



# IEEE Standard Adoption of Moving Picture, Audio and Data Coding by Artificial Intelligence (MPAI) Technical Specification Artificial Intelligence Framework (AIF) 1.1

**STANDARDS**

Developed by the  
Collaborative Activities Governance Board

IEEE SA Industry Affiliated Network (IAN)  
Base specification contributed by MPAI community



**IEEE Std 3301™-2022**

# IEEE Standard Adoption of Moving Picture, Audio and Data Coding by Artificial Intelligence (MPAI) Technical Specification Artificial Intelligence Framework (AIF) 1.1

Developed by the

**Collaborative Activities Governance Board**  
of the  
**IEEE SA Board of Governors**

Approved 3 December 2022

**IEEE SA Standards Board**

**Abstract:** This standard adopts MPAl AI Framework (MPAl-AIF) Technical Specification Version 1.1 as an IEEE Standard. The MPAl-AIF Technical Specification specifies architecture, interfaces, protocols, and Application Programming Interfaces (API) of an AI Framework (AIF), especially designed for the execution of AI-based implementation, but also suitable for mixed AI and traditional data processing workflow.

**Keywords:** AI Framework, AIFs, IEEE 3301™, MPAl-AIF

---

The Institute of Electrical and Electronics Engineers, Inc.  
3 Park Avenue, New York, NY 10016-5997, USA

Copyright © 2023 by The Institute of Electrical and Electronics Engineers, Inc.  
All rights reserved. Published 28 April 2023. Printed in the United States of America.

IEEE is a registered trademark in the U.S. Patent & Trademark Office, owned by The Institute of Electrical and Electronics Engineers, Incorporated.

The MPAl specification, Copyright ©MPAl 2021-2022. All rights reserved.

PDF: ISBN 978-1-5044-9334-5 STD25881  
Print: ISBN 978-1-5044-9335-2 STDPD25881

*IEEE prohibits discrimination, harassment, and bullying.*

For more information, visit <https://www.ieee.org/about/corporate/governance/p9-26.html>.

*No part of this publication may be reproduced in any form, in an electronic retrieval system or otherwise, without the prior written permission of the publisher.*

ii  
Copyright © 2023 IEEE. All rights reserved.

## Important Notices and Disclaimers Concerning IEEE Standards Documents

IEEE Standards documents are made available for use subject to important notices and legal disclaimers. These notices and disclaimers, or a reference to this page (<https://standards.ieee.org/ipr/disclaimers.html>), appear in all standards and may be found under the heading “Important Notices and Disclaimers Concerning IEEE Standards Documents.”

### Notice and Disclaimer of Liability Concerning the Use of IEEE Standards Documents

IEEE Standards documents are developed within IEEE Societies and subcommittees of IEEE Standards Association (IEEE SA) Board of Governors. IEEE develops its standards through an accredited consensus development process, which brings together volunteers representing varied viewpoints and interests to achieve the final product. IEEE Standards are documents developed by volunteers with scientific, academic, and industry-based expertise in technical working groups. Volunteers are not necessarily members of IEEE or IEEE SA and participate without compensation from IEEE. While IEEE administers the process and establishes rules to promote fairness in the consensus development process, IEEE does not independently evaluate, test, or verify the accuracy of any of the information or the soundness of any judgments contained in its standards.

IEEE makes no warranties or representations concerning its standards, and expressly disclaims all warranties, express or implied, concerning this standard, including but not limited to the warranties of merchantability, fitness for a particular purpose and non-infringement. In addition, IEEE does not warrant or represent that the use of the material contained in its standards is free from patent infringement. IEEE standards documents are supplied “AS IS” and “WITH ALL FAULTS.”

Use of an IEEE standard is wholly voluntary. The existence of an IEEE Standard does not imply that there are no other ways to produce, test, measure, purchase, market, or provide other goods and services related to the scope of the IEEE standard. Furthermore, the viewpoint expressed at the time a standard is approved and issued is subject to change brought about through developments in the state of the art and comments received from users of the standard.

In publishing and making its standards available, IEEE is not suggesting or rendering professional or other services for, or on behalf of, any person or entity, nor is IEEE undertaking to perform any duty owed by any other person or entity to another. Any person utilizing any IEEE Standards document, should rely upon his or her own independent judgment in the exercise of reasonable care in any given circumstances or, as appropriate, seek the advice of a competent professional in determining the appropriateness of a given IEEE standard.

IN NO EVENT SHALL IEEE BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO: THE NEED TO PROCURE SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE PUBLICATION, USE OF, OR RELIANCE UPON ANY STANDARD, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE AND REGARDLESS OF WHETHER SUCH DAMAGE WAS FORESEEABLE.

## Translations

The IEEE consensus development process involves the review of documents in English only. In the event that an IEEE standard is translated, only the English version published by IEEE is the approved IEEE standard.

## Official statements

A statement, written or oral, that is not processed in accordance with the IEEE SA Standards Board Operations Manual shall not be considered or inferred to be the official position of IEEE or any of its committees and shall not be considered to be, nor be relied upon as, a formal position of IEEE. At lectures, symposia, seminars, or educational courses, an individual presenting information on IEEE standards shall make it clear that the presenter's views should be considered the personal views of that individual rather than the formal position of IEEE, IEEE SA, the Standards Committee, or the Working Group. Statements made by volunteers may not represent the formal position of their employer(s) or affiliation(s).

## Comments on standards

Comments for revision of IEEE Standards documents are welcome from any interested party, regardless of membership affiliation with IEEE or IEEE SA. However, **IEEE does not provide interpretations, consulting information, or advice pertaining to IEEE Standards documents.**

Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments. Since IEEE standards represent a consensus of concerned interests, it is important that any responses to comments and questions also receive the concurrence of a balance of interests. For this reason, IEEE and the members of its Societies and subcommittees of the IEEE SA Board of Governors are not able to provide an instant response to comments, or questions except in those cases where the matter has previously been addressed. For the same reason, IEEE does not respond to interpretation requests. Any person who would like to participate in evaluating comments or in revisions to an IEEE standard is welcome to join the relevant IEEE working group. You can indicate interest in a working group using the Interests tab in the Manage Profile & Interests area of the [IEEE SA myProject system](#).<sup>1</sup> An IEEE Account is needed to access the application.

Comments on standards should be submitted using the [Contact Us](#) form.<sup>2</sup>

## Laws and regulations

Users of IEEE Standards documents should consult all applicable laws and regulations. Compliance with the provisions of any IEEE Standards document does not constitute compliance to any applicable regulatory requirements. Implementers of the standard are responsible for observing or referring to the applicable regulatory requirements. IEEE does not, by the publication of its standards, intend to urge action that is not in compliance with applicable laws, and these documents may not be construed as doing so.

---

<sup>1</sup> Available at: <https://development.standards.ieee.org/myproject-web/public/view.html#landing>.

<sup>2</sup> Available at: <https://standards.ieee.org/content/ieee-standards/en/about/contact/index.html>.

## Data privacy

Users of IEEE Standards documents should evaluate the standards for considerations of data privacy and data ownership in the context of assessing and using the standards in compliance with applicable laws and regulations.

## Copyrights

IEEE draft and approved standards are copyrighted by IEEE under US and international copyright laws. They are made available by IEEE and are adopted for a wide variety of both public and private uses. These include both use, by reference, in laws and regulations, and use in private self-regulation, standardization, and the promotion of engineering practices and methods. By making these documents available for use and adoption by public authorities and private users, neither IEEE nor its licensors waive any rights in copyright to the documents.

## Photocopies

Subject to payment of the appropriate licensing fees, IEEE will grant users a limited, non-exclusive license to photocopy portions of any individual standard for company or organizational internal use or individual, non-commercial use only. To arrange for payment of licensing fees, please contact Copyright Clearance Center, Customer Service, 222 Rosewood Drive, Danvers, MA 01923 USA; +1 978 750 8400; <https://www.copyright.com/>. Permission to photocopy portions of any individual standard for educational classroom use can also be obtained through the Copyright Clearance Center.

## Updating of IEEE Standards documents

Users of IEEE Standards documents should be aware that these documents may be superseded at any time by the issuance of new editions or may be amended from time to time through the issuance of amendments, corrigenda, or errata. An official IEEE document at any point in time consists of the current edition of the document together with any amendments, corrigenda, or errata then in effect.

Every IEEE standard is subjected to review at least every 10 years. When a document is more than 10 years old and has not undergone a revision process, it is reasonable to conclude that its contents, although still of some value, do not wholly reflect the present state of the art. Users are cautioned to check to determine that they have the latest edition of any IEEE standard.

In order to determine whether a given document is the current edition and whether it has been amended through the issuance of amendments, corrigenda, or errata, visit [IEEE Xplore](#) or [contact IEEE](#).<sup>3</sup> For more information about the IEEE SA or IEEE's standards development process, visit the IEEE SA Website.

## Errata

Errata, if any, for all IEEE standards can be accessed on the [IEEE SA Website](#).<sup>4</sup> Search for standard number and year of approval to access the web page of the published standard. Errata links are located under the Additional Resources Details section. Errata are also available in [IEEE Xplore](#). Users are encouraged to periodically check for errata.

<sup>3</sup> Available at: <https://ieeexplore.ieee.org/browse/standards/collection/ieee>.

<sup>4</sup> Available at: <https://standards.ieee.org/standard/index.html>.

## Patents

IEEE Standards are developed in compliance with the [IEEE SA Patent Policy](#).<sup>5</sup>

Attention is called to the possibility that implementation of this standard may require use of subject matter covered by patent rights. By publication of this standard, no position is taken by the IEEE with respect to the existence or validity of any patent rights in connection therewith. If a patent holder or patent applicant has filed a statement of assurance via an Accepted Letter of Assurance, then the statement is listed on the IEEE SA Website at <https://standards.ieee.org/about/sasb/patcom/patents.html>. Letters of Assurance may indicate whether the Submitter is willing or unwilling to grant licenses under patent rights without compensation or under reasonable rates, with reasonable terms and conditions that are demonstrably free of any unfair discrimination to applicants desiring to obtain such licenses.

Essential Patent Claims may exist for which a Letter of Assurance has not been received. The IEEE is not responsible for identifying Essential Patent Claims for which a license may be required, for conducting inquiries into the legal validity or scope of Patents Claims, or determining whether any licensing terms or conditions provided in connection with submission of a Letter of Assurance, if any, or in any licensing agreements are reasonable or non-discriminatory. Users of this standard are expressly advised that determination of the validity of any patent rights, and the risk of infringement of such rights, is entirely their own responsibility. Further information may be obtained from the IEEE Standards Association.

This IEEE standard is based on a technical specification developed by a consortium. Patent declarations may have been provided to the consortium as a membership requirement, or in compliance with the consortium's intellectual property rights (IPR) policies and procedures. Normative references in this standard may contain material subject to patent claims that are also subject to the consortium's or other organizations' IPR policies and procedures. Additional information about declarations may be available from the consortium's website:

MOVING PICTURE, AUDIO AND DATA CODING BY ARTIFICIAL INTELLIGENCE (MPAI) -  
<https://mpai.community/standards/mpai-aif/>

## IMPORTANT NOTICE

IEEE Standards do not guarantee or ensure safety, security, health, or environmental protection, or ensure against interference with or from other devices or networks. IEEE Standards development activities consider research and information presented to the standards development group in developing any safety recommendations. Other information about safety practices, changes in technology or technology implementation, or impact by peripheral systems also may be pertinent to safety considerations during implementation of the standard. Implementers and users of IEEE Standards documents are responsible for determining and complying with all appropriate safety, security, environmental, health, and interference protection practices and all applicable laws and regulations.

---

<sup>5</sup> Available at: <https://standards.ieee.org/about/sasb/patcom/materials.html>.

## Participants

At the time this draft standard was completed, the Artificial Intelligence Framework Working Group had the following membership:

**Stephen Dukes, *Chair***  
**Andrea Basso, *Vice Chair***

| <i>Organization Represented</i>                                                          | <i>Name of Representative</i> |
|------------------------------------------------------------------------------------------|-------------------------------|
| 0xSenses Corporation.....                                                                | Yu Yuan                       |
| 1stCycle Corporation.....                                                                | Johnny Lin                    |
| Beijing Tsinghua Changgung Hospital.....                                                 | Bin Yang                      |
| Dreamerse.....                                                                           | Stephen Dukes                 |
| Electronics and Telecommunications Research Institute (ETRI).....                        | Miran Choi                    |
| Moving Picture, Audio and Data Coding by Artificial Intelligence (MPAI)/Synesthesia..... | Andrea Basso                  |

The following members of the entity Standards Association balloting group voted on this standard. Balloters may have voted for approval, disapproval, or abstention.

|                      |                         |                         |
|----------------------|-------------------------|-------------------------|
| 0xSenses Corporation | Dreamerse               | PetroChina Planning and |
| 1stCycle Corporation | Haier Group Corporation | Engineering Institute   |
|                      | MPAI Synthesia          |                         |

When the IEEE SA Standards Board approved this standard on 3 December 2022, it had the following membership:

**David J. Law, *Chair***  
**Ted Burse, *Vice Chair***  
**Gary Hoffman, *Past Chair***  
**Konstantinos Karachalios, *Secretary***

|                       |                      |                    |
|-----------------------|----------------------|--------------------|
| Edward A. Addy        | Johnny Daozhuang Lin | Mark Siira         |
| Ramy Ahmed Fathy      | Kevin Lu             | Dorothy V. Stanley |
| J. Travis Griffith    | Daleep C. Mohla      | Lei Wang           |
| Guido R. Hiertz       | Andrew Myles         | F. Keith Waters    |
| Yousef Kimiagar       | Damir Novosel        | Karl Weber         |
| Joseph L. Koepfinger* | Annette D. Reilly    | Sha Wei            |
| Thomas Koshy          | Robby Robson         | Philip B. Winston  |
| John D. Kulick        | Jon Walter Rosdahl   | Daidi Zhong        |

\*Member Emeritus

## Introduction

This introduction is not part of IEEE Std 3301-2022, IEEE Standard Adoption of Moving Picture, Audio and Data Coding by Artificial Intelligence (MPAI) Technical Specification Artificial Intelligence Framework (AIF) 1.1.

This IEEE standard is based on a technical specification developed by a consortium. Patent declarations may be provided to the consortium as a membership requirement, or in compliance with the consortium's intellectual property rights (IPR) policies and procedures. Normative references in this standard may contain material subject to patent claims that are also subject to the consortium's other organizations' IPR policies and procedures. Additional information about declaration may be available from consortium's website.

MOVING PICTURE, AUDIO AND DATA CODING BY ARTIFICIAL INTELLIGENCE (MPAI)—  
<https://mpai.community/standards/mpai-aif/>.



Moving Picture, Audio and Data Coding  
by Artificial Intelligence  
[www.mpai.community](http://www.mpai.community)

# MPAI Technical Specification

## AI Framework MPAI-AIF

V1.1

### 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 Annex 2 – Notices and Disclaimers.

# AI Framework

## Version 1.1

|         |                                                      |    |
|---------|------------------------------------------------------|----|
| 1       | Introduction (Informative).....                      | 4  |
| 2       | Scope (Normative) .....                              | 6  |
| 3       | Terms and definitions (Normative) .....              | 6  |
| 4       | References .....                                     | 7  |
| 4.1     | Normative references.....                            | 7  |
| 4.2     | Informative references .....                         | 8  |
| 5       | Architecture of the AI Framework (Normative) .....   | 8  |
| 5.1     | AI Framework Components .....                        | 8  |
| 5.2     | AI Framework Implementations.....                    | 10 |
| 5.3     | AIMs.....                                            | 10 |
| 5.3.1   | Implementation types .....                           | 10 |
| 5.3.2   | Combination .....                                    | 10 |
| 5.3.3   | Hardware-software compatibility.....                 | 11 |
| 5.3.4   | Actual implementations.....                          | 11 |
| 5.3.4.1 | Hardware.....                                        | 11 |
| 5.3.4.2 | Software .....                                       | 11 |
| 6       | Metadata .....                                       | 11 |
| 6.1     | Communication channels and their data types .....    | 11 |
| 6.1.1   | Type system.....                                     | 12 |
| 6.1.2   | Mapping the type to buffer contents .....            | 13 |
| 6.2     | AIF Metadata .....                                   | 13 |
| 6.3     | AIW/AIM Metadata .....                               | 14 |
| 7       | API .....                                            | 19 |
| 7.1     | General.....                                         | 19 |
| 7.2     | Conventions .....                                    | 19 |
| 7.2.1   | API types .....                                      | 19 |
| 7.2.2   | Return codes.....                                    | 20 |
| 7.2.3   | High-priority Messages .....                         | 20 |
| 7.3     | Store API called by Controller .....                 | 20 |
| 7.3.1   | Get and parse archive .....                          | 21 |
| 7.3.1.1 | MPAI_AIFS_GetAndParseArchive.....                    | 21 |
| 7.4     | Controller API called by User Agent .....            | 21 |
| 7.4.1   | General .....                                        | 21 |
| 7.4.1.1 | MPAI_AIFU_Controller_Initialize.....                 | 21 |
| 7.4.1.2 | MPAI_AIFU_Controller_Destroy .....                   | 21 |
| 7.4.2   | Start/Pause/Resume/Stop Messages to other AIWs ..... | 21 |
| 7.4.2.1 | MPAI_AIFU_AIW_Start.....                             | 21 |
| 7.4.2.2 | MPAI_AIFU_AIW_Pause.....                             | 21 |
| 7.4.2.3 | MPAI_AIFU_AIW_Resume .....                           | 22 |
| 7.4.2.4 | MPAI_AIFU_AIW_Stop.....                              | 22 |
| 7.4.3   | Inquire about state of AIWs and AIMs .....           | 22 |
| 7.4.3.1 | MPAI_AIFU_AIM_GetStatus.....                         | 22 |
| 7.4.4   | Management of Shared and AIM Storage for AIWs.....   | 22 |
| 7.4.4.1 | MPAI_AIFU_SharedStorage_Init .....                   | 22 |
| 7.4.4.2 | MPAI_AIFU_AIMStorage_Init.....                       | 22 |
| 7.4.5   | Communication management.....                        | 22 |

|          |                                                           |    |
|----------|-----------------------------------------------------------|----|
| 7.4.5.1  | MPAI_AIFU_Communication_Event .....                       | 22 |
| 7.4.6    | Resource allocation management .....                      | 22 |
| 7.4.6.1  | MPAI_AIFU_Resource_GetGlobal .....                        | 22 |
| 7.4.6.2  | MPAI_AIFU_Resource_SetGlobal .....                        | 23 |
| 7.4.6.3  | MPAI_AIFU_Resource_GetAIW .....                           | 23 |
| 7.4.6.4  | MPAI_AIFU_Resource_SetAIW .....                           | 23 |
| 7.5      | Controller API called by AIMs .....                       | 23 |
| 7.5.1    | General .....                                             | 23 |
| 7.5.2    | Resource allocation management .....                      | 23 |
| 7.5.2.1  | MPAI_AIFM_Resource_GetGlobal .....                        | 23 |
| 7.5.2.2  | MPAI_AIFM_Resource_SetGlobal .....                        | 23 |
| 7.5.2.3  | MPAI_AIFM_Resource_GetAIW .....                           | 23 |
| 7.5.2.4  | MPAI_AIFM_Resource_SetAIW .....                           | 24 |
| 7.5.3    | Register/deregister AIMs with the Controller .....        | 24 |
| 7.5.3.1  | MPAI_AIFM_AIM_Register_Local .....                        | 24 |
| 7.5.3.2  | MPAI_AIFM_AIM_Register_Remote .....                       | 24 |
| 7.5.3.3  | MPAI_AIFM_AIM_Deregister .....                            | 24 |
| 7.5.4    | Send Start/Pause/Resume/Stop Messages to other AIMs ..... | 24 |
| 7.5.4.1  | MPAI_AIFM_AIM_Start .....                                 | 24 |
| 7.5.4.2  | MPAI_AIFM_AIM_Pause .....                                 | 24 |
| 7.5.4.3  | MPAI_AIFM_AIM_Resume .....                                | 24 |
| 7.5.4.4  | MPAI_AIFM_AIM_Stop .....                                  | 25 |
| 7.5.4.5  | MPAI_AIFM_AIM_EventHandler .....                          | 25 |
| 7.5.5    | Register Connections between AIMs .....                   | 25 |
| 7.5.5.1  | MPAI_AIFM_Channel_Create .....                            | 25 |
| 7.5.5.2  | MPAI_AIFM_Channel_Destroy .....                           | 25 |
| 7.5.6    | Using Ports .....                                         | 25 |
| 7.5.6.1  | MPAI_AIFM_Port_Output_Read .....                          | 25 |
| 7.5.6.2  | MPAI_AIFM_Port_Input_Write .....                          | 25 |
| 7.5.6.3  | MPAI_AIFM_Port_Reset .....                                | 26 |
| 7.5.6.4  | MPAI_AIFM_Port_CountPendingMessages .....                 | 26 |
| 7.5.6.5  | MPAI_AIFM_Port_Probe .....                                | 26 |
| 7.5.6.6  | MPAI_AIFM_Port_Select .....                               | 26 |
| 7.5.7    | Operations on messages .....                              | 26 |
| 7.5.7.1  | MPAI_AIFM_Message_Copy .....                              | 26 |
| 7.5.7.2  | MPAI_AIFM_Message_Delete .....                            | 26 |
| 7.5.7.3  | MPAI_AIFM_Message_GetBuffer .....                         | 26 |
| 7.5.7.4  | MPAI_AIFM_Message_GetBufferLength .....                   | 26 |
| 7.5.7.5  | MPAI_AIFM_Message_Parse .....                             | 27 |
| 7.5.7.6  | MPAI_AIFM_Message_Parse_Get_StructField .....             | 27 |
| 7.5.7.7  | MPAI_AIFM_Message_Parse_Get_VariantType .....             | 27 |
| 7.5.7.8  | MPAI_AIFM_Message_Parse_Get_ArrayLength .....             | 27 |
| 7.5.7.9  | MPAI_AIFM_Message_Parse_Get_ArrayField .....              | 27 |
| 7.5.7.10 | MPAI_AIFM_Message_Parse_Delete .....                      | 27 |
| 7.5.8    | Functions specific to machine learning .....              | 28 |
| 7.5.8.1  | Support for model update .....                            | 28 |
| 7.5.8.2  | Support for model drift .....                             | 28 |
| 7.5.9    | Controller API called by Controller .....                 | 28 |
| 7.5.9.1  | MPAI_AIFM_External_List .....                             | 28 |
| 7.5.9.2  | MPAI_AIFM_External_Output_Read .....                      | 28 |

|         |                                                                       |    |
|---------|-----------------------------------------------------------------------|----|
| 7.5.9.3 | MPAI_AIFM_External_Input_Write .....                                  | 28 |
| 8       | Implementation Guidelines (Informative).....                          | 29 |
| 9       | Examples (Informative).....                                           | 29 |
| 9.1     | AIF Implementations .....                                             | 29 |
| 9.1.1   | Resource-constrained implementation .....                             | 29 |
| 9.1.2   | Non-resource-constrained implementation .....                         | 29 |
| 9.2     | Examples of types.....                                                | 30 |
| 9.3     | Examples of Metadata .....                                            | 30 |
| 9.3.1   | Metadata of Enhanced Audioconference Experience AIF .....             | 30 |
| 9.3.2   | Metadata of Enhanced Audioconference Experience AIW .....             | 31 |
| 9.3.3   | Metadata of CAE-EAE Analysis Transform AIM.....                       | 38 |
| 9.3.4   | Metadata of Enhanced Audioconference Experience AIW .....             | 39 |
| 9.3.5   | Metadata of CAE-EAE Analysis Transform AIM.....                       | 45 |
| 9.3.6   | Metadata of CAE-EAE Sound Field Description AIM.....                  | 46 |
| 9.3.7   | Metadata of CAE-EAE Speech Detection and Separation AIM.....          | 48 |
| 9.3.8   | Metadata of CAE-EAE Noise Cancellation AIM .....                      | 49 |
| 9.3.9   | Metadata of CAE-EAE Synthesis Transform AIM.....                      | 50 |
| 9.3.10  | Metadata of CAE-EAE Packager AIM .....                                | 51 |
| Annex 1 | MPAI-wide terms and definitions.....                                  | 53 |
| Annex 2 | Notices and Disclaimers Concerning MPAI Standards (Informative) ..... | 56 |
| Annex 3 | Patent declarations .....                                             | 58 |

## 1 Introduction (Informative)

In recent years, Artificial Intelligence (AI) and related technologies have been introduced in a broad range of applications, have started affecting the life of millions of people and are expected to do so even more in the future. As digital media standards have positively influenced industry and billions of people, so AI-based data coding standards are expected to have a similar positive impact. Indeed, research has shown that data coding with AI-based technologies is generally *more efficient* than with existing technologies for, e.g., compression and feature-based description.

However, some AI technologies may carry inherent risks, e.g., in terms of bias toward some classes of users. Therefore, the need for standardisation is more important and urgent than ever.

The international, unaffiliated, not-for-profit MPAI – Moving Picture, Audio and Data Coding by Artificial Intelligence Standards Developing Organisation has the mission to develop *AI-enabled data coding standards*. MPAI Application Standards enable the development of AI-based products, applications, and services.

As a rule, MPAI standards include four documents: Technical Specification, Reference Software Specifications, Conformance Testing Specifications, and Performance Assessment Specifications. The last type of Specification includes standard operating procedures to enable users of MPAI Implementations to make informed decision about their applicability based on the notion of Performance, defined as a set of attributes characterising a reliable and trustworthy implementation. In the following, Terms beginning with a capital letter are defined in *Table 1* if they are specific to this Standard and in *Table 2* if they are common to all MPAI Standards.

In general, MPAI Application Standards are defined as aggregations – called AI Workflows (AIW) – of processing elements – called AI Modules (AIM) – executed in an AI Framework (AIF). MPAI defines Interoperability as the ability to replace an AIW or an AIM Implementation with a functionally equivalent Implementation.

MPAI also defines 3 Interoperability Levels of an AIF that executes an AIW. The AIW and its AIMs may have 3 Levels:

*Level 1* – Implementer-specific and satisfying the MPAI-AIF Standard.

*Level 2* – Specified by an MPAI Application Standard.

*Level 3* – Specified by an MPAI Application Standard and certified by a Performance Assessor.

MPAI offers Users access to the promised benefits of AI with a guarantee of increased transparency, trust and reliability as the Interoperability Level of an Implementation moves from 1 to 3. Additional information on Interoperability Levels is provided in reference [12].

*Figure 1* depicts the MPAI-AIF Reference Model under which Implementations of MPAI Application Standards and user-defined MPAI-AIF Conforming applications operate.

MPAI Application Standards normatively specify the Syntax and Semantics of the input and output data and the Function of the AIW and the AIMs, and the Connections between and among the AIMs of an AIW.

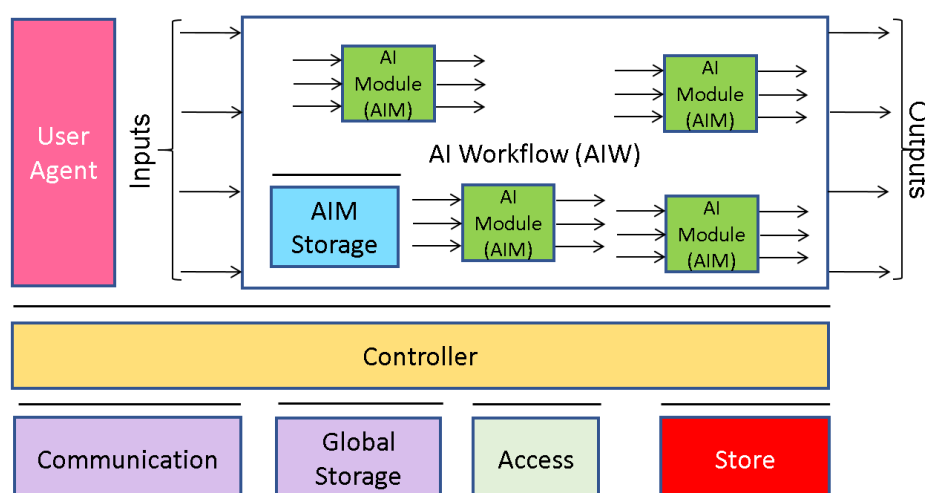


*Figure 1 – The AI Framework (AIF) Reference Model and its Components*

In particular, an AIM is defined by its Function and data, but not by its internal architecture, which may be based on AI or data processing, and implemented in software, hardware or hybrid software and hardware technologies.

MPAI Standards are designed to enable a User to obtain, via standard protocols, an Implementation of an AIW and of the set of corresponding AIMs, and execute it in an AIF Implementation. The Store in *Figure 1* is an entity from which Implementations are downloaded. MPAI Standards assume that the AIF, AIW, and AIM Implementations may have been developed by independent implementers. A necessary condition for this to be possible, is that any AIF, AIW, and AIM implementations be uniquely identified. MPAI has appointed an ImplementerID Registration Authority (IIDRA) to assign unique ImplementerIDs (IID) to Implementers.<sup>1</sup>

A necessary condition to make possible the operations described in the paragraph above is the existence of an ecosystem composed of Conformance Testers, Performance Assessors, and an instance of the IIDRA and of the Store. Reference [12] provides an informative example of such ecosystem.

The chapters and the annexes of this Technical Specification are Normative, unless they are labelled as Informative.

<sup>1</sup> At the time of publication of this standard, the MPAI Store was assigned as the IIDRA. Information about the assignment of ImplmenterIDs is available in Clause 1 of IEEE Std 3300-2022.

## 2 Scope (Normative)

The MPAAI *AI Framework* (MPAAI-AIF) Technical Specification specifies the architecture, interfaces, protocols and Application Programming Interfaces (API) of an AI Framework (AI specially designed for execution of AI-based implementations, but also suitable for mixed AI and traditional data processing workflows.

MPAAI-AIF possesses the following main features:

- Independent of the Operating System.
- Component-based modular architecture with specified interfaces.
- Interfaces encapsulate Components to abstract them from the development environment.
- Interface with the Store enables access to validated Components.
- Component can be Implemented as:
  - Software only, from MCUs to HPC.
  - Hardware only.
  - Hybrid hardware-software.
- Component system features are:
  - Execution in local and distributed Zero-Trust architectures [14].
  - Possibility to interact with other Implementations operating in proximity.
  - Direct support to Machine Learning functionalities.

The current version of the MPAAI-AIF Technical Specification has been developed by the MPAAI AI Framework Development Committee (AIF-DC). Future Versions may revise and/or extend the Scope of the Standard.

## 3 Terms and definitions (Normative)

The Terms used in this standard whose first letter is capital are defined in *Table 1*. The Terms of MPAAI-wide applicability are defined in *Table 1*.

*Table 1 – MPAAI-AIF Terms*

| Term               | Definition                                                                                                                                                                               |
|--------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Access             | Static or slowly changing data that are required by an application such as domain knowledge data, data models, etc.                                                                      |
| AI Framework (AIF) | The environment where AIWs are executed.                                                                                                                                                 |
| AI Module (AIM)    | A processing element receiving AIM-specific Inputs and producing AIM-specific Outputs according to its Function. An AIM may be an aggregation of AIMs. AIMs operate in the Trusted Zone. |
| AI Workflow (AIW)  | A structured aggregation of AIMs implementing a Use Case receiving AIM-specific inputs and producing AIM-specific outputs according to its Function. AIWs operate in the Trusted Zone.   |
| AIF Metadata       | The data set describing the capabilities of an AIF set by the AIF Implementer.                                                                                                           |
| AIM Metadata       | The data set describing the capabilities of an AIM set by the AIM Implementer.                                                                                                           |
| AIM Storage        | A Component to store data of individual AIMs. An AIM may only access its own data. The AIM Storage is part of the Trusted Zone.                                                          |
| AIW Metadata       | The data set describing the capabilities of an AIW set by the AIW Implementer.                                                                                                           |
| Channel            | A physical or logical connection between an output Port of an AIM and                                                                                                                    |

|                 |                                                                                                                                                                                          |
|-----------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                 | an input Port of an AIM. The term “connection” is also used as synonymous. Channels are part of the Trusted Zone.                                                                        |
| Communication   | The infrastructure that implements message passing between AIMs. Communication operates in the Trusted Zone.                                                                             |
| Component       | One of the 9 AIF elements: Access, AI Module, AI Workflow, Communication, Controller, AIM Storage, Shared Storage, Store, and User Agent.                                                |
| Controller      | A Component that manages and controls the AIMs in the AIWs, so that they execute in the correct order and at the time when they are needed. The Controller operates in the Trusted Zone. |
| Data Type       | An instance of the Data Types defined by 6.1.1.                                                                                                                                          |
| Device          | A hardware and/or software entity running at least one instance of an AIF.                                                                                                               |
| Event           | An occurrence acted on by an Implementation.                                                                                                                                             |
| External Port   | An input or output Port simulating communication with an external Controller.                                                                                                            |
| Knowledge Base  | Structured and/or unstructured information made accessible to AIMs via MPAI-specified interfaces.                                                                                        |
| Message         | A sequence of Records.                                                                                                                                                                   |
| MPAI Ontology   | A dynamic collection of terms with a defined semantics managed by MPAI.                                                                                                                  |
| MPAI Server     | A remote machine executing one or more AIMs.                                                                                                                                             |
| Store           | The repository of Implementations.                                                                                                                                                       |
| Port            | A physical or logical communication interface of an AIM.                                                                                                                                 |
| Record          | Data with a specified Format.                                                                                                                                                            |
| Resource policy | The set of conditions under which specific actions may be applied.                                                                                                                       |
| Shared Storage  | A Component to store data shared among AIMs. The Shared Storage is part of the Trusted Zone.                                                                                             |
| Status          | The set of parameters characterising a Component.                                                                                                                                        |
| Structure       | A composition of Records                                                                                                                                                                 |
| Swarm Element   | An AIF in a in a proximity-based scenario.                                                                                                                                               |
| Time Base       | The protocol specifying how Components can access timing information. The \Time Base is part of the Trusted Zone.                                                                        |
| Topology        | The set of Channels connecting AIMs in an AIW.                                                                                                                                           |
| Trusted Zone    | An environment that contains only trusted objects, i.e., object that do not require further authentication.                                                                              |
| User Agent      | The Component interfacing the user with an AIF through the Controller                                                                                                                    |
| Zero Trust      | A cybersecurity model primarily focused on data and service protection that assumes no implicit trust [14].                                                                              |

## 4 References

### 4.1 Normative references

MPAI-AIF normatively references the following documents:

1. GIT protocol, <https://git-scm.com/book/en/v2/Git-on-the-Server-The-Protocols>.
2. ZIP format, <https://pkware.cachefly.net/webdocs/casestudies/APPNOTE.TXT>.
3. Date and Time in the Internet: Timestamps; IETF RFC 3339; July 2002.
4. Uniform Resource Identifiers (URI): Generic Syntax, IETF RFC 2396, August 1998.
5. The JavaScript Object Notation (JSON) Data Interchange Format; <https://datatracker.ietf.org/doc/html/rfc8259>; IETF rfc8259; December 2017

6. JSON Schema; <https://json-schema.org/>.
7. BNF Notation for syntax; <https://www.w3.org/Notation.html>
8. MPAI; The MPAI Ontology; <https://mpai.community/standards/mpai-aif/mpai-ontology/>
9. MPAI; The MPAI Statutes; <https://mpai.community/statutes/>
10. MPAI; The MPAI Patent Policy; <https://mpai.community/about/the-mpai-patent-policy/>.
11. Framework Licence of the Artificial Intelligence Framework Technical Specification (MPAI-AIF); <https://mpai.community/standards/mpai-aif/framework-licence/>

## 4.2 Informative references

12. Technical Specification: The Governance of the MPAI Ecosystem V1, 2021; <https://mpai.community/standards/mpai-gme/>
13. Message Passing Interface (MPI), <https://www.mcs.anl.gov/research/projects/mpi/>
14. Rose, Scott; Borchert, Oliver; Mitchell, Stu; Connelly, Sean; "Zero Trust Architecture"; <https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-207.pdf>
15. MPAI Technical Specification; Context-based Audio Enhancement (MPAI-CAE) V1; <https://mpai.community/standards/resources/>.
16. MPAI Technical Specification; Compression and Understanding of Industrial Data (MPAI-CUI) V1; <https://mpai.community/standards/resources/>.
17. MPAI Technical Specification; Multimodal Conversation (MPAI-MMC) V1; <https://mpai.community/standards/resources/>.
18. W. Wang, J. Gao, M. Zhang, S. Wang, G. Chen, T. K. Ng, B. C. Ooi, J. Shao, and M. Reyad, "Rafiki: machine learning as an analytics service system," Proceedings of the VLDB Endowment, vol. 12, no. 2, pp. 128–140, 2018.
19. Y. Lee, A. Scolari, B.-G. Chun, M. D. Santambrogio, M. Weimer, and M. Interlandi; PRETZEL: Opening the black box of machine learning prediction serving systems; in 13th USENIX Symposium on Operating Systems Design and Implementation (OSDI18), pp. 611–626, 2018.
20. ML.NET [ONLINE]; <https://dotnet.microsoft.com/apps/machinelearning-ai/ml-dotnet>.
21. D. Crankshaw, X. Wang, G. Zhou, M. J. Franklin, J. E. Gonzalez, and I. Stoica; Clipper: A low-latency online prediction serving system; in NSDI, pp. 613–627, 2017.
22. S. Zhao, M. Talasila, G. Jacobson, C. Borcea, S. A. Aftab, and J. F. Murray; Packaging and sharing machine learning models via the acumos ai open platform; in 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA), pp. 841–846, IEEE, 2018.
23. Apache Prediction I/O; <https://predictionio.apache.org/>.
24. D. Sculley, G. Holt, D. Golovin, E. Davydov, T. Phillips, D. Ebner, V. Chaudhary, M. Young, J. Crespo, D. Dennison; Hidden technical debt in Machine learning systems Share; on NIPS'15: Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2; December 2015 Pages 2503–2511

## 5 Architecture of the AI Framework (Normative)

### 5.1 AI Framework Components

MPAI-AIF normatively specifies the Components of *Figure 1*. All Components, except Access, operate in a Trusted Zone. This is graphically depicted with a read frame around the Components. The specific functions of the Components are:

1. **Controller:**
  - a. Provides basic functionalities such as scheduling, communication between AIMs and with AIF Components such as Internal and Shared Storage.

- b. Acts as a resource manager, according to instructions given by the User.
  - c. Is connected by default to all the AIFs in a given AIF.
  - d. Activates/suspends/resumes/deactivates AIWs based on User's or other inputs.
  - e. Supports complex application scenarios by balancing load and resources.
  - f. Exposes three APIs:
    - i. *AIM APIs* enable AIFs to communicate with it (register themselves, communicate and access the rest of the AIF environment). An AIW is an AIF with additional metadata. Therefore, an AIW uses the same AIF API.
    - ii. *User APIs* enable User or other Controllers to perform high-level tasks (e.g., switch the Controller on and off, give inputs to the AIW through the Controller).
    - iii. *Controller-to-Controller API* enables interactions among Controllers.
  - g. Accesses the *Store APIs* enabling communication between the AIF and the Store.
  - h. May run one or more AIWs and on a different computing platform than the AIW.
  - i. May communicate with other Controllers. When different Controllers running on separate agents (Swarm Elements) interact with one another, they cooperate by requesting one or more Controllers in range to open Remote Ports. The Controllers on which the Remote Ports are opened can then react to information sent by other Controllers in range through the Remote Ports and implement a collective behaviour of choice. For instance: there is a main Controller and the other Controllers in the swarm react to the information it sends; or there is no main Controller and all Controllers in the swarm behave according to a collective logic specified in the programming of all Controllers.
2. **Communication:** connects the AIF Components via Events or Channels connecting an output Port of an AIF with an input Port of another AIF. Communication has the following characteristics:
- a. The Communication Component is turned on jointly with the Controller.
  - b. Communication needs not be persistent.
  - c. Channels are unicast and may be physical or logical.
  - d. Messages are transmitted via Channels. They are composed of sequences of Records and may be of two types:
    - i. High-Priority Messages expressed as up to 16-bit integers.
    - ii. Normal-Priority Messages expressed as MPAI-AIF defined types (6.1.1).
 Messages may be communicated through Channels or Events.
3. **AI Module (AIF):** a data processing element with a specified Function receiving AIF-specific inputs and producing AIF-specific outputs having the following characteristics:
- a. Communicates with other Components through Ports or Events.
  - b. Includes at least one AIF with an input Port and one AIF with an output Port.
  - c. May incorporate other AIFs.
  - d. May be hot-pluggable, and dynamically register and disconnect itself on the fly.
  - e. May be executed:
    - i. Locally, i.e., it encapsulates hardware physically accessible to the Controller.
    - ii. On different computing platforms, e.g., in the cloud or on swarms of drones, and encapsulates communication with a remote Controller.
4. **AI Workflow (AIW):** an organised aggregation of AIFs receiving AIF-specific inputs and producing AIF-specific outputs according to its Function implementing a use case that is proprietary or specified by an MPAI Application Standard.
5. **Shared Storage:** stores data shared by AIFs.
6. **AIF Storage:** stores data of individual AIFs.
7. **User Agent:** interfaces the User with an AIF through the Controller.
8. **Access:** provides access to static or slowly changing data that is required by AIFs such as domain knowledge data, data models, etc.

9. **Store:** stores Implementations for users to download by secure protocols.

## 5.2 AI Framework Implementations

MPAI-AIF enables a wide variety of Implementations:

1. AIF Implementations can be tailored to different execution environments, e.g., High-Performance Computing systems or resource-constrained computing boards. For instance, the Controller might be a process on a HPC system or a library function on a computing board.
2. There is always a Controller even if the AIF is a lightweight Implementation.
3. The API may have different MPAI-defined Profiles to allow for Implementations:
  - a. To run on different computing platforms and different programming languages.
  - b. To be based on different hardware and resources available.
4. AIMs may be Implemented in hardware, software and mixed-hardware and software.
5. Interoperability between AIMs is ensured by the way communication between AIMs is defined, irrespective of whether they are implemented in hardware or software.
6. Use of Ports and Channels ensures that compatible AIM Ports may be connected together irrespective of the AIMs' implementation technology.
7. Message generation and Event management is implementation independent.

## 5.3 AIMs

### 5.3.1 Implementation types

AIMs can be implemented in either hardware or software keeping the same interfaces independent of the implementation technology. However, the nature of the AIM might impose constraints on the specific values of certain API parameters and different Profiles may impose different constraints. For instance, Events (easy to accommodate in software but less so in hardware); and persistent Channels (easy to make in hardware, less so in software).

While software-software and hardware-hardware connections are homogeneous, a hybrid hardware-software scenario is inherently heterogeneous and requires the specification of additional communication protocols, which are used to wrap the hardware part and connect it to software. A list of such protocols is provided by the MPAI Ontology [8].

Examples of supported architectures are:

- *CPU-based devices* running an operating system.
- *Memory-mapped devices* (FPGAs, GPUs, TPUs) which are presented as accelerators.
- *Cloud-based frameworks*.
- *Naked hardware devices* (i.e., IP in FPGAs) that communicate through hardware Ports.
- *Encapsulated blocks of a hardware design* (i.e., IP in FPGAs) that communicate through a memory-mapped bus. In this case, the Metadata associated with the AIM (see 6.3) shall also specify the low-level communication protocol used by the Ports.

### 5.3.2 Combination

MPAI-AIF supports the following ways of combining AIMs:

- *Software AIMs* connected to other software AIMs resulting in a software AIM
- *Non-encapsulated hardware blocks* connected to other non-encapsulated hardware blocks, resulting in a larger, non-encapsulated hardware AIM
- *Encapsulated hardware blocks* connected to either other encapsulated hardware blocks or other software blocks, resulting in a larger software AIM.

Connection between a non-encapsulated hardware AIM and a software AIM is not supported as in such a case direct communication between the AIMs cannot be defined in any meaningful way.

### 5.3.3 Hardware-software compatibility

To achieve communication among AIMs irrespective of their implementation technology, the requirements considered in the following two cases should be satisfied:

1. *Hardware AIM to Hardware AIM*: Each named type in a Structure is transmitted as a separate channel. Vector types are implemented as two channels, one transmitting the size and the second transmitting the data.
2. *All other combinations*: Fill out a Structure by recursively traversing the definition (breadth-first). Sub-fields are laid down according to their type, in little-endian order.

### 5.3.4 Actual implementations

#### 5.3.4.1 Hardware

Metadata ensures that hardware blocks can be directly connected to other hardware/software blocks, provided the specification platforms for the two blocks have compatible interfaces, i.e., they have compatible Ports and Channels.

#### 5.3.4.2 Software

Software Implementations shall ensure that Communication among different constituent AIMs, and with other AIMs outside the block, is performed correctly.

In addition, AIM software Implementations shall contain a number of well-defined steps so as to ensure that the Controller is correctly initialised and remains in a consistent internal state, i.e.:

1. **Code registering the different AIMs** used by the AIW. The registration operation specifies where the AIMs will be executed, either locally or remotely. The AIM Implementations are archives downloaded from the Store containing source code, binary code and hardware designs executed on a local machine/HPC cluster/MPC machine or a remote machine.
2. **Code starting/stopping** the AIMs.
3. **Code registering the input/output Ports** for the AIM.
4. **Code instantiating unicast channels** between AIM Ports belonging to AIMs used by the AIW, and connections from/to the AIM being defined to/from remote AIMs.
5. **Registering Ports** and connecting them may result in a number of steps performed by the Controller – some suitable data structure (including, for instance, data buffers) will be allocated for each Port or Channel, in order to support the functions specified by the Controller API called by the AIM (7.5).
6. **Explicitly write/read data** to/from, any of the existing Ports.
7. In general, arbitrary functionality can be added to a software AIM. For instance, depending on the AIM Function, one would typically link libraries that allow a GPU or FPGA to be managed through Direct Memory Access (DMA), or link and use high-level libraries (e.g., TensorFlow) that implement AI-related functionality.
8. The API implementation depends on the architecture the Implementation is designed for.

## 6 Metadata

Metadata specifies static properties pertaining to the interaction between:

1. A Controller and its hosting hardware.
2. An AIW and its hosting Controller.
3. An AIM and the AIW it belongs to.

Metadata specified in the following Sections is represented in JSON Schema.

### 6.1 Communication channels and their data types

This Section specifies how Metadata pertaining to a communication Channel is defined.

### 6.1.1 Type system

The data interchange happening through buffers involves the exchange of structured data. Message data types exchanged through Ports and communication Channels are defined by the following Backus–Naur Form (BNF) specification [7]. Words in bold typeface are keywords; capitalised words such as NAME are tokens.

```
fifo_type :=
    | /* The empty type */
    | base_type NAME
recursive_type :=
    | recursive_base_type NAME
base_type :=
    | toplevel_base_type
    | recursive_base_type
    | ( base_type )
toplevel_base_type :=
    | array_type
    | toplevel_struct_type
    | toplevel_variant_type
array_type :=
    | recursive_base_type []
toplevel_struct_type :=
    | { one_or_more_fifo_types_struct }
one_or_more_fifo_types_struct :=
    | fifo_type
    | fifo_type ; one_or_more_fifo_types_struct
toplevel_variant_type :=
    | { one_or_more_fifo_types_variant }
one_or_more_fifo_types_variant :=
    | fifo_type | fifo_type
    | fifo_type | one_or_more_fifo_types_variant
recursive_base_type :=
    | signed_type
    | unsigned_type
    | float_type
    | struct_type
    | variant_type
signed_type :=
    | int8
    | int16
    | int32
    | int64
unsigned_type :=
    | uint8 | byte
    | uint16
    | uint32
    | uint64
float_type :=
    | float32
    | float64
struct_type :=
    | { one_or_more_recursive_types_struct }
one_or_more_recursive_types_struct :=
    | recursive_type
    | recursive_type ; one_or_more_recursive_types_struct
variant_type :=
    | { one_or_more_recursive_types_variant }
```