretation	Visual Scene Description	Portable Avatar Multiplexing
Entity and Context Understanding	Speaker Identity Recognition		
	Text and Speech Translation		
	Text-To-Speech		
	Text-to-Text Translation		

7.1.2 Reference Software

As a rule, MPAI provides Reference Software implementing the AI Modules released with the BSD-3-Clause licence and the following disclaimers:

1. The purpose of the Reference Software is to provide a working Implementation of an AIM, not a ready-to-use product.
2. MPAI disclaims the suitability of the Reference Software for any other purposes than those of the MPAI-HMC Standard and does not guarantee that it offers the best performance and that it is secure.
3. Users shall verify that they have the right to use any third-party software required by the Reference Software, e.g., by accepting the licences from third-party repositories.

This edition includes only part of the MPAI-HMC AIMs Reference Software Implementation.

7.1.3 Conformance Testing

An implementation of an AI Module conforms with MPAI-HMC if it accepts as input and produces as output Data and/or Data Objects (combination of Data of a certain Data Type and its Qualifier) conforming with those specified by all relevant MPAI Technical Specifications.

The Conformance of an instance of a Data is to be expressed by a sentence like "Data validates against the Data Type Schema". This means that:

- Any Data has the specified type.
- Any Data Sub-Type is as indicated in the Qualifier.
- The Data Format is indicated by the Qualifier.
- Any File and/or Stream have the Formats indicated by the Qualifier.

- Any Attribute of the Data is of the type or validates against the Schema specified in the Qualifier.

The method to Test the Conformance of a Data or Data Object instance is specified in the *Data Types* chapter.

7.1.4 Performance Assessment

Performance is a multidimensional entity because it can have various connotations. Therefore, the Performance Assessment Specification should provide methods to measure how well an AIM performs its function, using a metric that depends on the nature of the function, such as:

1. *Quality*: Performance Assessment measures how well an AIM performs its function, using a metric that depends on the nature of the function, e.g., the word error rate (WER) of an Automatic Speech Recognition (ASR) AIM.
2. *Bias*: Performance Assessment measures how well an AIM performs its function, using a metric that depends on a bias related to certain attributes of the AIM. For instance, an ASR AIM tends to have a higher WER when the speaker is from a particular geographic area.
3. *Legal* compliance: Performance Assessment measures how well an AIM performs its function, using a metric that assesses its accordance with a certain legal standard.
4. *Ethical* compliance: the Performance Assessment of an AIM can measure the compliance of an AIM to a target ethical standard.

Note that the current MPAI-HMC V2.0 Technical Specification provides AIM Performance Assessment methods for a limited number of AIMs.

7.2 AI modules from MPAI-CAE

7.2.1 Audio Analysis Transform

7.2.1.1 Functions

Audio Analysis Transform (CAE-AAT):

Receives	*Audio Object*	*As Multichannel Audio*
Transforms	Multichannel Audio	into frequency bands via a Fast Fourier Transform (FFT). The following operations are carried out in discrete frequency bands. When such a configuration is used, a 50% overlap between subsequent audio blocks needs to be employed. The output is a data structure comprising complex valued audio samples in the frequency domain.
Produces	*Audio* Object	In the Transform domain.

7.2.1.2 Reference Model

Figure 1 depicts the Reference Architecture of the Audio Analysis Transform (CAE-AAT) AIM.

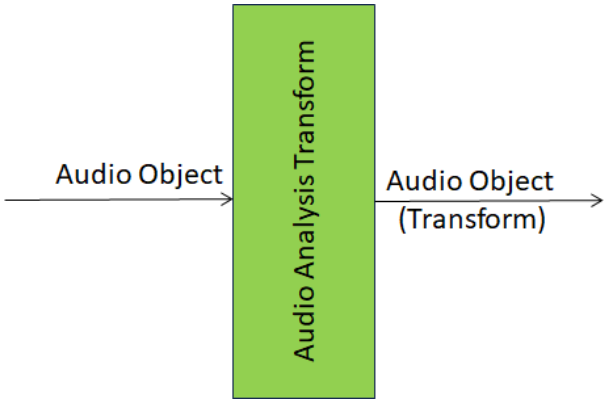

Figure 1 – Audio Analysis Transform (CAE-AAT) AIM

7.2.1.3 Input/Output Data

Table 1 specifies the Input and Output Data of the Audio Analysis Transform (CAE-AAT) AIM.

Table 1 – Audio Analysis Transform (CAE-AAT) AIM

Input	Description
Audio Object	Audio Object (with associated Microphone Array info)
Output	**Description**
Audio Object (Transform)	The result of the application of the Fast Fourier Transform to Multichannel Audio.

7.2.1.4 JSON Metadata

https://schemas.mpai.community/CAE1/V2.3/AIMs/AudioAnalysisTransform.json

7.2.1.5 Conformance Testing

The following steps shall be followed when testing the Conformance of a CAE-AAT AIM instance.

1. Use the following datasets:
    1. DS1: *n* Test Audio Object files including Multichannel Audio as Interleaved Multichannel Audio format.
    2. DS2: *n* Expected Audio Object Output files including data in Transform Interleaved Multichannel Audio format.
2. Feed the AIM under test with the Test files (DS1).
3. Perform the following steps to analyse the Audio Object (Transform) with the Expected Audio Objects (DS2):
    1. Check the data format of the Audio Object (Transform) with the format of the given Expected Audio Objects.
    2. Calculate the peak-to-peak Amplitude (A) of each Audio block in the Expected Audio Objects.
    3. Calculate the RMSE of each Audio block by comparing the Audio Object (Transform) (*x*) with the Expected Audio Objects (*y*).
    4. Accept the AIM under test if, for each audio block, these two conditions are satisfied:
        1. Data format of the Audio Object (Transform) is the same as the format of the Expected Audio Object and
        2. RMSE < A* 0.1%.
4. The Conformance Tester will provide the following matrix containing a limited number of input records (*n*) with the corresponding outputs. If an input record fails, the tester would specify the reason why the test case fails.

Input data (DS1)	Expected Output Data (DS2)	Data Format	RMSE
Audio Object (Microphone Array) ID_1	Audio Object (Transform) ID_1	T/F	*< A*0.1%*
Audio Object (Microphone Array) ID_2	Audio Object (Transform) ID_2	T/F	*< A*0.1%*
Audio Object (Microphone Array) ID_3	Audio Object (Transform) ID_3	T/F	*< A*0.1%*
…	…	…	…
Audio Object (Microphone Array) ID_n	Audio Object (Transform) ID_n	T/F	*< A*0.1%*

5. Final evaluation: T/F Denoting with *i*, the record number in DS1 and DS2, the matrices reflect the results obtained with input records i with the corresponding outputs i.

DS1	DS2	Audio Object (Transform) output value (from AIM under test)
DS1[*i*]	DS2[*i*]	Audio Object (Transform)[*i*]

Table 2 provides the Conformance Testing Method for the formats of the CAE-AAT AIM output.
Note: If a schema contains references to other schemas, conformance of data for the primary schema implies that any data referencing a secondary schema shall also validate against the relevant schema, if present and conform with the Qualifier, if present.

Table 2 – Conformance Testing Method for CAE-AAT AIM

Receives	Audio Object (Microphone Array)	Shall validate against Audio Object schema. Audio Data shall conform with Audio Qualifier.
Produces	Audio Object (Transform)	Shall validate against Audio Object schema. Audio Data shall conform with Audio Qualifier.

7.2.2 Audio Descriptors Multiplexing

7.2.2.1 Functions

Audio Descriptor Multiplexing (CAE-AMX):

Receives	*Enhanced Audio Objects*	Audio Objects with reduced noise.
	Audio Scene Geometry	The spatial arrangement of Audio Objects.
Multiplexes	*Enhanced Audio Objects*	
	Audio Scene Geometry	
Produces	*Audio Scene Descriptors*	The Descriptors of the Audio Scene.

7.2.2.2 Reference Model

Figure 1 depicts the Reference Architecture of the Audio Descriptor Multiplexing AIM.

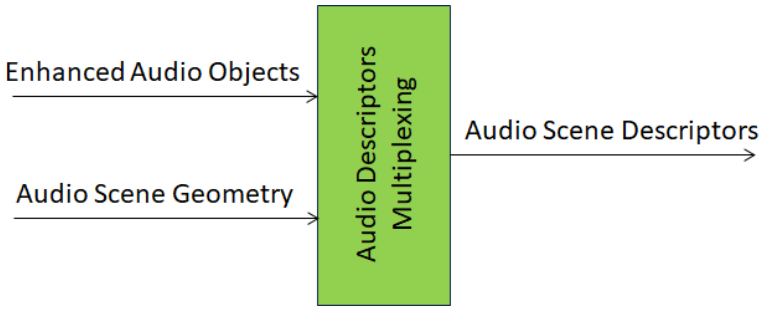

Figure 1 – The Audio Descriptor Multiplexing AIM

7.2.2.3 Input/Output Data

Table 1 specifies the Input and Output Data of the Audio Descriptor Multiplexing AIM.

Table 1 – I/O Data of Audio Descriptor Multiplexing

Input	Description

Enhanced Audio Object	Time-domain Audio Objects without noise.
Audio Scene Geometry	The Space-Time arrangement of Audio objects in an Audio Scene
Output	**Description**
Audio Scene Descriptors	The combination of Audio Scene Geometry and Audio Objects.

7.2.2.4 JSON Metadata

https://schemas.mpai.community/CAE1/V2.3/AIMs/AudioDescriptorsMultiplexing.json

7.2.2.5 Conformance Testing

The following steps shall be followed when testing the Conformance of a CAE-AMX AIM instance.

1. Use the following datasets:
    1. DS1: *n* Enhanced Audio Objects Test files.
    2. DS3: *n* Audio Scene Geometry of the Enhanced Audio Objects.
    3. DS4: *n* Expected Output Audio Scene Descriptors.
2. Feed the AIM under test with the Test files (DS1, DS3).
3. Analyse the Audio Scene Descriptors with the Expected Audio Scene Descriptors (DS4).
    1. Check the Audio Scene Descriptors with Expected given Audio Scene Descriptors.
    2. Calculate the peak-to-peak Amplitude (A) of each Audio block in the Expected Audio Scene Descriptors.
    3. Calculate the RMSE of each Audio block by comparing the output (*x*) with the Audio Scene Descriptors (*y*) Audio blocks.
    4. Accept the CAE-AMX AIM under test if, for each audio block, these the two conditions are satisfied:
        1. Data format of Audio Scene Descriptors is the same as the Expected Audio Scene Descriptors and
        2. RMSE < A * 0.1%
4. The Conformance Tester will provide the following matrix with a limited number of input records (*n*) with the corresponding outputs. If an input record fails, the tester would specify the reason why the test case fails.

Input data (DS1, DS3)	Expected Output Data (DS4)	Data Format	RMSE
Enhanced Audio Objects ID_1 Audio Scene Geometry ID_1	Audio Scene Descriptors ID_1	T/F	< A * 0.1%
Enhanced Audio Objects ID_2 Audio Scene Geometry ID_2	Audio Scene Descriptors ID_2	T/F	< A * 0.1%
Enhanced Audio Objects ID_3 Audio Scene Geometry ID_3	Audio Scene Descriptors ID_3	T/F	< A * 0.1%
…	…		…
Enhanced Audio Objects ID_n Audio Scene Geometry ID_n	Audio Scene Descriptors ID_n	T/F	< A * 0.1%

5. Final evaluation: T/F Denoting with *i*, the record number in DS1 and DS3, the matrices reflect the results obtained with input records i with the corresponding outputs i.

DS1	DS3	DS4	CAE-AMX output value (obtained through the AIM under test)

DS1[*i*]	DS3[i]	DS4[i]	Multiplexer[*i*]

Table 2 provides the Conformance Testing Method for the formats of the CAE-AMX AIM output. Note: If a schema contains references to other schemas, conformance of data for the primary schema implies that any data referencing a secondary schema shall also validate against the relevant schema, if present and conform with the Qualifier, if present.

Table 2 – Conformance Testing Method for CAE-AMX AIM

Receives	Enhanced Audio Objects	Shall validate against Audio Object schema. Audio Data shall conform with Audio Qualifier.
	Audio Scene Geometry	Shall validate against Audio Basic Scene Geometry schema.
Produces	Audio Scene Descriptors	Shall validate against Audio Basic Scene Descriptors schema.

7.2.3 Audio Object Identification

7.2.3.1 Functions

Audio Object Identification (CAE-AOI):

Receives	*Audio Scene Geometry*	The spatial arrangements of the Audio Objects.
	Audio Objects	*The individual input Audio Objects*
Identifies	The Audio Objects.	
Produces	The *Audio Instance IDs*	The Instance Identifier of the Audio Objects.

7.2.3.2 Reference Model

Figure 1 depicts the Reference Architecture of the Audio Object Identification AIM.

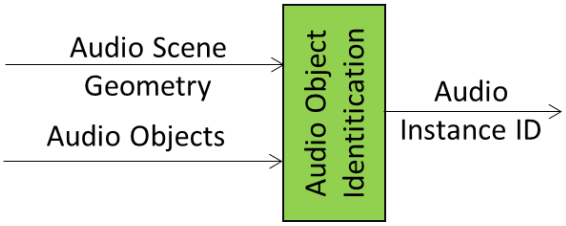

Figure 1 – Audio Object Identification AIM

The Audio Object Identification AIM shall be able to parse either an Audio-Visual Scene Geometry or its Audio Scene Geometry subset.

7.2.3.3 Input/Output Data

Table 1 specifies the Input and Output Data of the Audio Object Identification AIM.

Table 1 – I/O Data of the Audio Object Identification AIM

Input	Description
Audio Scene Geometry	The digital representation of the spatial arrangement of the Visual Objects of the Scene.
Audio Objects	The Audio Objects in the Audio Scene Geometry with an identifiable source target of identification.

Output	Description
Audio Instance Identifier	The Instance Identifier of the specific Audio Object.

7.2.3.4 JSON Metadata

https://schemas.mpai.community/CAE1/V2.3/AIMs/AudioObjectIdentification.json

7.2.3.5 Conformance Testing

Table 2 provides the Conformance Testing Method for the CAE-AOI AIM. Conformance Testing of the individual AIMs of the CAE-AOI AIM are given by the individual AIM Specification.

If a schema contains references to other schemas, conformance of data for the primary schema implies that any data referencing a secondary schema shall also validate against the relevant schema, if present and conform with the Qualifier, if present.

Table 2 – Conformance Testing Method for CAE-AOI AIM

Receives	Audio Scene Geometry	Shall validate against Audio Basic Scene Geometry schema.
	Audio Objects	Shall validate against Audio Object schema. Audio Data shall conform with Audio Qualifier.
Produces	The Audio Instance IDs	Shall validate against Instance ID schema.

7.2.4 Audio Scene Description

7.2.4.1 Functions

Audio Scene Description (CAE-ASD):

Receives	Space-Time	Audio Scene's Space-Time.
	Audio Object	Input Audio Object.
	Audio Scene Descriptors	Possible Audio Scene to be augmented.
Computes	Audio Objects	Inside the input Audio Object.
	Audio Scene Geometry	Possible Audio Scene to be augmented.
Adds	Audio Scene Descriptors	To make the new Audio Scene.
Produces	Audio Scene Descriptors (output)	The augmented Audio Scene.

Unlike CAE-ABS that can only describe an Audio Scene composed of Audio Objects, CAE-ASD can describe an Audio Scene in terms of Audio Objects *and* Audio Scenes. In their turn, Audio Scene can be composed of Objects and Scenes.

7.2.4.2 Reference Model

Figure 1 depicts the Reference Architecture of the Audio Scene Description AIM.

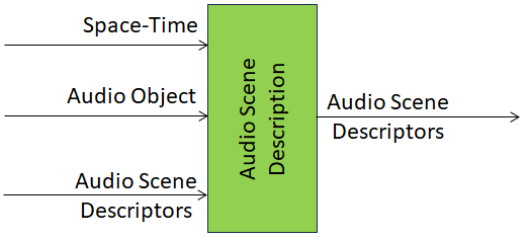

Figure 1 – The Audio Scene Description (CAE-ASD) AIM

7.2.4.3 Input/Output Data

Table 1 specifies the Input and Output Data of the Audio Scene Description AIM.

Table 1 – I/O Data of the Audio Scene Description (CAE-ASD) AIM

Input	Description
Space-Time	Audio Objects Space-Time information.
Audio Object	Multichannel Audio (with associated Microphone Array info)
Audio Scene Descriptors	Input Audio Scene Descriptors
Output	**Description**
Audio Scene Descriptors	Output Audio Scene Descriptors

7.2.4.4 SubAIMs

Audio Scene Description (CAE-ASD) is a Composite AIM with the structure depicted in Figure 2.

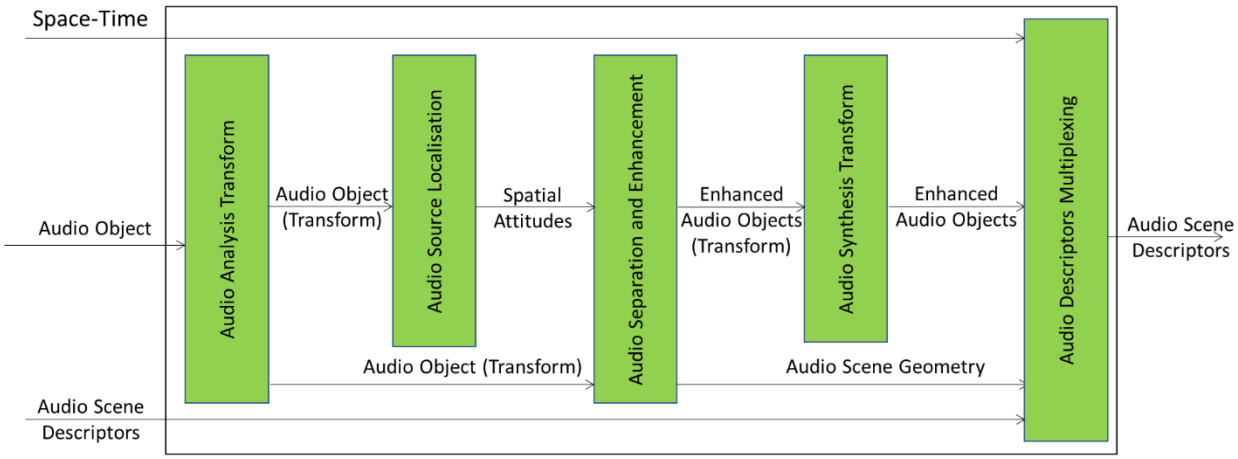

Figure 2 - The Audio Scene Description (CAE-ASD) Composite AIM

The specification of the CAE-ASD Basic AIMs are provided at the links of Table 2.

Table 2 – BASIC AIMs of Audio Scene Descriptors

AIW	AIMs	Names	JSON
CAE-ASD		Audio Scene Description	File
	CAE-AAT	Audio Analysis Transform	File
	CAE-ASL	Audio Source Localisation	File
	CAE-ASE	Audio Separation and Enhancement	File
	CAE-AST	Audio Synthesis Transform	File
	CAE-ADM	Audio Descriptors Multiplexing	File

7.2.4.5 JSON Metadata

https://schemas.mpai.community/CAE1/V2.3/AIMs/AudioSceneDescription.json

7.2.4.6 Conformance Testing

Table 2 provides the Conformance Testing Method for the CAE-ASD AIM. The Conformance Testing Method for the individual Basic AIMs of the CAE-ASD Composite AIM is provided by the individual Basic AIMs.

If a schema contains references to other schemas, conformance of data for the primary schema implies that any data referencing a secondary schema shall also validate against the relevant schema, if present and conform with the Qualifier, if present.

Table 2 – Conformance Testing Method for CAE-ASD AIM

Receives	Space-Time	Shall validate against Space-Time Schema.
	Audio Object	Shall validate against Audio-Object Schema. Audio Data shall conform with Audio Data Qualifier.
	Audio Scene Descriptors	Shall validate against Audio Scene Descriptors Schema.
Produces	Audio Scene Descriptors	Shall validate against Audio Scene Descriptors Schema.

7.2.5 Audio Separation and Enhancement

7.2.5.1 Functions

Audio Separation and Enhancement (CAE-ASE):

Receives	*Audio Object*	in the Transform domain.
	Audio Spatial Attitudes	Spatial Attitudes of the input Audio Objects.
Separates	*Audio Objects*	by using their Spatial Attitudes.
Produces	*Enhanced Audio Object*	in the Transform domain.
	Audio Scene Geometry	The Geometry of Audio Objects in the Scene.

7.2.5.2 Reference Model

Figure 1 depicts the Reference Architecture of the Audio Separation and Enhancement AIM.

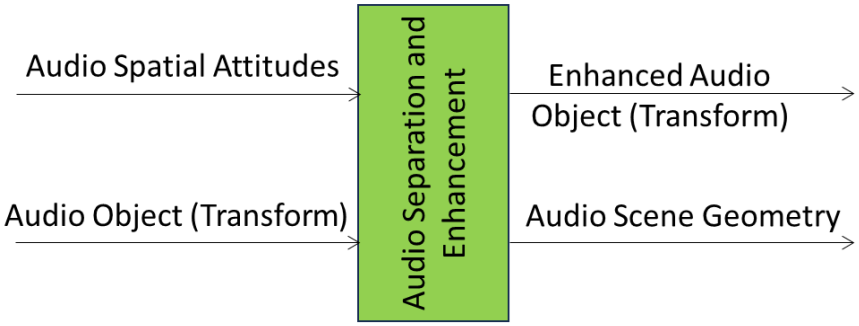

Figure 1 – Audio Separation and Enhancement AIM

7.2.5.3 Input/Output Data

Table 11 specifies the Input and Output Data of the Audio Separation and Enhancement AIM.

Table 1 – I/O Data of Audio Separation and Enhancement

Input	Description
Audio Object	The result of the application of the Fast Fourier Transform to the Multichannel Audio.
Audio Spatial Attitudes	The Spatial Attitudes of Audio Objects.
Output	**Description**
Enhanced Audio Object	Enhanced Multichannel Audio in the transform domain.
Audio Scene Geometry	The spatial arrangement of the Audio Objects.

7.2.5.4 JSON Metadata

https://schemas.mpai.community/CAE/V2.3/AIMs/AudioSeparationAndEnhancement.json

7.2.5.5 Conformance Testing

The following steps shall be followed when testing the Conformance of a CAE-ASE AIM instance.

1. Use the following datasets:
    1. DS1: *n* Test files containing Audio Objects (Transform).
    2. DS4: *n* Test files containing the Spatial Attitudes of Audio Objects.
    3. DS2: *n* Expected Enhanced Audio Objects (Transform) Files.
    4. DS3: *n* Expected Audio Scene Geometries.
2. Feed the AIM under test with the Test Audio Objects (Transform) and Spatial Attitudes.
3. Analyse the Audio Scene Geometry and Enhanced Audio (Transform).
    1. Control the Audio Scene Geometry with the Expected Audio Scene Geometry:
        1. Count the number of Audio Objects in the Audio Scene Geometry.
        2. Calculate the angle difference (AD) in degrees between the Audio Objects (u) in the Audio Scene Geometry and the Audio Objects (v) in the Expected Audio Scene Geometry.
    2. Compare the number of Audio Blocks in the Expected Audio Objects with the number of Audio Blocks in the Audio Objects (Transform).

3. Calculate Signal to Interference Ratio (SIR), Signal to Distortion Ratio (SDR), and Signal to Artefacts Ratio (SAR) between the Expected Audio Objects (Transform) and Output Audio Objects (Transform).
4. Accept the CAE-ASE AIM under test if these four conditions are satisfied:
    1. The number of Audio Objects (Transform) in the Audio Scene Geometry is equal to the number of Audio Objects (Transform) in the Expected Audio Scene Geometry.
    2. The number of Audio Blocks in the Audio Objects (Transform) is equal to the number of Audio Blocks in the Expected Audio Objects (Transform).
    3. Compare each Audio Objects (Transform) in the Audio Scene Geometry with the Audio Objects (Transform) in the Expected Audio Scene Geometry.
        1. Each Audio Objects (Transform)'s AD between the Expected and Output is less than 5 degrees.
    4. Compare each Audio Objects (Transform) with the Audio Objects (Transform) in the Expected Audio Objects (Transform).
        1. If the room reverb time (T60) is greater than 0.5 seconds.
            1. Each object's SIR between the Expected and Output is greater than or equal to 10 dB.
            2. Each object's SDR between the Expected and Output is greater than or equal to 3 dB.
            3. Each object's SAR between the Expected and Output is greater than or equal to 3 dB.
        2. If the room reverb time (T60) is less than 0.5 seconds.
            1. Each object's SIR between the Expected and Output is greater than or equal to 15 dB.
            2. Each object's SDR between the Expected and Output is greater than or equal to 6 dB.
4. The Conformance Tester will provide the following matrix containing a limited number of input records (*n*) with the corresponding outputs. If an input record fails, the tester would specify the reason why the test case fails.

Input data (DS1, DS4)	**Expected Output Data (DS2, DS3)**	**Data Format**	**Audio Scene Geometry**	**Source Separation Metrics**
Spatial Attitude (ID_1) Audio Object (Transform) ID_1	Enhanced Audio Object (Transform) ID_1 Audio Scene Geometry ID_1	T/F	T/F	T/F
Spatial Attitude (ID_2) Audio Object (Transform) ID_2	Enhanced Audio Object (Transform) ID_2 Audio Scene Geometry ID_2	T/F	T/F	T/F
Spatial Attitude (ID_3) Audio Object (Transform) ID_3	Enhanced Audio Object (Transform ID_3 Audio Scene Geometry ID_3	T/F	T/F	T/F
…	…	…	…	…
Spatial Attitude (ID_n) Audio Object (Transform) ID_n	Enhanced Audio Object (Transform ID_n Audio Scene Geometry ID_n	T/F	T/F	T/F

6. Final evaluation : T/F Denoting with *i*, the record number in DS1, DS2, and DS3, the matrices reflect the results obtained with input records i with the corresponding outputs i.

DS1	DS2	DS3	Sound Field Description output value (obtained through the AIM under test)
DS1[*i*]	DS2[*i*]	DS3[i]	SpeechDetectionandSeparation[*i*]

Table 2 provides the Conformance Testing Method for the formats of the CAE-ASE AIM output. Note: If a schema contains references to other schemas, conformance of data for the primary schema implies that any data referencing a secondary schema shall also validate against the relevant schema, if present and conform with the Qualifier, if present.

Table 2 – Conformance Testing Method for CAE-ASE AIM

Receives	Audio Object	Shall validate against Audio Object schema. Audio Data shall conform with Audio Qualifier.
	Audio Spatial Attitudes	Shall validate against Spatial Attitude schema.
Produces	Enhanced Audio Object	Shall validate against Audio Object schema. Audio Data shall conform with Audio Qualifier.
	Audio Scene Geometry	Shall validate against Audio Basic Scene Geometry schema.

7.2.6 Audio Source Localisation

7.2.6.1 Functions

Audio Source Localisation (CAE-ASL):

Receives	*Audio Objects*	With associated Microphone Array information.
Detects	*Audio Objects*	In the Audio Scene.
Determines	*Spatial Attitudes*	Of Audio Objects.
Produces	*Spatial Attitudes*	Of input Audio Objects.

7.2.6.2 Reference Model

Figure 1 depicts the Reference Architecture of the Audio Source Localisation (CAE-ASL) AIM.

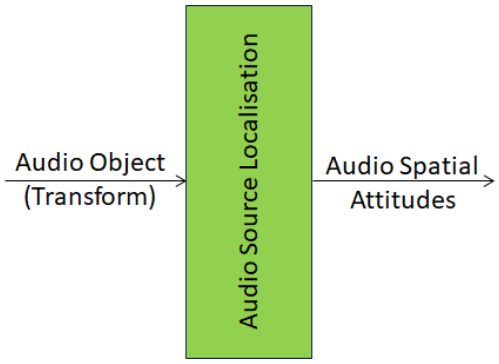

Figure 1 – Audio Source Localisation (CAE-ASL) AIM

7.2.6.3 Input/Output Data

Table 1 specifies the Input and Output Data of the Audio Source Localisation (CAE-ASL) AIM.

Table 1 – Audio Source Localisation (CAE-ASL) AIM

Input	Description
Audio Object	The result of the application of the Fast Fourier Transform to the Multichannel Audio (with associated Microphone Array info).
Output	**Description**
Audio Spatial Attitudes	The Orientations and Directions of Audio Objects.

7.2.6.4 JSON Metadata

https://schemas.mpai.community/CAE1/V2.3/AIMs/AudioSourceLocalisation.json

7.2.6.5 Conformance Testing

The following procedure shall be followed when testing the Conformance of a CAE-ASL AIM instance.

1. Use the following datasets:
    1. DS1: *n* Test files containing Audio Objects (Transform) .
    2. DS2: *n* Expected Spatial Attitudes.
2. Feed the AIM under test with the Test files.
3. Analyse the Spatial Attitudes produced by the CAE-ASL AIM instance.
4. Calculate the angle difference (AD) in degrees between the output Spatial Attitudes with the Expected Spatial Attitudes.

Input data (DS1)	**Expected Output Data (DS2)**	**Data Format**	**RMSE**
Audio Object (Microphone Array) ID_1	Spatial Attitude ID_1	T/F	< *A*0.1%*
Audio Object (Microphone Array) ID_2	Spatial Attitude (Transform) ID_2	T/F	< *A*0.1%*
Audio Object (Microphone Array) ID_3	Spatial Attitude (Transform) ID_3	T/F	< *A*0.1%*
…	…	…	…
Audio Object (Microphone Array) ID_n	Spatial Attitude (Transform) ID_n	T/F	< *A*0.1%*

5. Final evaluation: T/F Denoting with *i*, the record number in DS1 and DS2, the matrices reflect the results obtained with input records i with the corresponding outputs i.

DS1	DS2	Audio Object (Transform) output value (from AIM under test)
DS1[*i*]	DS2[*i*]	Audio Object (Transform)[*i*]

Table 2 provides the Conformance Testing Method for the formats of the CAE-ASL AIM output. Note: If a schema contains references to other schemas, conformance of data for the primary schema implies that any data referencing a secondary schema shall also validate against the relevant schema, if present and conform with the Qualifier, if present.

Table 2 – Conformance Testing Method for CAE-ASL AIM

Receives	*Audio Objects*	Shall validate against Audio Object schema. Audio Data shall conform with Audio Qualifier.
Produces	*Spatial Attitudes*	Shall validate against Spatial Attitude schema.

7.2.7 Audio Synthesis Transform

7.2.7.1 Functions

Audio Synthesis Transform (CAE-AST):

Receives	*Enhanced Audio Object (Transform)*	with associated Microphone Array info.
Transforms	*Enhanced Audio Object (Transform)*	from the frequency domain to the time domain via an Inverse Fast Fourier Transform (IFFT).
Produces	*Enhanced Audio Object*	with associated Microphone Array info.

7.2.7.2 Reference Model

Figure 1 depicts the Reference Architecture of the Audio Synthesis Transform (CAE-AST) AIM.

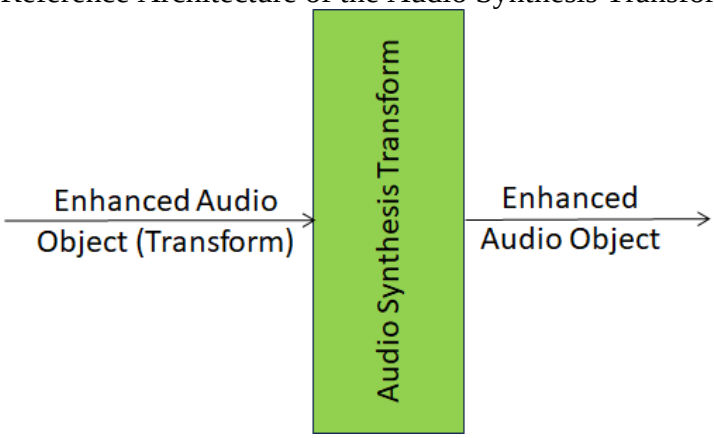

Figure 1 – The Audio Synthesis Transform (CAE-AST) AIM

7.2.7.3 Input/Output Data

Table 1 specifies the Input and Output Data of the Audio Synthesis Transform (CAE-AST) AIM.

Table 1 – I/O Data of Synthesis Transform (CAE-AST) AIM

Input	**Description**
Enhanced Audio Objects	Audio Objects in the time-frequency domain.
Output	**Description**
Enhanced Audio Objects	Audio Objects in the time domain.

7.2.7.4 JSON Metadata

https://schemas.mpai.community/CAE1/V2.3/AIMs/AudioSynthesisTransform.json

7.2.7.5 Reference Software

7.2.7.6 Conformance Testing

The following steps shall be followed when testing the Conformance of a CAE-AAT AIM instance.

1. Use the following datasets
    1. DS1: *n* Test files including data in Enhanced Audio Objects (Transform).
    2. DS2: *n* Expected Output files including data in Enhanced Audio Objects.
2. Feed the AIM under test with the Enhanced Audio Objects (Transform) Test files (DS1).
3. Analyse the Enhanced Audio Objects with the Expected Enhanced Audio Objects (DS2) with the following steps:
    1. Check the Enhanced Audio Objects data format with the given Expected Enhanced Audio Objects format.
    2. Calculate the peak-to-peak Amplitude (A) of each Audio block in the Expected Enhanced Audio Objects.
    3. Calculate the RMSE of each Audio block by comparing the output (*x*) with the Expected Enhanced Audio Objects (*y*) Audio blocks.
    4. Accept the AIM under test if, for each audio block, these the two conditions are satisfied:
        1. Data format of the Enhanced Audio Objects is the same with the Expected Enhanced Audio Objects and
        2. RMSE < A* 0.1%
4. The Conformance Tester will provide the following matrix with a limited number of input records (*n*) with the corresponding outputs. If an input record fails, the tester would specify the reason why the test case fails.

Input data (DS1)	Expected Output Data (DS2)	Data Format	RMSE
Enhanced Audio Objects ID_1	Enhanced Audio Objects ID_1	T/F	< A* 0.1%
Enhanced Audio Objects ID_2	Enhanced Audio Objects ID_2	T/F	< A* 0.1%
Enhanced Audio Objects ID_3	Enhanced Audio Objects ID_3	T/F	< A* 0.1%
…	…		…
Enhanced Audio Objects ID_n	Enhanced Audio Objects ID_n	T/F	< A* 0.1%

5. Final evaluation: T/F Denoting with *i*, the record number in DS1 and DS2, the matrices reflect the results obtained with input records i with the corresponding outputs i.

DS1	DS2	Synthesis Transform output value (obtained through the AIM under test)
DS1[*i*]	DS2[*i*]	SynthesisTransform[*i*]

Table 2 provides the Conformance Testing Method for the CAE-AST AIM for the formats of the CAE-AST AIM output.

Note: If a schema contains references to other schemas, conformance of data for the primary schema implies that any data referencing a secondary schema shall also validate against the relevant schema, if present and conform with the Qualifier, if present.

Table 2 – Conformance Testing Method for CAE-AST AIM

Receives	*Enhanced Audio Object (Transform)*	Shall validate against Audio Object schema. Audio Data shall conform with Audio Qualifier.
Produces	*Enhanced Audio Object*	Shall validate against Audio Object schema. Audio Data shall conform with Audio Qualifier.

7.3 AI modules from MPAI-HMC

7.3.1 Scene Integration and Description

7.3.1.1 Functions

AV Scene Integration and Description (HMC-SID) performs the following functions:

Receives	Portable Avatar
Adds	The Avatar in the Input Portable Avatar to the Audio-Visual Scene Descriptors conveyed by the Input Portable Avatar with an appropriate Spatial Attitude. If the Input Portable Avatar does not include a Scene, the AV Scene Integration and Description AIM uses a generic Scene.
Produces	The Audio-Visual Scene Descriptors of the resulting Audio-Visual Scene.

7.3.1.2 Reference Model

Figure 1 depicts the HMC-SID Reference Architecture.

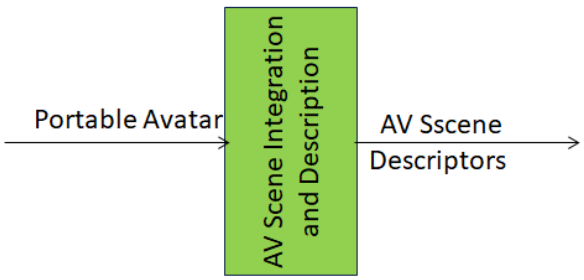

Figure 1 – The AV Scene Integration and Description AIM

7.3.1.3 I/O Data

The Input and Output Data of are specified in Table 1.

Table 1 – I/O Data of the AV Scene Integration and Description AIM

Input	Description
Portable Avatar	A Communication Item received from a Machine Entity.

Output	Description
Audio-Visual Scene Descriptors	The Descriptors of the AV Scene where the Avatar conveyed by the Input Portable Avatar has been added to the Scene with the appropriate Spatial Attitude.

7.3.1.4 JSON Metadata

https://schemas.mpai.community/HMC/V2.0/AIMs/AVSceneIntegrationAndDescription.json

7.3.1.5 Conformance Testing

Table 2 provides the Conformance Testing Method for HMC-SID AIM.

If a schema contains references to other schemas, conformance of data for the primary schema implies that any data referencing a secondary schema shall also validate against the relevant schema, if present and conform with the Qualifier, if present.

Table 2 – Conformance Testing Method for HMC-SID AIM

Receives	Portable Avatar	Shall validate against Portable Avatar Schema. Portable Avatar Data shall conform with respective Qualifiers.
Produces	Audio-Visual Scene Descriptors	Shall validate against AV Scene Descriptors Schema.

7.3.2 Entity and Context Understanding

7.3.2.1 Function

The function of Entity and Context Understanding (HMC-ECU) enables a Machine to understand the information conveyed by an Entity and its Context providing the information needed by the Entity Dialogue Processing AIM to produce a pertinent response composed of Machine Text and Machine Personal Status.

Therefore, Entity and Context Understanding (HMC-ECC)::

Receives	Audio-Visual Scene Descriptors	And separates into components.
Recognises	Speech	Of Entity.
	Audio Object and Visual Object.	Providing their Identities.
Understands	Natural Language	(Of Entity) expressed as Text being cognizant of the Audio and Visual Instances
Extracts	Personal Status.	Of Entity.
Translates	Text	Of Entity.
Produces:	Audio-Visual Scene Geometry	Geometry of the Scene.

	Entity ID	Entity producing Input Data.
	Audio Instance ID	Identified Instance.
	Visual Instance ID	Identified Instance.
	Personal Status	On Entity.
	Translated Text	Of Refined Text.
	Meaning	Of Refined Text.

7.3.2.2 Reference Model

Entity and Context Understanding is an AIM whose Reference Model is depicted in Figure 2.

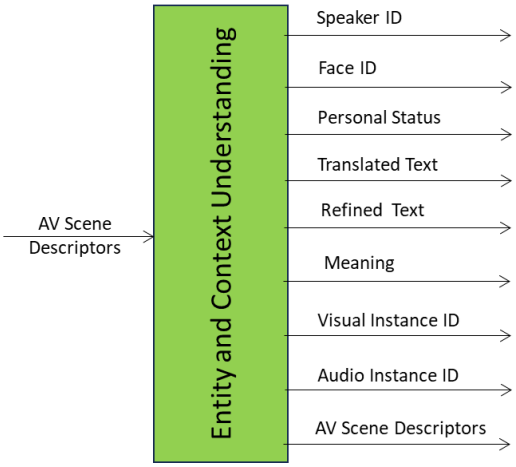

Figure 1 – The Entity and Context Understanding Composite AIM

7.3.2.3 Input and Output Data

Table 1 specifies the Input and Output Data of the of the Entity Context Understanding AIM.

Table 1 – I/O Data of the Entity Context Understanding Composite AIM

Input	Description
Body Descriptors	The Descriptors of the Body Objects of Entities in the Visual Scene.
Face Descriptors	The Descriptors of the Face Objects of Entities in the Visual Scene.
Speech Object	The digital representation of the speech emitted by the Entity.
Audio-Visual Scene Geometry	The digital representation of the spatial arrangement of the Audio, Visual, and Audio-Visual Objects of the Scene.
Visual Objects	The Visual Objects of the Scene.
Audio Object	The Audio Objects of the Scene.
Text Object	Text of Entity with Entity ID.
Output	**Description**

Personal Status	Personal Status of Entity having the Entity ID.
Translated Text	Translated Text of Text Object or of Text conveyed by Speech Object.
Refined Text	Refined Text of Speech Object.
Meaning	Other name for Refined Text Descriptors.
Visual Instance ID	The Identifier of the specific Visual Object belonging to a level in the taxonomy.
Audio-Visual Scene Geometry	As in Input
Audio Instance ID	The Identifier of the specific Audio Object belonging to a level in the taxonomy.

7.3.2.4 SubAIMs

Entity and Context Understanding is a Composite AIM whose Reference Model is depicted in Figure 2.

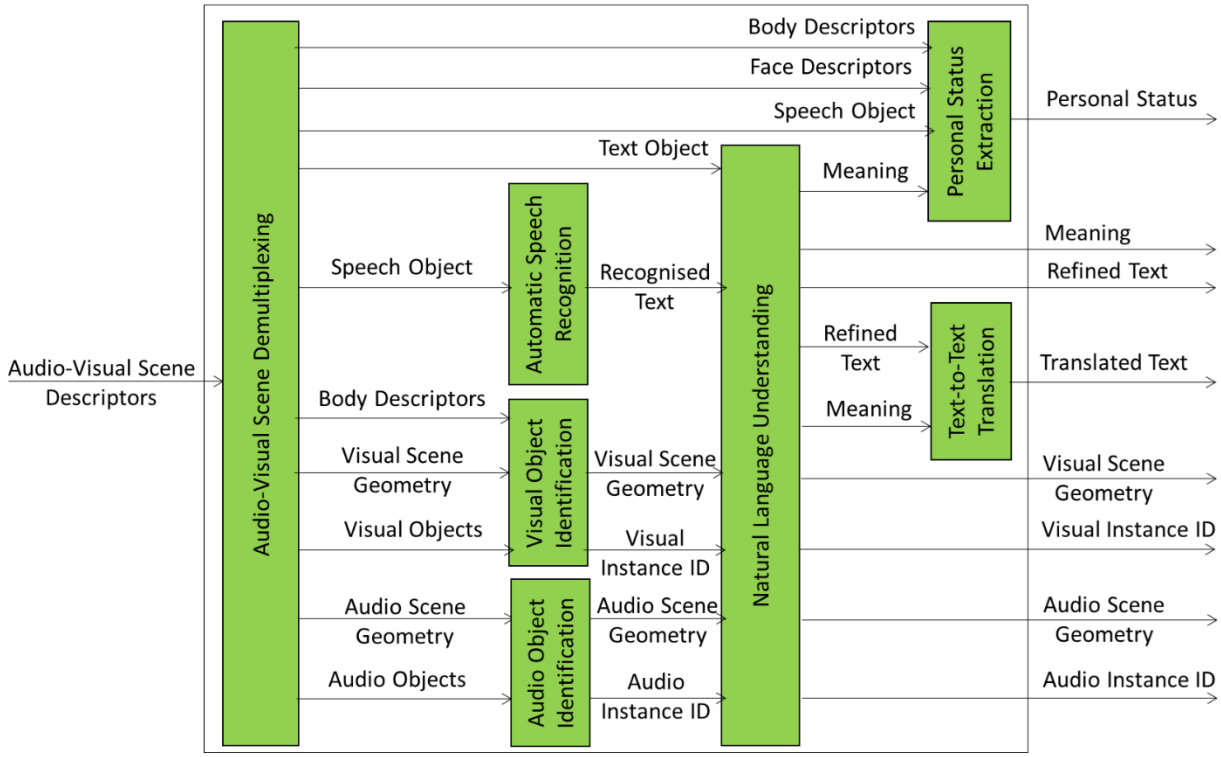

Figure 2 – The Entity and Context Understanding Composite AIM

Table 2 - AIMs and JSON Metadata

AIM	AIM/1	AIM/2	AIM Name	JSON
HMC-ECU			Entity and Context Understanding	X
	OSD-SDX		Audio-Visual Scene Demultiplexing	X
	MMC-ASR		Automatic Speech Recognition	X
	OSD-VOI		Visual Object Identification	X

	CAE-AOI		Audio Object Identification	X
	MMC-NLU		Natural Language Understanding	X
	MMC-PSE		Personal Status Extraction	X
		MMC-ETD	Entity Text Description	X
		MMC-ESD	Entity Speech Description	X
		PAF-EFD	Entity Face Description	X
		PAF-EBD	Entity Body Description	X
		MMC-PTI	PS-Text Interpretation	X
		MMC-PSI	PS-Speech Interpretation	X
		PAF-PFI	PS-Face Interpretation	X
		PAF-PGI	PS-Gesture Interpretation	X
		MMC-PMX	Personal Status Multiplexing	X
	MMC-TTT		Text-to-Text Translation	X

7.3.2.5 JSON Metadata

https://schemas.mpai.community/HMC/V2.0/AIMs/EntityAndContextUnderstanding.json

7.3.2.6 Profiles

Entity Context Understanding Profiles are defined.

7.3.2.7 Conformance Testing

Table 2 provides the Conformance Testing Method for the HMC-ECU AIM.

If a schema contains references to other schemas, conformance of data for the primary schema implies that any data referencing a secondary schema shall also validate against the relevant schema, if present and conform with the Qualifier, if present.

Table 2 – Conformance Testing Method for CAE-ECU AIM

Receives	Body Descriptors	Shall validate against Body Descriptors XML Schema.
	Face Descriptors	Shall validate against Face Descriptors Schema.
	Speech Object	Shall validate against Speech Object Schema. Speech Data shall conform with Speech Qualifier.
	Audio-Visual Scene Geometry	Shall validate against Audio-Visual Scene Geometry Schema.
	Visual Objects	Shall validate against Visual Object Schema. Visual Data shall conform with Visual Qualifier.
	Audio Object	Shall validate against Audio Object Schema. Audio Data shall conform with Visual Qualifier.
	Text Object	Shall validate against Text Object Schema. Text Data shall conform with Visual Qualifier.

Produces	Personal Status	Shall validate against Personal Status Schema.
	Translated Text	Shall validate against Text Object Schema. Text Data shall conform with Visual Qualifier.
	Refined Text	Shall validate against Text Object Schema. Text Data shall conform with Visual Qualifier.
	Meaning	Shall validate against Meaning schema
	Visual Instance ID	Shall validate against Instance ID schema.
	Audio-Visual Scene Geometry	Shall validate against Audio-Visual Scene Geometry Schema.
	Audio Instance ID	Shall validate against Instance ID schema.

7.4 AI modules from MPAI-MMC

7.4.1 Automatic Speech Recognition

7.4.1.1 Functions

Automatic Speech Recognition (MMC-ASR):

Receives	Language Selector	Signalling the language of the speech.
	Auxiliary Text	Text that may be used to provide context information.
	Speech Object	Speech to be recognised.
	Speaker ID	ID of speaker uttering speech.
	Speech Overlap	Data type providing information of speech overlap.
	Speaker Time	Time during which the speech is to be recognised.
Produces	Recognised Text	(Also called text transcript).

Recognised Text can be a Text Segment or just a string.

7.4.1.2 Reference Model

Figure 1 depicts the Reference Model of the Automatic Speech Recognition (MMC-ASR) AIM.

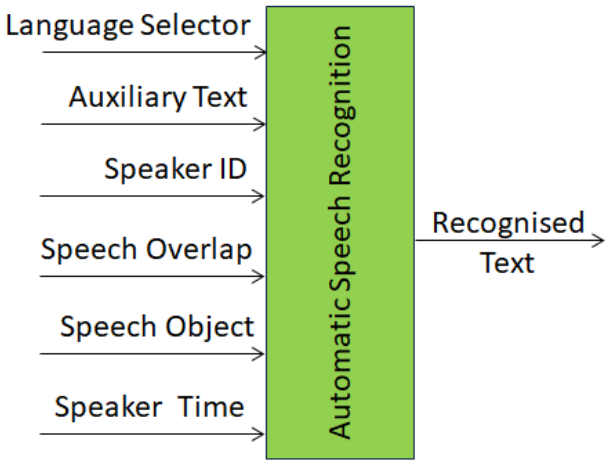

Figure 1 – The Automatic Speech Recognition (MMC-ASR) AIM

7.4.1.3 Input/Output Data

Table 1 specifies the Input and Output Data of the Automatic Speech Recognition (MMC-ASR) AIM.

Table 1 – I/O Data of the Automatic Speech Recognition (MMC-ASR) AIM

Input	Description
Language Selector	Selects input language
Auxiliary Text	Text Object with content related to Speech Object.
Speech Object	Speech Object emitted by Entity
Speaker ID	Identity of Speaker
Speech Overlap	Times and IDs of overlapping speech segments
Speaker Time	Time during which Speech is recognised
Output	**Description**
Recognised Text	Output of the Automatic Speech Recognition AIM, a Text Segment or just a string.

7.4.1.4 JSON Metadata

https://schemas.mpai.community/MMC/V2.3/AIMs/AutomaticSpeechRecognition.json

7.4.1.4.1 Disclaimers

1. This MMM-ASR Reference Software Implementation is released with the BSD-3-Clause licence.
2. The purpose of this Reference Software is to demonstrate a working Implementation of MMC-ASR, not to provide a ready-to-use product.
3. MPAI disclaims the suitability of the Software for any other purposes and does not guarantee that it is secure.
4. Use of this Reference Software may require acceptance of licences from the respective repositories. Users shall verify that they have the right to use any third-party software required by this Reference Software.

7.4.1.4.2 Guide to the ASR code #1

The code takes Speech Objects from MMC-AUS and generates Text Segments (called text transcripts). It uses the whisper-large-v3 model to convert an input Speech Object (speaker's turn) into a Text Segment (here called text transcript). Disfluencies (e.g., repetitions, repairs, filled pauses) are often omitted. The Whisper reference document is available.
The MMC-ASR Reference Software is found at the MPAI gitlab site. Use of this AI Modules is for developers who are familiar with Python, Docker, RabbitMQ, and downloading models from HuggingFace. The Reference Software contains:

1. src: a folder with the Python code implementing the AIM
2. Dockerfile: a Docker file containing only the libraries required to build the Docker image and run the container
3. requirements.txt: dependencies installed in the Docker image
4. README.md: commands for cloning https://huggingface.co/openai/whisper-large-v3

Library: https://github.com/linto-ai/whisper-timestamped

7.4.1.4.3 Guide to the ASR code #2

Use of this AI Modules is for developers who are familiar with Python and downloading models from HuggingFace,

A wrapper for the Whisper NN Module:
1. Manages input files and parameters: Speech Object
2. Performs Speech Recognition on each Speech Object by executing the Whisper Module.
3. Outputs Recognised Text.

The MMC-ASR Reference Software is found at the NNW gitlab site (registration required). It contains:
1. The python code implementing the AIM.
2. The required libraries are: pytorch and transformers (HuggingFace).

7.4.1.4.4 Acknowledgements

This version of the MMC-ASR Reference Software
1. #1 has been developed by the MPAI *AI Framework* Development Committee (AIF-DC).
2. #2 has been developed by the MPAI *Neural Network Watermarking* Development Committee (NNW-DC).

7.4.1.5 Conformance Testing

Table 2 provides the Conformance Testing Method for MMC-ASR AIM.

If a schema contains references to other schemas, conformance of data for the primary schema implies that any data referencing a secondary schema shall also validate against the relevant schema, if present and conform with the Qualifier, if present.

Table 2 - MMC-ASR AIM Conformance Testing

Input	Language Selector	Shall validate against the Language Selector part of the schema.
	Auxiliary Text	Shall validate against the Text Object schema. Text Data shall conform with the Text Qualifier.
	Speech Object	Shall validate against the Speech Object schema. Speech Data shall conform with the Speech Qualifier.
	Speaker ID	Shall validate against the Instance ID schema.
	Speech Overlap	Shall validate against the Speech Overlap schema.
	Speaker Time	Shall validate against the Time schema.
Output	Text Object	Shall validate against the Text Object schema. Text Data shall conform with the Text Qualifier, e.g. output language shall be that indicated by the Language Selector,

Table 3 provides an example of MMC-ASR AIM Conformance Testing.

Table 3 - An example of MMC-ASR AIM Conformance Testing

Input Data	**Data Format**	**Input Conformance Testing Data**
Speech Object	.wav	All input Speech files to be drawn from Speech files.
Output Data	**Data Format**	**Output Conformance Testing Criteria**
Recognised Text	Unicode	All Text files produced shall conform with Text files.

7.4.1.6 Performance Assessment

Performance Assessment of an ASR Implementation (ASRI) can be performed for a language for which there is a dataset of speech segments of various durations with corresponding Transcription Text. An MMC-ASR AIM Performance Assessment Report shall be based on the following steps and specify the input dataset used.

For each Recognised Text produced by the ASRI being Assessed for Performance in response to a speech segment provided as input:

1. Compare the Recognised Text with the Transcription Text
2. Compute the Word Error Rate (WER) defined as the sum of deletion, insertion, and substitution errors in the Recognised Text compared to the Transcription Text, divided by the total number of words in the Transcription Text.

This code can be used to compute the WER.

Performance Assessment of an ASRI for a language in a Performance Assessment Report is defined as "The WER computed on all speech segments included in the reported dataset".

7.4.2 Entity Dialogue Processing

7.4.2.1 Functions

Entity Dialogue Processing (MMC-EDP):

Receives	Text Object	Text of the entity upstream to be processed.
	Object Instance ID	Of an object in a scene.
	Personal Status	of the entity upstream.
	Text Descriptors	Descriptors of input Text Object.
	AV Scene Geometry	Geometry of the AV scene containing object whose ID is provided.
	Speaker ID	ID of speaker uttering the speech that contains the Text Object.
	Face ID	ID of the face of the speaker uttering the speech that contains the Text Object.
	Summary	A summary of the discussions being held in the environment.
Handles	One Text Object at a time	From an entity upstream.
Recognises	The identity	Of entity upstream using speech and/or face.
Takes into account	Past Text Objects	and their spatial arrangement.
Produces	Summary	Edited summary based on input data.
	Text Object	of Machine.
	Personal Status	of Machine.

7.4.2.2 Reference Model

Figure 1 depicts the Reference Model of the Entity Dialogue Processing (MMC-EDP) AIM.

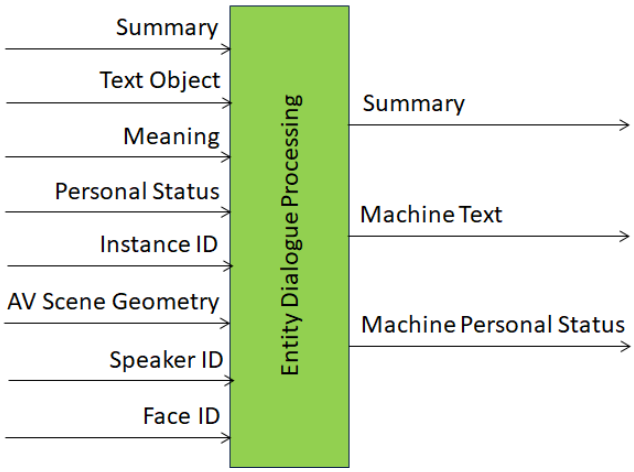

Figure 1 – Entity Dialogue Processing (MMC-EDP) AIM Reference Model

7.4.2.3 Input/Output Data

Table 1 specifies the Input and Output Data of the Entity Dialogue Processing (MMC-EDP) AIM.

Table 1 – I/O Data of the Entity Dialogue Processing (MMC-EDP) AIM

Input	Description
Summary	The summary in the current state.
Text Object	Text or Refined Text from the Entity the Machine is communicating with.
Meaning	Descriptors of Text and/or Translated Text of the Entity the Machine is communicating with.
Personal Status	Personal Status of the Entity the Machine is communicating with.
Instance ID	ID of the Audio of Visual Object the Entity refers to.
Audio-Visual Scene Geometry	The Geometry of the AV Scene.
Speaker ID	The ID of the Speaker.
Face ID	The ID of the Face.
Output	**Description**
Machine Text	Text produced by the Machine in response to input.
Machine Personal Status	The Personal Status the Machine intends to add to its Modalities.
Summary	The result of refining the input Summary taking comments into consideration.

7.4.2.4 JSON Metadata

https://schemas.mpai.community/MMC/V2.3/AIMs/EntityDialogueProcessing.json

7.4.2.5 Profiles

Profiles of Entity Dialogue Processing are specified.

7.4.2.6 Conformance Testing

Table 2 provides the Conformance Testing Method for MMC-EDP AIM.

If a schema contains references to other schemas, conformance of data for the primary schema implies that any data referencing a secondary schema shall also validate against the relevant schema, if present and conform with the Qualifier, if present.

Table 2 – MMC-EDP AIM Conformance Testing

Input	Text Object	Shall validate against Text Object schema. Text Data shall conform with Text Qualifier.
	Object Instance ID	Shall validate against Instance Identifier schema.
	Input Personal Status	Shall validate against Personal Status schema.
	Meaning	Shall validate against Text Descriptors schema.
	Audio-Visual Scene Geometry	Shall validate against AV Scene Geometry schema.
	Speaker ID	Shall validate against Instance ID schema.
	Face ID	Shall validate against Face ID schema.
	Summary	Shall validate against Summary schema. Shall validate against Text Object schema. Text Data shall conform with Text Qualifier.
Output	Edited Summary	Shall validate against Summary schema. Shall validate against Text Object schema. Text Data shall conform with Text Qualifier.
	Machine Text Object	Shall validate against Text Object schema. Text Data shall conform with Text Qualifier.
	Machine Personal Status	Shall validate against Personal Status schema.

Table 3 provides an example of MMC-EDP AIM Conformance Testing.

Table 3 – An example of MMC-EDP AIM Conformance Testing

Input Data	Data Type	Input Conformance Testing Data
Meaning	JSON	All input JSON Emotion files to be drawn from Meaning JSON Files

Recognised Text	Unicode	All input Text files to be drawn from Text files.
Input Emotion	JSON	All input JSON Emotion files to be drawn from Emotion JSON Files
Output Data	**Data Type**	**Output Conformance Testing Criteria**
Machine Text	Unicode	All Text files produced shall conform with Text.
Machine Emotion	JSON	Emotion JSON Files shall validate against Emotion Schema

The two attributes emotion_Name and emotion_SetName must be present in the output JSON file of Emotion. The value of either of the two attributes may be null.

7.4.3 Entity Speech Description

7.4.3.1 Functions

Entity Speech Description (MMC-ESD):

Receives	*Speech Object*	From an AIM or Entity.
Produces	*Speech Descriptors*	of the input Speech Object.

7.4.3.2 Reference Model

Figure 1 depicts the Reference Model of the Entity Speech Description (MMC-ESD) AIM.

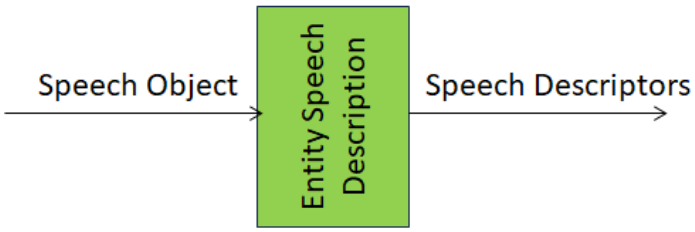

Figure 1 Entity Speech Description (MMC-ESD) AIM Reference Model

7.4.3.3 Input/Output Data

Table 1 specifies the Input and Output Data of the Entity Speech Description (MMC-ESD) AIM.

Table 1 – I/O Data of the Entity Speech Description (MMC-ESD) AIM

Input	**Description**
Speech Object	Speech of Entity or AIM
Output	**Description**
Speech Descriptors	Descriptors of Entity Speech

7.4.3.4 JSON Metadata

https://schemas.mpai.community/MMC/V2.3/AIMs/EntitySpeechDescription.json

7.4.3.5 Conformance Testing

Table 2 provides the Conformance Testing Method for MMC-ESD AIM.

If a schema contains references to other schemas, conformance of data for the primary schema implies that any data referencing a secondary schema shall also validate against the relevant schema, if present and conform with the Qualifier, if present.

Table 2 – Conformance Testing Method for MMC-ESD AIM

Input	Speech Object	Shall validate against Speech Object schema. Speech Data shall conform with Speech Qualifier.
Output	Speech Descriptors	Shall validate against Speech Descriptors schema.

7.4.4 Entity Text Description

7.4.4.1 Functions

Entity Text Description (MMC-ETD):

Receives	*Text Object*	Text Object from an entity.
Produces	*Text Descriptors*	Descriptors of Text Object's Text Data.

7.4.4.2 Reference Model

Figure 1 depicts the Reference Model of the Entity Text Description (MMC-ETD) AIM.

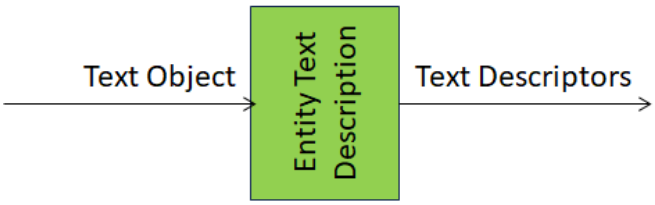

Figure 1 Entity Text Description (MMC-ETD) AIM Reference Model

7.4.4.3 Input/Output Data

Table 1 specifies the Input and Output Data of the Entity Text Description (MMC-ETD) AIM.

Table 1 – I/O Data of the Entity Text Description (MMC-ETD) AIM

Input	Description
Text Object	Text Object from and entity.

Output	Description
Text Descriptors	Descriptors of Descriptors of Text Data of Text Object.

7.4.4.4 JSON Metadata

https://schemas.mpai.community/MMC/V2.3/AIMs/EntityTextDescription.json

7.4.4.5 Conformance Testing

Table 2 provides the Conformance Testing Method for MMC-ETD AIM.

If a schema contains references to other schemas, conformance of data for the primary schema implies that any data referencing a secondary schema shall also validate against the relevant schema, if present and conform with the Qualifier, if present.

Table 2 – Conformance Testing Method for MMC-ETD AIM

Input	Text Object	Shall validate against Text Object schema. Text Data shall conform with Text Qualifier.
Output	Text Descriptors	Shall validate against Text Descriptors schema.

7.4.5 Natural Language Understanding

7.4.5.1 Functions

Natural Language Understanding (MMC-NLU):

Receives	Text Object directly input by the Entity.
	Recognised Text from an Automatic Speech Recognition AIM.
	The ID of an Instance.
	The Audio-Visual Scene Descriptors containing the Instance ID.
Refines	Input Text if coming from an Automatic Speech Recognition AIM
Extracts	Meaning (Text Descriptors) from Recognised Text or Entity's Text Object.
Produces	Refined Text.
	Text Descriptors (Meaning).
Enables	Personal Stats Display to produce a Portable Avatar.

7.4.5.2 Reference Model

Figure 1 specifies the Reference Model of the Natural Language Understanding (MMC-NLU) AIM.

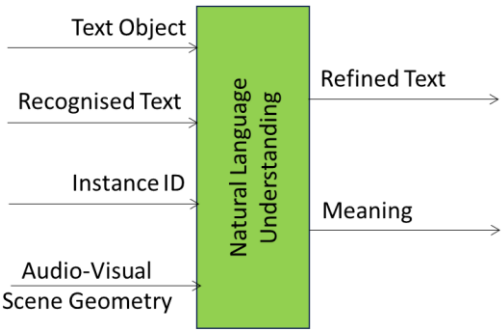

Figure 1 – The Natural Language Understanding (MMC-NLU) AIM Reference Model

7.4.5.3 Input/Output Data

Table 1 specifies the Input and Output Data of the Natural Language Understanding (MMC-NLU) AIM.

Table 1 – I/O Data of the Natural Language Understanding (MMC-NLU) AIM

Input	Description
Text Object	Input Text.
Recognised Text	Text from the Automatic Speech Recognition AIM.
Instance ID	The Identifier of the specific Audio or Visual Object belonging to a level in the taxonomy.
Audio-Visual Scene Geometry	The digital representation of the spatial arrangement of the Visual Objects of the Scene.
Visual Instance ID	The Identifier of the specific Visual Object belonging to a level in the taxonomy.
Output	**Description**
Meaning	Descriptors of the Refined Text.
Refined Text	The refined version of the Recognised Text from the NLU AIM.

7.4.5.4 JSON Metadata

https://schemas.mpai.community/MMC/V2.3/AIMs/NaturalLanguageUnderstanding.json

7.4.5.5 Profiles

The Profiles of the Natural Language Understanding (MMC-NLU) AIM are specified.

7.4.5.6 Conformance Testing

Table 2 provides the Conformance Testing Method for MMC-NLU AIM.

If a schema contains references to other schemas, conformance of data for the primary schema implies that any data referencing a secondary schema shall also validate against the relevant schema, if present and conform with the Qualifier, if present.

Table 2 – Conformance Testing Method for MMC-NLU AIM

Input	Text Object	Shall validate against Text Object schema. Text Data shall conform with Text Qualifier.
	Recognised Text	Shall validate against Text Object schema. Text Data shall conform with Text Qualifier.
	Instance ID	Shall validate against Instance ID schema.
	Audio-Visual Scene Geometry	Shall validate against AV Scene Descriptors schema.
Output	Refined Text	Shall validate against Text Object schema. Text Data shall conform with Text Qualifier.
	Meaning	Shall validate against Meaning schema.

Table 3 provides an example of MMC-NLU AIM conformance testing.

Table 3 – An example MMC-NLU AIM conformance testing

Input Data	**Data Type**	**Input Conformance Testing Data**
Input Selector	Binary data	All Input Selectors shall conform with Selector.
Text Object	Unicode	All input Text files to be drawn from Text files.
Recognised Text	Unicode	All input Text files to be drawn from Text files.
Output Data	**Data Type**	**Output Conformance Testing Criteria**
Meaning	JSON	All JSON files shall validate against Meaning Schema
Refined Text	Unicode	All Text files produced shall conform with Text.

The four taggings: POS_tagging, NE_tagging, dependency_tagging, and SRL_tagging must be present in the output JSON file of Meaning. Any of the four tagging values may be null.

7.4.6 Personal Status Extraction

7.4.6.1 Functions

Personal Status Extraction (MMC-PSE):

Receives	*Text Object* or *Text Descriptors*	
	Text Selector	indicating whether Text or Text Descriptors should be used.
	Speech Object or *Speech Descriptors*	

	Speech Selector	indicating whether Speech or Speech Descriptors should be used.
	Face or *Face Descriptors*	
	Face Selector	indicating whether Face or Face Descriptors should be used.
	Body or *Gesture Descriptors*	
	Body Selector	indicating whether Body or Gesture Descriptors should be used.
Computes and then Interprets	depending on reception of	the Descriptors of a Modality (Text, Speech, or Face).
	Text Descriptors	alternatively, Interprets the received Descriptors and produces Personal Status of the Text Object (PS-Text).
	Speech Descriptors;	alternatively, Interprets the received Descriptors and produces Personal Status of the Speech Object (PS-Speech).
	Face Descriptors	alternatively, Interprets the received Descriptors and produces Personal Status of the Face (PS-Face).
	Gesture Descriptors	alternatively, Interprets the received Gesture Descriptors of the Body.
Multiplexes	The results of the interpretations.	
Produces	Entity's Personal Status	

7.4.6.2 Reference Model

Figure 1 depicts the Reference Model of the Personal Status Extraction (MMC-PSE) AIM.

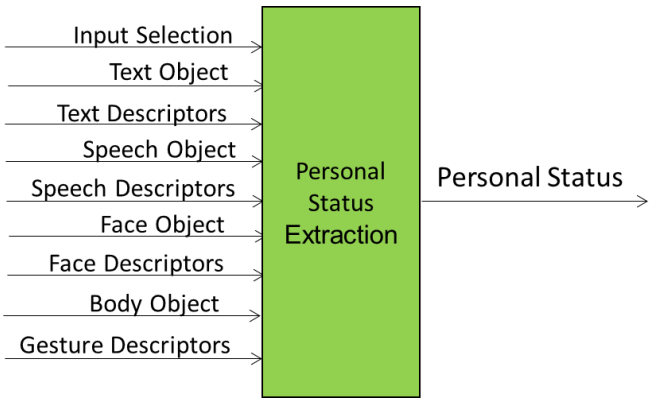

Figure 1 – The Personal Status Extraction Composite (MMC-PSE) AIM Reference Model

7.4.6.3 Input/Output Data

Table 1 specifies the Input and Output Data of the Personal Status Extraction (MMC-PSE) AIM.

Table 1 – I/O Data of the Personal Status Extraction (MMC-PSE) AIM

Input data	From	Description
Input Selector	An external signal	Media or Descriptors Selector
Text Object	Keyboard or AIM	Text or Recognised Text.
Text Descriptors	An upstream AIM	Functionally equivalent to Text Description.
Speech Object	Microphone/upstream AIM	Speech of Entity.
Speech Descriptors	An upstream AIM	Functionally equivalent to Speech Description.
Face Visual Object	Visual Scene Description	The face of the Entity.
Face Descriptors	An upstream AIM	Functionally equivalent to Face Description.
Body Visual Object	Visual Scene Description	The body of the Entity.
Gesture Descriptors	An upstream AIM	Functionally equivalent to Body Description.
Output data	**To**	**Description**
Personal Status	A downstream AIM	For further processing

7.4.6.4 SubAIMs

A Personal Status Extraction AIM instance can be implemented as a Composite AIM with different degrees of composition. The most extended composition if depicted by Figure 2

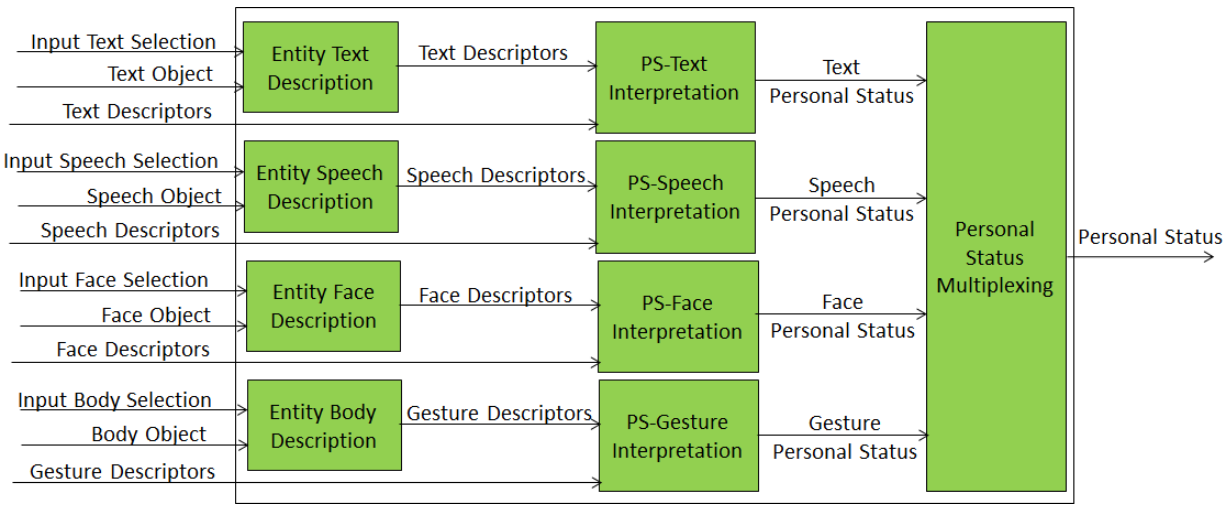

Figure 2 - The version of Personal Status Extraction AIM with the highest level of composition.

Table 2 gives the AIMs and their JSON Metadata of MMC-PSE.

Table 2 - AIMs and JSON Metadata

AIMs	AIMs	AIM Names	JSON
MMC-PSE		Personal Status Extraction	X
	MMC-ETD	Entity Text Description	X
	MMC-ESD	Entity Speech Description	X
	PAF-EFD	Entity Face Description	X
	PAF-EBD	Entity Body Description	X
	MMC-PTI	PS-Text Interpretation	X
	MMC-PSI	PS-Speech Interpretation	X
	PAF-PFI	PS-Face Interpretation	X
	PAF-PGI	PS-Gesture Interpretation	X
	MMC-PMX	Personal Status Multiplexing	X

7.4.6.5 JSON Metadata

https://schemas.mpai.community/MMC/V2.3/AIMs/PersonalStatusExtraction.json

7.4.6.6 Profiles

The Profiles of Personal Status Extraction are specified.

7.4.6.7 Conformance Testing

Table 3 provides the Conformance Testing Method for MMC-PSE AIM as a Basic AIM. Conformance Testing of the individual AIMs of the MMC-PSE Composite AIM are given by the individual AIM Specification.

If a schema contains references to other schemas, conformance of data for the primary schema implies that any data that refers to a secondary schema shall also validate against the relevant schema, if present and conform with the Qualifier, if present.

Table 3 – Conformance Testing Method for MMC-PSE AIM

Input	Text Object or	Shall validate against Text Object schema. Text Data shall conform with Text Qualifier.
	Text Descriptors	Shall validate against Text Descriptors schema.
	Text Selector	Shall validate against Text Selector schema.
	Speech Object or	Shall validate against Speech Object schema. Speech Data shall conform with Speech Qualifier.
	Speech Descriptors	Shall validate against Speech Descriptors schema.
	Speech Selector	Shall validate against Speech Selector schema.
	Face Visual Object or	Shall validate against Visual Object schema. Visual Data shall conform with Visual Qualifier.

	Face Descriptors	Shall validate against Face Descriptors schema.
	Face Selector	Shall validate against Face Selector schema.
	Body Visual Object	Shall validate against Visual Object schema. Visual Data shall conform with Visual Qualifier.
	Gesture Descriptors	Shall validate against Gesture Descriptors schema.
	Body Selector	Shall validate against Body Selector schema.
Output	Entity Personal Status	Shall validate against Personal Status schema.

7.4.7 Personal Status Multiplexing

7.4.7.1 Functions

Personal Status Multiplexing (MMC-PSM):

Receives	*PS-Text*	Personal Status of Text
	PS-Speech	Personal Status of Speech
	PS-Face	Personal Status of Face
	PS-Gesture	Personal Status of Gesture
Produces	*Personal Status*	Multiplexed Personal Status

7.4.7.2 Reference Model

Figure 1 depicts the Reference Model of the Personal Status Multiplexing (MMC-PSM) AIM.

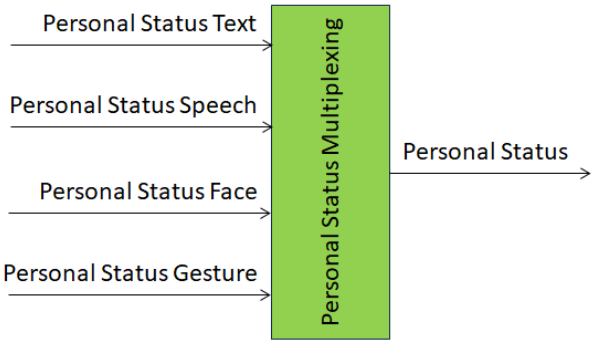

Figure 1 – The Personal Status Multiplexing (MMC-PSM) AIM Reference Model

7.4.7.3 Input/Output Data

Table 1 specifies the Input and Output Data of the Personal Status Multiplexing (MMC-PSM) AIM.

Table 1 – I/O Data of the Personal Status Multiplexing (MMC-PSM)

Input	Description
Text Personal Status	Personal Status of Text Object.
Speech Personal Status	Personal Status of Speech Object.

Face Personal Status	Personal Status of Face Object.
Gesture Personal Status	Personal Status of Gesture conveyed by Body Object.
Output	**Description**
Personal Status	Personal Status of Machine.

7.4.7.4 JSON Metadata

https://schemas.mpai.community/MMC/V2.3/AIMs/PersonalStatusMultiplexing.json

7.4.7.5 Conformance Testing

Table 2 provides the Conformance Testing Method for MMC-PSM AIM.

If a schema contains references to other schemas, conformance of data for the primary schema implies that any data referencing a secondary schema shall also validate against the relevant schema, if present and conform with the Qualifier, if present.

Table 2 – Conformance Testing Method for MMC-PSM AIM

Input	Text Personal Status	Shall validate against Text Personal Status schema.
	Speech Personal Status	Shall validate against Speech Personal Status schema.
	Face Personal Status	Shall validate against Face Personal Status schema.
	Gesture Personal Status	Shall validate against Gesture Personal Status schema.
Output	Personal Status	Shall validate against Personal Status schema.

7.4.8 PS-Speech Interpretation

7.4.8.1 Functions

PS-Speech Interpretation (MMC-PSI):

Receives	*Speech Descriptors*	To be interpreted.
Produces	*PS-Speech*	The Personal Status of the Speech Modality.

7.4.8.2 Reference Model

Figure 1 depicts the Reference Model of the PS-Speech Interpretation (MMC-PSI) AIM.

Figure 1 The PS-Speech Interpretation (MMC-PSI) AIM Reference Model

7.4.8.3 Input/Output Data

Table 1 specifies the Input and Output Data of the PS-Speech Interpretation (MMC-PSI) AIM.

Table 1 – I/O Data of the PS-Speech Interpretation (MMC-PSI) AIM

Input	Description
Speech Descriptors	Descriptors of Speech
Output	**Description**
Speech Personal Status	Personal Status of Speech

7.4.8.4 JSON Metadata

https://schemas.mpai.community/MMC/V2.3/AIMs/PSSpeechInterpretation.json

7.4.8.5 Conformance Testing

Table 2 provides the Conformance Testing Method for MMC-PSI AIM.

If a schema contains references to other schemas, conformance of data for the primary schema implies that any data referencing a secondary schema shall also validate against the relevant schema, if present and conform with the Qualifier, if present.

Table 2 – Conformance Testing Method for MMC-PSI AIM

Input	Speech Descriptors	Shall validate against Speech Descriptors schema
Output	Speech Personal Status	Shall validate against Speech Personal Status schema

7.4.9 PS-Text Interpretation

7.4.9.1 Functions

PS-Text Interpretation (MMC-PRI):

Receives	*Text Descriptors*	Either from Text Description or as a direct input to PS-Text Interpretation.
Produces	*PS-Text*	the Personal Status of the Text Modality.

7.4.9.2 Reference Model

Figure 1 depicts the Reference Model of the PS-Text Interpretation (MMC-PRI) AIM.

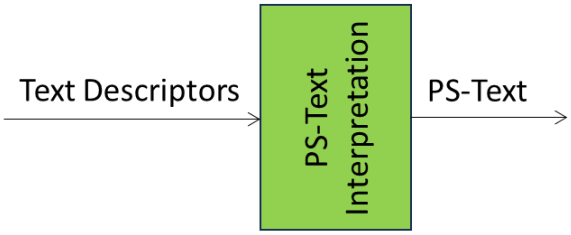

Figure 1 – The PS-Text Interpretation (MMC-PRI) AIM Reference Model

7.4.9.3 Input/Output Data

Table 1 specifies the Input and Output Data of the PS-Text Interpretation (MMC-PRI) AIM.

Table 1 – I/O Data of the PS-Text Interpretation (MMC-PRI) AIM

Input	**Description**
Text Descriptors	Descriptors of Text Data
Output	**Description**
Text Personal Status	Personal Status of Text Data

7.4.9.4 JSON Metadata

https://schemas.mpai.community/MMC/V2.3/AIMs/PSTextInterpretation.json

7.4.9.5 Conformance Testing

Table 2 provides the Conformance Testing Method for MMC-PRI AIM.

If a schema contains references to other schemas, conformance of data for the primary schema implies that any data referencing a secondary schema shall also validate against the relevant schema, if present and conform with the Qualifier, if present.

Table 2 – Conformance Testing Method for MMC-PRI AIM

Input	Text Descriptors	Shall validate against Text Descriptors schema
Output	Text Personal Status	Shall validate against Text Personal Status schema

7.4.10 Speaker Identity Recognition

7.4.10.1 Functions

Speaker Identity Recognition (MMC-SIR):

Receives	*Auxiliary Text*	Text related to the Speech.
	Speech Object	Speech of which the Speaker id requested.
	Speech Time	Time during whose duration Speaker ID is requested.

	Speech Overlap	Data signaling which parts of Speech Data have overlapping speech.
	Speech Scene Geometry	Disposition of Speech Data of the scene where the Speech whose speaker is to be identified is located.
Produces	*Speaker Identifier*	ID of speaker.

7.4.10.2 Reference Model

The Reference Architecture of Speaker Identity Recognition (MMC-SIR) is depicted in Figure 1.

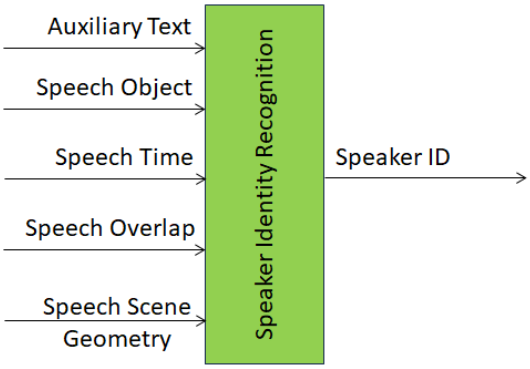

Figure 1 – The Speaker Identity Recognition (MMC-SIR) AIM

7.4.10.3 Input/Output Data

Table 1 specifies the Input and Output Data of the Speaker Identity Recognition (MMC-SIR) AIM.

Table 1 – I/O Data of the Speaker Identity Recognition (MMC-SIR) AIM

Input	Description
Auxiliary Text	Text with content related to Speaker ID.
Speech Object	Speech Object emitted by the Speaker.
Speech Time	The start and end time of the Speech.
Speech Overlap	Information about overlapping Speech.
Speech Scene Geometry	Information about Speech Object location.
Output	**Description**
Speaker Identifier	The Visual Descriptors of the Visual Scene.

7.4.10.4 JSON Metadata

https://schemas.mpai.community/MMC/V2.3/AIMs/SpeakerIdentityRecognition.json

7.4.10.5 Reference Software

7.4.10.5.1 Disclaimers

1. This MMC-SIR Reference Software Implementation is released with the BSD-3-Clause licence.
2. The purpose of this MMC-SIR Reference Software is to show a working Implementation of MMC-SIR, not to provide a ready-to-use product.
3. MPAI disclaims the suitability of the Software for any other purposes and does not guarantee that it is secure.

4. Use of this Reference Software may require acceptance of licences from the respective repositories. Users shall verify that they have the right to use any third-party software required by this Reference Software.

7.4.10.5.2 Guide to the MMC-SIR code

MMC-SIR performs speaker verification with a pretrained ECAPA-TDNN model; that is, it identifies the speaker of each speech segment by comparison with a dataset consisting of short clips of human speech.
The MMC-SIR Reference Software is found at the MPAI gitlab site. It contains:
1. src: a folder with the Python code implementing the AIM
2. Dockerfile: a Docker file containing only the libraries required to build the Docker image and run the container
3. requirements.txt: dependencies installed in the Docker image
4. README.md: commands for cloning https://huggingface.co/speechbrain/spkrec-ecapa-voxceleb

Library: https://github.com/speechbrain/speechbrain

7.4.10.5.3 Acknowledgements

This version of the MMC-SIR Reference Software has been developed by the MPAI *AI Framework* Development Committee (AIF-DC).

7.4.10.6 Conformance Testing

Table 2 provides the Conformance Testing Method for MMC-SIR AIM.
If a schema contains references to other schemas, conformance of data for the primary schema implies that any data referencing a secondary schema shall also validate against the relevant schema, if present and conform with the Qualifier, if present.

Table 2 – Conformance Testing Method for MMC-SIR AIM

Input	Text Object	Shall validate against Text Object schema. Auxiliary Text Data shall conform with Text Qualifier.
	Speech Object	Shall validate against Speech Object schema. Speech Data shall conform with Speech Qualifier.
	Speech Time	Shall validate against Time schema.
	Speech Overlap	Shall validate against Speech Overlap schema. Speech Data shall conform with Speech Qualifier.
	Speech Scene Geometry	Shall validate against Speech Scene Geometry schema.
Output	Speaker Identifier	Shall validate against Instance ID schema.

7.4.10.7 Performance Assessment

Performance Assessment of an MMC-SIR AIM Implementation shall be performed using a dataset of speech segments all in the same language for each segment of which the Identity of the Speaker is provided with reference to a Taxonomy.
The Performance Assessment Report of an MMC-SIR AIM Implementation shall include:
1. The Identifier of the MMC-SIR AIM.
2. The Identifier of the speech segment dataset.

3. The language of the speech segment dataset.
4. The Taxonomy of Speaker Identifiers.
5. The Performance of the MMC-SIR AIM expressed as the Accuracy of the Identifiers provided by MMC-SIR AIM computed on all speech segments of the dataset referenced in 2.

7.4.11 Text and Speech Translation

7.4.11.1 Functions

The Text and Speech Translation Composite (MMC-TST) AIM :

Receives	Selector	To chhose between:
		- The AIM output should be Text or Speech.
		- The output Speech should retain the input Speech Features.
	Language Preferences	as requested input and output language.
	Personal Status.	Use of Personal Status
	Text.	Use of Text
	Speech.	Use of Speech
Performs	A subset of) the following:	
	Conversion of input Speech	Into Text using Personal Status.
	Translation of Text	To the target language.
	Extraction of Features	From Speech.
	Conversion of Text	Into Speech adding the Input Speech's Features.
Produces	Translated Text.	Depends of Selector.
.	Translated Speech	Depends of Selector.

7.4.11.2 Reference Model

Figure 1 depicts the Reference Model of the Text-and-Speech Translation Composite (MMC-TST) AIM.

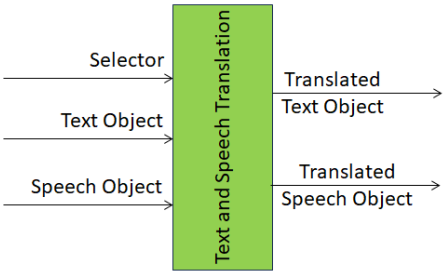

Figure 1 – Text-and-Speech Translation (MMC-TST) AIM Reference Model

7.4.11.3 Input/Output Data

Table 1 specifies the Input and Output Data of the Text-to-Text Translation (MMC-TST) AIM.

Table 1 – I/O Data of the Text-and-Speech Translation (MMC-TST) AIM

Input	Semantics
Selector	Signals: 1. Whether the input is Text or Speech 2. Whether the input Speech features are preserved in the output Speech. 3. The Input and output languages.
Speech Object	Speech produced in input language by a human desiring translation into output language
TextObject	Alternative textual source information to be translated into and pronounced in output language depending on the value of Input Selection.
Output	**Description**
Translated SpeechObject	Speech in input language translated into output language preserving the Input Speech features in the Output Speech, depending on Selector.
Translated TextObject	Text of Input Speech or Input Text translated into output language, depending on Selector.

7.4.11.4 SubAIMs

Text and Speech Translation is a Composite AIM whose Reference Model is depicted in Figure 2.

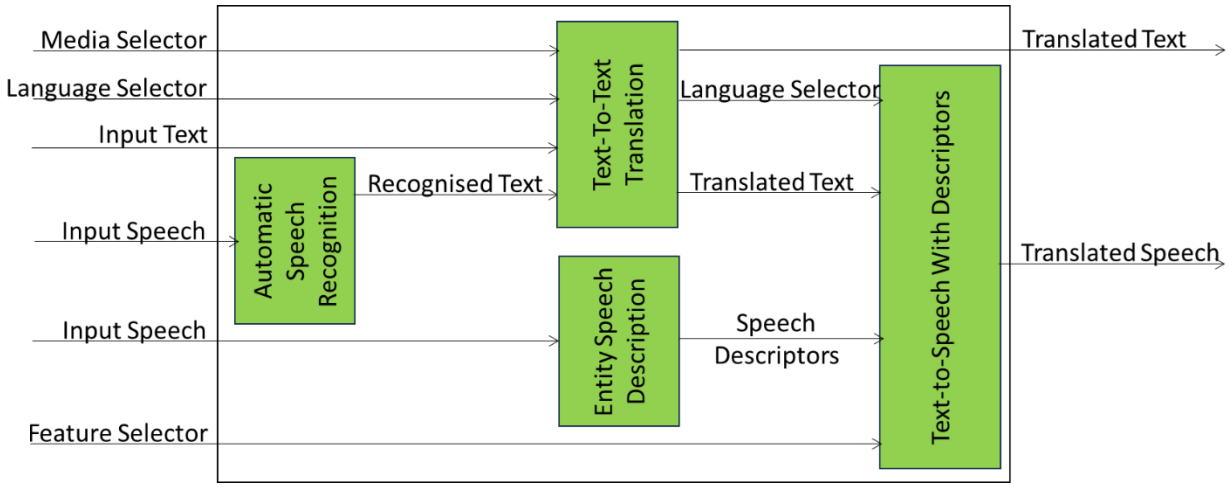

Figure 2 – Text-and-Speech Translation Composite (MMC-TST) AIM

Table 2 - AIMs of Text-and-Speech Translation Composite (MMC-TST) AIM

AIW	AIMs	AIM Names	JSON
MMC-TST		Text-and-Speech Translation	X
	MMC-ASR	Automatic Speech Recognition	X
	MMC-TTT	Text-to-Text Translation	X

	MMC-ISD	Entity Speech Description	X
	MMC-DTS	Descriptors Text-to-Speech	X

7.4.11.5 JSON Metadata

https://schemas.mpai.community/MMC/V2.3/AIMs/TextAndSpeechTranslation.json

7.4.11.6 Profiles

The Profiles of Text and Speech Translation are specified.

7.4.11.7 Conformance Testing

Table 3 provides the Conformance Testing Method for MMC-TST AIM as a Basic AIM. Conformance Testing of the individual AIMs of the MMC-TST Composite AIM are given by the individual AIM Specification.

If a schema contains references to other schemas, conformance of data for the primary schema implies that any data referencing a secondary schema shall also validate against the relevant schema, if present and conform with the Qualifier, if present.

Table 3 – Conformance Testing Method for MMC-TST AIM

Input	Selector	Shall validate against Selector schema.
	Text Object	Shall validate against Text Object schema. Text Data shall conform with Text Qualifier.
	Speech Object	Shall validate against Speech Object schema. Speech Data shall conform with Speech Qualifier.
Output	Translated Text Object	Shall validate against Text Object. Text Data shall conform with Text Qualifier.
	Translated Speech Object	Shall validate against Speech Object. Speech Data shall conform with Speech Qualifier.

Important note. This Conformance Testing Specification does not provide methods and datasets to Test the Conformance of the individual Speech Feature Extraction and Text-To-Speech Basic AIMs, only of their Descriptors Speech Translation Composite AIMs.

Table 4 provides an example of MMC-TSTAIM conformance testing.

Table 4 – An example MMC-TST AIM conformance testing

Input Data	Data Type	Input Conformance Testing Data
Input Selector	Selector	All Input Selectors to conform with Selector.
Requested Language	Selector	All Language Selectors to be drawn from Language Codes.
Input Text	Unicode	All input Text files shall be drawn from Text files.

Input Speech	.wav	All input Text files shall be drawn from Speech files.
Output Data	**Data Type**	**Conformance Test**
Machine Text	Unicode	All Text files produced shall conform with Text files.
Machine Speech	.wav	All Speech files produced shall conform with Speech files.

7.4.12 Text-To-Speech

7.4.12.1 Functions

Text-To-Speech (MMC-TTS):

Receives	Text Object	
	Personal Status	to be contained in the Synthesised Speech Object.
	Speech Model	used by AIM depending on Profile.
Feeds	Text Object and Personal Status	to Speech Model.
Produces	Synthesised Speech Object	output of AIM.

7.4.12.2 Reference Model

Figure 1 specifies the Reference Model of the Text-To-Speech (MMC-TTS) AIM.

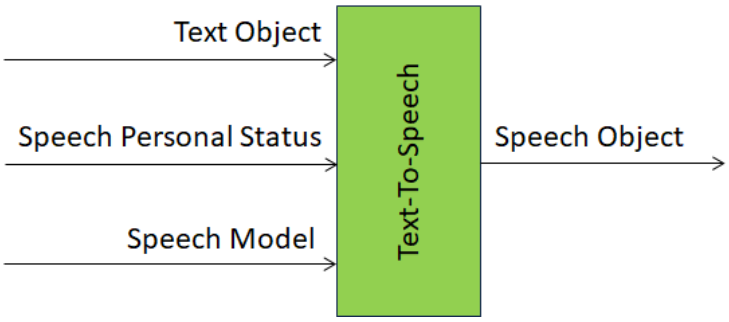

Figure 1 – The Text-To-Speech AIM Reference Model

7.4.12.3 Input/Output Data

Table 1 specifies the Input and Output Data of the Automatic Speech Recognition AIM.

Table 1 – I/O Data of the Automatic Speech Recognition AIM

Input	Description
Text Object	Input Text.
Personal Status	Input Personal Status of the Speech Modality.
Speech Model	NN Model used to produce Speech from Text and Personal Status.
Output	**Description**
Speech Object	Output of the Text-To-Speech AIM,

7.4.12.4 JSON Metadata

https://schemas.mpai.community/MMC/V2.3/AIMs/TextToSpeech.json

7.4.12.5 Profiles

The Text-To-Speech Profiles are specified.

7.4.12.6 Reference Software

7.4.12.6.1 Disclaimers

1. The purpose of this MMC-TTS Reference Software is to provide a working Implementation of MMC-TTS, not to provide a ready-to-use product.
2. MPAI disclaims the suitability of the Software for any other purposes and does not guarantee that it is secure.
3. Use of this Reference Software may require acceptance of licences from the respective repositories. Users shall verify that they have the right to use any third-party software required by this Reference Software.

7.4.12.6.2 Guide to the MMC-TTS code

Use of this AI Module is for developers who are familiar with Python and downloading models from HuggingFace,
A wrapper for the speech5 NN Module

1. Manages input files and parameters: Text Object
2. Executes the BLIP Module to perform the Speech Recognition on each individual pair of Text and Visual Object.
3. Outputs Speech Object as answer.

The MMC-TTS Reference Software is found at the MPAI-NNW gitlab site. It contains:

1. The python code implementing the AIM
2. Required libraries are: pytorch, transformers (HuggingFace), datasets (HuggingFace), and sound file.

7.4.12.6.3 Acknowledgements

This version of the MMC-TTS Reference Software has been developed by the MPAI *Neural Network Watermarking* Development Committee (NNW-DC).

7.4.12.7 Conformance Testing

Table 2 provides the Conformance Testing Method for MMC-TTS AIM.
If a schema contains references to other schemas, conformance of data for the primary schema implies that any data referencing a secondary schema shall also validate against the relevant schema, if present and conform with the Qualifier, if present.

Table 2 – Conformance Testing Method for MMC-TTS AIM

Input	Text Object	Shall validate against Text Object schema. Text Data shall conform with Text Qualifier.
	Personal Status	Shall validate against Personal Status schema.
	Speech Model	Shall validate against Machine Learning Model schema. Machine Learning Model Data shall conform with Machine Learning Model Qualifier.
Output	Synthesised Speech Object	Shall validate against Speech Object schema. Speech Data shall conform with Speech Qualifier.

Table 3 provides an example of MMC-TTS AIM conformance testing.

Table 3 – An example MMC-TTS AIM conformance testing

Input Data	Data Type	Input Conformance Testing Data
Machine Text	Unicode	All input Text files to be drawn from Text files.
Machine Emotion	JSON	All input JSON Emotion files to be drawn from Emotion JSON Files
Output Data	**Data Type**	**Output Conformance Testing Criteria**
Machine Speech	.wav	All Speech files produced shall conform with Speech.

7.4.13 Text-to-Text Translation

7.4.13.1 Functions

Text-to-Text Translation:

Receives	*Selector*	Determining the input and target language.
	Text Object	Text to be translated.
	Meaning	Input Text Meaning.
Produces	*Translated Text*	*Output Translates Text.*

7.4.13.2 Reference Model

Figure 1 depicts the Reference Model of the Text-to-Text Translation AIM.

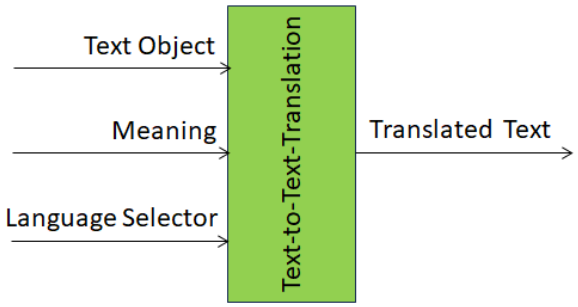

Figure 1 – Text-to-Text Translation AIM Reference Model

7.4.13.3 Input/Output Data

Table 1 specifies the Input and Output Data of the Text-to-Text Translation AIM.

Table 1 – I/O Data of the Text-to-Text Translation AIM

Input	Description
Text Object	Input Text Object.
Meaning	Meaning of Input Text
Language Selector	Input and target Language.
Output	**Description**
Translated Text	Translation of Text (or Refined Text).

7.4.13.4 JSON Metadata

https://schemas.mpai.community/MMC/V2.3/AIMs/TextToTextTranslation.json

7.4.13.5 Conformance Testing

Table 2 provides the Conformance Testing Method for MMC-TTT AIM.
If a schema contains references to other schemas, conformance of data for the primary schema implies that any data referencing a secondary schema shall also validate against the relevant schema, if present and conform with the Qualifier, if present.

Table 2 – Conformance Testing Method for MMC-TTT AIM

Input	Language Selector	Shall validate against Language Selector schema.
	Text Object	Shall validate against Text Object schema. Text Data shall conform with Text Qualifier.
	Meaning	Shall validate against Meaning schema.
Output	Translated Text Object	Shall validate against Text Object schema. Text Data shall conform with Text Qualifier.

Table 3 provides an example of MMC-TTT AIM conformance testing.

Table 3 – An example MMC-TTT AIM conformance testing

Input Data	**Data Type**	**Input Conformance Testing Data**
Input Text	Unicode	All input Text files to be drawn from Text files.
Output Data	**Data Type**	**Output Conformance Testing Criteria**
Translated Text	Unicode	All Text files produced shall conform with Text.

7.4.13.6 Performance Assessment

Performance Assessment of an MMC-TTT AIM Implementation shall be performed using a dataset of text sentences in a given language. Each text sentence shall have at least one translated text.
The Performance Assessment Report of an MMC-TTT AIM Implementation shall include:

1. The Identifier of the MMC-TTT AIM.
2. The Identifier of the dataset of text sentences.
3. The name of the input and output languages and their ISO 639 Set 3 three-letter code.
4. The number of text sentences in the data set and the average number of translated texts per input text.
5. The maximum value N of n-grams.
6. The BLEU Score of the MMC-TTT AIM defined as the Arithmetic Mean of the individual BLEU Scores computed over the dataset, where each BLEU Score is the product of the Brevity Penalty and the Geometric Mean Precision, where:
    1. *Brevity Penalty* of a candidate translation of length *c* to a reference translation of length *r* is min $(1,e^{(1-r/c)})$.
    2. *Sentence Precision* of a set of N n-grams is $\exp(\sum_{n=1,N}\log(p_n)/N$, where p_i is the precision of the i-th n-gram.

7.5 AI modules from MPAI-OSD

7.5.1 Audio-Visual Alignment

7.5.1.1 Functions

Audio-Visual Alignment (OSD-AVA):

Receives	*Speech Scene Descriptors*	Descriptors of potentially present Speech Scene.
	Audio Scene Descriptors	Descriptors of potentially present Audio Scene.
	Visual Scene Descriptors	Descriptors of Visual Scene.
Aligns	Speech, Audio, and Visual Objects	Sharing the same Spatial Attitude
Produces	*Audio-Visual Scene Descriptors*	Where Speech Objects, Audio Objects, and Visual Objects having the same Spatial Attitude have the same or compatible Identifiers.

7.5.1.2 Reference Model

Figure 1 specifies the Reference Model of the Audio-Visual Alignment (OSD-AVA) AIM.

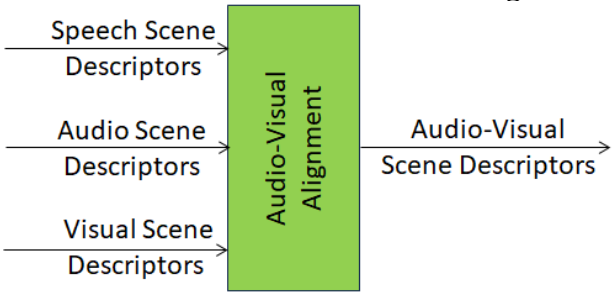

Figure 1 - Reference Model of the Audio-Visual Alignment (OSD-AVA) AIM

7.5.1.3 Input/Output Data

Table 1 specifies the Input and Output Data of the Audio-Visual Alignment (OSD-AVA) AIM.

Table 1 – I/O Data of the Audio-Visual Alignment AIM

Input	Description
Speech Scene Descriptors	The IDs and the geometry of the Speech Objects of the Scene.
Audio Scene Descriptors	The IDs and the geometry of the Audio Objects of the Scene.
Visual Scene Descriptors	The IDs and the geometry of the Audio Objects of the Scene.
Output	**Description**
Audio-Visual Scene Descriptors	The IDs and the geometry of the Audio, Visual and Audio-Visual Objects of the Scene.

7.5.1.4 JSON Metadata

https://schemas.mpai.community/OSD/V1.2/AIMs/AudioVisualAlignment.json

7.5.1.5 Reference Software

7.5.1.5.1 Disclaimers

1. This OSD-AVA Reference Software Implementation is released with the BSD-3-Clause licence.

2. The purpose of this Reference Software is to show a working Implementation of OSD-AVA, not to provide a ready-to-use product.
3. MPAI disclaims the suitability of this Reference Software for any other purposes and does not guarantee that it is secure.
4. Use of this Reference Software may require acceptance of licences from the respective repositories. Users shall verify that they have the right to use any third-party software required by this Reference Software.

7.5.1.5.2 Guide to OSD-AVA code

OSD-AVA arranges the output Visual Objects and Speech Objects with corresponding Time information: scene cuts/transitions and speakers' turns. Each Object is bounded by two adjacent times from a list of unique times that are either 1) scene cuts/transitions or 2) starts and ends of speakers' turns.
Use of this Reference Software for the OSD-AVA AI Module is for developers who are familiar with Python, Docker, and RabbitMQ.
OSD-AVA computes segments as unique intervals from scene bounds and from speech segments. Moreover, OSD-AVA outputs visual objects and speech objects.
The OSD-AVA Reference Software is found at the MPAI gitlab site. It contains:
1. src: a folder with the Python code implementing the AIM
2. Dockerfile: a Docker file containing only the libraries required to build the Docker image and run the container
3. requirements.txt: dependencies installed in the Docker image.

7.5.1.5.3 Acknowledgements

This version of the MMC-ASR Reference Software has been developed by the MPAI *AI Framework* Development Committee (AIF-DC).

7.5.1.6 Conformance Testing

Table 2 provides the Conformance Testing Method for OSD-AVA AIM.
If a schema contains references to other schemas, conformance of data for the primary schema implies that any data referencing a secondary schema shall also validate against the relevant schema, if present and conform with the Qualifier, if present.

Table 2 – Conformance Testing Method for OSD-AVA AIM

Receives	Speech Scene Descriptors	Shall validate against Speech Scene Descriptors schema
	Audio Scene Descriptors	Shall validate against Audio Scene Descriptors schema
	Visual Scene Descriptors	Shall validate against Visual Scene Descriptors schema
Produces	Audio-Visual Scene Descriptors	Shall validate against AV Scene Descriptors schema

7.5.2 Audio-Visual Event Description

7.5.2.1 Functions

Audio-Visual Event Description (OSD-AVE).

Receives	*Audio-Visual Scene Descriptors*.	A sequence.

Creates	Input *Audio-Visual Scene Descriptors*.	A file.
Produces	*Audio-Visual Event Descriptors*	

7.5.2.2 Reference Model

The Audio-Visual Event Description (OSD-AVE) AIM Reference Model is depicted in Figure 1.

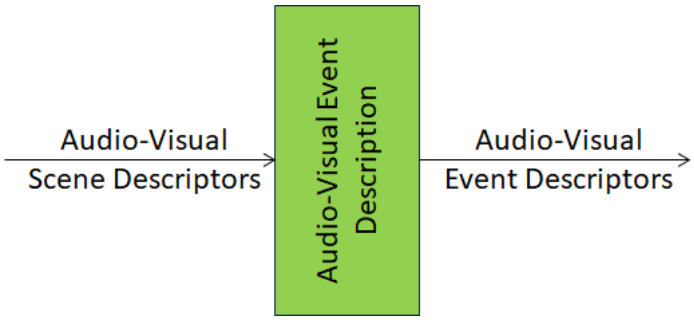

Figure 1 – The Audio-Visual Event Description (OSD-AVE) AIM Reference Model

7.5.2.3 Input/Output Data

Table 1 specifies the Input and Output Data of the Audio-Visual Event Description AIM. Links are to the Data Type specifications.

Table 1 – I/O Data of the Audio-Visual Event Description (OSD-AVE) AIM

Input	**Description**
Audio-Visual Scene Descriptors	Sequence of Audio-Visual Scene Descriptors.
Output	**Description**
Audio-Visual Event Descriptors	The Audio-Visual Event Descriptors of the Audio-Visual Scene.

7.5.2.4 JSON Metadata

https://schemas.mpai.community/OSD/V1.2/AIMs/AudioVisualEventDescription.json

7.5.2.5 Reference Software

1. This OSD-AVE Reference Software Implementation is released with the BSD-3-Clause licence.
2. The purpose of this Reference Software is to show a working Implementation of OSD-AVE, not to provide a ready-to-use product.
3. MPAI disclaims the suitability of the Software for any other purposes and does not guarantee that it is secure.
4. Use of this Reference Software may require acceptance of licences from the respective repositories. Users shall verify that they have the right to use any third-party software required by this Reference Software.

7.5.2.5.1 Guide to the OSD-AVE code

OSD-AVE arranges the audio-visual scene descriptors from OSD-AVS into Audio-Visual Event Descriptors.
Use of this Reference Software for the OSD-AVE AI Module is for developers who are familiar with Python, Docker, and RabbitMQ.
The OSD-AVE Reference Software is found at the MPAI gitlab site. It contains:

1. src: a folder with the Python code implementing the AIM
2. Dockerfile: a Docker file containing only the libraries required to build the Docker image and run the container
3. requirements.txt: dependencies installed in the Docker image.

7.5.2.5.2 Acknowledgements

This version of the OSD-AVE Reference Software has been developed by the MPAI *AI Framework* Development Committee (AIF-DC).

7.5.2.6 Conformance Testing

Table 2 provides the Conformance Testing Method for OSD-AVE AIM.
If a schema contains references to other schemas, conformance of data for the primary schema implies that any data referencing a secondary schema shall also validate against the relevant schema, if present and conform with the Qualifier, if present.

Table 2 – Conformance Testing Method for OSD-AVE AIM

Receives	Audio-Visual Scene Descriptors	Shall validate against AV Scene Descriptors schema
Produces	Audio-Visual Event Descriptors	Shall validate against AV Event Descriptors schema

7.5.3 Audio-Visual Scene Demultiplexing

7.5.3.1 Functions

Audio-Visual Scene Demultiplexing (OSD-SDX):

Receives	*Audio-Visual Scene Descriptors*
Demultiplexes	*Audio-Visual Scene Descriptors*
Produces	*Speech Scene Geometry*
	Audio Scene Geometry
	Visual Scene Geometry
	Speech Objects
	Audio Objects
	Visual Objects

7.5.3.2 Reference Model

Figure 1 depicts the Reference Model of the Audio-Visual Scene Demultiplexing (OSD-SDX) AIM.

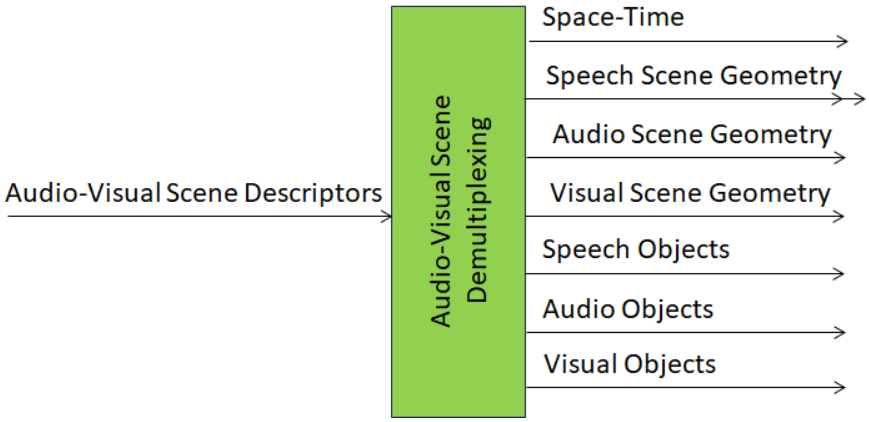

Figure 1 – Audio-Visual Scene Demultiplexing (OSD-SDX) AIM Reference Model

7.5.3.3 Input/Output Data

Table 1 specifies the Input and Output Data of the of the Audio-Visual Scene Demultiplexing (OSD-SDX) AIM.

Table 1 – I/O Data of the Audio-Visual Scene Demultiplexing (OSD-SDX) AIM

Input	Description
Audio-Visual Scene Descriptors	The Descriptors of the Audio-Visual Scene.
Output	**Description**
Space-Time	Space-Time information of the Audio-Visual Scene
Speech Scene Geometry	The Descriptors of the Speech Scene.
Audio Scene Geometry	The Descriptors of the Audio Scene.
Visual Scene Geometry	The Descriptors of the Visual Scene.
Audio Object	The Audio Objects in the Scene.
Speech Object	The Speech Objects in the Scene.
Visual Object	The Visual Objects in the Scene.

7.5.3.4 JSON Metadata

https://schemas.mpai.community/OSD/V1.2/AIMs/AudioVisualSceneDemultiplexing.json

7.5.3.5 Conformance Testing

Table 2 provides the Conformance Testing Method for the OSD-SDX AIM as a Basic AIM.

If a schema contains references to other schemas, conformance of data for the primary schema implies that any data referencing a secondary schema shall also validate against the relevant schema, if present and conform with the Qualifier, if present.

Table 2 – Conformance Testing Method for OSD-SDX AIM

Receives	Audio-Visual Scene Descriptors	Shall validate against AV Scene Descriptors schema
Produces	Speech Scene Geometry	Shall validate against Speech Scene Geometry schema
	Audio Scene Geometry	Shall validate against Audio Scene Geometry schema
	Visual Scene Geometry	Shall validate against Visual Scene Geometry schema
	Speech Objects	Shall validate against Speech Objects schema Speech Data shall conform with Qualifier
	Audio Objects	Shall validate against Audio Objects schema Audio Data shall conform with Qualifier
	Visual Objects	Shall validate against Visual Objects schema Visual Data shall conform with Qualifier

7.5.4 Visual Direction Identification

7.5.4.1 Functions

Visual Direction Identification (OSD-VDI):

Receives	Visual Scene Geometry	The Geometry of the Visual Scene
	Body Descriptors	The Descriptors of a Body.
Produces	Point of View	The direction of a line traversing a point of the forefinger of the Entity.

7.5.4.2 Reference Model

Figure 1 depicts the Reference Model of the Visual Direction Identification (OSD-VOI) AIM.

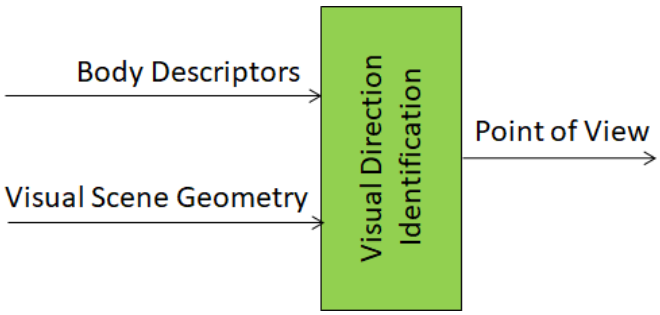

Figure 1 – The Visual Direction Identification (OSD-VOI) AIM Reference Model

7.5.4.3 Input/Output Data

Table 1 specifies the Input and Output Data of the Visual Direction Identification (OSD-VOI) AIM.

Table 1 – I/O Data of the Visual Direction Identification (OSD-VOI) AIM

Input	Description
Body Descriptors	The Descriptors of the Body Objects in the Visual Scene.
Visual Scene Geometry	The digital representation of the spatial arrangement of the Visual Objects of the Scene.
Output	**Description**
Point ov View	The direction of the line traversing the forefinger of the target Entity.

7.5.4.4 JSON Metadata

https://schemas.mpai.community/OSD/V1.2/AIMs/VisualDirectionIdentification.json

7.5.4.5 Conformance Testing

Table 2 provides the Conformance Testing Method for OSD-VDI AIM.

If a schema contains references to other schemas, conformance of data for the primary schema implies that any data referencing a secondary schema shall also validate against the relevant schema, if present and conform with the Qualifier, if present.

Table 2 – Conformance Testing Method for OSD-VDI AIM

Receives	Visual Scene Geometry	Shall validate against Visual Scene Geometry schema
	Body Descriptors	Shall validate against Body Descriptors XML schema
Produces	Point of View	Shall validate against Point of View schema

7.5.4.6 Performance Assessment

7.5.5 Visual Instance Identification

7.5.5.1 Functions

Visual Instance Identification (OSD-VII):

Receives	*Visual Object*	To be identified.
Produces	An *Instance ID*	Identifying an element of a set of Visual Objects belonging to a level in a taxonomy.

7.5.5.2 Reference Model

Figure 1 specifies the Reference Model of the Visual Instance Identification (OSD-VII) AIM.

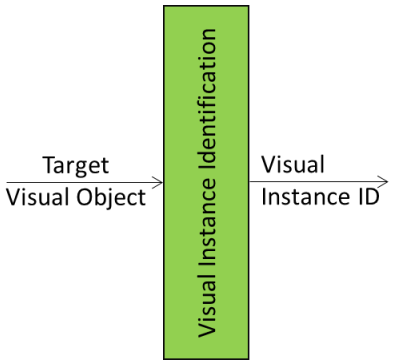

Figure 1 – The Visual Instance Identification (OSD-VII) AIM Reference Model

7.5.5.3 Input/Output Data

Table 1 specifies the Input and Output Data of the Visual Instance Identification (OSD-VII) AIM.

Table 1 – I/O Data of Visual Instance Identification (OSD-VII) AIM

Input	**Description**
Target Visual Object	The Visual Object crossed by the line traversing the forefinger of the Entity.
Output	**Description**
Visual Instance Identifier	The Identifier of the specific Visual Object belonging to a level in the taxonomy.

7.5.5.4 JSON Metadata

https://schemas.mpai.community/OSD/V1.2/AIMs/VisualInstanceIdentification.json

7.5.5.5 Conformance Testing

Table 2 provides the Conformance Testing Method for OSD-VII AIM.

Note that a schema may contain references to other schemas. In this case, validation of data for the primary schema implies that any data that refers to a secondary schema shall also validate.

Table 2 – Conformance Testing Method for OSD-VII AIM

Receives	Visual Object	Shall validate against Visual Object schema. Visual Data shall conform with Qualifier.
Produces	Instance ID	Shall validate against Instance ID schema.

Table 3 provides an example of MMC-AQM AIM conformance testing.

Table 3 – An example MMC-AQM AIM conformance testing

Input Data	Data Type	Input Conformance Testing Data
Input Image	JPEG	All input Text files to be drawn from Image files.
Output Data	**Data Type**	**Data Format**
Object Instance ID	Identifier	All Identifiers of Visual Objects shall be represented according to Instance Identifier

7.5.6 Visual Object Extraction

7.5.6.1 Functions

Visual Object Extraction (OSD-VOE):

Receives	*Visual Scene Geometry*	Description of object arrangement.
	Visual Objects	To be extracted for identification.
	Point of View	Crossed by line.
Extracts	*Visual Object*	Crossed by line from Point of View.

7.5.6.2 Reference Model

Figure 1 depicts the Reference Model of the Visual Object Extraction (OSD-VOE) AIM.

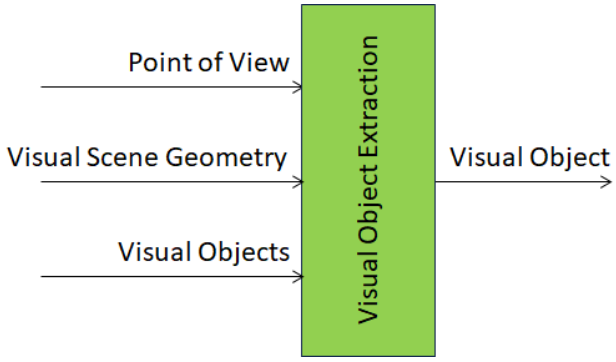

Figure 1 – The Visual Object Extraction (OSD-VOE) AIM Reference Model

7.5.6.3 Input/Output Data

Table 1 specifies the Input and Output Data of the Visual Object Extraction (OSD-VOE) AIM.

Table 1 – I/O Data of the Visual Object Extraction (OSD-VOE) AIM

Input	Description
Point of View	The direction of the line traversing the forefinger of the Entity.
Visual Scene Geometry	The digital representation of the spatial arrangement of the Visual Objects of the Scene.

Visual Objects	The Visual Objects of the Visual Scene Geometry.
Output	**Description**
Target Visual Object ID	The Visual Object crossed by the line traversing the forefinger of the Entity.

7.5.6.4 JSON Metadata

https://schemas.mpai.community/OSD/V1.2/AIMs/VisualObjectExtraction.json

7.5.6.5 Conformance Testing

Table 2 provides the Conformance Testing Method for MMC-VOE AIM.

Note that a schema may contain references to other schemas. In this case, validation of data for the primary schema implies that any data that refers to a secondary schema shall also validate.

Table 2 – Conformance Testing Method for MMC-VOE AIM

Receives	Visual Scene Geometry	Shall validate against Visual Scene Geometry schema.
	Visual Objects	Shall validate against Visual Object schema. Visual Data shall conform with Qualifier.
	Point of View	Shall validate against Point of View schema.
Extracts	Visual Object	Shall validate against Visual Object schema. Visual Data shall conform with Qualifier.

7.5.7 Visual Object Identification

7.5.7.1 Functions

Visual Object Identification (OSD-VOI):

Receives	*Visual Scene Geometry*	The arrangement of the objects in the Scene.
	Visual Objects	The Objects in the Scene.
	Body Descriptors	Descriptors of the Body indicating the object.
Produces	*Visual Instance ID*	Identifying a Visual Object in the Scene that belongs to some level in a taxonomy.

7.5.7.2 Reference Model

Figure 1 specifies the Reference Model of Visual Object Identification (OSD-VOI) AIM.

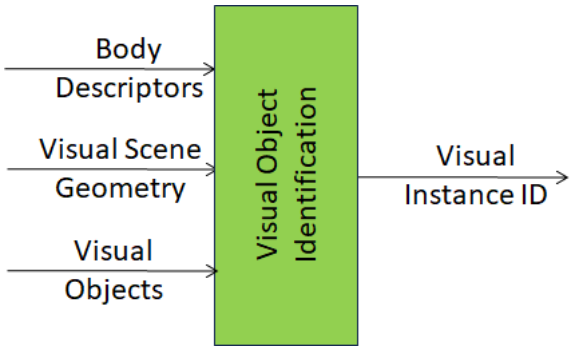

Figure 1- The Visual Object Identification (OSD-VOI) AIM Reference Model

7.5.7.3 Input/Output Data

Table 1 specifies the Input and Output Data of the Visual Object Identification (OSD-VOI) AIM.

Table 1 – I/O Data of the Visual Object Identification (OSD-VOI) AIM

Input	Description
Body Descriptors	The Descriptors of the Body Objects of Entities in the Visual Scene.
Visual Scene Geometry	The digital representation of the spatial arrangement of the Visual Objects of the Scene.
Visual Object	The Visual Objects in the Visual Scene that are not Entities.
Output	**Description**
Visual Instance Identifier	The Identifier of the specific Visual Object belonging to a level in the taxonomy.

7.5.7.4 SubAIMs

Visual Object Identification (OSD-VOI) is a Composite AIM specified by Figure 2.

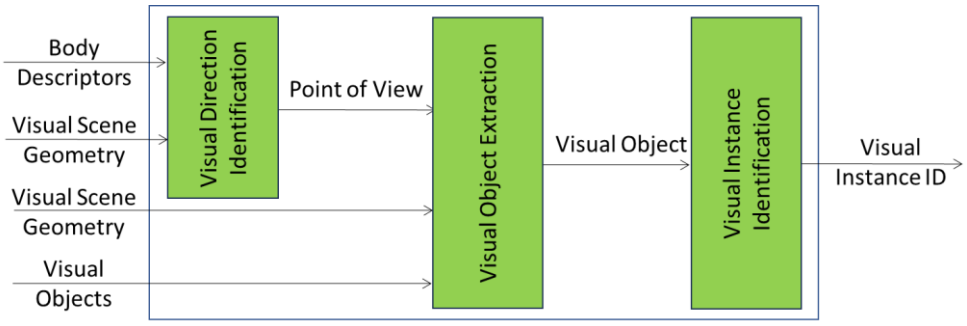

Figure 2 - The Visual Object Identification (OSD-VOI) Composite AIM

The Visual Direction Identification AIM shall be able to parse either an AV Scene Geometry or its Visual Scene Geometry subset.

The AIMs composing the Visual Object Identification (OSD-VOI) Composite AIM are:

AIM	AIMs	Names
OSD-VOI		Visual Object Identification
	OSD-VDI	Visual Direction Identification
	OSD-VOE	Visual Object Extraction
	OSD-VII	Visual Instance Identification

7.5.7.5 JSON Metadata

https://schemas.mpai.community/OSD/V1.2/AIMs/VisualObjectIdentification.json

7.5.7.6 Conformance Testing

Table 2 provides the Conformance Testing Method for OSD-VOI AIM. Conformance Testing of the individual AIMs of the OSD-VOI Composite AIM are given by the individual AIM Specification.

Note that a schema may contain references to other schemas. In this case, validation of data for the primary schema implies that any data that refers to a secondary schema shall also validate.

Table 2 – Conformance Testing Method for OSD-VOI AIM

Receives	Visual Scene Geometry	Shall validate against Visual Scene Geometry schema.
	Visual Objects	Shall validate against Visual Objects schema. Visual Data shall conform with Qualifier.
	Body Descriptors	Shall validate against Body Descriptors XML schema.
Produces	Visual Instance ID	Shall validate against Instance ID schema.

7.5.8 Visual Scene Description

7.5.8.1 Functions

Visual Scene Description (OSD-VSD):

Receives	Space-Time	of the Scene
	Visual Objects.	individual Visual Objects.
	Visual Scene Descriptors	Additional Visual Scene to be augmented.
Processes	Visual Objects	
Creates	Visual Scene Descriptors	from the Visual Objects, if possible.
Combines	The Visual Scene Descriptors	of the Visual Object and the input Visual Scene Descriptors.
Produces	Visual Scene Descriptors	

7.5.8.2 Reference Model

The Reference Architecture is depicted in Figure 1.

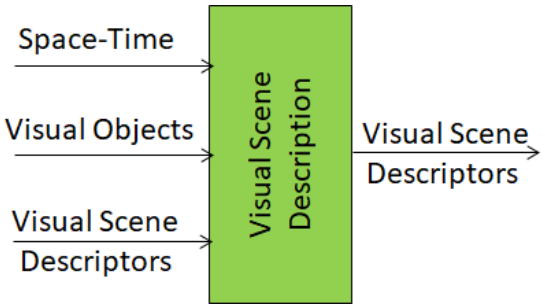

Figure 1 – The Visual Scene Description (OSD-VSD) AIM

7.5.8.3 Input/Output Data

Table 1 specifies the Input and Output Data of the Visual Scene Description (OSD-VSD) AIM. Links are to the Data Type specifications.

Table 1 – I/O Data of the Visual Scene Description (OSD-VSD) AIM

Input	**Description**
Space-Time	Visual Scene captured by Machine.
Visual Objects	Input Visual Objects.
Visual Scene Descriptors	Input Visual Scene Descriptors.
Output	**Description**
Visual Scene Descriptors	The output Visual Descriptors.

7.5.8.4 JSON Metadata

https://schemas.mpai.community/OSD/V1.2/AIMs/VisualSceneDescription.json

7.5.8.5 Conformance Testing

Table 2 provides the Conformance Testing Method for OSD-VSD AIM.

If a schema contains references to other schemas, conformance of data for the primary schema implies that any data referencing a secondary schema shall also validate against the relevant schema, if present and conform with the Qualifier, if present.

Table 2 – Conformance Testing Method for OSD-VSD AIM

Receives	SpaceTime	Shall validate against Space-Time schema.
	Visual Objects	Shall validate against Visual Object schema. Visual Data shall conform with Qualifier.
	Visual Scene Descriptors	Shall validate against Visual Scene Descriptors schema.
Produces	Visual Scene Descriptors	Shall validate against Visual Scene Descriptors schema.

7.6 AI modules from MPAI-PAF

7.6.1 Audio-Visual Scene Rendering

7.6.1.1 Functions

Audio-Visual Scene Rendering (PAF-AVR):

Receives	Point of View	To be used in rendering the scene and its objects.
	Audio-Visual Scene Descriptors	jointly with or alternatively with Portable Avatar.
	Portable Avatar	Jointly with or alternatively with AV Scene Descriptors.
Transforms	Portable Avatar	Into generic Audio-Visual Scene Descriptors if input is Portable Avatar.
Produces	Portable Avatar's Output Speech	Always integrated in the Audio-Visual Scene. Output Speech results from the rendering of Audio Scene Descriptors from human-selected Point of View.
	Output Audio	Resulting from the rendering of Audio Scene Descriptors from human-selected Point of View.
	Output Visual	Resulting from the rendering of Audio Scene Descriptors from human-selected Point of View. View Selector tells the OSD-AVR AIM where the visual components of the Portable Avatar should also be integrated.

7.6.1.2 Reference Model

Figure 1 specifies the Reference Model of the Audio-Visual Scene Rendering (PAF-AVR) AIM.

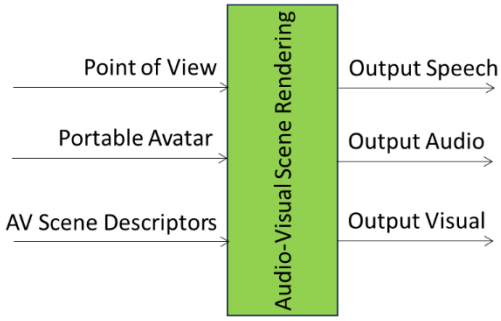

Figure 1 – The Audio-Visual Scene Rendering (PAF-AVR) AIM

7.6.1.3 Input/Output Data

Table 1 specifies the Input and Output Data of the Audio-Visual Scene Rendering (PAF-AVR) AIM.

Table 1 – I/O Data of the Audio-Visual Scene Rendering (PAF-AVR) AIM

Input	Description
Portable Avatar	Data produced, e.g., by Personal Status Display.
AV Scene Descriptors	Audio-Visual Scene Descriptors.
Point of View	Point from where an Entity perceives the Audio-Visual Scene
Output	**Description**
Output Speech	The Speech components of the Audio-Visual Scene.
Output Audio	The Audio components of the Audio-Visual Scene.
Output Visual	The Visual components of the Audio-Visual Scene.

7.6.1.4 JSON Metadata

https://schemas.mpai.community/PAF/V1.3/AIMs/AudioVisualSceneRendering.json

7.6.1.5 Profiles

The Profiles of Audio-Visual Scene Rendering are specified.

7.6.1.6 Conformance Testing

Table 2 provides the Conformance Testing Method for PAF-AVR AIM.

If a schema contains references to other schemas, conformance of data for the primary schema implies that any data referencing a secondary schema shall also validate against the relevant schema, if present and conform with the Qualifier, if present.

Table 2 – Conformance Testing Method for PAF-AVR AIM

Receives	Portable Avatar	Shall validate against Point of View Schema.
	AV Scene Descriptors	Shall validate against AV Scene Descriptors Schema.
	Point of View	Shall validate against Portable Avatar Schema. Portable Avatar Data shall conform with respective Qualifiers.
Produces	Output Speech	Shall validate against Speech Object Schema. Speech Data shall conform with Speech Qualifier.
	Output Audio	Shall validate against Audio Object Schema. Audio Data shall conform with Audio Qualifier.
	Output Visual	Shall validate against Visual Object or 3D Model Schema. Visual Data shall conform with Visual Object.

7.6.2 Face Identity Recognition

7.6.2.1 Functions

Face Identity Recognition (PAF-FIR):

Receives	*Text Object*	Text that is related with the Face to be identified.
	Image Visual Object	Image containing Face to be identified.
	Face Time	Time when the face should be identified.
	Visual Scene Geometry	Of the scene where the Face is located.
Searches for	*Bounding Boxes*	That include faces
Finds	best match	Between the Faces and those in a database.
Produces	*Face Identities*	Face Instance Identifiers.
	Bounding Boxes	Bounding Boxes that include faces.

7.6.2.2 Reference Model

Figure 1 depicts the Reference Model of the Face Identity Recognition AIM.

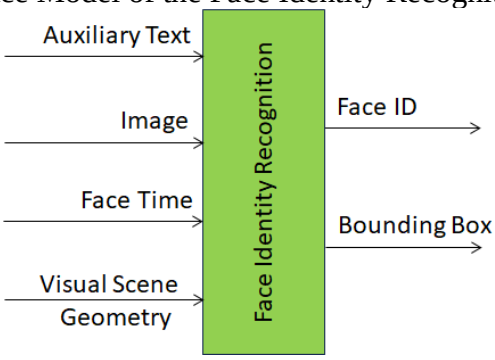

Figure 1 – Face Identity Recognition AIM

7.6.2.3 Input/Output Data

Table 1 specifies the Input and Output Data of the of the Face Identity Recognition AIM.

Table 1 – I/O Data of Face Identity Recognition AIM

Input	Description
Auxiliary Text	Text with a content related to Face ID.
Image Visual Object	An image containing the Face to be identified.
Face Time	The Time during which the Face should be identified.
Visual Scene Geometry	The Geometry of the Scene where the Face is located.
Output	**Description**
Face Identifiers	Associate strings to elements belonging to some levels in a hierarchical classification (taxonomy).
Bounding Boxes	The box containing the Face identified.

7.6.2.4 JSON Metadata

https://schemas.mpai.community/PAF/V1.3/AIMs/FaceIdentityRecognition.json

7.6.2.5 Reference Software

7.6.2.5.1 Disclaimers

1. This PAF-FIR Reference Software Implementation is released with the BSD-3-Clause licence.
2. The purpose of this PAF-FIR Reference Software is to show a working Implementation of PAF-FIR, not to provide a ready-to-use product.
3. MPAI disclaims the suitability of the Software for any other purposes and does not guarantee that it is secure.
4. Use of this Reference Software may require acceptance of licences from the respective repositories. Users shall verify that they have the right to use any third-party software required by this Reference Software.

7.6.2.5.2 Guide to the PAF-FIR code

Use of this Reference Software for the PAF-FIR AI Module is for developers who are familiar with Python, Docker, RabbitMQ, and downloading models from HuggingFace
PAF-FIR performs face identity recognition with a pretrained FaceNet model; that is, it identifies the faces in a given number of frames per scene by comparison with a dataset of faces.
The PAF-FIR Reference Software is found at the MPAI gitlab site. It contains:

1. src: a folder with the Python code implementing the AIM
2. Dockerfile: a Docker file containing only the libraries required to build the Docker image and run the container
3. requirements.txt: dependencies installed in the Docker image
4. README.md: where to find and save weights of face recognition model FaceNet512.

Library: https://github.com/serengil/deepface

7.6.2.5.3 Acknowledgements

This version of the PAF-FIR Reference Software has been developed by the MPAI *AI Framework* Development Committee (AIF-DC).

7.6.2.6 Conformance Testing

Table 2 provides the Conformance Testing Method for PAF-FIR AIM.
If a schema contains references to other schemas, conformance of data for the primary schema implies that any data referencing a secondary schema shall also validate against the relevant schema, if present and conform with the Qualifier, if present.

Table 2 – Conformance Testing Method for PAF-FIR AIM

Receives	Text Object	Shall validate against Text Object Schema.
	Visual Object (Image)	Shall validate against Visual Object Schema. Image Data shall conform with Visual Qualifier.
	Face Time	Shall validate against Time Schema.
	Visual Scene Geometry	Shall validate against Visual Scene Geometry Schema.
Produces	Face Instance IDs	Shall validate against Instance ID Schema.
	Visual Object (Bounding Box)	Shall validate against Bounding Box Schema. Bounding Box Data shall conform with Visual Qualifier.

7.6.2.7 Performance Assessment

Performance Assessment of an PAF-FIR AIM Implementation shall be performed using a dataset of faces for each face of which the Identity of the face is provided with reference to a Taxonomy.
The Performance Assessment Report of an PAF-FIR AIM Implementation shall include:

1. The Identifier of the PAF-FIR AIM.
2. The identifier of the face dataset.
3. The identifier of the Taxonomy of face identifiers.
4. The Performance of the PAF-FIR AIM Implementation expressed by the Accuracy of the Identifiers provided by the output of the PAF-FIR AIM computed on all faces of the dataset referenced in 2 using the Taxonomy referenced in 3.

7.6.3 Entity Body Description

7.6.3.1 Functions

Entity Body Description (PAF-EBD):

Receives	*Body Visual Object*	Body of Entity or from upstream AIM.
Produces	*Body Descriptors*	Descriptors of Body Visu*al Object*

7.6.3.2 Reference Model

Figure 1 specifies the Reference Model of the Entity Body Description (PAF-EBD) AIM.

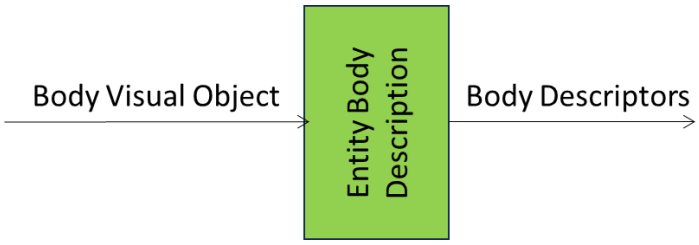

Figure 1 Entity Body Description (PAF-EBD) AIM

7.6.3.3 Input/Output Data

Table 1 specifies the Input and Output Data of Entity Body Description (PAF-EBD) AIM.

Table 1 – I/O Data of the Entity Body Description (PAF-EBD) AIM

Input	Description
Body Visual Object	Body of Entity.
Output	**Description**
Body Descriptors	Descriptors of Body

7.6.3.4 JSON Metadata

https://schemas.mpai.community/PAF/V1.3/AIMs/EntityBodyDescription.json

7.6.3.5 Conformance Testing

Table 2 provides the Conformance Testing Method for MMC-EBD AIM.

If a schema contains references to other schemas, conformance of data for the primary schema implies that any data referencing a secondary schema shall also validate against the relevant schema, if present and conform with the Qualifier, if present.

Table 2 – Conformance Testing Method for MMC-EBD AIM

Receives	Body Visual Object	Shall validate against Visual Object Schema. Body Data shall conform with Visual Object.
Produces	Body Descriptors	Shall validate against Body Descriptors XML Schema.

7.6.4 Face Identity Recognition

7.6.4.1 Functions

Face Identity Recognition (PAF-FIR):

Receives	*Text Object*	Text that is related with the Face to be identified.
	Image Visual Object	Image containing Face to be identified.
	Face Time	Time when the face should be identified.
	Visual Scene Geometry	Of the scene where the Face is located.
Searches for	*Bounding Boxes*	That include faces
Finds	best match	Between the Faces and those in a database.
Produces	*Face Identities*	Face Instance Identifiers.
	Bounding Boxes	Bounding Boxes that include faces.

7.6.4.2 Reference Model

Figure 1 depicts the Reference Model of the Face Identity Recognition AIM.

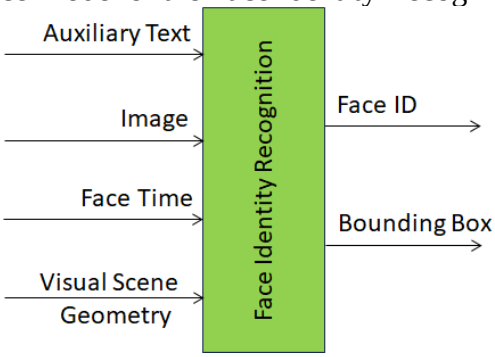

Figure 1 – Face Identity Recognition AIM

7.6.4.3 Input/Output Data

Table 1 specifies the Input and Output Data of the of the Face Identity Recognition AIM.

Table 1 – I/O Data of Face Identity Recognition AIM

Input	Description
Auxiliary Text	Text with a content related to Face ID.

Image Visual Object	An image containing the Face to be identified.
Face Time	The Time during which the Face should be identified.
Visual Scene Geometry	The Geometry of the Scene where the Face is located.
Output	**Description**
Face Identifiers	Associate strings to elements belonging to some levels in a hierarchical classification (taxonomy).
Bounding Boxes	The box containing the Face identified.

7.6.4.4 JSON Metadata

https://schemas.mpai.community/PAF/V1.3/AIMs/FaceIdentityRecognition.json

7.6.4.4.1 Disclaimers

1. This PAF-FIR Reference Software Implementation is released with the BSD-3-Clause licence.
2. The purpose of this PAF-FIR Reference Software is to show a working Implementation of PAF-FIR, not to provide a ready-to-use product.
3. MPAI disclaims the suitability of the Software for any other purposes and does not guarantee that it is secure.
4. Use of this Reference Software may require acceptance of licences from the respective repositories. Users shall verify that they have the right to use any third-party software required by this Reference Software.

7.6.4.4.2 Guide to the PAF-FIR code

Use of this Reference Software for the PAF-FIR AI Module is for developers who are familiar with Python, Docker, RabbitMQ, and downloading models from HuggingFace
PAF-FIR performs face identity recognition with a pretrained FaceNet model; that is, it identifies the faces in a given number of frames per scene by comparison with a dataset of faces.
The PAF-FIR Reference Software is found at the MPAI gitlab site. It contains:

1. src: a folder with the Python code implementing the AIM
2. Dockerfile: a Docker file containing only the libraries required to build the Docker image and run the container
3. requirements.txt: dependencies installed in the Docker image
4. README.md: where to find and save weights of face recognition model FaceNet512.

Library: https://github.com/serengil/deepface

7.6.4.4.3 Acknowledgements

This version of the PAF-FIR Reference Software has been developed by the MPAI *AI Framework* Development Committee (AIF-DC).

7.6.4.5 Conformance Testing

Table 2 provides the Conformance Testing Method for PAF-FIR AIM.
If a schema contains references to other schemas, conformance of data for the primary schema implies that any data referencing a secondary schema shall also validate against the relevant schema, if present and conform with the Qualifier, if present.

Table 2 – Conformance Testing Method for PAF-FIR AIM

Receives	Text Object	Shall validate against Text Object Schema.
	Visual Object (Image)	Shall validate against Visual Object Schema. Image Data shall conform with Visual Qualifier.
	Face Time	Shall validate against Time Schema.
	Visual Scene Geometry	Shall validate against Visual Scene Geometry Schema.
Produces	Face Instance IDs	Shall validate against Instance ID Schema.
	Visual Object (Bounding Box)	Shall validate against Bounding Box Schema. Bounding Box Data shall conform with Visual Qualifier.

7.6.4.6 Performance Assessment

Performance Assessment of an PAF-FIR AIM Implementation shall be performed using a dataset of faces for each face of which the Identity of the face is provided with reference to a Taxonomy. The Performance Assessment Report of an PAF-FIR AIM Implementation shall include:

1. The Identifier of the PAF-FIR AIM.
2. The identifier of the face dataset.
3. The identifier of the Taxonomy of face identifiers.
4. The Performance of the PAF-FIR AIM Implementation expressed by the Accuracy of the Identifiers provided by the output of the PAF-FIR AIM computed on all faces of the dataset referenced in 2 using the Taxonomy referenced in 3.

7.6.5 Portable Avatar Demultiplexing

7.6.5.1 Functions

Portable Avatar Demultiplexing (PAF-PDX):

Receives	Portable Avatar
Demultiplexes	Elements in Portable Avatar.
Produces	- Portable Avatar ID
	- Avatar Space-Time
	- Avatar
	- Language Selector
	- Speech Object
	- Text Object
	- Speech Model
	- Personal Status
	- Audio Visual Scene Descriptors
	- Audio Visual Scene Space Time

7.6.5.2 Reference Model

Figure 1 specifies the Reference Model of the Personal Avatar Demultiplexing (PAF-PDX) AIM.

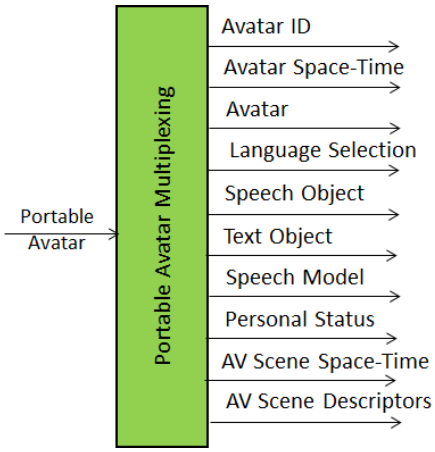

Figure 1– The Personal Avatar Demultiplexing (PAF-PDX) AIM

7.6.5.3 Input/Output Data

Table 1 specifies the Input and Output Data of the Portable Avatar Demultiplexing (PAF-PDX) AIM.

Table 1 – Portable Avatar Demultiplexing (PAF-PDX) AIM

Input	Description
Portable Avatar	From an upstream AIM or another AIW.
Output	**Description**
AvatarID	Avatar ID.
Avatar Space-Time	Portable Avatar Time.
Avatar	Avatar in Portable Avatar.
Language Selector	Language of Avatar.
Speech Object	The Speech in the time when the PA is valid.
Text Object	The Time in the time when the PA is valid.
Speech Model	The NN Model used to synthesise text.
Avatar Personal Status	The Avatar's Personal Status.
AV Scene Descriptors	Descriptors of AV Scene.
AV Scene Space-Time	Space-Time info of AV Scene.

7.6.5.4 JSON Metadata

https://schemas.mpai.community/PAF/V1.3/AIMs/PortableAvatarDemultiplexing.json

7.6.5.5 Conformance Testing

Table 2 provides the Conformance Testing Method for PAF-PDX AIM.

Note that a schema may contain references to other schemas. In this case, validation of data for the primary schema implies that any data that refers to a secondary schema shall also validate.

Table 2 – Conformance Testing Method for PAF-PDX AIM

Receives	Portable Avatar	Shall validate against Portable Avatar Schema. Portable Avatar Data shall conform with respective Qualifiers.
Produces	Portable Avatar ID	Shall be string or validate against Instance ID Schema.
	Avatar Space-Time	Shall validate against Space-Time Schema.
	Avatar	Shall validate against Avatar Schema. Avatar Model Data shall conform with 3D Model Qualifier.
	Language Selector	Shall validate against "Language" Selector Schema.
	Text Object	Shall validate against Text Object Schema. Text Data shall conform with Text Qualifier.
	Speech Object	Shall validate against Speech Object Schema. Speech Data shall conform with Speech Qualifier.
	Speech Model	Shall validate against Machine Learning Model Schema. Speech Model Data shall conform with Machine Learning Model Qualifier.
	Personal Status	Shall validate against Personal Status Schema.
	Audio Visual Scene Descriptors	Shall validate against AV Scene Descriptors Schema.
	Audio Visual Scene Space-Time	Shall validate against Space-Time Schema.

7.6.6 PS-Face Interpretation

7.6.6.1 Functions

PS-Face Interpretation (PAF-PFI):

Receives	*Face Descriptors*	from Face Description or as input to PS-Face Interpretation
Produces	*Face Personal Status*	the Personal Status of the Face Modality

7.6.6.2 Reference Model

Figure 1 specifies the Reference Architecture of the PS-Face Interpretation (PAF-PFI) AIM.

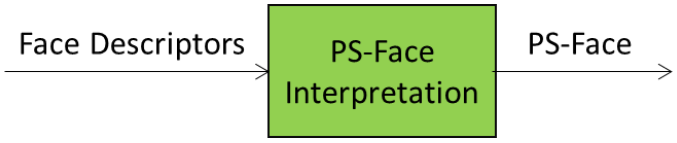

Figure 1– The PS-Face Interpretation (PAF-PFI) AIM Reference Model

7.6.6.3 Input/Output Data

Table 1specifies the Input and Output Data of the PS-Face Interpretation (PAF-PFI) AIM.

Table 1– I/O Data of the PS-Face Interpretation (PAF-PFI) AIM

Input	**Description**
Face Descriptors	Descriptors of Face
Output	**Description**
Face Personal Status	Personal Status of Face

7.6.6.4 JSON Metadata

https://schemas.mpai.community/PAF/V1.3/AIMs/PSFaceInterpretation.json

7.6.6.5 Conformance Testing

Table 2 provides the Conformance Testing Method for PAF-PFI AIM.

Note that a schema may contain references to other schemas. In this case, validation of data for the primary schema implies that any data that refers to a secondary schema shall also validate.

Table 2 – Conformance Testing Method for PAF-PFI AIM

Receives	Face Descriptors	Shall validate against Face Descriptors Schema
Produces	Face Personal Status	Shall validate against Face Personal Status Schema

7.6.7 PS-Gesture Interpretation

7.6.7.1 Functions

PS-Body Interpretation (PAF-PGI):

Receives	*Gesture Descriptors*	from a Gesture Description AIM or as input to PS-Gesture Interpretation
Produces	*PS-Gesture*	the Personal Status conveyed by the Body Modality.

7.6.7.2 Reference Model

Figure 1 specifies the Reference Architecture of the PS-Gesture Interpretation (PAF-PGI) AIM.

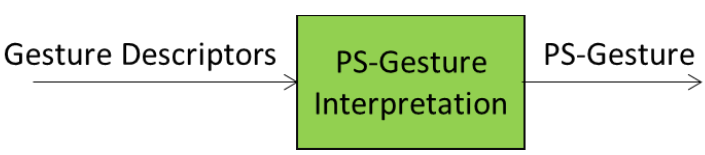

Figure 1– The PS-Gesture Interpretation (PAF-PGI) AIM

7.6.7.3 Input/Output Data

Table 1 specifies the Input and Output Data of the PS-Gesture Interpretation (PAF-PGI) AIM.

Table 1– I/O Data of the PS-Gesture Interpretation (PAF-PGI) AIM

Input	Description
Gesture Descriptors	Descriptors of Body
Output	**Description**
Gesture Personal Status	Personal Status of Body

7.6.7.4 JSON Metadata

https://schemas.mpai.community/PAF/V1.3/AIMs/PSGestureInterpretation.json

7.6.7.5 Conformance Testing

Table 2 provides the Conformance Testing Method for PAF-PGI AIM.

If a schema contains references to other schemas, conformance of data for the primary schema implies that any data referencing a secondary schema shall also validate against the relevant schema, if present and conform with the Qualifier, if present.

Table 2 – Conformance Testing Method for PAF-PGI AIM

Receives	Gesture Descriptors	Shall validate against Gesture Descriptors Schema
Produces	Gesture Personal Status	Shall validate against Gesture Personal Status Schema

7.6.8 Personal Status Display

7.6.8.1 Functions

Personal Status Display (PAF-PSD):

Receives	Machine ID	ID to be used to identify the Avatar in Portable Avatar.
	Text Object	Text associated to Avatar in Portable Avatar.
	Personal Status	Personal Status associated to Avatar in Portable Avatar.
	Avatar Model	3D Model associated to Avatar in Portable Avatar.
	Speech Model	Speech Model Associated to Avatar in Portable Avatar.
Produces	Portable Avatar	Output Portable Avatar.
Enables	PAF-AVR	To render the Portable Avatar produced by PAF-PSD.

7.6.8.2 Reference Model

Figure 1 depicts the AIMs implementing the Personal Status Display (PAF-PSD) AIM.

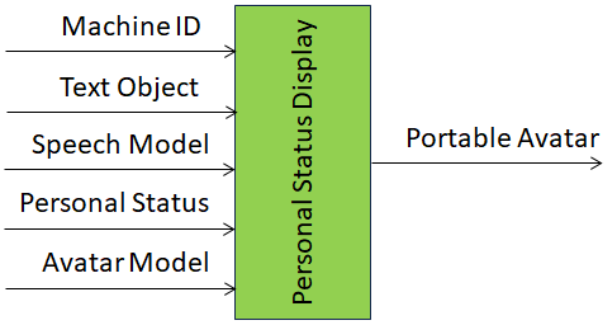

Figure 1 – Reference Model of Personal Status Display (PAF-PSD) AIM

7.6.8.3 Input/Output Data

Table 1 gives the Input/Output Data of Personal Status Display (PAF-PSD).

Table 1 – I/O Data of Personal Status Display

Input data	From	Description
Avatar ID	Upstream AIM	Portable Avatar's ID
Avatar Model	Upstream AIM or embedded in PSD	Part of Portable Avatar
Text Object	Keyboard or upstream AIM	Texts of Portable Avatar
Personal Status	Personal Status Extraction or Machine	To add PS to Speech, Face, and Gesture
Speech Model	Upstream AIM or embedded in PSD	Neural Network
Output data	**To**	**Description**
Portable Avatar	Downstream AIM or renderer	As Portable Avatar

7.6.8.4 SubAIMs

Figure 2 gives the Reference Model of the the Personal Status Display Composite AIM.

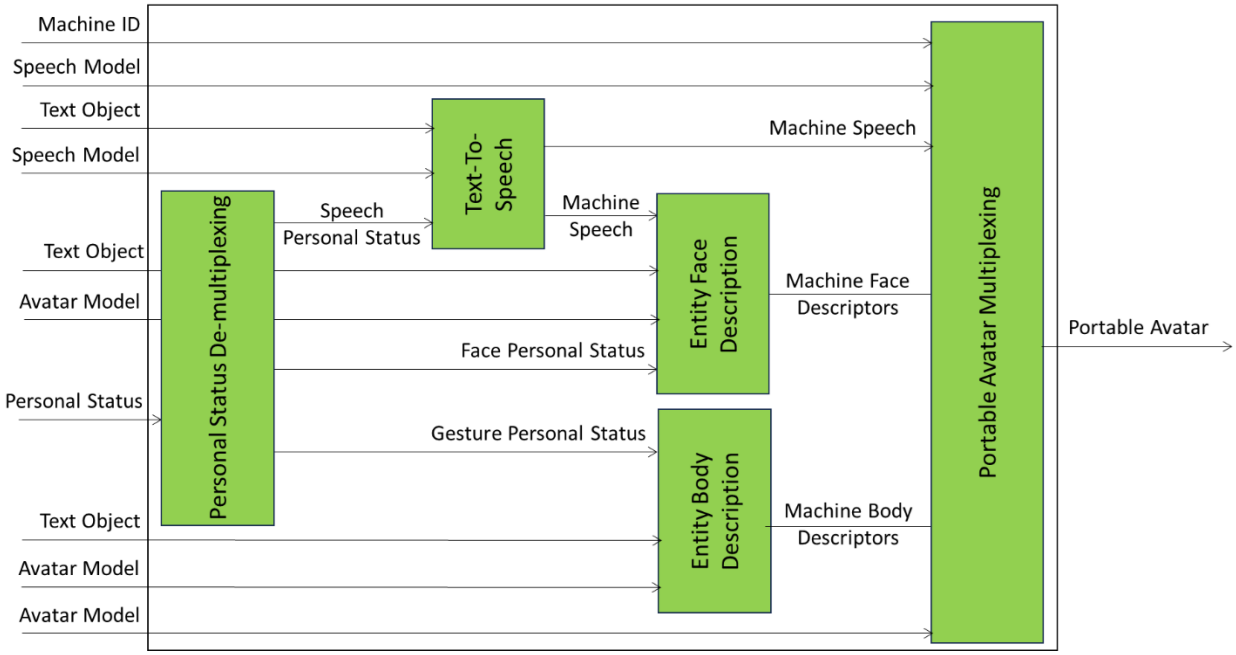

Figure 2 – Reference Model of Personal Status Display Composite AIM

The Personal Status Display Composite AIM operates as follows:

1. Avatar ID is the ID of the Portable Avatar.
2. Personal Status Demultiplexing makes available the component PS-Speech, PS-Face, and PS-Gesture Modalities.
3. Machine Text is synthesised as Speech using a Speech Model in a format specified by NN Format and the Personal Status provided by PS-Speech.
4. Machine Speech and PS-Face are used to produce the Machine Face Descriptors.
5. PS-Gesture and Text are used for Machine Body Descriptors using the Avatar Model.
6. Portable Avatar Multiplexing produces the Portable Avatar.

Table 2 gives the list of PSD AIMs with their input and output Data.

Table 2 –AIMs of Personal Status Display Composite AIM and JSON Metadata

AIW	AIMs	Name and Specification	JSON
PAF-PSD		Personal Status Display	X
	MMC-PDX	Personal Status Demultiplexing	X
	MMC-TTS	Text-to-Speech	X
	PAF-EFD	Entity Face Description	X
	PAF-EBD	Entity Body Description	X
	PAF-PMX	Portable Avatar Multiplexing	X

7.6.8.5 JSON Metadata

https://schemas.mpai.community/PAF/V1.3/AIMs/PersonalStatusDisplay.json

7.6.8.6 Profiles

The Profiles of Personal Status Display are specified.

7.6.8.7 Conformance Testing

The Conformance Testing Method for the PAF-PSD Basic AIM is provided here. The Conformance Testing Method for the individual Basic AIMs of the PAF-PSD Composite AIM is provided by the individual Basic AIMs.
Table 2 provides the Conformance Testing Method for PAF-PSD AIM.
If a schema contains references to other schemas, conformance of data for the primary schema implies that any data referencing a secondary schema shall also validate against the relevant schema, if present and conform with the Qualifier, if present.

Table 2 – Conformance Testing Method for PAF-PSD AIM

Receives	Machine ID	Shall be string or validate against Instance ID Schema
	Text Object	Shall validate against Text Object Schema. Text Data shall conform with Speech Qualifier.
	Personal Status	Shall validate against Personal Status Schema.
	Avatar Model	Shall validate against 3D Model Schema. Avatar Model Data shall conform with 3D Model Qualifier.
	Speech Model	Shall validate against Machine Learning Model Schema. Speech Model Data shall conform with Machine Learning Model Qualifier.

Produces	Portable Avatar	Shall validate against Portable Avatar Schema. Portable Avatar Data shall conform with respective Qualifiers.

7.6.9 Portable Avatar Multiplexing

7.6.9.1 Functions

Portable Avatar Multiplexing (PSD-PMX):

Receives	An arbitrary number of elements in Portable Avatar out of:
	- Portable Avatar ID
	- Avatar Space-Time
	- Avatar
	- Language Selector
	- Text Object
	- Speech Model
	- Speech Object
	- Personal Status
	- Audio- Visual Scene Space-Time
	- Audio-Visual Scene Descriptors
	- An existing Portable Avatar
Changes	Existing with Input Data
Adds	Input Data that is not in the Input Portable Avatar
Produces	Portable Avatar

7.6.9.2 Reference Model

Figure 1 specifies the Reference Model of the Portable Avatar Multiplexing (PSD-PMX) AIM.

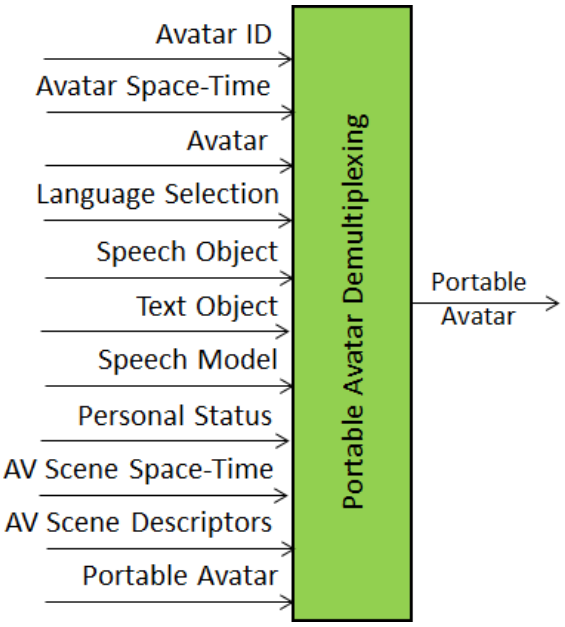

Figure 1 – The Portable Avatar Multiplexing (PSD-PMX) AIM

7.6.9.3 Input/Output Data

Table 1 specifies the Input and Output Data of the Portable Avatar Multiplexing (PSD-PMX) AIM.

Table *1* – I/O Data of the Portable Avatar Multiplexing (PSD-PMX) AIM

Input	Description
AvatarID	Avatar ID.
Avatar Space-Time	Portable Avatar Time.
Avatar	Avatar in Portable Avatar.
Language Selector	Language of Avatar.
Speech Object	The Speech in the time when the PA is valid.
Text Object	The Time in the time when the PA is valid.
Speech Model	The NN Model used to synthesise text.
Avatar Personal Status	The Avatar's Personal Status.
AV Scene Descriptors	Descriptors of AV Scene.
AV Scene Space-Time	Space-Time info of AV Scene.
Portable Avatar	The input Portable Item.
Output	**Description**
Portable Avatar	The output Portable Item.

7.6.9.4 JSON Metadata

https://schemas.mpai.community/PAF/V1.3/AIMs/PortableAvatarMultiplexing.json

7.6.9.5 Conformance Testing

Table 2 provides the Conformance Testing Method for PAF-PMX AIM.

If a schema contains references to other schemas, conformance of data for the primary schema implies that any data referencing a secondary schema shall also validate against the relevant schema, if present and conform with the Qualifier, if present.

Table 2 – Conformance Testing Method for PAF-PMX AIM

Receives	Portable Avatar ID	Shall be string or validate against Instance ID Schema
	Avatar Space-Time	Shall validate against Space-Time Schema
	Avatar	Shall validate against Avatar Schema. Avatar Model Data shall conform with 3D Model Qualifier.
	Language Selector	Shall validate against Selector Schema
	Text Object	Shall validate against Text Object Schema. Text Data shall conform with Text Qualifier.
	Speech Object	Shall validate against Speech Object Schema. Speech Data shall conform with Speech Qualifier.
	Speech Model	Shall validate against Machine Learning Model Schema. Speech Model Data shall conform with Machine Learning Model Qualifier.
	Personal Status	Shall validate against Personal Status Schema.
	Audio Visual Scene Descriptors	Shall validate against AV Scene Descriptors Schema.
	Audio Visual Scene Space-Time	Shall validate against Space-Time Schema.
Produces	Portable Avatar	Shall validate against Portable Avatar Schema. Portable Avatar Data shall conform with respective Qualifiers.

8 Data Types

MPAI-HMC V2.0 only uses Data Types defined by other MPAI Technical Specifications. Table 1 provides the full list with web links of the Data Types utilised by HMC-CEC organised according to the Technical Specifications that specify them.

Table 1 - AI Modules utilised by HMC-CEC

MPAI-AIF	MPAI-MMC	MPAI-OSD	MPAI-PAF
Machine Learning Model	Cognitive State	Audio-Visual Basic Scene Descriptors	3D Model
MPAI-CAE	Emotion	Audio-Visual Basic Scene Geometry	Avatar

Audio Basic Scene Descriptors	Intention	Audio-Visual Event Descriptors	Body Descriptors
Audio Basic Scene Geometry	Meaning	Audio-Visual Object	Face Descriptors
Audio Object	Personal Status	Audio-Visual Scene Descriptors	Portable Avatar
Audio Scene Descriptors	Social Attitude	Audio-Visual Scene Geometry	
Audio Scene Geometry	Speech Descriptors	Instance Identifier	
	Speech Object	Point of View	
	Text Descriptors	Selector	
	Text Object	Space-Time	
		Spatial Attitude	
		Time	
		Visual Basic Scene Descriptors	
		Visual Basic Scene Geometry	
		Visual Object	
		Visual Scene Descriptors	
		Visual Scene Geometry	

8.1 Data Types from MPAI-AIF

8.1.1 Machine Learning Model

8.1.1.1 Definition

A Data Type an instance of which results from the application of training data to a process.

8.1.1.2 Functional Requirements

A Machine Learning Model enables the performance of specific functions such as classification most of which are target of MPAI Technical Specifications.

8.1.1.3 Syntax

https://schemas.mpai.community/AIF/V2.1/data/MLModel.json

8.1.1.4 Semantics

Label	Size	Description
Header	N1 Bytes	Machine Learning Model Header
Standard-MachineLearningModel	9 Bytes	The characters “AIF-MLM-V”
- Version	N2 Bytes	Major version – 1 or 2 characters
- Dot-separator	1 Byte	The character “.”
- Subversion	N3 Bytes	Minor version – 1 or 2 characters
MInstanceID	N4 Bytes	Identifier of M-Instance.

MLModelID	N5 Bytes	Identifier of Machine Learning Model.
MLModelQualifier	N6 Bytes	Qualifier of Machine Learning Model
MLModelDataLength	N7 Bytes	Length in Bytes of ML Model, i.e., Base Image that is required to operate the ML Model.
MLModeDataURI	N8 Bytes	URI of Machine Learning Model Data
DescrMetadata	N9 Bytes	Descriptive Metadata.

8.2 Data Types from MPAI-CAE

8.2.1 Audio Basic Scene Descriptors

8.2.1.1 Definition

A Data Type describing the Audio Objects and their spatial arrangement in an Audio Scene.

8.2.1.2 Functional Requirements

The Audio Basic Scene Descriptors Data Type includes:

1. The ID of a Virtual Space where the Audio Basic Scene are or will be located.
2. The ID of the Audio Basic Scene Descriptors.
3. The Space-Time information of the Audio Basic Scene.
4. The number of Audio Objects in the Audio Basic Scene.
5. The Audio Objects of the Audio Basic Scene including, for each Audio Object:
    1. The Audio Object Space-Time.
    2. The Audio Object ID or the Audio Object.

8.2.1.3 Syntax

https://schemas.mpai.community/CAE1/V2.3/data/AudioBasicSceneDescriptors.json

8.2.1.4 Semantics

Label	Size	Description
Header	N1 Bytes	Audio Basic Scene Descriptors Header
- Standard-AudioBasicSceneDescriptors	9 Bytes	The characters “OSD-ABS-V”
- Version	N2 Bytes	Major version – 1 or 2 characters
- Dot-separator	1 Byte	The character “.”
- Subversion	N3 Bytes	Minor version – 1 or 2 characters
MInstanceID	N4 Bytes	Identifier of M-Instance.
AudioBasicSceneDescriptorsID	N5 Bytes	Identifier of the Audio Basic Scene Descriptors.
AudioObjectCount	N6 Bytes	Number of Objects in Scene.
AudioBasicSceneSpaceTime	N7 Bytes	Data about the Audio Scene's Space-Time.
AudioBasicSceneDescriptorsData[]	N8 Bytes	Set of Audio Objects in Scene.
- AudioObjectSpaceTime	N9 Bytes	Space-Time info of Audio Object.
- AudioObjectID and/or AudioObject	N10 Bytes	Audio Object ID and/or Audio Object.
DescrMetadata	N11 Bytes	Descriptive Metadata.

8.2.1.5 Conformance Testing

A Data instance Conforms with CAE-USC V2.3 Audio Basic Scene Descriptors (CAE-ABS) if:

1. JSON Data validate against the Audio Basic Scene Descriptors' JSON Schema.
2. All Data in the Audio Basic Scene Descriptors JSON Schema
    1. Have the types specified.
    2. Validate against their JSON Schemas.
    3. Conform with their Data Qualifiers if present.

8.2.2 Audio Basic Scene Geometry

8.2.2.1 Definition

A Data Type including the arrangement of Audio Objects' in an Audio Scene and their Qualifiers.

8.2.2.2 Functional Requirements

The Audio Basic Scene Geometry Data Type includes:

1. The ID of a Virtual Space (M-Instance) where the Audio Basic Scene is or will be located.
2. The ID of the Audio Basic Scene Geometry.
3. The number of Audio Objects in the Audio Basic Scene.
4. The Space-Time Attributes of the Audio Basic Scene Geometry.
5. For each Audio Object in the Audio Basic Scene:
    1. The Space-Time information.
    2. The Audio Object Qualifiers.

8.2.2.3 Syntax

https://schemas.mpai.community/CAE1/V2.3/data/AudioBasicSceneGeometry.json

8.2.2.4 Semantics

Label	Size	Description
Header	N1 Bytes	Audio Basic Scene Geometry Header
– Standard-AudioBasicSceneGeometry	9 Bytes	The characters “OSD-ABG-V”
– Version	N2 Bytes	Major version – 1 or 2 characters
– Dot-separator	1 Byte	The character “.”
– Subversion	N3 Bytes	Minor version – 1 or 2 characters.
MInstanceID	N4 Bytes	Identifier of M-Instance.
AudioBasicSceneGeometryID	N5 Bytes	Identifier of the Audio Basic Scene Geometry.
AudioObjectCount	N6 Bytes	Number of Audio Objects in Audio Basic Scene.
AudioBasicSceneSpaceTime	N8 Bytes	Space and Time of Audio Basic Scene Geometry.
AudioBasicSceneAudioObjects[]	N9 Bytes	Set of Audio Objects.
– AudioObjectSpaceTime	N10 Bytes	Space and Time of Audio Object.
– AudioObjectQualifiers	N12 Bytes	Qualifiers of Audio Object.
DescrMetadata	N17 Bytes	Descriptive Metadata.

8.2.2.5 Conformance Testing

A Data instance Conforms with CAE-USC V2.3 Data Types Audio Basic Scene Geometry (CAE-ABG) if:

1. The JSON Data validate against the Audio Basic Scene Geometry's JSON Schema.
2. All JSON Data in the Audio Basic Scene Geometry's JSON Schema:
    1. Have the types specified.
    2. Validate against their JSON Schemas.
    3. Conform with their Data Qualifiers if present.

8.2.3 Audio Object

8.2.3.1 Definition

A Data Type including:

1. Audio Data whose rendering is perceived by a hearing device or audible to a human with attached Qualifier.
2. Descriptive Data regarding Sub-Types, Formats and Attributes of the Audio Data (optionally).
3. Space-Time information.

8.2.3.2 Functional Requirements

An Audio Object includes:

1. The ID of a Virtual Space (M-Instance) where it is or will be located.
2. The ID of the Audio Object.
3. The ID(s) of the Parent Object(s) supporting two cases:
    1. The Parent Object has spawned two (or more) Objects.
    2. Two (or more) Parent Objects have merged into one.
4. The Space-Time information of all Parent Objects in the M-Instance.
5. The Space-Time information of the Visual Data in an M-Instance.
6. The Visual Data Qualifier.
7. The Audio Data Annotations, including:
    1. Annotation
    2. Annotation Space-Time
    3. Process Action IDs
8. The Audio Object-specific Data:
    1. Visual Data Qualifier.
    2. Visual Data Annotation.
    3. Visual Data length in Bytes.
    4. Visual Data URI.

8.2.3.3 Syntax

https://schemas.mpai.community/CAE1/V2.3/data/AudioObject.json

8.2.3.4 Semantics

Label	Size	Description
Header	N1 Bytes	Audio Object Header
– Standard-AudioObject	9 Bytes	The characters “CAE-AUO-V”
– Version	N2 Bytes	Major version – 1 or 2 characters
– Dot-separator	1 Byte	The character “.”
– Subversion	N3 Bytes	Minor version – 1 or 2 characters

MinstanceID	N4 Bytes	Identifier of M-Instance.
AudioObjectID	N5 Bytes	Identifier of the Audio Object.
ParentAudioObjects[]	N6 Bytes	Identifier(s) of Parent Audio Objects.
- ParentAudioObjectID	N7 Bytes	ID of a Parent Audio Object
- ParentAudioDataSpaceTime	N8 Bytes	Space Time info of Parent Audio Data
AudioDataSpaceTime	N9 Bytes	Space-Time info of Audio Data.
AudioDataQualifier	N10 Bytes	Audio Data Qualifier.
SpeechDataAnnotations[]	N11 Bytes	Annotations of Speech Data
– Annotation	N12 Bytes	ID of Annotation
– AnnotationSpaceTime	N13 Bytes	Where/when Annotation is attached.
– ProcessActionIDs	N14 Bytes	What is possible to do with the Annotation
AudioDataLength	N15 Bytes	Number of Bytes of Audio Data
AudioDataURI	N16 Bytes	URI of Data of Audio Data
DescrMetadata	N17 Bytes	Descriptive Metadata

8.2.3.5 Conformance Testing

A Data instance Conforms with CAE-USC V2.3 Audio Object (CAE-AUO) if:

1. JSON Data validate against the Audio Object's JSON Schema.
2. All Data in the Audio Object's JSON Schema
    1. Have the specified types.
    2. Validate against their JSON Schemas.
    3. Conform with their Data Qualifiers if present.

8.2.4 Audio Scene Descriptors

8.2.4.1 Definition

A Data Type including an Audio Scene's:

1. Audio Data and their spatial arrangement.
2. Audio Sub-Scenes and their spatial arrangement.

8.2.4.2 Functional Requirements

Audio Scene Descriptors include:

1. The ID of a Virtual Space (M-Instance) where it is or will be located.
2. The ID of the Audio Scene Descriptors.
3. The number of Audio Data in the Audio Scene.
4. The number of Audio Sub-Scenes in the Audio Scene
5. The Space-Time of the Audio Scene.
6. The Audio Data and their Spatial Attitudes.
7. The Audio Sub-Scenes and their Spatial Attitudes.

Compared to Audio Basic Scene Descriptors, Audio Scene Descriptors may additionally include Audio Sub-Scenes.

8.2.4.3 Syntax

https://schemas.mpai.community/CAE1/V2.3/data/AudioSceneDescriptors.json

8.2.4.4 Semantics

Label	Size	Description
Header	N1 Bytes	Audio Scene Descriptors Header
- Standard - AudioSceneDescriptors	9 Bytes	The characters “OSD-ASD-V”
- Version	N2 Bytes	Major version – 1 or 2 characters
- Dot-separator	1 Byte	The character “.”
- Subversion	N3 Bytes	Minor version – 1 or 2 characters
MinstanceID	N4 Bytes	Identifier of M-Instance.
AudioSceneDescriptorsID	N5 Bytes	Identifier of the Audio Scene Descriptors.
AudioSceneSpaceTime	N6 Bytes	Space-Time of Audio Scene.
AudioDataCount	N7 Bytes	# of Audio Data in Audio Scene
AudioDataSet[]	N8 Bytes	Set of Audio Data that do not belong to an Audio Sub-Scene
- AudioData	N9 Bytes	Audio Data in the Scene to an Audio Sub-Scene.
- AudioDataSpaceTime	N10 Bytes	Space-Time information of Audio Data.
AudioSubSceneCount	N11 Bytes	# of Audio Sub-Scenes in Audio Scene
AudioSubScenes[]	N12 Bytes	Set of Audio Sub-Scenes in Audio Scene
- AudioSubSceneData	N13 Bytes	Data of Audio Sub-Scene
- AudioSubSceneSpaceTime	N14 Bytes	Audio Sub-Scene's Space-Time info
DescrMetadata	N15 Bytes	Descriptive Metadata

8.2.4.5 Conformance Testing

A Data instance Conforms with CAE-USC V2.3 Data Types Audio Scene Descriptors (CAE-ASD) if

1. The JSON Data validate against the Audio Scene Descriptors' JSON Schema.
2. The Data in the Audio Scene Descriptors' JSON Schema:
    1. Have the specified types.
    2. Validate against their JSON Schemas
    3. Conform with their Data Qualifiers if present.

8.2.5 Audio Scene Geometry

8.2.5.1 Definition

A Data Type including an Audio Scene's:

1. Spatial arrangement of the Audio Data represented by their Qualifiers.
2. Spatial arrangement of the Audio Sub-Scenes'.

8.2.5.2 Functional Requirements

The Audio Scene Geometry Data Type includes:

1. The ID of a Virtual Space (M-Instance) where it is or will be located.
2. The ID of the Audio Scene Geometry.
3. The number of Audio Objects in the Audio Scene.

4. The number of Audio Scenes in the Audio Scene
5. The Space-Time Attributes of the Audio Scene.
6. The Spatial Attitudes and Audio Data Qualifiers.
7. The Spatial Attitudes of Audio Scenes.

8.2.5.3 Syntax

https://schemas.mpai.community/CAE1/V2.3/data/AudioSceneGeometry.json

8.2.5.4 Semantics

Label	Size	Description
Header	N1 Bytes	Audio Scene Geometry Header
– Standard-AudioSceneGeometry	9 Bytes	The characters "OSD-ASG-V"
– Version	N2 Bytes	Major version – 1 or 2 characters
– Dot-separator	1 Byte	The character "."
– Subversion	N3 Bytes	Minor version – 1 or 2 characters.
MinstanceID	N4 Bytes	Identifier of M-Instance.
AudioSceneGeometryID	N5 Bytes	Identifier of the Audio Scene Geometry.
AudioDataCount	N6 Bytes	Number of Audio Data in Audio Scene.
AudioSceneSpaceTime	N7 Bytes	Space and Time of Audio Scene Geometry.
AudioDataSet[]	N8 Bytes	Set of Audio Data.
– AudioDataSpaceTime	N9 Bytes	Space and Time of Audio Data.
– AudioDataQualifiers	N10 Bytes	Qualifiers of Audio Data.
AudioSubSceneCount	N11 Bytes	Number of Audio Sub-Scenes in Audio Scene.
AudioSubSceneData[]	N12 Bytes	Set of Audio Sub-Scenes.
– AudioSubSceneSpaceTime	N13 Bytes	Space and Time of Audio Scene.
DescrMetadata	N14 Bytes	Descriptive Metadata.

8.2.5.5 Conformance Testing

A Data instance Conforms with CAE-USC V2.3 Audio Scene Geometry (CAE-ASG) if
1. The JSON Data validate against the Audio Scene Geometry's JSON Schema.
2. All the Data in the Audio Scene Geometry's JSON Schema:
    1. Have the specified types.
    2. Validate against their JSON Schemas.
    3. Conform with their Data Qualifiers if present.

8.3 Data Types from MPAI-MMC

8.3.1 Cognitive State

8.3.1.1 Definition

Cognitive State is a Personal Status Factor representing the internal state of an Entity such as "surprised" or "interested".

8.3.1.2 Functional Requirements

Cognitive State can be expressed via several *Modalities*: Text, Speech, Face, and Gestures. (Other Modalities, such as body posture, may be handled in future MPAI Versions.)

Within a given Modality, Cognitive State can be analysed and interpreted via various *Descriptors*. For example, when expressed via Speech, the elements may be expressed through combinations of such features as prosody (pitch, rhythm, and volume variations); separable speech effects (such as degrees of voice tension, breathiness, etc.); and vocal gestures (laughs, sobs, etc.).

Cognitive State is represented by a standard set of labels and associated semantics by two tables:

- A *Label Set Table* containing descriptive labels relevant to the Factor in a three-level format:
  - The CATEGORIES column specifies the relevant categories using nouns (e.g., "ANGER").
  - The GENERAL ADJECTIVAL column gives adjectival labels for general or basic labels within a category (e.g., "angry").
  - The SPECIFIC ADJECTIVAL column gives more specific (sub-categorised) labels in the relevant category (e.g., "furious").
- A *Label Semantics Table* providing the semantics for each label in the GENERAL ADJECTIVAL and SPECIFIC ADJECTIVAL columns of the Label Set Table. For example, for "angry" the semantic gloss is "emotion due to perception of physical or emotional damage or threat."

Table 1 gives the standardised three-level Basic Cognitive State Label Set.

Table 1 – Basic Cognitive State Label Set

COGNITIVE CATEGORIES	GENERAL ADJECTIVAL	SPECIFIC ADJECTIVAL
AROUSAL	aroused/excited/energetic	cheerful playful lethargic sleepy
ATTENTION	attentive	expectant/anticipating thoughtful distracted/absent-minded vigilant hopeful/optimistic
BELIEF	credulous	sceptical
INTEREST	interested	fascinated curious bored
SURPRISE	surprised	astounded startled
UNDERSTANDING	comprehending	uncomprehending bewildered/puzzled

Table 2 provides the semantics for each label in the GENERAL ADJECTIVAL and SPECIFIC ADJECTIVAL columns above.

Table 2 – Basic Cognitive State Semantics Set

ID	Cognitive State	Meaning
1	aroused/excited/energetic	cognitive state of alertness and energy

2	Astounded	high degree of surprised
3	Attentive	cognitive state of paying attention
4	bewildered/puzzled	high degree of incomprehension
5	Bored	not interested
6	Cheerful	energetic combined with and communicating happiness
7	comprehending	cognitive state of successful application of mental models to a situation
8	Credulous	cognitive state of conformance to mental models of a situation
9	Curious	interest due to drive to know or understand
10	distracted/absent-minded	not attentive to present situation due to competing thoughts
11	expectant/anticipating	attentive to (expecting) future event or events
12	Fascinated	high degree of interest
13	Interested	cognitive state of attentiveness due to salience or appeal to emotions or drives
14	Lethargic	not aroused
15	Playful	energetic and communicating willingness to play
16	Sceptical	not credulous
17	Sleepy	not aroused due to need for sleep
18	Surprised	cognitive state due to violation of expectation
19	Startled	surprised by a sudden event or perception
20	Surprised	cognitive state due to violation of expectation
21	thoughtful	attentive to thoughts
22	uncomprehending	not comprehending

These sets have been compiled in the interests of basic cooperation and coordination among AIM submitters and vendors complemented by a procedure whereby AIM submitters may propose extended or alternate sets for their purposes.

An Implementer wishing to extend or replace a *Label Set Table* for one of the three Factors is requested to do the following:

1. Create a new Label Set Table where:
    1. Proposed additions are clearly marked (in case of extension).
    2. b. All the elements of the target Cognitive State and levels (up to 3) are listed (in case of replacement).
2. Create a new Label Semantics Table where the semantics of elements of the Cognitive State is:
    1. Added to the semantics of the existing Cognitive State (in case of extension).
    2. Provided (in case of replacement). The submitted semantics should have a level of detail comparable to the semantics given in the current *Label Semantics Table*.
3. Submit both tables to the MPAI Secretariat.

The appropriate MPAI Development Committee will examine the proposed extension or replacement. Only the adequacy of the proposed new tables in terms of clarity and completeness will be considered. In case the new tables are not clear or complete, a revision of the tables will be requested.

The accepted Cognitive State Set will be identified as proposed by the submitter and reviewed by the appropriate MPAI Committee and posted to the MPAI web site.
The versioning system is based on a name – MPAI for MPAI-generated versions or "organisation name" for the proposing organisation – with a suffix m.n where m indicates the version and n indicated the subversion.

8.3.1.3 Syntax

https://schemas.mpai.community/MMC/V2.3/data/CognitiveState.json

8.3.1.4 Semantics

Label	Size	Description
Header	N1 Bytes	Entity Cognitive State Header
- Standard-EntityCognitiveState	9 Bytes	The characters "MMC-ECS-V"
- Version	N2 Bytes	Major version – 1 or 2 characters
- Dot-separator	1 Byte	The character "."
- Subversion	N3 Bytes	Minor version – 1 or 2 characters
MInstanceID	N4 Bytes	Identifier of M-Instance.
EntityCognitiveStateID	N5 Bytes	Identifier of CogState.
EntityCognitiveStateSpaceTime	N7 Bytes	Space-Time info of CogState.
EntityCognitiveStateData	N8 Bytes	Data associated to CogState.
- FusedCogState	N9 Bytes	Integrated CogState Value.
- TextCogState	N10 Bytes	Text CogState Value.
- SpeechCogState	N11 Bytes	Speech CogState Value.
- FaceCogState	N12 Bytes	Face CogState Value.
- GestureCogState	N13 Bytes	Gesture CogState Value.
DescrMetadata	N14 Bytes	Descriptive Metadata

8.3.1.5 Conformance Testing

A Data instance Conforms with MPAI-MMC V2.3 Entity Cognitive State (MMC-ECS) if:
1. The Data validates against the Entity Cognitive State 's JSON Schema.
2. All Data in the Entity Cognitive State 's JSON Schema
    1. Have the specified type
    2. Validate against their JSON Schemas
    3. Conform with their Data Qualifiers if present.

8.3.2 Emotion

8.3.2.1 Definition

Emotion is a Factor of Personal Status representing the internal state of an Entity such as that results from its interaction with the Context, such as "Angry", "Sad", "Determined".

8.3.2.2 Functional Requirements

Emotion can be expressed via several *Modalities*: Text, Speech, Face, and Gestures. (Other Modalities, such as body posture, may be handled in future MPAI Versions.)

Within a given Modality, Emotion can be analysed and interpreted via various *Descriptors*. For example, when expressed via Speech, the elements may be expressed through combinations of such features as prosody (pitch, rhythm, and volume variations); separable speech effects (such as degrees of voice tension, breathiness, etc.); and vocal gestures (laughs, sobs, etc.).

Emotion is represented by a standard set of labels and associated semantics by two tables:

- A *Label Set Table* containing descriptive labels relevant to the Factor in a three-level format:
  - The CATEGORIES column specifies the relevant categories using nouns (e.g., "ANGER").
  - The GENERAL ADJECTIVAL column gives adjectival labels for general or basic labels within a category (e.g., "angry").
  - The SPECIFIC ADJECTIVAL column gives more specific (sub-categorised) labels in the relevant category (e.g., "furious").
- A *Label Semantics Table* providing the semantics for each label in the GENERAL ADJECTIVAL and SPECIFIC ADJECTIVAL columns of the Label Set Table. For example, for "angry" the semantic gloss is "emotion due to perception of physical or emotional damage or threat."

Table 1 gives the standardised three-level Basic Emotion Set partly based on Paul Eckman [19].

Table 1 – Basic Emotion Label Set

EMOTION CATEGORIES	GENERAL ADJECTIVAL	SPECIFIC ADJECTIVAL
ANGER	angry	furious irritated frustrated
CALMNESS	calm	peaceful/serene resigned
DISGUST	disgusted	repulsed
FEAR	fearful/scared	terrified anxious/uneasy
HAPPINESS	happy	joyful content delighted amused
HURT	hurt jealous	insulted/offended resentful/disgruntled bitter
PRIDE/SHAME	proud ashamed	guilty/remorseful/sorry embarrassed
RETROSPECTION	nostalgic	homesick
SADNESS	sad	lonely grief-stricken depressed/gloomy disappointed

Table 2 provides the semantics for each label in the GENERAL ADJECTIVAL and SPECIFIC ADJECTIVAL columns above.

Table 2 – Basic Emotion Semantics Set

ID	Emotion	Meaning
1	Amused	positive emotion combined with interest (cognitive state)
2	Angry	emotion due to perception of physical or emotional damage or threat
3	Anxious/uneasy	low or medium degree of fear, often continuing rather than instant
4	Ashamed	emotion due to awareness of violating social or moral norms
5	Bitter	persistently angry due to disappointment or perception of hurt or injury
6	Calm	relatively lacking emotion
7	Content	medium or low degree of happiness, continuing rather than instant
8	Delighted	high degree of happiness, often combined with surprise
9	Depressed/ Gloomy	high degree of sadness, continuing rather than instant, combined with lethargy (see AROUSAL)
10	Disappointed	sadness due to failure of desired outcome
11	Disgusted	emotion due to urge to avoid, often due to unpleasant perception or disapproval
12	Embarrassed	shame due to consciousness of violation of social conventions
13	Fearful/scared	emotion due to anticipation of physical or emotional pain or other undesired event or events
14	Frustrated	angry due to failure of desired outcome
15	Furious	high degree of angry
16	Grief-stricken	sadness due to loss of an important social contact
17	Happy	positive emotion, often continuing rather than instant
18	Homesick	sad due to absence from home
19	Hurt	emotion due to perception that others have caused social pain or embarrassment
20	Insulted/offended	emotion due to perception that one has been improperly treated socially
21	Irritated	low or medium degree of angry
22	Jealous	emotion due to perception that others are more fortunate or successful
23	Joyful	high degree of happiness, often due to a specific event
24	Repulsed	high degree of disgusted
25	Lonely	sad due to insufficient social contact
26	Mortified	high degree of embarrassment
27	Nostalgic	emotion associated with pleasant memories, usually of long before
28	Peaceful/serene	calm combined with low degree of happiness
29	Proud	emotion due to perception of positive social standing
30	Resentful/disgruntled	emotion due to perception that one has been improperly treated
31	Resigned	calm due to acceptance of failure of desired outcome, often combined with low degree of sadness

32	Sad	negative emotion, often continuing rather than instant, often associated with a specific event
33	Terrified	high degree of fear

These sets have been compiled in the interests of basic cooperation and coordination among AIM submitters and vendors complemented by a procedure whereby AIM submitters may propose extended or alternate sets for their purposes.
An Implementer wishing to extend or replace a *Label Set Table* for Emotion is requested to do the following:

1. Create a new Label Set Table where:
    1. Proposed additions are clearly marked (in case of extension).
    2. b. All the elements of the Emotion and levels (up to 3) are listed (in case of replacement).
2. Create a new Label Semantics Table where the semantics of elements of the Emotion is:
    1. Added to the semantics of the existing Emotion (in case of extension).
    2. Provided (in case of replacement). The submitted semantics should have a level of detail comparable to the semantics given in the current *Label Semantics Table*.
3. Submit both tables to the MPAI Secretariat.

The appropriate MPAI Development Committee will examine the proposed extension or replacement. Only the adequacy of the proposed new tables in terms of clarity and completeness will be considered. In case the new tables are not clear or complete, a revision of the tables will be requested.
The accepted Emotion Set will be identified as proposed by the submitter and reviewed by the appropriate MPAI Committee and posted to the MPAI web site.
The versioning system is based on a name – MPAI for MPAI-generated versions or "organisation name" for the proposing organisation – with a suffix m.n where m indicates the version and n indicated the subversion.

8.3.2.3 Syntax

https://schemas.mpai.community/MMC/V2.3/data/Emotion.json

8.3.2.4 Semantics

Label	Size	Description
Header	N1 Bytes	Entity Emotion Header
- Standard-EntityEmotion	9 Bytes	The characters "MMC-EEM-V"
- Version	N2 Bytes	Major version – 1 or 2 characters
- Dot-separator	1 Byte	The character "."
- Subversion	N3 Bytes	Minor version – 1 or 2 characters
MInstanceID	N4 Bytes	Identifier of M-Instance.
EntityEmotionID	N5 Bytes	Identifier of the Emotion.
EntityEmotionSpaceTime	N7 Bytes	Space-Time info of Emotion
EntityEmotionData	N8 Bytes	Data associated to Emotion.
- FusedEmotion	N9 Bytes	Integrated Emotion Value.
- TextEmotion	N10 Bytes	Text Emotion Value.

- SpeechEmotion	N11 Bytes	Speech Emotion Value.
- FaceEmotion	N12 Bytes	Face Emotion Value.
- GestureCogState	N13 Bytes	Gesture Emotion Value.
DescrMetadata	N14 Bytes	Descriptive Metadata

8.3.2.5 Conformance Testing

A Data instance Conforms with MPAI-MMC V2.3 Entity Emotion (MMC-EEM) if:

1. The Data validates against the Entity Emotion 's JSON Schema.
2. All Data in the Entity Emotion 's JSON Schema
    1. Have the specified type
    2. Validate against their JSON Schemas
    3. Conform with their Data Qualifiers if present.

8.3.3 Intention

8.3.3.1 Definition

Data Type expressing the result of analysis of the goal of a question.

8.3.3.2 Functional Requirements

Intention provides abstracts of Intention of User Question using properties: qtopic, qfocus, qLAT, qSAT and qdomain.

8.3.3.3 Syntax

https://schemas.mpai.community/MMC/V2.3/data/Intention.json

8.3.3.4 Semantics

Label	Size	Description
Header	N1 Bytes	The Intention Header
- Standard	9 Bytes	The characters "MMC-INT-V"
- Version	N2 Bytes	Major version – 1 or 2 characters
- Dot-separator	1 Byte	The character "."
- Subversion	N3 Bytes	Minor version – 1 or 2 characters
MInstanceID	N4 Bytes	Identifier of M-Instance.
IntentionID	N5 Bytes	ID of Intention
IntentionData	N6 Bytes	Data included in Intention.
- qtopic	N7 Bytes	Indicates the topic of the question. Question topic is the object or event that the question is about. Ex. of Qtopic is King Lear in "Who is the author of King Lear?".
- qfocus	N8 Bytes	Indicates the focus of the question, which is the part of the question that, if replaced by the answer, makes the question a stand-alone statement. Ex. What, where, who, what policy. Which river, etc. Example: - Question: Who is the president of USA? (The word "Who" is the focus of the question and it will be replaced by "Biden" in the

		Answer.) - Answer: Biden is the president of USA.
- qLAT	N8 Bytes	Indicates the lexical answer type of the question.
- qSAT	N9 Bytes	Indicates the semantic answer type of the question. QSAT corresponds to Named Entity type of the language analysis results.
- qdomain		Indicates the domain of the question such as “science”, “weather”, “history”. Example: Who is the third king of Yi dynasty in Korea? (qdomain: history)
DescrMetadata	N10 Bytes	Descriptive Metadata

8.3.3.5 Conformance Testing

A Data instance Conforms with MPAI-MMC V2.3 Intention (OSD-INT) if:
1. The Data validates against the Intention’s JSON Schema.
2. All Data in the Intention’s JSON Schema have the specified type.

8.3.4 Meaning

8.3.4.1 Definition

A Data Type representing the syntactic and semantic information of an input text. Meaning is synonym of Text Descriptors.

8.3.4.2 Functional Requirements

Meaning is used to extract information from text to help the Entity Dialogue Processing AIM to produce a response.

8.3.4.3 Syntax

https://schemas.mpai.community/MMC/V2.3/data/Meaning.json

8.3.4.4 Semantics

Label	Size	Description
Header	N1 Bytes	Meaning Header
- Standard-Meaning	9 Bytes	The characters “MMC-TXD-V”
- Version	N2 Bytes	Major version – 1 or 2 characters
- Dot-separator	1 Byte	The character “.”
- Subversion	N3 Bytes	Minor version – 1 or 2 characters
MInstanceID	N4 Bytes	Identifier of M-Instance.
MeaningID	N5 Bytes	Identifier of Meaning.
Meaning	N6 Bytes	Data set of Meaning
- POS_tagging	N7 Bytes	Results of POS (Part of Speech, e.g., noun, verb, etc.) tagging including information on the question’s POS tagging set and tagged results.
- NE_tagging	N8 Bytes	Results of NE (Named Entity e.g., Person, Organisation, Fruit, etc.) tagging results including information on the question’s tagging set and tagged results.

- Dependency_tagging	N9 Bytes	Results of dependency (structure of the sentence, e.g., subject, object, head of relation, etc.) tagging including information on the question's dependency tagging set and tagged results.
- SRL_tagging	N10 Bytes	Results of SRL (Semantic Role Labelling) tagging results including information on the question's SRL tagging set and tagged results. SRL indicates the semantic structure of the sentence such as agent, location, patient role, etc.
DescrMetadata	N11 Bytes	Descriptive Metadata

8.3.4.5 Conformance Testing

A Data instance Conforms with MPAI-MMC V2.3 Meaning (MMC-MEA) if:
1. The Data validates against the Meaning's JSON Schema.
2. All Data in the Meaning's JSON Schema have the specified type.

8.3.5 Personal Status

8.3.5.1 Definition

A Data Type representing the information internal to an Entity that characterises their behaviour.

8.3.5.2 Functional Requirements

Personal Status is a Data Type composed of three *Factors*:
1. *Emotion* (such as "angry" or "sad").
2. *Cognitive State* (such as "surprised" or "interested").
3. *Social Attitude* (such as "polite" or "arrogant").

Factors are expressed by *Modalities*: Text, Speech, Face, and Gestures. (Other Modalities, such as body posture, may be handled in future MPAI Versions.)

Within a given Modality, the Factors can be analysed and interpreted via various *Descriptors*. For example, when expressed via Speech, the elements may be expressed through combinations of such features as prosody (pitch, rhythm, and volume variations); separable speech effects (such as degrees of voice tension, breathiness, etc.); and vocal gestures (laughs, sobs, etc.).

Each of Emotion, Cognitive State, and Social Attitude Factors is represented by a standard set of labels and associated semantics. For each of these Factors, two tables are provided:

- A *Label Set Table* containing descriptive labels relevant to the Factor in a three-level format:
  - The CATEGORIES column specifies the relevant categories using nouns (e.g., "ANGER").
  - The GENERAL ADJECTIVAL column gives adjectival labels for general or basic labels within a category (e.g., "angry").
  - The SPECIFIC ADJECTIVAL column gives more specific (sub-categorised) labels in the relevant category (e.g., "furious").
- A *Label Semantics Table* providing the semantics for each label in the GENERAL ADJECTIVAL and SPECIFIC ADJECTIVAL columns of the Label Set Table. For example, for "angry" the semantic gloss is "emotion due to perception of physical or emotional damage or threat."

These sets have been compiled in the interests of basic cooperation and coordination among AIM submitters and vendors complemented by a procedure whereby AIM submitters may propose extended or alternate sets for their purposes.

An Implementer wishing to extend or replace a *Label Set Table* for one of the three Factors is requested to do the following:

1. Create a new Label Set Table where:
    1. Proposed additions are clearly marked (in case of extension).
    2. b. All the elements of the target Factor and levels (up to 3) are listed (in case of replacement).
2. Create a new Label Semantics Table where the semantics of elements of the target Factor is:
    1. Added to the semantics of the existing target Factor (in case of extension).
    2. Provided (in case of replacement).
       The submitted semantics should have a level of detail comparable to the semantics given in the current *Label Semantics Table*.
3. Submit both tables to the MPAI Secretariat (secretariat@mpai.community).

The appropriate MPAI Development Committee will examine the proposed extension or replacement. Only the adequacy of the proposed new tables in terms of clarity and completeness will be considered. In case the new tables are not clear or complete, a revision of the tables will be requested.

The accepted External Factor Set will be identified as proposed by the submitter and reviewed by the appropriate MPAI Committee and posted to the MPAI web site.

The versioning system is based on a name – MPAI for MPAI-generated versions or "organisation name" for the proposing organisation – with a suffix m.n where m indicates the version and n indicated the subversion.

8.3.5.3 Syntax

https://schemas.mpai.community/MMC/V2.3/data/PersonalStatus.json

8.3.5.4 Semantics

Label	Size	Description
Header	N1 Bytes	Personal Status Header
- Standard-PersonalStatus	9 Bytes	The characters "MMC-EPS-V"
- Version	N2 Bytes	Major version – 1 or 2 characters
- Dot-separator	1 Byte	The character "."
- Subversion	N3 Bytes	Minor version – 1 or 2 characters
MInstanceID	N4 Bytes	Identifier of M-Instance.
PersonalStatusID	N5 Bytes	Identifier of Meaning.
PersonalStatusSpaceTime	N6 Bytes	Space-Time info of PersonalStatus
PersonalStatus	N7 Bytes	Personal Status
- CognitiveState	N8 Bytes	Cognitive State component of Personal Status
- Emotion	N9 Bytes	Emotion component of Personal Status
- SocialAttitude	N10 Bytes	Social Attitude component of Personal Status
DescrMetadata	N11 Bytes	Descriptive Metadata

8.3.5.5 Conformance Testing

A Data instance Conforms with MPAI-MMC V2.3 Personal Status (MMC-EPS) if:

1. The Data validates against the Personal Status's JSON Schema.
2. All Data in the Personal Status's JSON Schema
    1. Have the specified type.
    2. Validate against their JSON Schemas.
    3. Conform with their Data Qualifiers if present.

8.3.6 Social Attitude

8.3.6.1 Definition

Social Attitude is a Personal Status Factor representing the internal state of an Entity related to the way it intends to position itself vis-à-vis the Context, e.g., "Respectful", "Confrontational", "Soothing".

8.3.6.2 Functional Requirements

Social Attitude can be expressed via several *Modalities*: Text, Speech, Face, and Gestures. (Other Modalities, such as body posture, may be handled in future MPAI Versions.)
Within a given Modality, Social Attitude can be analysed and interpreted via various *Descriptors*. For example, when expressed via Speech, the elements may be expressed through combinations of such features as prosody (pitch, rhythm, and volume variations); separable speech effects (such as degrees of voice tension, breathiness, etc.); and vocal gestures (laughs, sobs, etc.).
Social Attitude is represented by a standard set of labels and associated semantics by two tables:

- A *Label Set Table* containing descriptive labels relevant to the Social Attitude in a three-level format:
    - The CATEGORIES column specifies the relevant categories using nouns (e.g., "ANGER").
    - The GENERAL ADJECTIVAL column gives adjectival labels for general or basic labels within a category (e.g., "angry").
    - The SPECIFIC ADJECTIVAL column gives more specific (sub-categorised) labels in the relevant category (e.g., "furious").
- A *Label Semantics Table* providing the semantics for each label in the GENERAL ADJECTIVAL and SPECIFIC ADJECTIVAL columns of the Label Set Table. For example, for "angry" the semantic gloss is "emotion due to perception of physical or emotional damage or threat."

These sets have been compiled in the interests of basic cooperation and coordination among AIM submitters and vendors complemented by a procedure whereby AIM submitters may propose extended or alternate sets for their purposes.
An Implementer wishing to extend or replace a *Label Set Table* for Social Attitude is requested to do the following:

1. Create a new Label Set Table where:
    1. Proposed additions are clearly marked (in case of extension).
    2. b. All the elements of the target Social Attitude and levels (up to 3) are listed (in case of replacement).
2. Create a new Label Semantics Table where the semantics of elements of the Social Attitude is:
    1. Added to the semantics of the existing Social Attitude (in case of extension).
    2. Provided (in case of replacement). The submitted semantics should have a level of detail comparable to the semantics given in the current *Label Semantics Table*.
3. Submit both tables to the MPAI Secretariat.

Table 1 gives the standardised three-level Basic Social Attitude Set.

Table 1 – Basic Social Attitude Label Set

SOCIAL ATTITUDE CATEGORIES	GENERAL ADJECTIVAL	SPECIFIC ADJECTIVAL

ACCEPTANCE	accepting exclusive/cliquish	welcoming/inviting friendly unfriendly/hostile
AGREEMENT, DISAGREEMENT	like-minded argumentative/disputatious	sarcastic
AGGRESSION	aggressive peaceful submissive	combative/belligerent passive-aggressive mocking
APPROVAL, DISAPPROVAL	admiring/approving disapproving indifferent	awed contemptuous
ACTIVITY, PASSIVITY	assertive passive	controlling permissive/lenient
COOPERATION	cooperative/agreeable uncooperative	flexible subversive/undermining uncommunicative stubborn disagreeable
RESPONSIVENESS	responsive/demonstrative emotional/passionate unresponsive/undemonstrative unemotional/detached	enthusiastic unenthusiastic passionate dispassionate
EMPATHY	empathetic/caring kind uncaring/callous	sympathetic merciful merciless/ruthless self-absorbed selfish/self-serving selfless/altruistic generous
EXPECTATION	optimistic pessimistic	positive sanguine negative/defeatist cynical
EXTROVERSION, INTROVERSION	outgoing/extroverted uninhibited/unreserved	sociable approachable
DEPENDENCE	dependent independent	helpless
MOTIVATION	motivated apathetic/indifferent	inspired excited/stimulated discouraged/dejected dismissive
OPENNESS, TRUST	open honest/sincere reasonable trusting	candid/frank closed/distant dishonest/deceitful responsible/trustworthy/dependable

		irresponsible distrustful
PRAISING, CRITICISM	laudatory critical	congratulatory flattering belittling
RESENTMENT, FORGIVENESS	forgiving unforgiving/vindictive /spiteful/ vengeful	understanding petty
SELF-PROMOTION	boastful modest/humble/ unassuming	
SELF-ESTEEM	conceited/vain self-deprecating/self-effacing	smug
SOCIAL DOMINANCE, CONFIDENCE	arrogant confident submissive	overconfident forward/presumptuous brazen
SEXUALITY	seductive lewd/bawdy/indecent prudish/priggish	suggestive/risqué/naughty
SOCIAL RANK	polite/courteous/respectful rude/disrespectful commanding/domineering pompous/pretentious obedient rebellious/defiant	condescending/patronizing/snobbish pedantic unaffected servile/obsequious

Table 56 provides the semantics for each label in the GENERAL ADJECTIVAL and SPECIFIC ADJECTIVAL columns above.

Table 2 – Basic Social Attitude Semantics Set

ID	Social attitude	Meaning
1	Accepting	attitude communicating willingness to accept into relationship or group
2	Admiring/approving	attitude due to perception that others' actions or results are valuable
3	Aggressive	tending to physically or metaphorically attack
4	Apathetic/indifferent	showing lack of interest
5	Approachable	sociable and not inspiring inhibition
6	Argumentative	tending to argue or dispute
7	Arrogant	emotion communicating social dominance
8	Assertive	taking active role in social situations
9	Awed	approval combined with incomprehension or fear
10	Belittling	criticising by understating victim’s achievements, personal attributes, etc.
11	Boastful	tending to praise or promote self

12	Brazen	high degree of forwardness/presumption
13	Candid/frank	open in linguistic communication
14	Closed/distant	not open
15	Commanding/domineering	tending to assert right to command
16	Combative/belligerent	high degree of aggression, often physical
17	Communicative	evincing willingness to communicate as needed
18	Conceited/vain	evincing undesirable degree of self-esteem
19	Condescending / patronizing / snobbish	disrespectfully asserting superior social status, experience, knowledge, or membership
20	Confident	attitude due to belief in own ability
21	Congratulatory	wishing well related to another's success or good luck
22	Contemptuous	high degree of disapproval and perceived superiority
23	Controlling	undesirably assertive
24	Cool	repressing outward reaction, often to indicate confidence or dominance, especially when confronting aggression, panic, etc.
25	Cooperative/agreeable	communicating willingness to cooperate
26	Critical	attitude expressing disapproval
27	Cynical	habitually negative, reflecting disappointment or disillusionment
28	Dependent	evincing inability to function without aid
29	Discouraged/dejected	unmotivated because goals or rewards were not achieved
30	Disagreeable	not agreeable
31	Disapproving	not approving
32	Dishonest/deceitful/insincere	not honest
33	Dismissive	actively indicating lack of interest or motivation
34	Distrustful	not trusting
35	Emotional/passionate	high degree of responsiveness to emotions
36	Empathetic/caring	interested in or vicariously feeling others' emotions
37	Enthusiastic	high degree of positive response, especially to specific occurrence
38	Excited/stimulated	attitude indicating cognitive and emotional arousal
39	Exclusive/cliquish	not welcoming into a social group
40	Flattering	praising with intent to influence, often insincere
41	Flexible	willing to adjust to changing circumstances or needs
42	Forward/presumptuous	not observing norms related to intimacy or rank
43	Forgiving	tending to forgive improper behaviour
44	Friendly	welcoming or inviting social contact
45	Generous	tending to give to others, materially or otherwise

46	Guilty/remorseful/sorry	regret due to consciousness of hurting or damaging others
47	Helpless	high degree of dependence
48	Honest/sincere	tending to communicate without deception
49	Independent	not dependent
50	Indifferent	neither approving nor disapproving
51	Inhibited/ reserved/ introverted/ withdrawn	unable or unwilling to participate socially
52	Inspired	motivated by some person, event, etc.
53	Irresponsible	not responsible
54	Kind	tending to act as motivated by empathy or sympathy
55	Laudatory	praising
56	Lewd/bawdy/indecent	evoking sexual associations in ways beyond social norms
57	Like-minded	attitude expressing agreement
58	Melodramatic	high or excessive degree of responsiveness or demonstrativeness
59	Merciful	tending to avoid punishing others, often motivated by empathy or sympathy
60	Merciless/ruthless	not merciful
61	Mocking	communicating non-physical aggression, often by imitating a disapproved aspect of the victim
62	Modest/humble/unassuming	not boastful
63	Motivated	communicating goal-directed emotion and cognitive state
64	Negative/defeatist	expressing pessimism, often habitually
65	Obedient	evincing tendency to obey commands
66	Open	tending to communicate without inhibition
67	Optimistic	tending to expect positive events or results
68	Outgoing/ extroverted/ uninhibited/ unreserved	not inhibited
69	Passive	not assertive
70	Passive-aggressive	covertly and non-physically aggressive
71	Peaceful	not aggressive
72	Pedantic	excessively displaying knowledge or academic status
73	Permissive	allowing activity that social norms might restrict
74	Pessimistic	tending to expect negative events or results
75	Petty	unforgiving concerning small matters
76	Polite/courteous/respectful	tending to respect social norms
77	Pompous/pretentious	excessively displaying social rank, often above actual status

78	Positive	expressing optimism, often habitually
79	Prudish/priggish	expressing disapproval of even minor social transgressions, especially related to sex
80	Reasonable	evincing willingness to resolve issues through reasoning
81	Rebellious/defiant	evincing unwillingness to obey
82	Responsible/trustworthy/ dependable	evincing characteristics or behaviour that encourage trust
83	Responsive/demonstrative	tending to outwardly react to emotions and cognitive states, often as prompted by others
84	Rude/disrespectful	not polite or respectful
85	Sanguine	low degree of optimism, often expressed calmly
86	Sarcastic	communicating disagreement by pretending agreement in an obviously insincere manner
87	Seductive	communicating interest in sexual or related contact
88	Self-absorbed	not empathetic due to excessive interest in self
89	Self-deprecating/self-effacing	tending to criticize, or fail to praise or promote, self
90	Selfish/self-serving	not generous due to excessive interest in own benefit
91	Selfless/altruistic	tending to act for others' benefit, sometimes exclusively
92	Servile/obsequious	excessively and demonstrably obedient
93	Shy	low degree of social inhibition
94	Smug	evincing undesirable degree of self-esteem related to perceived triumph
95	Stubborn	unwilling to change one's mind or behaviour
96	Sociable	comfortable in social situations
97	Submissive	tending to submit to social dominance
98	Subversive/undermining	communicating intention to work against a victim's goals
99	Suggestive/risqué/naughty	evoking sexual associations within social norms
100	Supportive	communicating willingness to support as needed
101	Sympathetic	empathetic related to others' hurt or suffering
102	Trusting	tending to trust others
103	Unaffected	not pompous
104	Uncaring/callous	not empathetic or caring
105	Uncommunicative	not communicative
106	Uncooperative	not cooperative
107	Understanding	forgiving due to ability to understand motivations
108	Unemotional/dispassionate/ detached	not emotional, even when emotion is expected
109	Unenthusiastic	not enthusiastic
110	Unfriendly/hostile	not friendly

111	Unresponsive/ undemonstrative	not responsive or demonstrative
112	Welcoming/inviting	high degree of acceptance with emotional warmth

The appropriate MPAI Development Committee will examine the proposed extension or replacement. Only the adequacy of the proposed new tables in terms of clarity and completeness will be considered. In case the new tables are not clear or complete, a revision of the tables will be requested.
The accepted Social Attitude Set will be identified as proposed by the submitter and reviewed by the appropriate MPAI Committee and posted to the MPAI web site.
The versioning system is based on a name – MPAI for MPAI-generated versions or "organisation name" for the proposing organisation – with a suffix m.n where m indicates the version and n indicated the subversion.

8.3.6.3 Syntax

https://schemas.mpai.community/MMC/V2.3/data/SocialAttitude.json

8.3.6.4 Semantics

Label	Size	Description
Header	N1 Bytes	Entity Social Attitude Header
- Standard-SocialAttitude	9 Bytes	The characters "MMC-ESA-V"
- Version	N2 Bytes	Major version – 1 or 2 characters
- Dot-separator	1 Byte	The character "."
- Subversion	N3 Bytes	Minor version – 1 or 2 characters
MInstanceID	N4 Bytes	Identifier of M-Instance.
SocialAttitudeID	N5 Bytes	Identifier of the Social Attitude.
SocialAttitudeSpaceTime	N7 Bytes	Space-Time info of Social Attitude.
SocialAttitudeData	N8 Bytes	Data associated to Social Attitude.
- FusedSocAtt	N9 Bytes	Integrated Social Attitude Value.
- TextSocAtt	N10 Bytes	Text Social Attitude Value.
- SpeechSocAtt	N11 Bytes	Speech Social Attitude Value.
- FaceSocAtt	N12 Bytes	Face Social Attitude Value.
- GestureSocAtt	N13 Bytes	Gesture Social Attitude Value.
DescrMetadata	N14 Bytes	Descriptive Metadata

8.3.6.5 Conformance Testing

A Data instance Conforms with MPAI-MMC V2.3 Entity Social Attitude (MMC-ESA) if:
1. The Data validates against the Entity Social Attitude's JSON Schema.
2. All Data in the Entity Social Attitude's JSON Schema
    1. Have the specified type.
    2. Validate against their JSON Schemas.
    3. Conform with their Data Qualifiers if present.

8.3.7 Speech Descriptors

8.3.7.1 Definition

A Data Type representing characteristic elements extracted from the input speech, specifically Pitch, Intensity, Tempo, Personal Status, and NNSpeechFeatures in a period of time.

8.3.7.2 Functional Requirements

Speech Descriptors may include Neural Network Descriptors.

8.3.7.3 Syntax

https://schemas.mpai.community/MMC/V2.3/data/SpeechDescriptors.json

8.3.7.4 Semantics

Label	Size	Description
Header	N1 Bytes	Speech Descriptors Header
- Standard - SpeechDescriptors	9 Bytes	The characters "MMC-SPD-V"
- Version	N2 Bytes	Major version – 1 or 2 characters
- Dot-separator	1 Byte	The character "."
- Subversion	N3 Byte	Minor version – 1 or 2 characters
MInstanceID	N4 Bytes	ID of the Metaverse Instance.
SpeechDescriptorsID	N5 Bytes	ID of Speech Descriptors.
SpeechDescriptorsData	N7 Bytes	Data associated with Input Text.
NNSpeechFeatures	N8 Bytes	The output vector of a neural-network using Speech as input.
Duration	N9 Bytes	The Time in which the Speech Descriptors are computed.
Pitch	N10 Bytes	Real number measuring the fundamental frequency of Speech in Hz (Hertz).
Intensity	N11 Bytes	Real number measuring the Energy of Speech in dBs (decibel).
Tempo	N12 Byte	Real number measuring the rate at which specified linguistic units (Phonemes, Syllables, or Words) are produced.
Personal Status	N13 Byte	The Speech Personal Status carried by the input speech.

8.3.7.5 Conformance Testing

A Data instance Conforms with MPAI-MMC V2.3 Speech Descriptors (MMC-SPD) if:

1. The Data validates against the Speech Descriptors' JSON Schema.
2. All Data in the Speech Descriptors' JSON Schema
    1. Have the specified type
    2. Validate against their JSON Schemas
    3. Conform with their Data Qualifiers if present.

8.3.8 Speech Object

8.3.8.1 Definition

A Data Type composed of:
1. Content represented as Data whose rendering has vocal attributes.
2. Speech Qualifier.
3. Space-Time information.

8.3.8.2 Functional Requirements

A Speech Object includes:
1. The ID of a Virtual Space (M-Instance) where it is or is intended to be located.
2. The ID of the Speech Object.
3. The ID(s) of Parent Object(s) supporting two cases:
    1. The Parent Object has spawned two (or more) Objects. That is, two Objects are now distinguished where only one was before.
    2. Two (or more) Parent Objects have merged into one.
4. The Space-Time information of all Parent Objects in the M-Instance.
5. The Speech Object Space-Time information.
6. The Speech Data Qualifier.
7. The Speech Data Annotations, including:
    1. Annotation
    2. Annotation Space-Time
    3. Process Action IDs
8. The Speech Data Length and URI:
    1. The length in Bytes of the Speech Data.
    2. The URI of the Speech Data.

8.3.8.3 Syntax

https://schemas.mpai.community/MMC/V2.3/data/SpeechObject.json

8.3.8.4 Semantics

Label	Size	Description
Header	N1 Bytes	Speech Object Data Header
– Standard-SpeechObject	9 Bytes	The characters “MMC-SPO-V”
– Version	N2 Bytes	Major version – 1 or 2 characters
– Dot-separator	1 Byte	The character “.”
– Subversion	N3 Bytes	Minor version – 1 or 2 characters
MInstanceID	N4 Bytes	Identifier of M-Instance.
SpeechObjectID	N5 Bytes	Identifier of the Speech Object.
ParentSpeechObjects[]	N6 Bytes	Identifier(s) of Parent Speech Objects.
- ParentAudioObjectID	N7 Bytes	ID of a Parent Audio Object
- ParentAudioDataSpaceTime	N8 Bytes	Space Time info of Parent Audio Data
ChildSpeechObjects[]	N6 Bytes	Identifier(s) of Child Speech Objects.
- ChildAudioObjectID	N7 Bytes	ID of a ChildAudio Object
- ChildAudioDataSpaceTime	N8 Bytes	Space Time info of ChildAudio Data

SpeechDataSpace-Time	N9 Bytes	Space-Time info of Data Object.
SpeechDataQualifier	N10 Bytes	Speech Data Qualifier.
SpeechDataAnnotations[]	N11 Bytes	Annotations of Speech Data
– Annotation	N12 Bytes	ID of Annotation
– AnnotationSpaceTime	N13 Bytes	Where/when Annotation is attached.
– RightsID	N14 Bytes	What is possible to do with the Annotation
DescrMetadata	N17 Bytes	Descriptive Metadata

8.3.8.5 Conformance Testing

A Data instance Conforms with MPAI-MMC V2.3 Speech Object (MMC-SPO) if:

1. The Data validates against the Speech Object's JSON Schema.
2. All Data in the Speech Object's JSON Schema
    1. Have the specified type
    2. Validate against their JSON Schemas
    3. Conform with their Data Qualifiers if present.

8.3.9 Text Descriptors

8.3.9.1 Definition

A Data Type representing the syntactic and semantic information of a Text.

8.3.9.2 Functional Requirements

Meaning is an extract of the information from text to help an Entity Dialogue Processing AIM to produce a response.

8.3.9.3 Syntax

https://schemas.mpai.community/MMC/V2.3/data/TextDescriptors.json

8.3.9.4 Semantics

Label	Size	Description
Header	N1 Bytes	Text Descriptors Header
- Standard - TextDescriptors	9 Bytes	The characters "MMC-TXD-V"
- Version	N2 Bytes	Major version – 1 or 2 characters
- Dot-separator	1 Byte	The character "."
- Subversion	N3 Bytes	Minor version – 1 or 2 characters
MInstanceID	N4 Bytes	Identifier of M-Instance.
TextDescriptorsID	N5 Bytes	ID of Text Descriptors
TextDescriptors	N6 Bytes	Identifier of the AV Object.
- POS_tagging	N7 Bytes	Results of POS (Part of Speech, e.g., noun, verb, etc.) tagging including information on the question's POS tagging set and tagged results.

- NE_tagging	N8 Bytes	Results of NE (Named Entity e.g., Person, Organisation, Fruit, etc.) tagging results including information on the question's tagging set and tagged results.
- Dependency_tagging	N9 Bytes	Results of dependency (structure of the sentence, e.g., subject, object, head of relation, etc.) tagging including information on the question's dependency tagging set and tagged results.
- SRL_tagging	N10 Bytes	Results of SRL (Semantic Role Labelling) tagging results including information on the question's SRL tagging set and tagged results. SRL indicates the semantic structure of the sentence such as agent, location, patient role, etc.
DesrMetadata	N11 Bytes	Descriptive Metadata

8.3.9.5 Conformance Testing

A Data instance Conforms with MPAI-MMC V2.3 Text Descriptors (MMC-TXD) if:

1. The Data validates against the Text Descriptors' JSON Schema.
2. All Data in the Text Descriptors' JSON Schema
    1. Have the specified type
    2. Validate against their JSON Schemas
    3. Conform with their Data Qualifiers if present.

8.3.10 Text Object

8.3.10.1 Definition

A Data Type composed of

1. Content represented by characters from a character set (Text Data).
2. Text Qualifier.
3. Text Space-Time information.

8.3.10.2 Functional Requirements

A Text Object includes:

1. The ID of a Virtual Space (M-Instance) where it is or it will be located.
2. The ID of the Text Object.
3. The Text Object Space-Time information.
4. The Text Data Qualifier.
5. The Text Data as:
    1. A string, or
    2. A Text Payload represented as
        1. Length in Bytes of the Text Data.
        2. URI of the Text Data.

8.3.10.3 Syntax

https://schemas.mpai.community/MMC/V2.3/data/TextObject.json

8.3.10.4 Semantics

Label	Size	Description
Header	N1 Bytes	Text Object Data Header
– Standard-Visual Object	9 Bytes	The characters "MMC-TXO-V"

– Version	N2 Bytes	Major version – 1 or 2 characters
– Dot-separator	1 Byte	The character "."
– Subversion	N3 Bytes	Minor version – 1 or 2 characters
MInstanceID	N4 Bytes	Identifier of M-Instance.
TextObjectID	N5 Bytes	Identifier of the Text Object.
TextObjectSpace-Time	N6 Nytes	Space-Time info of Text Object.
TextDataQualifier	N7 Bytes	Text Data Qualifier
TextDataPayload	N8 Bytes	Data of Text Object.
– Text as string, or	N9 Bytes	Text represented by a string.
– TextDataPayload	N10 Bytes	The Payload of the Text Object
– TextObjectLength	N11 Bytes	Number of Bytes in Text Payload
– TextObjectDataURI	N12 Bytes	URI of Data of Text Payload
DescrMetadata	N13 Bytes	Descriptive Metadata

8.3.10.5 Conformance Testing

A Data instance Conforms with MPAI-MMC V2.3 Text Object (MMC-TXO) if:

1. The Data validates against the Text Object's JSON Schema.
2. All Data in the Text Object 's JSON Schema
    1. Have the specified type
    2. Validate against their JSON Schemas
    3. Conform with their Data Qualifiers if present.

8.4 Data Types from MPAI-OSD

8.4.1 Audio-Visual Basic Scene Descriptors

8.4.1.1 Definition

A Data Type including the Objects of an Audio-Visual Scene and their arrangement in the Scene.

8.4.1.2 Functional Requirements

Audio-Visual Basic Scene Descriptors includes:

1. The ID of a Virtual Space where the Audio-Visual Basic Scene is or will be located.
2. The ID of the Audio-Visual Basic Scene Descriptors.
3. The number of
    1. Speech Objects in the Audio-Visual Basic Scene.
    2. Audio Objects in the Audio-Visual Basic Scene.
    3. Visual Objects in the Audio-Visual Basic Scene.
    4. Audio-Visual Objects in the Audio-Visual Basic Scene.
4. The Audio-Visual Basic Scene Space-Time info.
5. The Audio Objects including, for each Speech Object:
    1. The Speech Object Space-Time.
    2. The Speech Object.
6. The Audio Objects including, for each Audio Object:
    1. The Audio Object Space-Time.
    2. The Audio Object.

7. The Visual Objects including, for each Visual Object:
    1. The Visual Object Space-Time.
    2. The Visual Object.
8. The Audio-Visual Objects including, for each Audio-Visual Object:
    1. The Audio-Visual Object Space-Time.
    2. The Audio-Visual Object.

8.4.1.3 Syntax

https://schemas.mpai.community/OSD/V1.2/data/AudioVisualBasicSceneDescriptors.json

8.4.1.4 Semantics

Label	Size	Description
Header	N1 Bytes	Audio-Visual Basic Scene Descriptors Header
- Standard-AVScene	9 Bytes	The characters “OSD-BSD-V”
- Version	N2 Bytes	Major version – 1 or 2 characters
- Dot-separator	1 Byte	The character “.”
- Subversion	N3 Bytes	Minor version – 1 or 2 characters
MInstanceID	N4 Bytes	Identifier of M-Instance.
AVBasicSceneDescriptorsID	N5 Bytes	Identifier of the AV Object.
AVBasicSceneSpaceTime	N7 Bytes	Data about AVScene's Space and Time
AudioObjectCount	N6 Bytes	Number of Audio Objects in Scene
AudioObjectsData[]	N8 Bytes	Set of Audio Objects
- AudioObjectID and/or Object	N9 Bytes	Audio Object ID and/or Object
- AudioObjectSpaceTime	N10 Bytes	Space-Time of Audio Object
SpeechObjectCount	N6 Bytes	Number of Speech Objects in Scene
SpeechObjectsData[]	N11 Bytes	Set of Speech Objects
- SpeechObjectID and/or Object	N12 Bytes	Speech Object ID and/or Object
- SpeechObjectSpaceTime	N13 Bytes	Space-Time of Speech Object
VisualObjectCount	N6 Bytes	Number of Visual Objects in Scene
VisualObjectsData[]	N14 Bytes	Set of Visual Objects
- VisualObjectID and/or Object	N15 Bytes	Visual Object ID and/or Object
- VisualObjectSpaceTime	N16 Bytes	Space-Time of Visual Object
AudioVisualObjectCount	N6 Bytes	Number of Audio-Visual Objects in Scene
AudioVisualObjectsData[]	N17 Bytes	Set of Audio-Visual Objects
- AudioVisualObjectID and/or Object	N18 Bytes	Audio-Visual Object ID and/or Object
- AudioObjectSpaceTime	N19 Bytes	Space-Time of Audio-Visual Object
DescrMetadata	N20 Bytes	Descriptive Metadata

8.4.1.5 Conformance Testing

A Data instance Conforms with MPAI-OSD V1.2 Audio-Visual Basic Scene Descriptors (OSD-BSD) if:

1. The Data validates against the Audio-Visual Basic Scene Descriptors' JSON Schema.
2. All Data in the Audio-Visual Basic Scene Descriptors' JSON Schema
    1. Have the specified type
    2. Validate against their JSON Schemas
    3. Conform with their Data Qualifiers if present.

8.4.2 Audio-Visual Basic Scene Geometry

8.4.2.1 Definition

A Data Type including the arrangement of the Objects of an Audio-Visual Scene.

8.4.2.2 Functional Requirements

Audio-Visual Basic Scene Geometry include:

1. The ID of a Virtual Space (M-Instance) where it is or will be located.
2. The ID of the Audio-Visual Scene Geometry.
3. The number of Objects in the Scene.
4. The number of Scenes in the Scene.
5. The Space-Time Attributes of the Scene.
6. The Spatial Attitudes and Qualifiers of
    1. Audio Objects.
    2. Speech Objects.
    3. Visual Objects.
7. The Spatial Attitude of Audio-Visual Objects.

8.4.2.3 Syntax

https://schemas.mpai.community/OSD/V1.2/data/AudioVisualBasicSceneGeometry.json

8.4.2.4 Semantics

Label	Size	Description
Header	N1 Bytes	AV Basic Scene Geometry Header
– Standard-AVBasicSceneGeometry	9 Bytes	The characters "OSD-BSG-V"
– Version	N2 Bytes	Major version – 1 or 2 characters
– Dot-separator	1 Byte	The character "."
– Subversion	N3 Bytes	Minor version – 1 or 2 characters.
MInstanceID	N4 Bytes	Identifier of M-Instance.
AVBasicSceneGeometryID	N5 Bytes	Identifier of the AV Object.
ObjectCount	N6 Bytes	Number of Objects in Scene.
AVBasicSceneSpaceTime	N8 Bytes	Space and Time of AV Basic Scene Geometry.
AVSceneSpeechObjects[]	N9 Bytes	Set of Audio Objects.
– SpeechObjectSpaceTime	N10 Bytes	Space and Time of SpeechObject.
– SpeechObjectQualifiers	N11 Bytes	Qualifier of SpeechObject.
AVSceneAudioObjects[]	N12 Bytes	Set of Audio Objects.

– AudioObjectSpaceTime	N13 Bytes	Space and Time of Audio Object.
– AudioObjectQualifiers	N14 Bytes	Qualifier of Audio Object.
AVSceneVisualObjects[]	N15 Bytes	Set of Visual Objects
– VisualObjectSpaceTime	N16 Bytes	Space and Time of Visual Object.
– VisualObjectQualifiers	N17 Bytes	Qualifier of Visual Object.
AVSceneAVObjects[]	N18 Bytes	Set of AV Objects
– AVObjectSpaceTime	N19 Bytes	ID of Attribute of AV Scene Object
DescrMetadata	N20 Bytes	Descriptive Metadata

8.4.2.5 Conformance Testing

A Data instance Conforms with MPAI-OSD V1.2 Audio-Visual Basic Scene Geometry (OSD-BSG) if:

1. The Data validates against the Audio-Visual Basic Scene Geometry's JSON Schema.
2. All Data in the Audio-Visual Basic Scene Geometry's JSON Schema
    1. Have the specified type
    2. Validate against their JSON Schemas
    3. Conform with their Data Qualifiers if present.

8.4.3 Audio-Visual Event Descriptors

8.4.3.1 Definition

An Item represented by a series of Audio-Visual Scene Descriptors for a certain duration (from a start Time to an end Time).

8.4.3.2 Functional Requirements

Audio-Visual Event Descriptors contains Audio-Visual Scene Descriptors between two Times.

8.4.3.3 Syntax

https://schemas.mpai.community/OSD/V1.2/data/AudioVisualEventDescriptors.json

8.4.3.4 Semantics

Label	Size	Description
Header	N1 Bytes	Audio-Visual Event Descriptors Header
· Standard	9 Bytes	The characters "OSD-AVE-V"
· Version	N2 Bytes	Major version – 1 or 2 characters
· Dot-separator	1 Byte	The character "."
· Subversion	N3 Bytes	Minor version – 1 or 2 characters
MInstanceID	N4 Bytes	Identifier of M-Instance.
AVEventID	N5 Bytes	Identifier of the Event.
AVEventSpaceTime	N6 Bytes	Data about start and end Space-Time.
AVSceneDescriptors[]	N7 Bytes	Collection of AV Scene Descriptors in Start Time and End Time
· AVSceneDescriptors	N8 Bytes	Set of AV Scene Descriptors of IDs.
DescrMetadata	N9 Bytes	Descriptive Metadata

8.4.3.5 Conformance Testing

A Data instance Conforms with Audio-Visual Event Descriptors (OSD-AVE) V1.2 if:

1. The Data validates against the Audio-Visual Event Descriptors' JSON Schema.
2. All Data in the Audio-Visual Event Descriptors' JSON Schema
    1. Have the specified type
    2. Validate against their JSON Schemas
    3. Conform with their Data Qualifiers if present.

8.4.3.6 Definition

Data whose rendering has both Audio and Visual perceptibility attributes.

8.4.3.7 Functional Requirements

Audio-Visual Object includes:

1. The ID of a Virtual Space (M-Instance) where it is or will be located.
2. The Speech-Audio-Visual Objects' Space-Time location.
3. The IDs of the Speech, Audio, and Visual Objects' and their Space-Time information.

8.4.3.8 Syntax

https://schemas.mpai.community/OSD/V1.2/data/AudioVisualObject.json

8.4.3.9 Semantics

Label	Size	Description
Header	N1 Bytes	Audio-Visual Object Header
- Standard-AudioVisualObject	9 Bytes	The characters "OSD-AVO-V"
- Version	N2 Byte	Major version – 1 or 2 Bytes
- Dot-separator	1 Byte	The character "."
- Subversion	N3 Bytes	Minor version – 1 or 2 Bytes
MInstanceID	N4 Bytes	Identifier of M-Instance.
AudioVisualObjectID	N5 Bytes	Identifier of Audio-Visual Object.
AudioVisualObjectSpaceTime	N6 Bytes	Space-Time of Audio-Visual Object
AudioVisualQualifier	N7 Bytes	Qualifier of the Audio-Visual Object
SpeechObjectData	N8 Bytes	Speech Object Data
- SpeechObjectID and/or Speech Object	N9 Bytes	Speech Object ID and/or Object
- SpeechObjectSpaceTime	N10 Bytes	Space-Time of Speech Object
AudioObjectData	N11 Bytes	Audio Object Data
- AudioObjectID and/or Audio Object	N12 Bytes	Audio Object ID and/or Object
- AudioObjectSpaceTime	N13 Bytes	Space-Time of Audio Object
VisualObjectData	N14 Bytes	Visual Object Data
- VisualObjectID and/or Visual Object	N15 Bytes	Visual Object ID and/or Object
- VisualObjectSpaceTime	N16 Bytes	Space-Time of Visual Object
DescrMetadata	N17 Bytes	Descriptive Metadata

8.4.3.10 Conformance Testing

A Data instance Conforms with V1.2 (OSD-AVO) if:

1. The Data validates against the Audio-Visual Object's JSON Schema.
2. All Data in the Audio-Visual Object's JSON Schema
    1. Have the specified type
    2. Validate against their JSON Schemas
    3. Conform with their Data Qualifiers if present.

8.4.4 Audio-Visual Scene Descriptors

8.4.4.1 Definition

A Data Type including the Audio-Visual Scene's Objects and Sub-Scenes and their arrangement in the Scene.

8.4.4.2 Functional Requirements

Audio-Visual Scene Descriptors includes Scenes in addition to Objects.

8.4.4.3 Syntax

https://schemas.mpai.community/OSD/V1.2/data/AudioVisualSceneDescriptors.json

8.4.4.4 Semantics

Label	Size	Description
Header	N1 Bytes	Audio-Visual Scene Descriptors Header
- Standard-AVSceneDescriptors	9 Bytes	The characters "OSD-AVS-V"
- Version	N2 Bytes	Major version – 1 or 2 characters
- Dot-separator	1 Byte	The character "."
- Subversion	N3 Bytes	Minor version – 1 or 2 characters
MinstanceID	N4 Bytes	Identifier of M-Instance.
AVBasicSceneDescriptorsID	N5 Bytes	Identifier of the AV Object.
ObjectCount	N6 Bytes	Number of Objects in Scene
AVSceneSpaceTime	N7 Bytes	Data about Space and Time
AudioObjectsData[]	N8 Bytes	Set of Audio Objects
- AudioObject	N9 Bytes	ID of Audio Object
- AudioObjectSpaceTime	N10 Bytes	Space-Time of Audio Object
- AudioObjectPayload	N11 Bytes	Length in Bytes and URI of Audio Object Payload
SpeechObjectsData[]	N12 Bytes	Set of SpeechObjects
- SpeechObject	N13 Bytes	Speech Object
- SpeechObjectSpaceTime	N14 Bytes	Space-Time of Speech Object
VisualObjectsData[]	N15 Bytes	Set of Visual Objects
- VisualObjectID	N16 Bytes	ID of Visual Object
- VisualObjectSpaceTime	N17 Bytes	Space-Time of Visual Object
- VisualObjectPayload	N18 Bytes	Length in Bytes and URI of Visual Object Payload
AudioVisualObjectsData[]	N19 Bytes	Set of Audio-Visual Objects

- AudioVisualObjectID	N18 Bytes	ID of Audio-Visual Object
- AudioObjectSpaceTime	N19 Bytes	Space-Time of Audio-Visual Object
SubSceneCount	N20 Bytes	Number of Sub-Scenes in Scene
SubSceneData[]	N21 Bytes	Set of Sub-Scenes
- SubSceneID	N22 Bytes	ID of Sub-Scene
- SubSceneSpaceTime	N23 Bytes	Space-Time of Sub-Scenes
- Payload	N24 Bytes	Length in Bytes and URI of Sub-Scene Payload
DescrMetadata	N25 Bytes	Descriptive Metadata

8.4.4.5 Conformance Testing

A Data instance Conforms with Audio-Visual Scene Descriptors (OSD-AVS) V1.2 if:

1. The Data validates against the Audio-Visual Scene Descriptors' JSON Schema.
2. All Data in the Audio-Visual Scene Descriptors' JSON Schema
    1. Have the specified type
    2. Validate against their JSON Schemas
    3. Conform with their Data Qualifiers if present.

8.4.5 Audio-Visual Scene Geometry

8.4.5.1 Definition

A Data Type including the Qualifiers of the Objects of an Audio-Visual Scene and the arrangement of the Objects and Audio-Visual Scenes in the Scene.

8.4.5.2 Functional Requirements

Audio-Visual Scene Geometry may include Scenes in addition to Objects.

8.4.5.3 Syntax

https://schemas.mpai.community/OSD/V1.2/data/AudioVisualSceneGeometry.json

8.4.5.4 Semantics

Label	Size	Description
Header	N1 Bytes	Audio-Visual Scene Geometry Header
- Standard-AVSceneGeometry	9 Bytes	The characters "OSD-AVG-V"
- Version	N2 Bytes	Major version – 1 or 2 characters
- Dot-separator	1 Byte	The character "."
- Subversion	N3 Bytes	Minor version – 1 or 2 characters
MinstanceID	N4 Bytes	Identifier of M-Instance.
AVBasicSceneGeometryID	N5 Bytes	Identifier of the AV Object.
ObjectCount	N6 Bytes	Number of Objects in Scene
AVSceneSpaceTime	N7 Bytes	Data about Space and Time
AudioObjectsData[]	N8 Bytes	Set of Audio Objects
- AudioObjectSpaceTime	N9 Bytes	ID of Space-Time of Audio Object
- AudioObjectQualifier	N10 Bytes	Qualifier of Audio Object

VisualObjectsData[]	N11 Bytes	Set of Visual Objects
- VisualObjectSpaceTime	N12 Bytes	ID of Space-Time of Visual Object
- VisualObjectQualifier	N13 Bytes	Qualifier of Visual Object
AudioVisualObjectsData[]	N14 Bytes	Set of Audio-Visual Objects
- AudioVisualObjectID	N15 Bytes	ID of Audio-Visual Object
- AudioObjectSpaceTime	N16 Bytes	ID of Space-Time of Audio-Visual Object
DescrMetadata	N17 Bytes	Descriptive Metadata

8.4.5.5 Conformance Testing

A Data instance Conforms with Audio-Visual Scene Geometry (OSD-AVG) V1.2 if:

1. The Data validates against the Audio-Visual Scene Geometry's JSON Schema.
2. All Data in the Audio-Visual Scene Geometry's JSON Schema
    1. Have the specified type
    2. Validate against their JSON Schemas
    3. Conform with their Data Qualifiers if present.

8.4.6 Instance Identifier

8.4.6.1 Definition

A Data Type associating a string (Identifier) with an element of a set of entities – Speech, Objects, Visual Objects, User IDs etc. – belonging to some levels in a hierarchical classification (taxonomy).

8.4.6.2 Functional Requirements

Instance Identifier includes:

1. ID of Virtual Space (M-Instance)
2. Instance Label
3. Confidence level of the association between Instance Label and Instance.
4. Taxonomy
5. Confidence level of the association between Taxonomy and the Instance.

8.4.6.3 Syntax

https://schemas.mpai.community/OSD/V1.2/data/InstanceIdentifier.json

8.4.6.4 Semantics

Label	Size	Description
Header	N1 Bytes	Instance Identifier Header
– Standard-InstanceIdentifier	9 Bytes	The characters "OSD-IID-V"
– Version	N2 Bytes	Major version – 1 or 2 characters
– Dot-separator	1 Byte	The character "."
– Subversion	N3 Bytes	Minor version – 1 or 2 characters
MinstanceID	N4 Bytes	Identifier of M-Instance
InstanceID	N5 Bytes	Identifier of Instance.
InstanceSpaceTime	N6 Bytes	Data about Space-Time
InstanceIdentifierData	N7 Bytes	Data set of Instance Identifier.

InstanceLabel	N8 Bytes	Instance identified by Instance Identifier.
LabelConfidenceLevel	N9 Bytes	Confidence of Instance Label and Instance association.
TaxonomyLabel	N10 Bytes	Taxonomy Instance Identifier belongs to.
TaxonomyConfidenceLevel	N11 Bytes	Confidence of Taxonomy Label .
TaxonomyDataLength	N12 Bytes	Number of Bytes
TaxonomyDataURI	N13 Bytes	URI of Taxonomy.
DescrMetadata	N14 Bytes	Descriptive Metadata

8.4.6.5 Conformance Testing

A Data instance Conforms with Instance Identifier (OSD-IID) V1.2 if:

1. The Data validates against the Instance Identifier's JSON Schema.
2. All Data in the Instance Identifier's JSON Schema
    1. Have the specified type
    2. Validate against their JSON Schemas
    3. Conform with their Data Qualifiers if present.

8.4.7 Point of View

8.4.7.1 Definition

Position and Orientation of an Object in a Virtual Environment excluding velocity and acceleration.

8.4.7.2 Functional Requirements

- An Object may have one of the following attributes: Speech, Audio; Visual; 3D Model, Audio-Visual; Haptic; Smell; RADAR; LiDAR; Ultrasound.
- Accuracy is the estimated absolute difference between the measured spatial and angular values of each of CartPosition, SpherPosition, Orientation, and their true value.

8.4.7.3 Syntax

https://schemas.mpai.community/OSD/V1.2/data/PointOfView.json

8.4.7.4 Semantics

Table 1 provides the semantics of the components of Point of View. The following should be noted:

1. Each of Position, Velocity, and Acceleration is provided either in Cartesian (X,Y,Z) or Spherical (r,φ,θ) Coordinates.
2. The Euler angles are indicated by (α,β,γ).

Table 1 – Semantics of Point of View

Label	Size	Description
Header	N1 Bytes	Point of View Header
- Standard-Point of View	9 Bytes	The characters "OSD-OPV-V"
- Version	N2 Bytes	Major version – 1 or 2 characters
- Dot-separator	1 Byte	The character "."
- Subversion	N3 Bytes	Minor version – 1 or 2 characters

Minstance	N4 Bytes	ID id Virtual space Orientation refers tu
PointOfViewID	N5 Bytes	Identifier of Object Point of View.
General	N6 Bytes	Set of general data.
- CoordType	N7 Bytes	One of Cartesian, Spherical, Geodesic, Toroidal.
- ObjectType	N8 Bytes	One of Digital Human, Generic.
- MediaType	N9 Bytes	One of Speech, Audio, Visual, Audio-Visual, Haptic, Smell, RADAR, LiDAR, Ultrasound.
PositionAndOrientation		
- CartPosition (X,Y,Z)	N10 Bytes	Array (in metres)
- CartPositionAccuracy (X,Y,Z)	N11 Bytes	Array Of CartPositionAccuracy
- SpherPosition (r,φ,θ)	N12 Bytes	Array (in metres and degrees)
- SpherPositionAccuracy (r,φ,θ)	N13 Bytes	Array of - SpherPositionAccuracy
- Orient (α,β,γ)	N14 Bytes	Array (in degrees)
- OrientAccuracy (α,β,γ)	N15 Bytes	Array of OrientAccuracy
DescrMetadata	N16 Bytes	Descriptive Metadata

8.4.7.5 Conformance Testing

A Data instance Conforms with MPAI-OSD V1.2 Point of View (OSD-OPV) V2.3 if:

1. The Data validates against the Point of View's JSON Schema.
2. All Data in the Point of View's JSON Schema.
    1. Have the specified type.
    2. Validate against their JSON Schemas.

8.4.8 Selector

8.4.8.1 Definition

A Data Type used to indicate specific operating values of an AIW or AIM.

8.4.8.2 Functional Requirements

Selector informs an AIW/AIM that a communicating Entity uses/requests to use:

1. Specific media – Text, Speech, Visual, or Gesture – as input or output.
2. Specific Language – as input or output.
3. Media or their Descriptors.
4. View an Avatar or a Scene

8.4.8.3 Syntax

https://schemas.mpai.community/OSD/V1.2/data/Selector.json

8.4.8.4 Semantics

Label	Size	Description
Header	N1 Bytes	Selector Header
- Standard-Selector	9 Bytes	The characters "OSD-SEL-V"

- Version	N2 Bytes	Major version – 1 or 2 characters
- Dot-separator	1 Byte	The character "."
- Subversion	N3 Bytes	Minor version – 1 or 2 characters
InputMedia	N4 Bytes	One or more of Text, Speech, Visual, or Gesture.
OutputMedia	N5 Bytes	One or more of Text, Speech, Visual, or Gesture.
InputLanguage	N6 Bytes	One of a list of languages.
OutputLanguage	N6 Bytes	One of a list of languages.
MediaOrDescriptors	N7 Bytes	One of Text, Speech, Face, Body for MMC-TST
SpeechDescriptors	N8 Bytes	One of No, Yes for MMC-PSE
View	N9 Bytes	One of Avatar or Scene
DescrMetadata	N9 Bytes	Descriptive Metadata

8.4.8.5 Conformance Testing

A Data instance Conforms with Selector (OSD-SEL) V1.2 if:

1. The Data validates against the Selector's JSON Schema.
2. All Data in the Selector's JSON Schema have the specified types.

8.4.9 Space Time

8.4.9.1 Definition

Data Type representing Space (i.e., Spatial Attitude) and Time information.

8.4.9.2 Functional Requirements

Space-Time includes Spatial Attitude and Time .

8.4.9.3 Syntax

https://schemas.mpai.community/OSD/V1.2/data/SpaceTime.json

8.4.9.4 Semantics

Label	Size	Description
Header	N1 Bytes	Space-Time Header
- Standard-Object	9 Bytes	The characters "OSD-SPT-V"
- Version	N2 Bytes	Major version – 1 or 2 characters
- Dot-separator	1 Byte	The character "."
- Subversion	N3 Bytes	Minor version – 1 or 2 characters
MInstance	N4 Bytes	Identifier of Virtual Space.
SpaceTimeID	17 Bytes	Identifier of Space-Time.
Space	N6 Bytes	Spatial Attitudes at T_0 and T_1
Time	N7 Bytes	Time interval between T_0 and T_1
DescrMetadata	N8 Bytes	Descriptive Metadata

8.4.9.5 Conformance Testing

A Data instance Conforms with Space-Time (OSD-SPT) V1.2 if:

1. The Data validates against the Space-Time's JSON Schema.
2. All Data in the Space-Time's JSON Schema
    1. Have the specified type.
    2. Validate against their JSON Schemas.
    3. Conform with their Data Qualifiers if present.

8.4.10 Spatial Attitude

8.4.10.1 Definition

An Item representing the Position and Orientation of an Object, and their velocities and accelerations.

8.4.10.2 Functional Requirements

- The As Spatial Attitude are defined as the combination of Position and orientation, the Functional Requirements are defined by Position and Orientation.

8.4.10.3 Syntax

https://schemas.mpai.community/OSD/V1.2/data/SpatialAttitude.json

8.4.10.4 Semantics

Table 1 provides the semantics of the components of the Spatial Attitude.

Table 1 – Semantics of the Spatial Attitude

Label	Size	Description
Header	N1 Bytes	Spatial Attitude Header
- Standard-SpatialAttitude	9 Bytes	The characters "OSD-OSA-V"
- Version	N2 Bytes	Major version – 1 or 2 characters
- Dot-separator	1 Byte	The character "."
- Subversion	N3 Bytes	Minor version – 1 or 2 characters
MinstanceID	N4 Bytes	ID of Virtual Space Objectrefers to.
ObjectSpatialAttitudeID	N5 Bytes	Identifier of Object Spatial Attitude.
General	N6 Bytes	Set of general data
- CoordinateType	N7 Bytes	One of Cartesian, Spherical, Geodesic, Toroidal.
- ObjectType	N8 Bytes	One of Digital Human, Generic.
- MediaType	N9 Bytes	One of Speech, Audio, Visual, Audio-Visual, Haptic, Smell, RADAR, LiDAR, Ultrasound.
Position	N10 Bytes	As specified by Position
Orientation	N11 Bytes	As specified by Orientation
DescrMetadata	N12 Bytes	Descriptive Metadata

8.4.10.5 Conformance Testing

A Data instance Conforms with V1.2 Spatial Attitude V1.2 (OSD-OSA) if:

1. The Data validates against the Spatial Attitude's JSON Schema.
2. All Data in the Spatial Attitude 's JSON Schema have the specified type.

8.4.11 Time

8.4.11.1 Definition

The start time and the end time of a duration.

8.4.11.2 Functional Requirements

Origin of Time can be Absolute (from 1970/01/01) or relative to a user-selected value.

8.4.11.3 Syntax

https://schemas.mpai.community/OSD/V1.2/data/Time.json

8.4.11.4 Semantics

Label	Size	Description
Header	N1 Bytes	Time Header
- Standard-Object	9 Bytes	The characters “OSD-TIM-V”
- Version	N2 Bytes	Major version – 1 or 2 characters
- Dot-separator	1 Byte	The character “.”
- Subversion	N3 Bytes	Minor version – 1 or 2 characters
MInstanceID	N4 Bytes	Identifier of M-Instance
TimeID	N5 Bytes	Identifier of M-Instance.
TimeData	17 Bytes	Data about Time
- TimeType	0 bit	0=Relative: start at 0000/00/00T00:00 1=Absolute: start at 1970/01/01T00:00.
- TimeUnit	1-5	reserved
- Reserved	6-7 bits	00=seconds, 01=milliseconds, 10=microseconds, 11=nanoseconds.
- StartTime	8 Bytes	Start of Time.
- EndTime	8 Bytes	End of Time.
DescrMetadata	N6 Bytes	Descriptive Metadata

8.4.11.5 Conformance Testing

A Data instance Conforms with MPAI-OSD V1.2 (OSD-) if:

1. The Data validates against the Times’s JSON Schema.
2. All Data in the Times’s JSON Schema have the specified type.

8.4.12 Visual Basic Scene Descriptors

8.4.12.1 Definition

A Data Type representing the Objects of a Visual Basic Scene and their arrangement in the Scene.

8.4.12.2 Functional Requirements

Visual Basic Scene Descriptors includes:

1. The ID of a Virtual Space where it is or will be located.
2. The ID of the Visual Scene Descriptors.
3. The number of Visual Objects in the Scene.
4. The Visual Basic Scene Space-Time.

5. The Visual Objects that include, for each Visual Object:
    1. Space-Time values potentially different from their intrinsic Space Times values.
    2. The Visual Object ID or the Visual Object.

8.4.12.3 Syntax

https://schemas.mpai.community/OSD/V1.2/data/VisualBasicSceneDescriptors.json

8.4.12.4 Semantics

Label	Size	Description
Header	N1 Bytes	Visual Basic Scene Descriptors Header
- Standard-VisualBasicSceneDescriptors	9 Bytes	The characters “OSD-VBS-”
- Version	N2 Bytes	Major version – 1 or 2 characters
- Dot-separator	1 Byte	The character “.”
- Subversion	N3 Bytes	Minor version – 1 or 2 characters
MInstanceID	N4 Bytes	Identifier of M-Instance.
VisualBasicSceneDescriptorsID	N5 Bytes	Identifier of the Visual Object.
ObjectCount	N6 Bytes	Number of Objects in Scene
VisualBasicSceneSpaceTime	N7 Bytes	Data about Space-Time of of Visual Scene
VisualBasicSceneVisualObjects[]	N16 Bytes	Set of Visual Objects
- VisualObjectID and/or VisualObject	N17 Bytes	Visual ObjectID or Visual Object
- VisualObjectSpaceTime	N18 Bytes	Space-Time of Visual Object
DescrMetadata	N32 Bytes	Descriptive Metadata

8.4.12.5 Conformance Testing

A Data instance Conforms with Visual Basic Scene Descriptors (OSD-VBS) V1.2 if:
1. The Data validates against the Visual Basic Scene Descriptors’ JSON Schema.
2. All Data in the Visual Basic Scene Descriptors' JSON Schema
    1. Have the specified type.
    2. Validate against their JSON Schemas.
    3. Conform with their Data Qualifiers.

8.4.13 Visual Basic Scene Geometry

8.4.13.1 Definition

A Data Type representing the spatial arrangement of the Visual Objects in a Visual Basic Scene.

8.4.13.2 Functional Requirements

Visual Basic Scene Geometry includes:
1. The ID of a Virtual Space where it is will be located.
2. The ID of the Visual Basic Scene Geometry.
3. The number of Visual Objects in the Scene.
4. The Visual Basic Scene Space-Time info.
5. For each Visual Object:
    1. The Visual Object Space-Time Attributes.
    2. The Visual Object Qualifier.

8.4.13.3 Syntax

https://schemas.mpai.community/OSD/V1.2/data/VisualBasicSceneGeometry.json

8.4.13.4 Semantics

Label	Size	Description
Header	N1 Bytes	Visual Basic Scene Geometry Header
- Standard-VisualBasicSceneGeometry	9 Bytes	The characters “OSD-VBG-V”
- Version	N2 Bytes	Major version – 1 or 2 characters
- Dot-separator	1 Byte	The character “.”
- Subversion	N3 Bytes	Minor version – 1 or 2 characters
MInstanceID	N4 Bytes	Identifier of M-Instance.
VisualBasicSceneGeometryID	N5 Bytes	Identifier of the Visual Basic Scene Geometry.
ObjectCount	N6 Bytes	Number of Objects in Visual Basic Scene
VisualBasicSceneSpaceTime	N7 Bytes	Space and Time info of Visual Basic Scene Geometry
VisualBasicSceneGeometryData[]	N8 Bytes	Set of Visual Basic Scene Geometry Data
- VisualDataSpaceTime	N9 Bytes	Space-Time info of Visual Data
- VisualDataQualifier	N10 Bytes	Visual Data Qualifier
DescrMetadata	N11 Bytes	Descriptive Metadata

8.4.13.5 Conformance Testing

A Data instance Conforms with Visual Basic Scene Geometry (OSD-VBG) V1.2 if:

1. The Data validates against the Visual Basic Scene Geometry’s JSON Schema.
2. All Data in the Visual Basic Scene Geometry’s JSON Schema
    1. Have the specified type
    2. Validate against their JSON Schemas
    3. Conform with their Data Qualifiers.

8.4.14 Visual Object

8.4.14.1 Definition

A Data Type including:

1. Visual Data perceptible by a visual device or visible to a human when rendered.
2. Descriptive Data regarding Sub-Types, Formats and Attributes of the Visual Data (optionally).

8.4.14.2 Functional Requirements

A Visual Object may include:

1. The ID of a Virtual Space (M-Instance) where it is or is intended to be located.
2. The ID of the Visual Object.
3. The ID(s) of Parent Object(s) supporting two cases:
    1. The Parent Object has spawned two (or more) Objects.
    2. Two (or more) Parent Objects have merged into one.
4. The Space-Time information of Parent Objects in an M-Instance.

5. The Space-Time information of the Visual Data in an M-Instance.
6. The Visual Data Qualifier.
7. The Visual Data Annotations, including:
    1. Annotation
    2. Annotation Space-Time
    3. Process Action ID
8. The Visual Object-specific Data:
    1. Visual Data Qualifier.
    2. Visual Data Annotation.
    3. Visual Data length in Bytes.
    4. Visual Data URI.

8.4.14.3 Syntax

https://schemas.mpai.community/OSD/V1.2/data/VisualObject.json

8.4.14.4 Semantics

Label	Size	Description
Header	N1 Bytes	Visual Object Header
– Standard-VisualObject	9 Bytes	The characters “OSD-VIO-V”
– Version	N2 Bytes	Major version – 1 or 2 characters
– Dot-separator	1 Byte	The character “.”
– Subversion	N3 Bytes	Minor version – 1 or 2 characters
MInstanceID	N4 Bytes	Identifier of M-Instance.
VisualObjectID	N5 Bytes	Identifier of the Visual Object.
ParentVisualObjects[]	N6 Bytes	Identifier(s) of Parent Visual Objects.
- ParentVisualObjectID	N7 Bytes	Identifier of a Parent Visual Object.
- ParentVisualDataSpaceTime	N8 Bytes	SpaceTime of a Parent Visual Object.
ChildVisualObjects[]	N9 Bytes	Identifier(s) of Parent Visual Objects.
- ChildVisualObjectID	N10 Bytes	Identifier of a Child Visual Object.
- ChildVisualDataSpaceTime	N11 Bytes	SpaceTime of a Child Visual Object.
VisualDataSpace-Time	N12 Bytes	Space-Time info of Visual Data.
VisualDataQualifier	N13 Bytes	Qualifier of Visual Data.
VisualDataAnnotations[]	N14 Bytes	Annotations of Visual Data
– Annotation	N15 Bytes	ID of Annotation
– AnnotationSpaceTime	N16 Bytes	Where/when Annotation is attached.
– ProcessActionID	N17 Bytes	What is possible to do with the Annotation
DescrMetadata	N18 Bytes	Descriptive Metadata

8.4.14.5 Conformance Testing

A Data instance Conforms with Visual Object (OSD-VIO) V1.2 if:

1. The Data validates against the Visual Object’s JSON Schema.

2. All Data in the Visual Object's JSON Schema
    1. Have the specified type
    2. Validate against their JSON Schemas
    3. Conform with their Data Qualifiers, if present.

8.4.15 Visual Scene Descriptors

8.4.15.1 Definition

A Data Type including the Objects of a Visual Scene, the Visual Sub-Scenes, and their arrangement in the Scene.

8.4.15.2 Functional Requirements

Visual Scene Descriptors include Sub-Scene Descriptors in addition to Objects.

8.4.15.3 Syntax

https://schemas.mpai.community/OSD/V1.2/data/VisualSceneDescriptors.json

8.4.15.4 Semantics

Label	Size	Description
Header	N1 Bytes	Visual Scene Descriptors Header
- Standard-VisualSceneDescriptors	9 Bytes	The characters "OSD-VSD-V"
- Version	N2 Bytes	Major version – 1 or 2 characters
- Dot-separator	1 Byte	The character "."
- Subversion	N3 Bytes	Minor version – 1 or 2 characters
MInstanceID	N4 Bytes	Identifier of M-Instance.
VisualSceneDescriptorsID	N5 Bytes	Identifier of Visual Scene Descriptors.
ObjectCount	N6 Bytes	Number of Visual Objects in Scene.
SubSceneCount	N7 Bytes	Number of Visual Scenes in Scene.
VisualSceneDescriptorsSpaceTime	N8 Bytes	Space and Time of Visual Scene Descriptors.
VisualSceneObjects[]	N9 Bytes	Set of Visual Objects.
- VisualSceneObject	N10 Bytes	Visual Object.
- VisualSceneObjectSpaceTime	N11 Bytes	Space Time of Visual Object.
VisualSceneSubScenes[]	N12 Bytes	Set of Visual Sub-Scenes.
- VisualSceneSubScene	N13 Bytes	Visual Sub-Scene.
- VisualSceneSubSceneSpaceTime	N14 Bytes	Space Time of Visual Sub-Scene.
DescrMetadata	N15 Bytes	Descriptive Metadata

8.4.15.5 Conformance Testing

A Data instance Conforms with Visual Scene Descriptors (OSD-VSD) V1.2 if:

1. The Data validates against the Visual Scene Descriptors' JSON Schema.
2. All Data in the Visual Scene Descriptors' JSON Schema
    1. Have the specified type
    2. Validate against their JSON Schemas
    3. Conform with their Data Qualifiers.

8.4.16 Visual Scene Geometry

8.4.16.1 Definition

An Object with Visual perceptibility attributes.

8.4.16.2 Functional Requirements

Visual Scene Descriptors include Scenes in addition to Objects.

8.4.16.3 Syntax

https://schemas.mpai.community/OSD/V1.2/data/VisualSceneGeometry.json

8.4.16.4 Semantics

Label	Size	Description
Header	N1 Bytes	Visual Scene Geometry Header
- Standard-VisualSceneGeometry	9 Bytes	The characters “OSD-VSG-V”
- Version	N2 Bytes	Major version – 1 or 2 characters
- Dot-separator	1 Byte	The character “.”
- Subversion	N3 Bytes	Minor version – 1 or 2 characters
MInstanceID	N4 Bytes	Identifier of M-Instance.
VisualSceneGeometryID	N5 Bytes	Identifier of Visual Scene Geometry.
ObjectCount	N6 Bytes	Number of Visual Objects in Scene.
SubSceneCount	N7 Bytes	Number of Visual SubScenes in Scene.
VisualSceneGeometrySpaceTime	N8 Bytes	Space and Time of Visual Scene Geometry.
VisualSceneObjects[]	N9 Bytes	Set of Visual Objects.
- VisualSceneObjectQualifiers	N10 Bytes	Qualifiers of Visual Object.
- VisualSceneObjectSpaceTime	N11 Bytes	Space Time of Visual Object.
VisualSceneScenes[]	N9 Bytes	Set of Visual Scenes.
- VisualSceneSceneQualifier	N10 Bytes	Qualifiers of Visual Scene.
- VisualSceneSceneSpaceTime	N11 Bytes	Space Time of Visual Scene.
DescrMetadata	N16 Bytes	Descriptive Metadata

8.4.16.5 Conformance Testing

A Data instance Conforms with Visual Scene Geometry (OSD-VSG) V1.2 if:

1. The Data validates against the Visual Scene Geometry’s JSON Schema.
2. All Data in the Visual Scene Geometry’s JSON Schema
    1. Have the specified type
    2. Validate against their JSON Schemas
    3. Conform with their Data Qualifiers.

8.5 Data Types from MPAI-PAF

8.5.1 Avatar

8.5.1.1 Definition

A Data Type that includes ID, Space-Time, Model, Body Descriptors, and Face Descriptors of an avatar.

8.5.1.2 Functional Requirements

Avatar conveys the following information:

1. The ID of the Virtual Space.
2. The ID of the Avatar.
3. The Time and Spatial Attitude of the Avatar is in the M-Instance.
4. The set of Data characterising a Avatar:
    1. Avatar Model.
    2. Face Descriptors.
    3. Body Descriptors.

8.5.1.3 Syntax

https://schemas.mpai.community/PAF/V1.3/data/Avatar.json

8.5.1.4 Semantics

Label	Size	Description
Header	N1 Bytes	Avatar Header.
– Standard-Avatar	9 Bytes	The characters “PAF-AVT-V”
– Version	N2 Bytes	Major version
– Dot-separator	1 Byte	The character "."
– Subversion	N3 Byte	Minor version
MInstanceID	N4 Bytes	ID of Virtual Space the Avatar belongs to.
AvatarSpaceTime	N5 Bytes	The inherent Space-Time info of the Avatar.
AvatarID	N6 Bytes	Identifier of Avatar.
AvatarData	N7 Bytes	Set of Data related to Avatar
- AvatarModel	N8 Bytes	Model of Avatar.
- BodyDescriptors	N9 Bytes	Avatar Body Descriptors.
- FaceDescriptors	N10 Bytes	Avatar Face Descriptors of Avatar.
DescrMetadata	N17 Bytes	Descriptive Metadata

8.5.1.5 Conformance Testing

A Data instance Conforms with PAF-AVA V1.3 Audio Object (CAE-AVA) if:

1. JSON Data validate against the 3D Model Object’s JSON Schema.
2. All Data in the 3D Model’s JSON Schema
    1. Have the specified type.
    2. Validate against their JSON Schemas.

8.5.2 Body Descriptors

8.5.2.1 Definition

Body Descriptors is a Data Type digitally representing a human or a humanoid.

Gesture Descriptors is a Data Type representing the subset of Body Descriptors selected by an application to convey Gesture information.

8.5.2.2 Functional Requirements

Body Descriptors should enable the representation of the joints of a body.

8.5.2.3 Syntax

Syntax is given by Reference. The Body Descriptors XML Syntax is given by: https://www.web3d.org/x3d/content/examples/X3dResources.html

8.5.2.4 Semantics

The semantics of Body Descriptors is provided by https://www.web3d.org/content/hanim-architecture-v2.

8.5.2.5 Conformance Testing

A Data instance Conforms with Body Descriptors (PAF-BDD) V1.3 if the Data instance validates against the Body Descriptors XML Schema.

8.5.3 Face Descriptors

8.5.3.1 Definition

Face Descriptors is a Data Type representing the features of the Face of an Entity.

8.5.3.2 Functional Requirements

The Face Descriptors represent the effect of the motion of the muscles of a human face.
The Face Descriptors Syntax represents the Actions Units of the Facial Action Coding System (FACS) originally developed by Carl-Herman Hjortsjö, adopted by Paul Ekman and Wallace V. Friesen (1978) and updated by Ekman, Friesen, and Joseph C. Hager (2002).

8.5.3.3 Syntax

https://schemas.mpai.community/PAF/V1.3/data/FaceDescriptors.json

8.5.3.4 Semantics

Label	Size	Description
Header	N1 Bytes	Face Descriptors Header
- Standard-FaceDescriptors	9 Bytes	The characters “OSD-FCD-V”
- Version	N2 Bytes	Major version – 1 or 2 characters

- Dot-separator		1 Byte	The character “.”
- Subversion		N3 Bytes	Minor version – 1 or 2 characters
FaceDescriptorsID		N4 Bytes	Identifier of Face Descriptors.
AU	**Description**	N5 Bytes	Facial muscle generating the Action
1	Inner Brow Raiser	1 Byte	Frontalis, pars medialis
2	Outer Brow Raiser	1 Byte	Frontalis, pars lateralis
4	Brow Lowerer	1 Byte	Corrugator supercilii, Depressor supercilii
5	Upper Lid Raiser	1 Byte	Levator palpebrae superioris
6	Cheek Raiser	1 Byte	Orbicularis oculi, pars orbitalis
7	Lid Tightener	1 Byte	Orbicularis oculi, pars palpebralis
9	Nose Wrinkler	1 Byte	Levator labii superioris alaquae nasi
10	Upper Lip Raiser	1 Byte	Levator labii superioris
11	Nasolabial Deepener	1 Byte	Zygomaticus minor
12	Lip Corner Puller	1 Byte	Zygomaticus major
13	Cheek Puffer	1 Byte	Levator anguli oris (a.k.a. Caninus)
14	Dimpler	1 Byte	Buccinator
15	Lip Corner Depressor	1 Byte	Depressor anguli oris (a.k.a. Triangularis)
16	Lower Lip Depressor	1 Byte	Depressor labii inferioris
17	Chin Raiser	1 Byte	Mentalis
18	Lip Puckerer	1 Byte	Incisivii labii superioris and Incisivii labii inferioris
20	Lip stretcher	1 Byte	Risorius with platysma
22	Lip Funneler	1 Byte	Orbicularis oris
23	Lip Tightener	1 Byte	Orbicularis oris
24	Lip Pressor	1 Byte	Orbicularis oris
25	Lips part	1 Byte	Depressor labii inferioris or relaxation of Mentalis, or Orbicularis oris
26	Jaw Drop	1 Byte	Masseter, relaxed Temporalis and internal Pterygoid
27	Mouth Stretch	1 Byte	Pterygoids, Digastric
28	Lip Suck	1 Byte	Orbicularis oris
41	Lid droop	1 Byte	Relaxation of Levator palpebrae superioris
42	Slit	1 Byte	Orbicularis oculi
43	Eyes Closed	1 Byte	Relaxation of Levator palpebrae superioris; Orbicularis oculi, pars palpebralis
44	Squint	1 Byte	Orbicularis oculi, pars palpebralis
45	Blink	1 Byte	Relaxation of Levator palpebrae superioris; Orbicularis oculi, pars palpebralis

46	Wink	1 Byte	Relaxation of Levator palpebrae superioris; Orbicularis oculi, pars palpebralis
61	Eyes turn left	1 Byte	Lateral rectus, medial rectus
62	Eyes turn right	1 Byte	Lateral rectus, medial rectus
63	Eyes up	1 Byte	Superior rectus, Inferior oblique

8.5.3.5 Conformance Testing

A Data instance Conforms with Face Descriptors (PAF-FCD) V1.3 if t

1. The Data validates against the Face Descriptors' JSON Schema.
2. All Data in the Face Descriptors' JSON Schema
    1. Have the specified type.
    2. Validate against their JSON Schemas.

8.5.3.6 Definition

A Data Type including

1. 3D Model Data representing the surface and relevant features of objects from the real world, or Media Data (Speech, Audio, Visual, 3D Model, LiDAR, RADAR, Ultrasound), or a mixture of the two.
2. Descriptive Data regarding Sub-Types, Formats and Attributes of the 3D Model Data (optionally).

3D Model Data may be rendered to be perceived as signals in the human-visible range (380 to 700 nm).

8.5.3.7 Functional Requirements

A 3D Model Object may include:

1. The ID of a Virtual Space (M-Instance) where it is or is intended to be located.
2. The ID of the 3D Model Object.
3. The 3D Model Data Qualifier.
4. The 3D Model Data Annotations, including:
    1. Annotation
    2. Annotation Space-Time
    3. Process Action ID
5. The 3D Model-specific Data:
    1. 3D Model Data Qualifier.
    2. 3D Model Data Annotation.
    3. 3D Model Data length in Bytes.
    4. 3D Model Data URI.

A 3D Model of a human is called Persona and may:

1. Faithfully reproduce the visual appearance of the human.
2. Have their visual appearance altered, compared to that of the human.
3. Have an unrelated visual appearance.
4. Display a presumptive Personal Status in speech, face, and gesture.
5. Be driven by
    1. The movements of the human.
    2. A Process.

8.5.3.8 Syntax

https://schemas.mpai.community/PAF/V1.3/data/3DModelObject.json

8.5.3.9 Semantics

Label	Size	Description
Header	N1 Bytes	3D Model Object Header
– Standard-3DModelObject	9 Bytes	The characters “OSD-3MO-V”
– Version	N2 Bytes	Major version – 1 or 2 characters
– Dot-separator	1 Byte	The character “.”
– Subversion	N3 Bytes	Minor version – 1 or 2 characters
MInstanceID	N4 Bytes	Identifier of M-Instance.
3DModelObjectID	N5 Bytes	Identifier of the 3D Model Object.
3DModelDataSpace-Time	N6 Bytes	Space-Time info of 3D Model Data.
3DModelDataQualifier	N7 Bytes	Qualifier of 3D Model Data.
3DModelDataAnnotations[]	N8 Bytes	Annotations of 3D Model Data
– Annotation	N9 Bytes	ID of Annotation
– AnnotationSpaceTime	N10 Bytes	Where/when Annotation is attached.
– ProcessActionID	N11 Bytes	What is possible to do with the Annotation
3DModelDataLength	N12 Bytes	Number of Bytes of 3D Model Data
3DModelDataURI	N13 Bytes	URI of Data of 3D Model Data
DescrMetadata	N14 Bytes	Descriptive Metadata

8.5.3.10 Conformance Testing

A Data instance Conforms with 3D Model Object (PAF-3MO) V1.3 if:

1. The Data validates against the 3D Model Object’s JSON Schema.
2. All Data in the 3D Model Object’s JSON Schema
    1. Have the specified type
    2. Validate against their JSON Schemas
    3. Conform with their Data Qualifiers, if present.

8.5.4 Portable Avatar

8.5.4.1 Definition

A Data Type that includes:

1. A set of avatar-related Data: M-Instance ID, Avatar ID, Space-Time, Avatar, Language Selector, Text, Speech Object, Personal Status, and
2. Descriptors of the Audio-Visual Scene where the Avatar is embedded and its Space-Time information.

8.5.4.2 Functional Requirements

Portable Avatar provides the following information:

1. The ID of the Virtual Space
2. The set of Data characterising a speaking avatar.

1. The M-Instance in which the the Avatar is located.
2. The Space-Time of the Avatar.
3. The Language Preference of the Avatar.
4. The Text the Avatar is associated with, or which will be converted into a Speech Object.
5. The Speech Model used to synthesise Text.
6. The Speech Object that the Avatar utters.
7. The Personal Status of the Avatar.
8. The Space-Time information of the Avatar embedded in the Audio-Visual Scene.

8.5.4.3 Syntax

https://schemas.mpai.community/PAF/V1.3/data/PortableAvatar.json

8.5.4.4 Semantics

Label	Size	Description
Header	N1 Bytes	The Header of the Portable Avatar Data.
– Standard-PortableAvatar	9 Bytes	The characters “PAF-PAV-V”
– Version	N2 Bytes	Major version
– Dot-separator	1 Byte	The character "."
– Subversion	N3 Byte	Minor version
MInstanceID	N4 Bytes	The ID of the M-Instance.
PortableAvatarID	N5 Bytes	Identifier of the Portable Avatar.
PortableAvatarData	N6 Bytes	Set of Data related to Avatar.
- Avatar	N7 Bytes	Avatar's Model and Face and Body Descriptors.
- PortableAvatarSpaceTime	N8 Bytes	Space-Time of Avatar instance in AV Scene.
- LanguageSelector	N9 Bytes	Avatar's Language Preference.
- TextObject	N10 Bytes	Text associated with Avatar.
- SpeechObject	N11 Bytes	Set of Data related to Speech Object.
- SpeechModel	N12 Bytes	Neural Network Model for Speech Synthesis.
- PersonalStatus	N13 Bytes	Personal Status of Avatar.
- AudioVisualSceneDescriptors	N14 Bytes	AV Scene Descriptors.
- AudioVisualSceneSpaceTime	N15 Bytes	Space and Time info of AV Scene in M-instance.
DescrMetadata	N16 Bytes	Descriptive Metadata

8.5.4.5 Conformance Testing

A Data instance Conforms with Portable Avatar (PAF-PAV) V1.3 if:
1. JSON Data validate against the Portable Avatar 's JSON Schema.
2. All Data in the Portable Avatar 's JSON Schema
    1. Have the specified type.
    2. Validate against their JSON Schemas.
    3. Conform with their Data Qualifiers if present.

9 Profiles

HMC-CEC uses six groups of capability classes to process a Communication Item:

Receives	Communication Items from a Machine or Audio-Visual Scenes from a real space.
Extracts	Personal Status from the Modalities (Text, Speech, Face, or Gesture) in the Communication Item received.
Understands	The Communication Item from the Modalities and the extracted Personal Status, with or without use of the spatial information embedded in the Communication Item.
Translates	Using the set of Modalities available to the Machine.
Generates	Response.
Renders	The response using available Modalities.

Table 1 defines the Attributes and Sub-Attributes of the HMC-CEC Profiles. The Sub-Attributes are expressed with three characters where the first two representing the medium are followed by O representing Object:

1. The Audio-Visual Scene represent Text (TXO), Speech (SPO), Audio (AUO), Visual (VIO), 3D Model, and Portable Avatar (PAF) Sub-Attributes, respectively.
2. The Personal Status, Understanding, Translation, and Display Response represent Text (TXO), Speech (SPO), Face (FCO), and Gesture (GSO), respectively.

The SPC Sub-Attribute of Understanding represents Spatial Information (SPaCe), i.e., the capability of an HMC-CEC implementation to use also Spatial Information to understand a Communication Item.

Table 1 – Attribute and Sub-Attribute Codes of HMC-CEC.

Attributes	**Codes**	**Sub-Attribute Codes**					
Audio-Visual Scene	AVS	TXO	SPO	AUO	VIO	3MO	PAF
Personal Status	EPS	TXO	SPO	FCO	GSO		
Understanding	UND	TXO	SPO	FCO	GSO		SPC
Translation	TRN	TXO	SPO	FCO	GSO		
Display Response	RES	TXO	SPO	FCO	GSO		

The JSON file provides the formal specification of MPAI-HMC Profiles.}
The Regular Expression can be interpreted by factoring it into its component rules:

```
^(ALL|NUL)([+-](AVS|UND|TRN|EPS|DIS)|
            @AVS#(TXO|SPO|AUO|VIO|3MO
                    PAF(#(AVA|LNG|NNM|EPS|SPC|
                            (TXO|SPO)#([a-z]{3})(<->|->)([a-z]{3})))?)|
           @UND#(TXO|SPO|FCO|GSO|SPC)|
           @TRN#(TXO|SPO|FCO|GSO)#([a-z]{3})(<->|->)([a-z]{3})|
           @EPS#(TXO|PSO|FCO|GSO)|
           @DIS#(TXO|PSO|FCO|GSO))+$
```

An additional rule not specified by the Regular Expression is that ALL be always followed by "-" and NUL be always followed by "+".