

Australian Standard™

**Information technology—Generic
coding of moving pictures and
associated audio information**

Part 11: IPMP on MPEG-2 systems



This Australian Standard was prepared by Committee IT-029, Coded Representation of Picture, Audio and Multimedia/Hypermedia Information. It was approved on behalf of the Council of Standards Australia on 27 October 2004.
This Standard was published on 3 December 2004.

The following are represented on Committee IT-029:

Australian Broadcasting Authority
Australian Broadcasting Corporation
Australian Consumer Association
Australian Subscription Television
CSIRO Information and Communication Technology Centre
DSTC
Department of Defence (Australia)
Free TV Australia
Special Broadcasting Service
The University of New South Wales
University of Sydney
University of Wollongong
Victoria University of Technology

Keeping Standards up-to-date

Standards are living documents which reflect progress in science, technology and systems. To maintain their currency, all Standards are periodically reviewed, and new editions are published. Between editions, amendments may be issued. Standards may also be withdrawn. It is important that readers assure themselves they are using a current Standard, which should include any amendments which may have been published since the Standard was purchased.

Detailed information about Standards can be found by visiting the Standards Web Shop at www.standards.com.au and looking up the relevant Standard in the on-line catalogue.

Alternatively, the printed Catalogue provides information current at 1 January each year, and the monthly magazine, *The Global Standard*, has a full listing of revisions and amendments published each month.

Australian Standards™ and other products and services developed by Standards Australia are published and distributed under contract by SAI Global, which operates the Standards Web Shop.

We also welcome suggestions for improvement in our Standards, and especially encourage readers to notify us immediately of any apparent inaccuracies or ambiguities. Contact us via email at mail@standards.org.au, or write to the Chief Executive, Standards Australia International Ltd, GPO Box 5420, Sydney, NSW 2001.

This Standard was issued in draft form for comment as DR 04409.

Australian Standard™

**Information technology—Generic
coding of moving pictures and
associated audio information**

Part 11: IPMP on MPEG-2 systems

First published as AS ISO/IEC 13818.11-2004.

COPYRIGHT

© Standards Australia International

All rights are reserved. No part of this work may be reproduced or copied in any form or by any means, electronic or mechanical, including photocopying, without the written permission of the publisher.

Published by Standards Australia International Ltd GPO Box 5420, Sydney, NSW 2001, Australia

ISBN 0 7337 6396 0

PREFACE

This Standard was prepared by the Standards Australia Committee IT-029, Coded Representation of Picture, Audio and Multimedia/Hypermedia Information.

This Standard is identical with, and has been reproduced from, ISO/IEC 13818-11:2004, *Information technology—Generic coding of moving pictures and associated audio information—Part 11: IPMP on MPEG-2 systems*.

The objective of this Standard is to provide users of MPEG-2 systems with the IPMP (Intellectual Property Management and Protection) guidelines, including the syntax and semantics for IPMP control, signalling and data extension.

This Standard is Part 11 of AS 13818, *Information technology—Generic coding of moving pictures and associated audio information*, which is published in parts as follows:

Part 1: Systems

Part 2: Video

Part 3: Audio

Part 4: Conformance testing

Part 5: Software simulation

Part 6: Extensions for DSM-CC

Part 7: Advanced audio coding

Part 9: Extension for real time interface for systems decoders

Part 10: Conformance extensions for Digital Storage Media Command and Control (DSM-CC)

Part 11: IPMP on MPEG-2 systems (this Standard)

The terms ‘normative’ and ‘informative’ are used to define the application of the annex to which they apply. A normative annex is an integral part of a standard, whereas an informative annex is only for information and guidance.

As this Standard is reproduced from an international standard, the following applies:

- (a) Its number appears on the cover and title page while the international standard number appears only on the cover.
- (b) In the source text ‘this International Standard’ should read ‘this Australian Standard’.
- (c) A full point substitutes for a comma when referring to a decimal marker.

References to International Standards should be replaced by references to Australian or Australian/New Zealand Standards, as follows:

<i>Reference to International Standard</i>		<i>Australian Standard/New Zealand Standard</i>	
ISO/IEC		AS/NZS	
13818	Information technology—Generic coding of moving pictures and associated audio information	13818	Information technology—Generic coding of moving pictures and associated audio information
13818-1	Part 1: Systems	13818.1	Part 1: Systems
14496	Information technology—Coding of audio-visual objects	14496	Information technology—Coding of audio-visual objects
14496-1	Part 1: Systems	14496.1	Part 1: Systems

Only referenced documents that have been adopted as Australian or Australian/New Zealand Standards have been listed.

CONTENTS

Page

1	Scope.....	1
2	Normative references	2
3	Terms and definitions.....	2
4	Compatibility with Conditional Access framework (Informative)	4
4.1	The existing MPEG-2 CA framework.....	4
4.2	Backward compatibility – the carriage of IPMP in the CA framework.....	4
4.3	Forward compatibility – the carriage of CA data within MPEG-2 IPMP.....	5
4.4	Co-existence of MPEG-2 IPMP and Conditional Access.....	5
5	Overview of MPEG-2 IPMP (Informative)	6
5.1	IPMP Architecture	6
5.1.1	IPMP tool acquisition and protection signaling.....	6
5.1.2	IPMP Information carriage in MPEG-2 Content.....	7
5.1.3	Messaging.....	7
5.1.4	Mutual Authentication	7
5.1.5	IPMP Tool connection and disconnection.....	8
5.1.6	Notification of IPMP Tool connection and disconnection	9
5.1.7	Common IPMP processing	9
5.1.8	IPMP tool to/from User interaction.....	9
6	Specifications (Normative).....	10
6.1	Overview	10
6.1.1	MPEG-2 IPMP architecture	10
6.1.2	Structure of IPMP protected MPEG-2 content	11
6.2	IPMP Control Information.....	11
6.2.1	IPMP Control Information in Transport Stream	11
6.2.2	IPMP Control Information in Program Stream	18
6.3	IPMP Protection Signalling	21
6.3.1	IPMP Protection Signalling in Transport Stream.....	21
6.3.2	IPMP Protection Signalling in Program Stream.....	22
6.3.3	IPMP Descriptor	22
6.4	IPMP Stream	24
6.4.1	IPMP Stream Specification.....	24
6.4.2	IPMP Stream Syntax	25
6.4.3	IPMP Stream Decoder Model	25
7	IPMP Data and Messages (Normative).....	27
7.1	IPMP_Data_BaseClass	27
7.1.1	Syntax.....	27
7.1.2	Semantics	27
7.1.3	Extension tags for the IPMP_Data_BaseClass message.....	28
7.2	Mutual Authentication	29
7.2.1	IPMP_InitAuthentication.....	29
7.2.2	IPMP_Mutual_Authentication	29
7.2.3	IPMP_TrustSecurityMetadata	34
7.2.4	DateClass	35
7.2.5	IPMP_SecureContainer.....	35
7.3	IPMP Tool connection and disconnection.....	36
7.3.1	IPMP_GetTools.....	36
7.3.2	IPMP_GetToolsResponse	36
7.3.3	IPMP Tool Parametric Capabilities Query	36

7.3.4	IPMP Tool Parametric Capabilities Query Response	37
7.3.5	IPMP_ConnectTool.....	37
7.3.6	IPMP_DisconnectTool.....	37
7.4	IPMP Tool notification.....	38
7.4.1	IPMP_AddToolNotificationListener	38
7.4.2	IPMP_RemoveToolNotificationListener	39
7.4.3	IPMP_NotifyToolEvent	39
7.5	IPMP Processing	40
7.5.1	IPMP_CanProcess	40
7.5.2	IPMP Opaque data	40
7.5.3	IPMP_KeyData	40
7.5.4	IPMP_RightsData.....	41
7.5.5	IPMP_SelectiveDecryptionInit.....	41
7.5.6	IPMP_AudioWatermarkingInit.....	41
7.5.7	IPMP_SendAudioWatermark.....	41
7.5.8	IPMP_VideoWatermarkingInit	41
7.5.9	IPMP_SendVideoWatermark	41
7.6	User Interaction Messages.....	41
7.6.1	IPMP_UserQuery	42
7.6.2	IPMP_UserQueryResponse	44
7.7	IPMP Information Delivery Functions.....	44
7.7.1	IPMP_ToolMessageBase	45
7.7.2	IPMP_MessageFromBitstream.....	45
7.7.3	IPMP_DescriptorFromBitstream	46
7.7.4	IPMP_MessageFromTool.....	46
Annex A	(normative) Tool/Content Transfer Messages among Distributed IPMP Devices	47
A.1	Introduction.....	47
A.2	Addressing of distributed devices	47
A.3	IPMP_DeviceMessageBase	47
A.3.1	Syntax.....	47
A.3.2	Semantics.....	47
A.4	Device to Device IPMP Message.....	48
A.4.1	IPMP_MessageFromDevice.....	48
A.5	Content Transfer Messages	48
A.5.1	IPMP_RequestContent.....	48
A.5.2	IPMP_ResponseToContentRequest	49
A.5.3	IPMP_ContentTransfer.....	49
A.6	Tool Transfer Messages	50
A.6.1	IPMP_RequestTool.....	50
A.6.2	IPMP_ResponseToToolRequest	50
A.7	Device ID messages	51
A.7.1	IPMP_DeviceID_Notification Message	51
Annex B	(normative) Schema for Terminal Platform	52
Annex C	(normative) Selective Decryption Configuration Data	55
C.1	Introduction.....	55
C.2	IPMP_SelectiveDecryptionInit.....	55
C.2.1	Syntax.....	55
C.2.2	Semantics.....	56
C.3	An example of a selective decryption configuration data (Informative).....	58
Annex D	(normative) Audio Watermarking Configuration and Notification	61
D.1	Introduction.....	61
D.2	IPMP_AudioWatermarkingInit.....	61
D.2.1	Syntax.....	61
D.2.2	Semantics.....	62
D.3	IPMP_SendAudioWatermark	63
D.3.1	Syntax.....	63
D.3.2	Semantics.....	63

Annex E (normative) Video Watermarking Configuration and Notification Data.....	64
E.1 Introduction	64
E.2 IPMP_VideoWatermarkingInit	64
E.2.1 Syntax.....	64
E.2.2 Syntax.....	65
E.3 IPMP_SendVideoWatermark	65
E.3.1 Syntax.....	65
E.3.2 Semantics	66
Annex F (informative) An example of a simple use case of MPEG-2 IPMP	67
F.1 Content authoring	67
F.2 MPEG-2 IPMP Terminal behaviour	67
Annex G (normative) List of Registration Authorities	70
G.1 Registered Data	70
G.2 Procedure for the request of Registered Data	70
G.3 Responsibilities of the Registration Authority	70
G.4 Contact information for the Registration Authority	71
G.5 Responsibilities of Parties Requesting Registered Data	71
G.6 Appeal Procedure for Denied Applications.....	71
G.7 Registration Application Form	72
G.7.1 Contact Information of organization requesting a RID	72
G.7.2 Request for specific registered data.....	72
G.7.3 Short description of the Registered Data that is in use and date system was implemented	72
G.7.4 Statement of an intention to apply the assigned Registered Data	72
G.7.5 Date of intended implementation of the Registered Data	73
G.7.6 Authorized representative.....	73
G.7.7 For official use of the Registration Authority.....	73
Annex H (informative) Patent statements	74

NOTES

AUSTRALIAN STANDARD

Information technology — Generic coding of moving pictures and associated audio information —

Part 11: IPMP on MPEG-2 systems

1 Scope

This International Standard specifies IPMP (Intellectual Property Management and Protection) on the MPEG-2 system, including:

- a) syntax and semantics for IPMP control information which includes tool list, tool container and rights container;
- b) syntax and semantics for IPMP descriptors, which facilitates IPMP protection signalling;
- c) syntax and semantics of IPMP data extending from the common base class IPMP_Data_BaseClass to support the following functionalities:
 - mutual authentication for IPMP tool to IPMP tool as well as IPMP tool to terminal communication,
 - the requesting by IPMP tools of the connection/disconnection to requested IPMP tools,
 - the notification to IPMP tools of the connection/disconnection of IPMP tools,
 - common IPMP processing,
 - IPMP tool to/from user interaction;
- d) syntax and semantics for IPMP information carriage to and from IPMP tools;
- e) syntax and semantics for the request and transfer of content and IPMP tools between terminals;
- f) XML syntax and semantics for the description of the environment in which the MPEG-2 Terminal/application is operating;
- g) a list of registration authorities required for the support of the specifications found herein.

This document is organized as follows.

Clause 1 provides an introduction to the document. Clause 4 explains the compatibility between the Conditional Access framework and MPEG-2 IPMP framework. Clause 5 provides an overview of the process supported by the IPMP framework, and identifies different normative elements in this process. Clause 6 provides specifications for components identified in Clause 5. Clause 7 provides specifications for the messaging architecture and all supported messages.

2 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 10646-1:2000, *Information technology — Universal Multiple-Octet Coded Character Set (UCS) — Part 1: Architecture and Basic Multilingual Plane*

ISO/IEC 13818-1, *Information technology — Generic coding of moving pictures and associated audio information: Systems*

ISO/IEC 14496-1:2001, *Information technology — Coding of audio-visual objects — Part 1: Systems*

ISO/IEC 14496-13¹⁾, *Information technology — Coding of audio-visual objects — Part 13: Intellectual Property Management and Protection (IPMP) extensions*

XML Schema Part 0: Primer, Part 1: Structures, and Part 2: Datatypes, W3C Recommendation, 2 May 2001, available at <<http://www.w3.org/TR/2001/REC-xmlschema-0-20010502>>, <<http://www.w3.org/TR/2001/REC-xmlschema-1-20010502>>, and <<http://www.w3.org/TR/2001/REC-xmlschema-2-20010502>>, respectively

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

3.1

Binary Representation

In the context of an IPMP Tool, this is the format of the implementation of that IPMP Tool. Examples: Platform Dependent Native Code, Java™ bytecode

3.2

Content

This implies part or whole of an MPEG presentation

3.3

Content Consumption

Any experience of given Content implies consumption of that content. Access, Playback, Denial of Access and Creation of a Copy are all types of content consumption

3.4

Content Stream

This is the incoming content, of MPEG-2 system

3.5

IPMP Device

An implemented application that implements an MPEG-2 Terminal supporting the use of MPEG-2 IPMP

3.6

IPMP Information

Information directed to a given IPMP Tool to enable, assist or facilitate its operation

3.7

IPMP Tool

IPMP tools are modules that perform (one or more) IPMP functions such as authentication, decryption, watermarking, etc. A given IPMP Tool may coordinate other IPMP Tools

1) To be published.

3.8**IPMP Tool Identifier**

This refers to the IPMP Tool ID. It identifies a Tool in an unambiguous way, at the presentation level or at a universal level

3.9**IPMP Tool List**

The IPMP Tool List identifies, and enables selection of, the IPMP Tools required to process the Content

3.10**IPMP Tool Manager**

The IPMP Tool Manager is a conceptual entity within the Terminal that processes IPMP Tool List(s) and retrieves the Tools that are specified therein

3.11**IPMP Tool Message**

A message passed between any combination of IPMP Tool or Terminal

3.12**Message Router**

A conceptual entity within the Terminal that implements the Terminal-side behaviour of the Terminal-Tool interface

3.13**Mutual Authentication**

Protocols carried out to determine the proper and correct identity of a communicating partner and the securing of communication channels between communicating partners

3.14**Parametric Description**

Information that enables a Terminal to choose a specific Tool implementation that will support all functionality required by a presentation

3.15**Renewability**

Renewability implies that a Terminal may acquire Missing IPMP Tools, or update an existing IPMP Tool with an updated version

3.16**Representation Format**

The binary format, platform and communication mechanisms applicable to a given implementation of an IPMP Tool or Terminal

3.17**Scope of Protection**

Scope of protection refers to the elementary stream and/or program governed by a given IPMP Tool instance

3.18**Secure Communication**

Communication between two entities is said to be secure to a sufficient level when the communication between the two entities can not be attacked by being altered, removed, forged or evesdropped on to the levels required by the two entities

3.19**Terminal**

A Terminal is an environment that consumes possibly protected content in compliance with the usage rules

3.20

User

A hardware, software or human entity that is the initiator and/or target of content consumption

3.21

Verification

Establishing, to an acceptable level of certainty, that a certain condition that was claimed to be true is true

4 Compatibility with Conditional Access framework (Informative)

4.1 The existing MPEG-2 CA framework

MPEG-2 systems provides a framework for the protection of content carried in Transport Streams and Program Streams. In the case of Transport Streams, the framework is based on these three distinct elements:

- A 2 bit **transport_scrambling_control** field in the TS packet header that is used to notify the receiver that the contents of the TS packet are scrambled. In practice, these bits may identify which of two “control words” is used as the scrambling key for the packet. It is left to the CA vendor to manage the transport and availability of the control words in the receiver. The same field is present in the PES packet header and is used to signal scrambling at the PES level.
- Conditional Access descriptors are located in Program Map sections to identify the content streams which are protected by the CA system and the streams carrying the ECM private data. These descriptors may optionally carry CA parameters associated with the protected content streams.
- The Conditional Access sections carry further Conditional Access descriptors that identify the Conditional Access system(s) associated with the entire Transport Stream and notify the receiver of the streams containing EMM private data.

For Program Streams, the following framework is defined:

- A 2 bit **PES_scrambling_control** field in the PES packet header that is used to notify the receiver that the contents of the PES packet are scrambled. In practice, these bits may identify which of two “control words” is used as the scrambling key for the packet. It is left to the CA vendor to manage the transport and availability of the control words in the receiver.
- Conditional Access descriptors are located in the Program Stream Map to identify the content streams which are protected by the CA system and the streams carrying the ECM private data. These descriptors may optionally carry CA parameters associated with the protected content streams.
- The Program Stream Map carry further Conditional Access descriptors that identify the Conditional Access system(s) associated with the entire Transport Stream and notify the receiver of the streams containing EMM private data.

These elements are fully described in ISO/IEC 13818-1.

4.2 Backward compatibility – the carriage of IPMP in the CA framework

In order to allow CA systems to recognize the presence of IPMP protection of content in Transport and Program Streams, the CA framework described above must be used to signal the presence of IPMP protection. IPMP enabled receivers may choose to locate IPMP streams using either the CA framework or the traditional IPMP mechanisms described elsewhere in this document. Systems designed to work only with the CA framework will be able to identify the use of a protection mechanism that renders the content inaccessible and deal with it accordingly.

The presence of IPMP protection is identified using the Conditional Access descriptor defined in ISO/IEC 13818-1 and which is carried in the Conditional Access sections and Program Map sections within Transport Streams and in the Program Stream Map in a Program Stream.

The Conditional Access descriptor is duplicated below and the relevant fields amended below the table to show how IPMP presence is signalled in the descriptor.

Syntax	No. of bits	Mnemonic
CA_descriptor() {		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
CA_system_ID	16	uimsbf
Reserved	3	bslbf
CA_PID	13	uimsbf
For (i=0; i<N; i++) {		
private_data_byte	8	uimsbf
}		
}		

CA_system_ID – This is a 16 bit field indicating the type of CA system applicable for either the associated ECM and/or EMM streams. The coding of this is privately defined and is not specified by ITU-T | ISO/IEC.

CA_PID – This is a 13 bit field indicating the PID of the Transport Stream packets which shall contain either ECM or EMM information for the CA systems as specified with the associated CA_system_ID. The contents (ECM or EMM) of the packets indicated by the CA_PID is determined from the context in which the CA_PID is found, i.e. a TS_program_map_section or the CA table in the Transport Stream, or the stream_id field in the Program Stream.

For Transport Streams, the CA_PID value of 0x0003 shall be used to indicate that there is IPMP protection within the system, since this PID is reserved for the IPMP Control Information Table.

4.3 Forward compatibility – the carriage of CA data within MPEG-2 IPMP

If a CA system is required to have data transported in MPEG-2 IPMP, an IPMP Tool ID should be registered. If the CA system is used to descramble the scrambled elementary stream, its tool ID should be included in the IPMP Tool List in IPMP Control Information.

There should be an IPMP Descriptor [6.3.3] created for this CA system. The IPMP Tool ID shall be set to the IPMP Tool ID registered. The control point should be set to 0x01, being the Control Point after Transport Buffer TB_n in STD if the content is scrambled at Transport Stream level and at control point 0x02, being the Control Point between the decode buffer (B_n or ES_n) and the decoder (D_n) in STD in the case of PES layer scrambling for Program or Transport Streams..

The ECMs and EMMs can be carried in either IPMP Descriptors or IPMP_StreamDataUpdate in IPMP Streams [6.4], in the form of IPMP_OpaqueData [7.5.2] extending from IPMP_Data_BaseClass. This information will then be routed by the terminal to the IPMP Tool (CA system) with the specified IPMP Tool ID.

In this way, an IPMP compliant terminal is able to set up the CA system at the indicated Control Point and descrambling can take place as it does in Conditional Access systems that otherwise employ the CA framework.

4.4 Co-existence of MPEG-2 IPMP and Conditional Access

A terminal can be designed in such a way as to allow the co-existence of MPEG-2 IPMP and Conditional Access systems and content as the two systems are able to operate independently or in cooperation with one another.

Content can be protected by both Conditional Access systems and MPEG-2 IPMP. Doing this means having hybrid content with IPMP Descriptors, CA descriptors, ECMs, EMMs, IPMP Stream, IPMP Control Information Table and Conditional Access Table.

The presence of CA_Descriptors in the CAT and/or Program Map sections (or the Program Stream Map in the case of Program Streams) that use a CA_system_id supported by the terminal (and CA_PID not equal to 0x0003) indicates that the stream may be received and decoded by a terminal regardless of any IPMP functions that are applied to the content. If the originator of the stream wishes to enforce the application of IPMP tools, then only CA_PID of 0x0003 should appear in any CA_descriptors applicable to that stream. In this case, conventional CA is achieved by using an IPMP Tool ID as described in 4.3 above.

There are at least two example scenarios envisaged.

- A CA system with a descrambler running at the TS control point, together with an IPMP watermarking tool running after the decoder, governed by an IPMP rights management tool to provide extended functionality and security beyond the CA system.
- Another example would be signalling the method for descrambling content using either a CA system or an IPMP Tool by the use of alternative tools in the tool list, if both of them are operating at the same control point 0x01 (or 0x02 in the case of PES level scrambling).

Handling this hybrid content demands a terminal enabled with both CA system and IPMP. Designing this type of terminal is feasible, but outside the scope of this specification.

5 Overview of MPEG-2 IPMP (Informative)

5.1 IPMP Architecture

This clause describes an IPMP architecture that may operate in cooperation with or independently of the IPMP specifications.

5.1.1 IPMP tool acquisition and protection signaling

The IPMP Tool Manager is a conceptual entity in a given IPMP Terminal that manages tools.

IPMP Control Information [6.2] carries IPMP Tool List Class [6.2.1.3.1] which conveys the list of IPMP tools required to access the content protected by IPMP, and may include a list of alternate IPMP tools or parametric descriptions of tools required to access the content. The conceptual entity Tool Manager parses the tool list, makes sure all tools are available, and retrieves missing tools if any.

IPMP Control Information may also carry IPMP Tool Container Class [6.2.1.3.8] which enables carriage of IPMP tool inside MPEG-2 system. Tool Manager may retrieve the IPMP tool from the content stream, load it, instantiate it and immediately use it in order to play out the content.

A missing IPMP Tool can also be acquired from a neighbouring IPMP device via tool transfer messages defined in Annex A, or it can be acquired by Tool Manager by connecting to a remote server and providing the terminal description as defined in Annex B.

The entire IPMP protection scheme can be described by embedding IPMP Descriptor's [6.3.3] into Program Map Table (PMT) in Transport stream or Program Stream Map (PSM) in Program Stream. It associates IPMP tool with each individual stream under its protection. It also indicates the control point at which a specific IPMP tool should be running.

The presence of IPMP Descriptor within a program loop but outside the elementary stream loop indicates that all elementary streams of this program is protected by tools given in the IPMP Descriptor.

The presence of IPMP Descriptor within the loop of an elementary stream indicates that the given elementary stream is protected by tools given in the IPMP descriptor.

5.1.2 IPMP Information carriage in MPEG-2 Content

IPMP Information can be carried in MPEG-2 content, which will then be directed to a given IPMP Tool to enable, assist or facilitate its operation.

All IPMP Information should be derived from IPMP_Data_BaseClass [7.1]. Syntax and semantics are defined for carriage of opaque data, tool initialization related data, etc. IPMP Information can be carried in either IPMP Stream [6.4] or IPMP Descriptors [6.3.3].

When IPMP Information is carried in IPMP Descriptor, the destination tool to which the IPMP Information should be directed is specified by the Tool ID specified therein. When IPMP Information is carried in IPMP Stream as a payload of IPMP_StreamDataUpdate [6.4.2], the destination tool is indirectly specified by the IPMP_Descriptor_ID specified therein.

5.1.3 Messaging

To facilitate the cooperation of multiple tools in the protection and governance of content, a message based architecture is provided. The message based architecture has three advantages over functional interface type architectures. The first is that security can more easily be maintained as messages are less difficult to protect in an open framework than parameters in a function parameters list. The second is that the only entities that need be concerned with a given message's definition are those that need to generate or act upon a given message and so additional functionality can be created and supported simply through the addition of required messages. The third is that full interoperability with IPMP tools can be easily achieved by defining a single messaging API by a third-party forum who adopts IPMP.

Physical routing of information is handled by a conceptual entity called the Message Router. The Message Router abstracts all platform-dependent routing and delivery issues, from the IPMP Tools. The interface between the Message Router and the tools is non-normative and is not defined in this specification.

All IPMP Tool interactions take place via the Terminal. IPMP Tools do not communicate directly with each other within the scope of this standard.

The delivery of both bit stream sourced IPMP information as well as IPMP tool and Terminal generated information is supported through the use of three separate messages which are passed via the Message Router to IPMP tools. The three messages are, IPMP_MessageFromBitstream [7.7.2], which is used to deliver IPMP stream data, IPMP_DescriptorFromBitstream [7.7.3], which is used to deliver IPMP Descriptors [6.3.3] and IPMP_MessageFromTool [7.7.4], which is used to deliver messages from either other IPMP tools or the Terminal itself.

5.1.4 Mutual Authentication

The most important aspect of a secure messaging architecture is the use of cryptographic algorithms and protocols that allow one to perform a number of important security functions.

At any point in IPMP Information or content processing, IPMP Tools may be required to communicate with one another or the Terminal. The degree of security required for such communication is determined by a number of variables including information that may be included by the content provider in the Content and conditions of trust established between tool providers a priori and out of band. It is generally the case that a given elementary stream is protected by multiple tools but that certain types of tools are complex (e.g. Rights Management tools) and others are utilities (e.g. Decryption engines). Complex tools may control the instantiation of other tools or make decisions about content use in response to usage queries from the terminal. Mutual authentication may occur between any pair of tools but the level of security required for this communication will in part be dictated by data contained in the bitstream in an opaque manner. The mechanism for making the determination of this security level is non-normative.

Mutual authentication is executed as follows:

1. The Tool that initiates mutual authentication with another tool determines the conditions of trust to be achieved by such authentication, i.e. the initiating tool determines whether it needs integrity protected communication or full secure, authenticated communication. This level may or may not be dictated by IPMP Information in the Content.
2. The communicating tools then engage in a message exchange to determine which authentication protocol will be used. In some cases, this protocol will have been determined by an a priori out of band negotiation between the tool providers in their security audits of one another.

Messages are defined to support the identified authentication functionalities. The first message IPMP_InitMutualAuthentication [7.2.1] may be used and delivered to a given IPMP tool such that the receiving IPMP tool is informed as to its required communication partner as well as security measures that must be in place. Following this message or the absence thereof, IPMP tools required to do so will use the IPMP_MutualAuthentication [7.2.2] message as required to determine or create secure channels of communication as needed based on the application.

As one purpose of Mutual Authentication is the verification of trust relationships existing between two entities these specifications provide for the carriage of trust and security metadata. This metadata may include zero or more certificates, credentials or integrity verification information. The creation or establishment of trust relationships are established by out of band relationships between the different entities involved in protecting and managing the content. However, the trust metadata that results from such relationships needs to be made available to permit static and dynamic verification of trust. IPMP_TrustSecurityMetadata [7.2.3] is defined to provide these additional security related data usable by IPMP tools to determine trust related information.

Once Mutual Authentication is performed, the IPMP_SecureContainer [7.2.5] may be used to pass information securely between IPMP tools and IPMP tools and Terminal.

5.1.5 IPMP Tool connection and disconnection

In the IPMP architecture, IPMP tool may be connected as the result of the presence of IPMP Descriptor in PMT or PSM and in addition may be connected due to requests from already connected IPMP tools.

Instantiation of the Tools to be connected is implementation dependent. IPMP tools may use the IPMP_GetTools [7.3.1] and IPMP_GetToolsResponse [7.3.2] messages to request a list of tools available for connection that exist as well as to respond to the request, respectively.

IPMP tools as well as the Terminal may also query a given IPMP tool as to its capabilities and functionality by using the IPMP_ToolParamCapabilitiesQuery [7.3.3] message with the tool being queried using the IPMP_ToolParamCapabilitiesResponse [7.3.4] message as a reply.

Knowing that a given tool is needed for processing, an IPMP tool may request the connection of another IPMP tool by using the IPMP_ConnectTool [7.3.5] message and may request the disconnection of another IPMP tool by using the IPMP_DisconnectTool [7.3.6] message. A connection may require the actual instantiation of a tool or may be accomplished through physical/electronic means.

The IPMP_ConnectTool message contains control point and sequence information to determine the exact location to connect the requested tool. Tools connected at the request of other tools inherit the same scope of protection as the requesting tool. Note that some control points are not associated with any known point on an Elementary Stream.

Each instantiation of an IPMP Tool shall establish a new logical instance of the tool, for a particular scope of protection. This logical instance can be uniquely identified using the IPMP_Descriptor_ID specified in the IPMP Descriptor which causes the instantiation. The binding between IPMP_Descriptor_ID and the logical instance of a tool ensures unambiguous message addressing.

The instantiation of an IPMP Tool should also resolve in a link between the MR and the Tool instance.

The tool requesting connection shall receive an IPMP_NotifyToolEvent [7.4.3] message indicating the instantiation of the IPMP Tool. The requesting tool and the instantiated tool may perform mutual authentication thereafter.

5.1.6 Notification of IPMP Tool connection and disconnection

During the processing of IPMP protected content, a number of IPMP tools may be involved, for communication and various security purposes, notification messages are supplied to notify IPMP tools when other IPMP tools have either been connected, disconnected or processed watermark information. Additionally IPMP tools may request at any time a list of all the IPMP tools currently connected at various specifiable scopes of protection.

The IPMP_AddToolNotificationListener [7.4.1] message may be used to indicate the sending IPMP tools intent to receive notifications of events and scopes of protection as specified in the IPMP_AddToolNotificationListener message. To remove oneself from being notified in the future for specified events, an IPMP tool may use the IPMP_RemoveToolNotificationListener [7.4.2] to do so.

As events occur for which notifications have been requested, the IPMP_NotifyToolEvent [7.4.3] message is sent to requesting IPMP tools.

5.1.7 Common IPMP processing

Direct support has been provided for the carrying out of common IPMP processing operations. Specific support and the related messages are as follows:

- IPMP_CanProcess [7.5.1] for the notification by an IPMP tool to the Terminal that stream processing may begin. All tools connected and processing data within a given stream must send permission before any tools in the stream receive stream data;
- IPMP_Opaquedata [7.5.2] for the carriage of user defined data;
- IPMP_KeyData [7.5.3] for the carriage of decryption key data as well as timing information to determine the validity period of time varying keys;
- IPMP_RightsData [7.5.4] for the carriage of rights expressions;
- IPMP_SelectiveDecryptionInit [Annex C] for the configuration of a decryption tool;
- IPMP_AudioWatermarkingInit [Annex D] for the configuration of an audio watermarking tool;
- IPMP_SendAudioWatermark [Annex D] for the sending of information to be embedded or that was extracted;
- IPMP_VideoWatermarkingInit [Annex E] for the configuration of a video watermarking tool;
- IPMP_SendVideoWatermark [Annex E] for the sending of information to be embedded or that was extracted.

5.1.8 IPMP tool to/from User interaction

During IPMP processing, direct interaction between IPMP tools and a User may be required. The IPMP_UserQuery [7.6.1] message is used to provide information to be relayed to a User and to request information as well. The IPMP_UserQueryResponse [7.6.2] is used to relay information provided by a User back to the originator of the User query.

6 Specifications (Normative)

6.1 Overview

6.1.1 MPEG-2 IPMP architecture

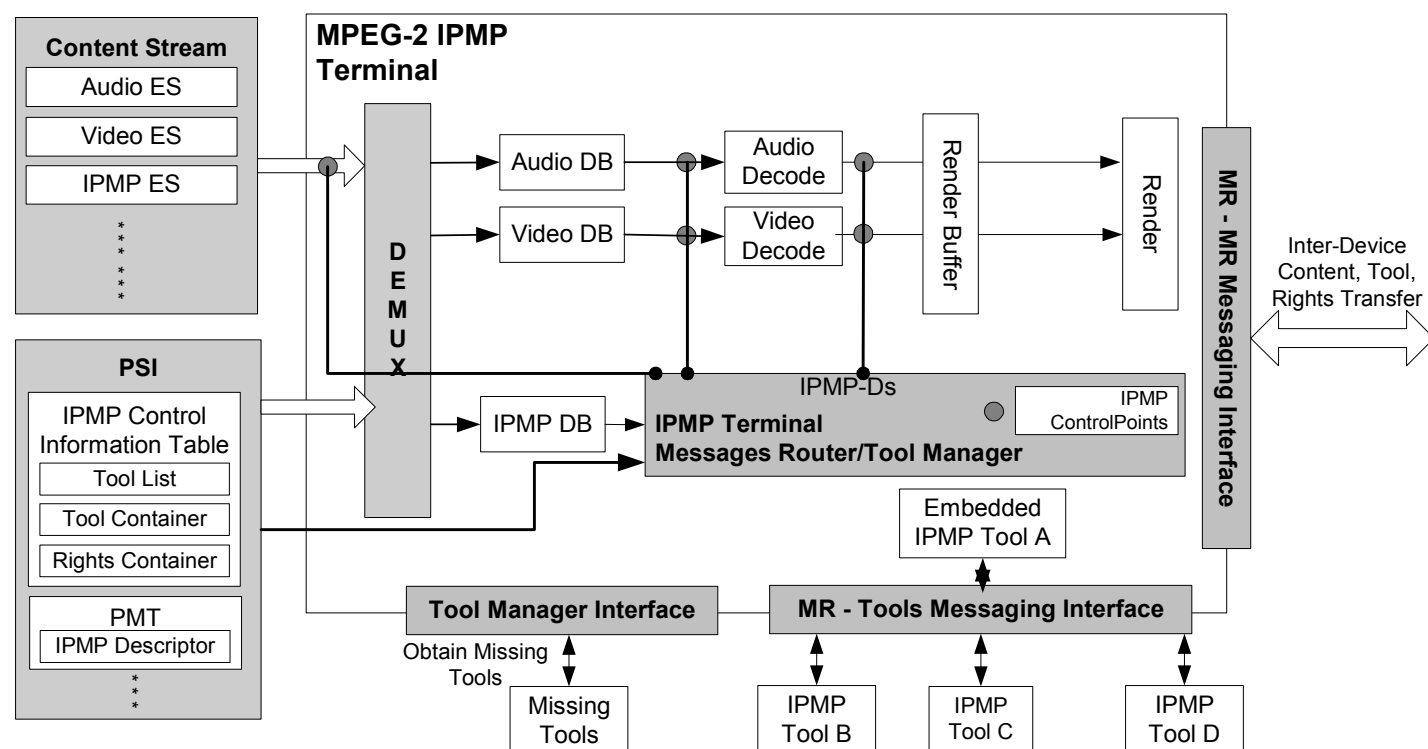


Figure 1 — MPEG-2 IPMP Architecture

The mapping of this diagram to a simplified use domain is shown in the informative Annex F.

6.1.2 Structure of IPMP protected MPEG-2 content

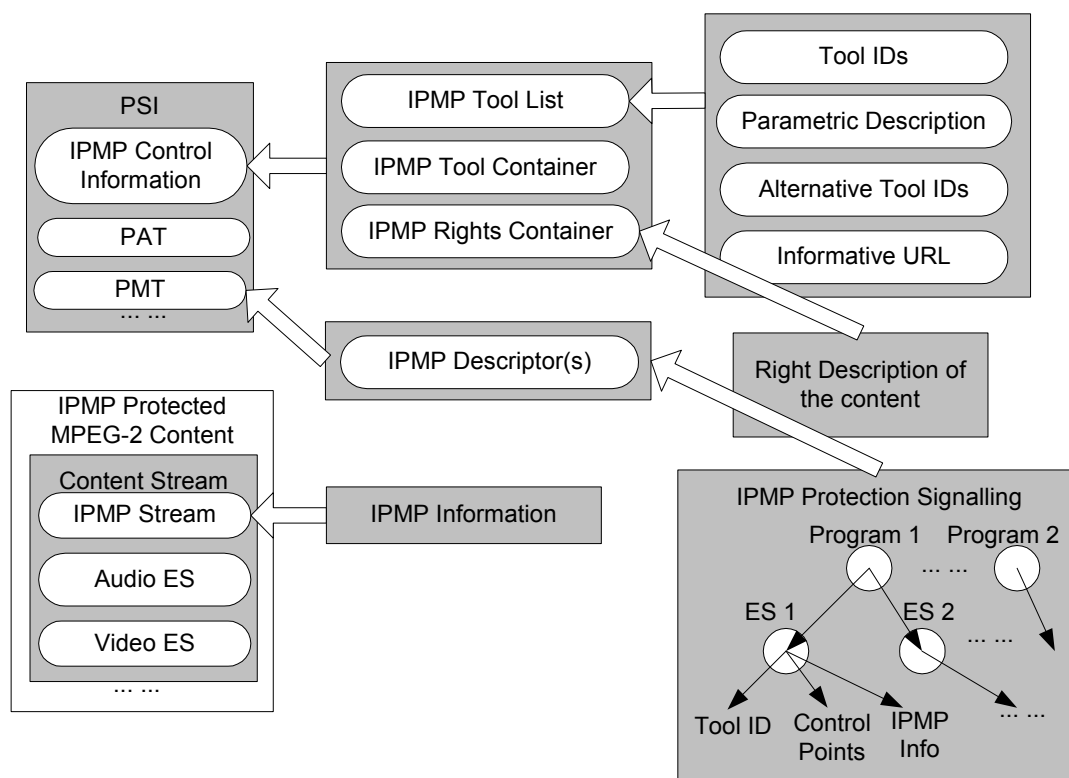


Figure 2 — Structure of IPMP protected MPEG-2 content

6.2 IPMP Control Information

IPMP Control Information contains information including Tool List, IPMP Rights Container and Tool Container. The IPMP Tool List identifies and enables selection of, the IPMP Tools required to process the Content. Tool Container enables the carriage of binary tool in content streams. The IPMP Rights Container contains rights description that describes usage rules associated with the IPMP protected content.

In short, the IPMP Control Information describes the IPMP tools needed to play the content, and the usage rights associated with the content. It could possibly carry the binary tool itself.

The entire IPMP Control Information can be optionally signed to ensure its integrity.

In Transport Stream, it exists in the form of the IPMP Control Information Table. While in Program Stream, it exists as one or more PES packets where the stream_id is extended_stream_id 0xFD and stream_id_extension of 0b000 0000, as defined in ISO/IEC 13818-1.

6.2.1 IPMP Control Information in Transport Stream

IPMP Control Information Table exists in PSI. The PID assigned to IPMP Control Information Table is 0x03 as defined in ISO/IEC 13818-1. The IPMP Control Information table may be segmented into one or more sections, before insertion into Transport Stream packets, with the following syntax.

6.2.1.1 Syntax of IPMP Control Information Section in Transport Stream

Syntax	No. of bits	Mnemonic
IPMP_Control_Info_section() {		
table_id	8	Uimbsbf
section_syntax_indicator	1	Bslbf
'0'	1	Bslbf
Reserved	2	Bslbf
section_length	12	uimbsbf
Reserved	2	bslbf
IPMP_control_info_version	5	uimbsbf
current_next_indicator	1	bslbf
section_number	8	uimbsbf
last_section_number	8	uimbsbf
IPMP_control_info_classes_total_len	16	uimbsbf
IPMP_control_info_classes_len_in_this_section	16	uimbsbf
for (i=0; i<N;i++) {		
IPMP_control_info_class()		
}		
If (section_number==last_section_number) {		
IsSigned	1	bslbf
Reserved	7	bslbf
if (isSigned)		
Signature		ByteArray
NumCerts	8	uimbsbf
for (i=0; i<numCerts;i++) {		
CertType	8	uimbsbf
Certificate		ByteArray
}		
Verifying_Tool_ID	128	uimbsbf
}		
CRC_32	32	rpchof
}		

6.2.1.2 Semantics

table_id -- This is an 8 bit field, which shall be always set to 0x07.

section_syntax_indicator -- The section_syntax_indicator is a 1 bit field which shall be set to '1'.

section_length -- This is a twelve bit field. It specifies the number of bytes of the section, starting immediately following the section_length field, and including the CRC. The value in this field shall not exceed 4093.

IPMP_control_info_version -- This 5 bit field is the version number of the whole IPMP Control Information Table. The version number shall be incremented by 1 modulo 32 when a change in the information carried within the IPMP control info table occurs. When the current_next_indicator is set to '1', then the version_number shall be that of the currently applicable IPMP Control Information Table. When the current_next_indicator is set to '0', then the version_number shall be that of the next applicable IPMP Control Information Table.

current_next_indicator -- A 1 bit indicator, which when set to '1' indicates that the IPMP Control Information Table sent is currently applicable. When the bit is set to '0', it indicates that the IPMP Control Information Table sent is not yet applicable and shall be the next IPMP Control Information Table to become valid.

section_number -- This 8 bit field gives the number of this section. The section_number of the first section in the IPMP Control Information Table shall be 0x00. It shall be incremented by 1 modulo 256 with each additional section in the IPMP Control Information Table.

last_section_number -- This 8 bit field specifies the number of the last section (that is, the section with the highest section_number) of the IPMP Control Information Table.

IPMP_Control_Info_classes_total_len -- This 16 bit field specifies the total length of the IPMP Control Information Classes, which may be carried in more than one IPMP Control Information sections.

IPMP_Control_Info_classes_len_in_this_section -- This 16 bit field specifies the length of the IPMP Control Information immediately following this field in this section.

IPMP_Control_Info_class -- The IPMP Control Information Classes, including Tool List, Tool Container, Rights Container. Since the length of IPMP Control Information class could possibly exceeds 4093, it may split into more than one IPMP Control Information sections. When the IPMP Control Information sections are assembled back to form the complete IPMP Control Information Table, the IPMP Control Information classes can be assembled back as well, with help of IPMP_Control_Info_classes_total_len.

isSigned -- This 1 bit field indicates the presence of a signature in the IPMP Control Information table.

Signature -- The signature of the entire IPMP Control Information classes including Tool List, IPMP Rights Container and Tool Container. ByteArray is defined in [7.2.2.4].

CertType -- The type of certification mechanism being used, value assigned by a Registration Authority.

NumCerts -- The number of certificates included.

Certificate -- The array of certificates.

Verifying_Tool_Id -- The ID of the Tool that is required to verify the certificate(s). A value of 0 indicates the device.

CRC_32 -- This is a 32 bit field that contains the CRC value that gives a zero output of the registers in the decoder defined in Annex B in ISO/IEC 13818-1 after processing the entire IPMP Control Information section.

6.2.1.3 IPMP Control Information Classes

The IPMP Control Information includes the following IPMP Control Information Classes, with different IPMP Control Information class tags.

IPMP Control Information Class Tag	Class Name
0x00	Forbidden
0x01	IPMP Tool List Class
0x02	IPMP Tool Info Class
0x03	IPMP Tool Container Class
0x04	IPMP Rights Container Class
0x05	IPMP Parametric Description Class
0x06-0xC0	ISO/IEC 13818-1 Reserved
0xC1-0xFE	User private
0xFF	Forbidden

6.2.1.3.1 IPMP Tool List Class

The IPMP Tool List Class includes a list of IPMP tools. It is used to specify all IPMP tools that should be used in order to play back the content.

Syntax	No. of bits	Mnemonic
IPMPToolList () {		
Control_info_class_tag	8	uimsbf
Length	16	uimsbf
NumTools	8	uimsbf
for (i=0; i<numTools; i++) {		
IPMPTool_Info ()		
}		
}		

6.2.1.3.2 Semantics of fields in IPMP Tool List Class

Control_info_class_tag – this 8 bit field should be set to 0x01 to indicate this is an IPMP Tool List Class.

Length – number of bytes of this class, starting immediately following this “Length” field.

NumTools – This 16 bit field specified how many tools are specified in this IPMP Tool List Class.

IPMPTool_Info – This class carries information about a tool including its tool ID, possible alternate tools, etc. It is defined in the following subclause.

6.2.1.3.3 IPMP Tool Info Class

IPMPTool_Info_Class contains information for a logical IPMP Tool required by the Device. The logical tool may be one of the following:

- 1) A vendor-specific IPMP Tool specified by IPMP_ToolID,
- 2) One of a list of alternate IPMP Tools,
- 3) An IPMP Tool specified by a parametric description.

Syntax	No. of bits	Mnemonic
IPMPTool_Info () {		
Control_info_class_tag	8	uimsbf
Length	16	uimsbf
IPMP_ToolID	128	uimsbf
isAltGroup	1	uimsbf
isParametric	1	uimsbf
Reserved (0b111111)	6	uimsbf
if (isAltGroup) {		
NumAlternatives	8	uimsbf
for (i=0; i< numAlternatives; i++) {		
Specific_Tool_ID	128	uimsbf
}		
}		
if (isParametric) {		
IPMP_ParametricDescription		
}		
NUMURLS	8	uimsbf
ToolURL[numURLs]		ByteArray
}		

6.2.1.3.4 Semantic definitions of fields in IPMP Tool Info Class

Each IPMPTool_Info_Class identifies one IPMP Tool that is required by the terminal to consume the content. This Tool shall be specified either as a unique implementation, as one of a list of alternatives, or through a parametric description.

A unique implementation is indicated by the isAltGroup and isParametric fields both set to zero. In this case, the IPMP_ToolID shall be from the range reserved for specific implementations of an IPMP Tool and shall directly indicate the required Tool.

In all other cases, the IPMP_ToolID serves as a Content-specific abstraction for an IPMP Tool ID since the actual IPMP Tool ID of the Tool is not known at the time of authoring the Content, and will depend on the Terminal implementation at a given time for a given piece of Content.

A parametric description is indicated by setting the isParametric field to one. In this case, the Terminal shall select an IPMP Tool that meets the criteria specified in the following parametric description. In this case, the IPMP_ToolID shall be from the range reserved for Parametric Tools or Alternative Tools. The actual IPMP Tool ID of the Tool that the terminal implementation selects to fulfill this parametric description is known only to the Terminal. All the Content, and other tools, will refer to this Tool, for this Content, via the IPMP_ToolID specified.

A list of alternative Tools is indicated by setting the isAltGroup flag to "1". The subsequent specific ToolIDs indicate the Tools that are equivalent alternatives to each other. If the isParametric field is also set to one, any Tool that is selected under the conditions for parametric tools (as discussed in the paragraph above) shall be considered by the Terminal to be another equivalent alternative to those specified via specific ToolIDs. The Terminal shall choose one from these equivalent alternatives at its discretion. The actual IPMP Tool ID of this Tool is known only to the Terminal.

Control_info_class_tag – this 8 bit field should be set to 0x02 to indicate this is an IPMP Tool Info Class.

Length – number of bytes of this class, starting immediately following this "Length" field.

IPMP_ToolID – the identifier of the IPMP Tool.

isAltGroup – if set to one, this IPMP_Tool contains a list of alternate IPMP Tools.

numAlternates – the number of alternative IPMP Tools specified in IPMP_Tool.

Specific_Tool_ID – an array of the IDs of specific alternative IPMP Tools that can allow consumption of the content.

isParametric – IPMP_Tool contains a parametric description of an IPMP Tool. In this case, IPMP_ToolID is an identifier for the parametrically described IPMP Tool, and the Terminal shall route information specified in the bitstream for IPMP_ToolID to the specific IPMP Tool instantiated by the terminal.

IPMP_ParametricDescription -- IPMP Parametric Description as defined in the following subclause.

ToolURL – An array of numURL informative URLs from which one or more tools specified in this class may be obtained.

6.2.1.3.5 IPMP Parametric Description Class

Using a parametric description, the content provider can now describe what type of IPMP tool is required to playback the content, instead of using fixed tool IDs. For example, the content provider can specify that an AES tool, with block size of 128 bits is required to decrypt video stream. The IPMP terminal, upon receiving such description specifying this tool, can then choose an optimised AES tool from the embedded tools.

This clause contains an illustration of the hierarchy that a parametric description would follow. It does not attempt to define any specific scheme for any specific Tool type. It is anticipated that such definitions will be added over time to the overall scheme as a need is identified and an optimal schema developed. We anticipate that only a basic framework will appear in the current version of the specification, and enhancements to the same will be left for future addendums and/or versions.

- Optional comment
- Version of parametric description syntax
- Class of Tool
- e.g. Decryption, Rights Language Parser
- Sub-class of Tool
 - e.g. for Decryption: AES, DES, NESSIE etc
 - e.g. for Watermarking: “Panos’s watermarking tool” etc
 - e.g. for Rights Language Parser: “Fred’s Rights Parser”
 - e.g. for Protocol Parser: “Mary’s Protocol Parser”
- Sub-class-specific information
 - e.g. for DES: number of bits, stream and/or block decipher capability
 - e.g. for Rights Language Parser : version

The parametric description is defined to allow a generic description of any type of IPMP tool, no matter the type of tool.

Syntax	No. of bits	Mnemonic
IPMP_ParametricDescription () {		
Control_info_class_tag	8	uimsbf
Length	16	uimsbf
descriptionComment		ByteArray
majorVersion	8	uimsbf
minorVersion	8	uimsbf
numOfDescriptions	8	uimsbf
for (i=0; i< numOfDescriptions; i++) {		
class		ByteArray
subClass		ByteArray
typeData		ByteArray
type		ByteArray
addedData		ByteArray
}		
}		

6.2.1.3.6 Semantic definitions of fields in IPMP Tool Info Class

class - class of the parametrically described tool, for example, decryption.

subClass - sub-class of the parametrically described tool, for example, AES under decryption class.

typeData - specific type data to describe a particular type of tool, for example, Block_length, to further specify a AES decryption tool.

type - value of the type data above, for example, 128 for the Block_length.

addedData - Any additional data which may help to further describe the parametrically defined tool.

6.2.1.3.7 IPMP Tool ID

The IPMP Tool Identifier is 128-bits long, platform independent, and shall contain a unique identification number for the IPMP Tool. A registration authority for IPMP Tools that use a unique ID is required. The registration authority may further maintain an association of the download URLs for various implementations of the given tool for various platforms. These platforms will be described to adequate detail using a structured representation. The IPMP ToolID identifies a specific IPMP Tool, unless in the reserved range for parametrically defined tools or alternative tools. Specific values within this 128-bit space are reserved for indicating parametric tools, the bitstream, the terminal, and other special addresses. These values may not be assigned to registered Tools.

IPMP_ToolID	Semantics
0x0000	Forbidden
0x0001	Content
0x0002	Terminal
0x0003 - 0x2000	Reserved for ISO use
0x2001 - 0xFFFF	Carry over from 14496-1 RA
0x10000 - 0x100FF	Parametric Tools or Alternative Tools
0x100FF - 2 ¹²⁸ -2	Open for registration
2 ¹²⁸ -1	Forbidden

6.2.1.3.8 IPMP Tool Container Class

IPMP Tool Container Class enables carriage of IPMP tool inside MPEG-2 system. The device may retrieve the IPMP tool from the content stream, load it, instantiate it and immediately use it in order to play out the content.

One implementation of a given tool is carried as the payload of one IPMP Tool Container Class, the representation format, packaging information and IPMP Tool ID of which is also specified in the container.

Syntax	No. of bits	Mnemonic
IPMPToolContainer() {		
Control_info_class_tag	8	uimsbf
Length	16	uimsbf
IPMP_ToolID	128	uimsbf
Tool_Format_ID	32	uimsbf
Tool_Package_ID	16	uimsbf
SizeofTool	16	uimsbf
for (i=0; i<sizeofTool; i++) {		
Toolbody	8	uimsbf
}		
}		

Handling and usage of Tools carried in Tool Container Class is implementation dependant.

6.2.1.3.9 Semantic definitions of fields in IPMP Tool Container Class

Control_info_class_tag – this 8 bit field should be set to 0x03 to indicate this is an IPMP Tool Container class.

Length – number of bytes of this class, starting immediately following this “Length” field.

IPMP_ToolID – the ID of the Tool carried in this stream.

Tool_Format_ID - This is defined as 0x0001 for a structurally described tool. Otherwise, the Tool_Format_ID indicates the Binary Representation of the Tool and is assigned by a registration authority.

NOTE A structurally described tool implies a description of the IPMP Tool in terms of a network of primitives that can be combined to provide some or all IPMP functionalities required for content consumption. For example, a DES decryption algorithm could be described as a sequence of opcodes calls receiving the ciphertext as input and providing the plaintext as output.

Tool_Package_ID indicates the details of the package of the Tool – examples are CAB or a Winzip self-install executable. Values are assigned by a registration authority.

6.2.1.3.10 IPMP Rights Container Class

IPMP Rights Container Class conveys the usage rules & states associated with the IPMP protected content.

Syntax	No. of bits	Mnemonic
IPMP_Rights_Container () {		
Control_info_class_tag	8	uimbsf
Length	16	uimbsf
rights_data		ByteArray
}		

6.2.1.3.11 Semantic Definitions of Fields in IPMP Rights Container Class

Control_info_class_tag – this 8 bit field should be set to 0x04 to indicate this is a IPMP Rights Container.

Length – number of bytes of this class, starting immediately following this “Length” field.

rights_data – This contains the details of usage rights information.

6.2.2 IPMP Control Information in Program Stream

In Program Stream, the IPMP Control Information is present as one or more PES packets where the stream_id is extended_stream_id 0xFD and stream_id_extension of 0b000 0000, as defined in ISO/IEC 13818-1.