ETSI

ETSI TECHNICAL REPORT

ETR 313

October 1996

Source: ETSI TC-TE

Reference: DTR/TE-01070

ICS: 33.020

Key words: MMI, multimedia, retrieval, terminal, videotex, VEMMI

Terminal Equipment (TE); End-to-end protocols for multimedia information retrieval services; Interworking between Digital Audio Visual Council (DAVIC) compliant broadband systems and Videotex systems

ETSI

European Telecommunications Standards Institute

ETSI Secretariat

Postal address: F-06921 Sophia Antipolis CEDEX - FRANCE
Office address: 650 Route des Lucioles - Sophia Antipolis - Valbonne - FRANCE
X.400: c=fr, a=atlas, p=etsi, s=secretariat - **Internet:** secretariat@etsi.fr

Tel.: +33 4 92 94 42 00 - Fax: +33 4 93 65 47 16

Contents

Blank page

Foreword

This ETSI Technical Report (ETR) has been produced by the Terminal Equipment (TE) Technical Committee of the European Telecommunications Standards Institute (ETSI).

ETRs are informative documents resulting from ETSI studies which are not appropriate for European Telecommunication Standard (ETS) or Interim European Telecommunication Standard (I-ETS) status. An ETR may be used to publish material which is either of an informative nature, relating to the use or the application of ETSs or I-ETSs, or which is immature and not yet suitable for formal adoption as an ETS or an I-ETS.

Blank page

1 Scope

The purpose of this ETSI Technical Report is to investigate the possibility of interworking between Digital Audio Visual Council (DAVIC) compliant broadband services and Versatile MultiMedia Interface (VEMMI) based information retrieval services on a technical level. The following interworking scenarios are considered:

- access from a Set Top Unit (STU) in a DAVIC system to VEMMI applications;
- access from a VEMMI terminal in a VEMMI-system to a Multimedia/Hypermedia Experts Group (MHEG)-5 application in a DAVIC system.

The document first introduces both system architectures (DAVIC, Videotex) briefly as it is mainly targeted to experts in either one of the domains.

NOTE: Readers being familiar with DAVIC may skip the introductory part to DAVIC-compliant systems. Readers being familiar with Videotex may skip the introductory part to Videotex systems.

However, both areas are described briefly and key technology is introduced.

2 References

This ETR incorporates by dated or undated reference, provisions from other publications. These references are cited at the appropriate places in the text and the publications are listed hereafter. For dated references, subsequent amendments to or revisions of any of these publications apply to this ETR only when incorporated in it by amendment or revision. For undated references the latest edition of the application referred to applies.

[1] DAVIC 1.0 Specification Part 01 to Part 12.

[2] ETR 173 (1995): "Terminal Equipment (TE); A Functional model for multimedia applications".

[3] ETS 300 709 (1996): "Terminal Equipment (TE); Enhanced Man Machine Interface for Videotex and Multimedia/Hypermedia Information Retrieval Services".

[4] ISO/IEC DIS 13522–5: "Information Technology - Coding of Multimedia and Hypermedia Information - Part 5: Support for Base-Level Interactive Applications".

[5] ITU-T Recommendation H.220.0 (1995)/ISO/IEC 13818-1 (1995): "Information technology - Generic coding of moving pictures and associated audio information - Part 1: Systems (MPEG-2)".

[6] ISO/IEC DIS 13818–6, "Information technology - Generic coding of moving pictures and associated audio - Part 6: Digital Storage Media - Command and Control (DSM-CC)".

[7] Common Object Request Broker: Architecture and Specification, Revision 2.0 (July 1995) (OMG CORBA 2.0).

[8] ISO/IEC 14750-1: "Information technology - The Interface Definition Language (IDL)".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of this ETR, the definitions in DAVIC 1.0 [1], ETS 300 709 [3], ISO/IEC 13522-5 [4] and ISO/IEC 13818-6 [6] apply.

3.2 Abbreviations

For the purposes of this ETR, the following abbreviations apply:

A0-A11	DAVIC Interface Location
AAL	ATM Adaptation Layer
AAL5	ATM Adaptation Layer Five
ADSL	Asynchronous Digital Subscriber Line
AP	Access Point
API	Application Programming Interface
ARP	Address Resolution Procedure
ASN.1	Abstract Syntax Notation One
ATM	Asynchronous Transfer Mode
BER	Basic Encoding Rules
BIN	Bitmap Identification Numbers
BRI	Basic Rate Interface
BUS	Broadcast and Unknown Server
CCITT	Comité Consultatif International Télégraphique et Téléphonique
CD	Committee Draft
CD	Compact Disc
CD-I	Compact Disc-Interactive
CDR	Common Data Representation
CIN	Component Identification Number
CLUT	Colour Look-Up Table
CORBA	Common Object Request Broker Architecture
CRC	Cyclic Redundancy Check
DAB	Digital Audio Broadcasting
DAVIC	Digital Audio Visual Council
DDA	Defined Display Area
DIS	Draft International Standard
DNS	Domain Name Server
DSM	Digital Storage Media
DSM-CC	Digital Storage Media - Control Commands
DTR	Draft Technical Report
DVB	Digital Video Broadcasting
EBNF	Extended Backus Naur Form
ESC	End Service Consumer
ESP	End Service Provider
FIN	Font Identification Number
FTP	File Transfer Protocol
FTTC	Fibre To The Curb
GUI	Graphical User Interface
HFC	Hybrid Fibre Coax
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
IAG	Internet Access Gateway
IDL	Interface Definition Language
IEC	International Electrotechnical Commission
IETF	Internet Engineering Task Force
IIOP	Internet Inter-ORB Protocol
IP	Internet Protocol
ISDN	Integrated Services Digital Network
ISO	International Organization For Standardization
ISP	Internet Service Provider
L1GW	Level One Gateway
LAN	Local Area Network
LES	LAN Emulation server
LLC	Logical Link Control
MAC	Medium Access Control
MHEG	Generic Reference to ISO 13522
MHEG-3	Generic Reference to ISO 13522-3
MHEG-5	Generic Reference to ISO 13522-5
MIN	Multimedia Identification Number

MIRS	Multimedia Information Retrieval Service
MOD	Movies On Demand
MP	Multilink PPP
MPEG	Moving Picture Experts Group
MPEG1	Generic reference to ISO
MPEG2	Generic reference to ISO 13818
MPOA	Multiprotocol over ATM
MUX	Multiplex
NHRP	Next Hop Resolution Protocol
NIU	Network Interface Unit
NVoD	Near Video on Demand
OIN	Object Identification Number
OMG	Object Management Group
ORB	Object Request Broker
OSI	Open Systems Interconnection
OSPF	Open Shortest Path First
PES	Packetized Elementary Stream
PC	Personal Computer
PCM	Pulse Code Modulation
PDH	Plesiochronous Digital Hierarchy
PID	Packet Identification
PID	Programme Identification
PIN	Personal Identification Number
PPP	Point to Point Protocol
PS	Program Stream
PSTN	Public Switched Telephone Network
RDP	Route Distribution Protocol
RFC	Request for Comment (Internet Standard)
ROM	Read Only Memory
RP2	Reference Point Two
RPC	Remote Procedure Call
RTE	Runtime Engine
S1-S4	DAVIC information flows
SBV	Syntax Based Videotex
SDH	Synchronous Digital Hierarchy
SIR	Script Interchange Representation
SL0	Principal Service Layer Identifier
SL1	Application Service Layer Identifier
SL2	Session and Transport Service Layer Identifier
SL3	Network Service Layer Identifier
SONET	Synchronous Optical NETwork
SRM	Session and Resource Manager
STB	Set Top Box
STC	Sub-Technical Committee
STU	Set Top Unit
TCP	Transmission Control Protocol
TE	Terminal Equipment
TIN	Text Identification Number
TS	Transport Stream
TV	Television
UCS	Universal Multiple-Octet Character Set
UDP	User Datagram Protocol
UN	User to Network
UNO	Universal Networked Objects
VDSL	Very high speed Digital Subscriber Line
VEMMI	Generic reference to ETS 300 709 (Versatile MultiMedia Interface)
VIN	Videotex Identification Number
VOD	Video On Demand
VPDE	Videotex Presentation Data Element
VSG	VEMMI Service Gateway
VTX	Videotex
WWW	World Wide Web

4 DAVIC Systems

NOTE: The relevant DAVIC reference documents for the purpose of this ETR are the DAVIC 1.0 Specifications, Part 01 - Part 12 [1].

In general DAVIC does not specify systems but tools that can be used to build DAVIC compliant systems. Systems in general tend to be application-specific. DAVIC-specified tools tend to be non-"system-specific" because they have to be usable by different industries in different systems.

Another main concept of DAVIC is "one functionality - one tool". To solve a given problem there is only one tool specified. This principle gives substantial benefits in terms of interoperability and availability of technology thanks to the easier achievement of a critical mass because of a wider field of applicability of the technology.

A DAVIC compliant system consists of the following major components:

- a DAVIC server;
- a server access network;
- the core network;
- the access network;
- a Network Interface Unit (NIU);
- a Set Top Unit (STU).

A DAVIC server is the host for contents and applications. It describes its physical organization through the use of an interface defined in Interface Definition Language (IDL). The interface is defined in Digital Storage Media - Control Commands (DSM-CC), User-to-User. The interface describes the file hierarchy as well as the access methods. A server can be a host for a Session Gateway, Services, Applications stored as files containing multimedia objects and real-time content information organized as streams.

A DAVIC compliant server access network and core network uses Asynchronous Transfer Mode (ATM) technology. The ATM Adaptation Layer selected in DAVIC is Adaptation Layer 5 (AAL 5). The recommended physical interface to carry ATM cells is Synchronous Digital Hierarchy (SDH)/Synchronous Optical NETwork (SONET) (155,52 Mbit/s or 622,08 Mbit/s). However, Plesiochronous Digital Hierarchy (PDH) plays nowadays an important role in some existing networks. For that reason PDH was identified as possible short term solution.

To map ATM cells to the physical medium in the Access Network the following methods have been identified:

- Asynchronous Digital Subscriber Line (ADSL);
- Very high speed Digital Subscriber Line (VDSL);
- Fibre To The Curb (FTTC);
- Hybrid Fibre Coax (HFC);
- symmetrical Physical Layer interfaces for copper pairs and fibre optic cables.

In addition Herzian access network interfaces (satellite), Integrated Services Digital Network (ISDN) and Public Switched Telephone Network (PSTN) interfaces for return channels have been identified and profiled.

A DAVIC compliant NIU abstracts the specificities of a given access network to the users terminal equipment. It offers the so-called DAVIC interface Location A0 interface that allows a user to connect a network independent STU. It provides a standardized quality of communication parameters (de-jittering, etc.) and may also contain access control functions. Following the relocation of tools principle, the NIU can be integrated with the STU to form a network dependant Set Top Box (STB).

The basic components of a DAVIC compliant STU are:

- a Network independent interface (A0);
- MPEG2 facilities allowing to access and decode MPEG2 datastreams;
- DSM-CC User-to-Network and DSM-CC User-to-User protocol stacks;
- a MHEG-5 multimedia engine;
- an interface to the TV-set at Reference Point 2 (RP2).

The following example of a DAVIC system is used for the purpose of this document. The system is conformant to DAVIC Profile 2 (see subclause 4.3.2). The figure representing the system structure and information flows is more detailed then necessary, especially for the lower layers. Nevertheless it is useful to see a DAVIC system with all its components. The different information flows are handled by different functional units in a STB. The reference architecture for a Terminal Function (STB) is introduced in ETR 173 [2]. The figure shows how the functional model of ETR 173 [2] maps on this DAVIC model.

Figure 1 shows the assignment of different information flows to different functional units in the terminal function. This assignment is not mandatory for a STB but a STB implementation could reasonably have a similar assignment and consequently a similar functional architecture.

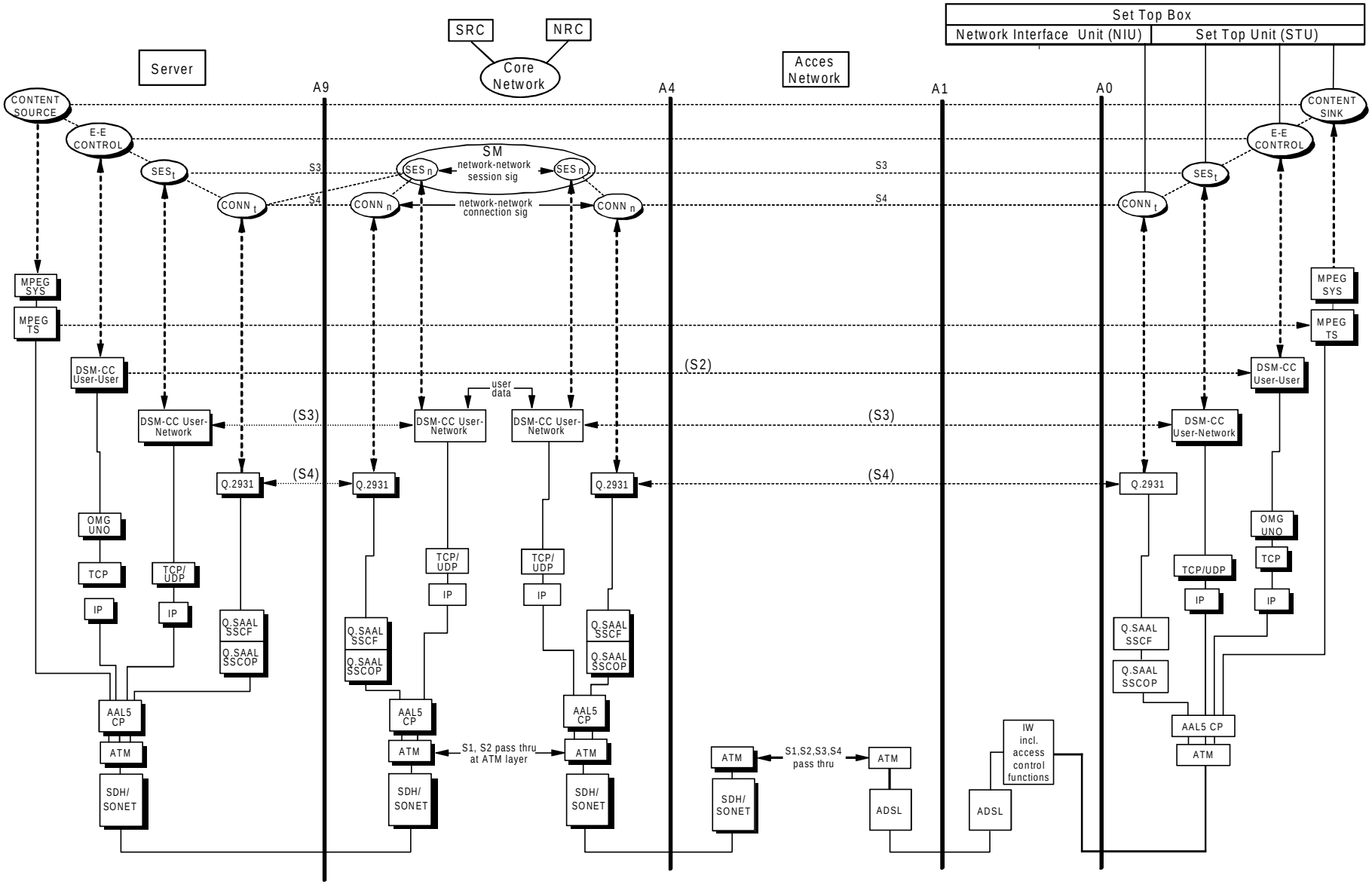


Figure 1: DAVIC system example

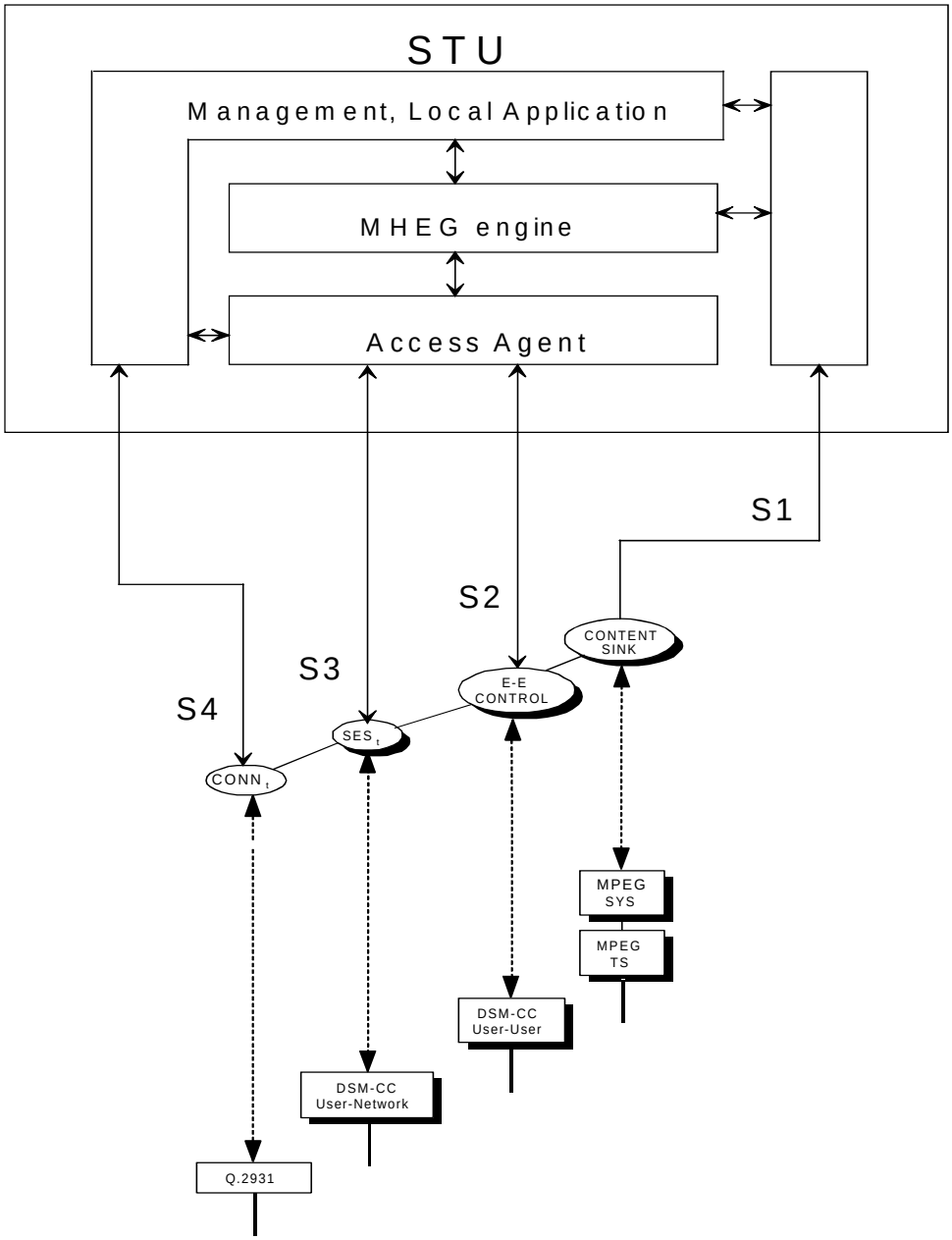


Figure 2: Assignment of the information flows to functional entities of the STU

4.1 Information flows

Figure 1 identifies the information flows in a DAVIC compliant environment. Figure 2 shows the functional units within a MHEG-engine that are responsible for handling the information flows. The following information flows are identified:

- S1, content-information flow;
- S2,S3,S4 control-information flows.

All information flows are carried over ATM and the ATM Adaptation Layer 5 (AAL5). In this example in the core network the ATM cells are mapped on a physical network using SDH technology. In the access network the ATM cells are mapped on a physical network using ADSL technology.

4.1.1 S1 information flow

S1 is content-information flow from a source object to a destination object, normally in the User Plane. The flow is transparent to any intermediate object through which the flow passes. The flow does not appear to alter the behaviour of the information source and destination objects. e.g., audio, video, or data transferred from an End Service Provider (ESP) to an End Service Consumer (ESC). In the STU the S1

information flow is handled by the presentation agent. It is responsible for rendering video and playing audio information. The S1 information flow is unidirectional (from server to terminal) and implemented using Moving Picture Experts Group (MPEG) Transport Stream (TS).

4.1.2 S2 information flow

S2 is control-information flow, normally in the User Plane, from an Application Service Layer source object to a peer destination object. The flow is transparent to any intermediate object through which the flow passes (it is an S1-category information flow as far as the intermediate object is concerned). In the usual case, the end-to-end information source and destination objects are Control Plane objects.

The behaviour of the source or destination object may change as a result of the flow. For example:

- an application may understand and accept a request to perform an operation that results in a change in its state, or
- it may reject a message because it does not understand the flow, or is unable to execute a service request for some reason. The destination object may reject the flow silently, i.e., without informing the source object that anything is wrong, and return to its pre-message state, or
- it may reject a message for the reasons above, and let the source know that its message was rejected (and perhaps why).

Initializing or reconfiguring a downloaded application, while it is idle, is an example S2 flow that changes the state of an application to enable it to provide a service to an SL0 client. Once the application is serving a client, S2 messages may also affect S1 information flows such as messages to Play or Stop a movie.

The S2 information flow is handled by the Access Agent. The Access Agent issues requests for files and the stream control primitives on behalf of the MHEG engine (DSM-CC - MHEG5 mapping). The S2 information flow consists of DSM-CC User-to-User commands compiled down to protocol primitives using the rules defined in Object Management Group (OMG) Universal Networked Objects (UNO) on a Transmission Control Protocol (TCP)/User Datagram Protocol (UDP)-Internet Protocol (IP) protocol stack.

4.1.3 S3 information flow

S3 is control-information flow, normally in the Control Plane, from a Session and Transport Service Layer source object to a peer destination object. Other than the change of service layer, S3 is similar to S2. Examples of S3 flows are messages to establish, modify, or terminate a session; negotiate resource requirements, and report exceptions.

The S3 information flow is handled by the Access Agent. The Access Agent issues requests for session establishment either on behalf of the local application or on behalf of the MHEG engine (DSM CC - MHEG5 mapping). The S3 information flow also supports resource negotiation and all other functionality required to set up and manage a session between two entities. The S3 information flow consists of DSM-CC User-to-Network commands on a TCP/UDP-IP protocol stack.

4.1.4 S4 information flow

S4 is control-information flow, normally in the Control Plane, from a Network Service Layer source object to a peer destination object. Other than the change of service layer, S4 is similar to S2 and S3. Examples of S4 information flows include messages to establish or release connections, communicate addresses, port information, and other routeing data. The S3 information flow is handled by the local application. It consists of ITU-T Recommendation Q.2931 (see annex A) protocol elements.

The following subclauses present first the two main profiles of the DAVIC 1.0 Specifications and second the application-oriented key technologies and their relevance for interworking.

4.2 Assumptions on the service architecture and on the execution model

The following tasks are performed by different functional units during the execution of a Multimedia Application in a DAVIC compliant system.

1) After the STB is switched on, the local applications process starts and initializes the STB resident software.

2) The local application process offers a local system menu using the services of the presentation agent. Alternatively the local application process could itself be an MHEG-5 application. This application could either be expressed using the DAVIC high level API (MHEG-5 objects) or it could make use of the services offered by the MHEG engine in a STB dependent manner. The latter case should be closer investigated in the MHEG-5 execution model.

3) The user selects now one of the services offered by the local application on the STB. These services could be pre-installed by the user or the STU- manufacturer. The list of services should be editable. The advantage of implementing the local system application using the high level API would be that it could be updated remotely and it could be adapted to the look-and-feel from a service provider. From a human factors point of view this is a very desirable feature because it allows the use of the same user interface for service applications and local applications.

4) When the user triggers the connection establishment, the local application should establish a session with the service gateway. If the local application is a MHEG-5 application a mapping of MHEG-5 actions on the DSM-CC-session establishment procedure (Primitive and all necessary parameters should be mapped) should be provided. ETS 300 777-2 (see annex A) should provide the required mapping.

   NOTE: It should also be checked how a generic way to invoke DSM-CC functions from an MHEG environment could be provided using the MHEG Script Interchange Representation (SIR). However, as scripting language support is not a mandatory feature of a DAVIC 1.0 system and as it is not decided yet that MHEG SIR will be the selected scripting language, an alternative option to access the DSM-CC environment should be studied.

5) Once the terminal has established a session with the service gateway, the terminal should retrieve the MHEG object representing the root of the MHEG application. There should be a naming convention for the file representing the root object of the service application and it should be specified in the execution model of the MHEG-5 engine that after session establishment the MHEG-5 engine should "prepare" the first object of an application (this will result in a DSM-CC file read primitive).

6) When the MHEG-5 application of the Service Gateway is running, the user can select the desired service or application. The DSM-CC MHEG-5 mapping should provide a way to map MHEG-5 actions to DSM-Service and application selection methods.

7) The user selects a service and the main application within the service is started.

4.3 Main DAVIC 1.0 profiles

Profiles are a set of functions supporting a range of similar applications. DAVIC identifies 3 main profiles. Profile 1 and Profile 2 are covered in the DAVIC 1.0 specification. Profile 3 (Conference Profile) will be covered in future versions of the DAVIC specifications. The following subclauses introduce Profile 1 and Profile 2.

4.3.1 Profile 1 (distribution profile)

This profile supports a range of applications which are characterized by a point-to-multipoint configuration and by the fact that the destination user, e.g. the end-user receiving the information, or the information sink cannot control the flow of information itself. The following example applications can be supported by this profile:

- Near Video on Demand (NVoD);
- broadcast;
- delayed broadcast;
- games;
- TV listings.

Other applications may also be supported by this profile. A DAVIC Broadcast system can interwork with Videotex services. A detailed technical description and the set of constraints and requirements are presented in subclause 7.2.

4.3.2 Profile 2 (retrieval profile)

This profile supports a range of applications which are characterized by a point-to-point configuration and by the fact that the destination user, e.g. the end-user receiving the information, is provided with the necessary tools to control the flow of information. The following example applications can be supported by this profile:

- MoD;
- teleshopping;
- delayed broadcast;
- games;
- telemedicine;
- internet access;
- news on demand;
- virtual CD ROM;
- TV listings;
- distance learning;
- homebanking;
- Karaoke on demand;
- transaction services.

Other applications may also be supported by this profile. A DAVIC retrieval system can interwork with Videotex services. A detailed technical description and the set of constraints and requirements are presented in clause 7.

4.4 Key technology

The following subclauses describe the key technology used in a DAVIC system.

4.4.1 MHEG-5

NOTE: The relevant MHEG-5 reference document for the purpose of this ETR is ISO/IEC DIS 13522-5 [4].

MHEG-5 was developed to support the distribution of interactive multimedia applications in a client/server architecture[1)] across platforms of different types and brands. ISO/IEC DIS13522-5 [4] defines a final-form representation for application interchange. The applications consist mainly of declarative code, but provisions for calling procedural code have been made. MHEG-5 applications need only be authored once and can run on any platform that is MHEG-5 compliant. The developed applications would reside on the server, and as portions of the application are needed, they will be downloaded to the client. In a broadcast environment, this download mechanism could rely, for instance, on cyclic rebroadcasting of all portions of the application. It is the responsibility of the client to have a runtime that interprets the application parts, presents the application to the user, and handles the local interaction with the user.

The applications are run on the client. The server is only downloading required parts of the application on request. There is no MHEG engine running MHEG applications on the server.

The differences between VEMMI and MHEG-5 are highlighted in clause 6.

---

1) The server can be a *virtual* server, such as the collection of a number of broadcast channels on a broadcast network.

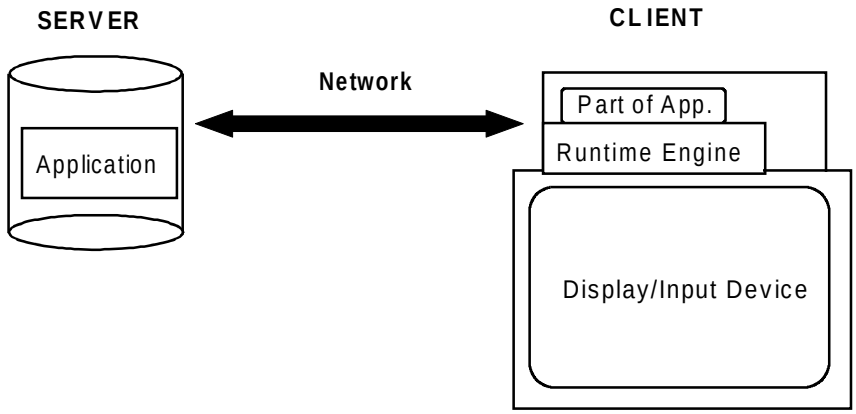


Figure 3: MHEG Application on server with part of the application downloaded to the client in a client/server configuration

An MHEG-5 application is made up of scenes and objects that are common to all scenes. A scene contains a group of objects used to present information (graphics, sound, video, etc.) along with localized behaviour based on events firing (e.g. the left button being pushed activates a sound). At most, one scene is active at any one time. Navigation in an application is done by making transitions between scenes.

The interactive system has the ability to display visual objects in a rectangular co-ordinate system with a fixed size, and to play audible objects. User input devices (e.g. remote control, game controller, etc.) can be used with the runtime to allow interaction with the applications.

MHEG-5 provides the syntax and semantics of a class structure for building presentation-oriented applications. An instance of an MHEG class is called an MHEG object.

The figures in this clause depict the class diagram of the object classes defined by MHEG-5. Their meaning is explained below.

The next part of this overview introduces the concepts defined by MHEG-5 by explaining the classes shown in figure 5 and their subclasses. For MHEG-5 classes that have similar semantics to VEMMI presentation objects, this is introduced.

NOTE: As VEMMI is based on presentation objects rather than on object oriented technology, abstract MHEG-5 classes do not have an equivalent in VEMMI.

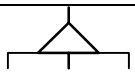
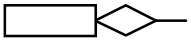
Link	The boxes depict an MHEG-5 class. Class names in bold depict a concrete class.
Visible	Class names in normal style depict an abstract class.
	Triangles depict inheritance relationships.
	Diamonds depict composition relationships.
	Dark circles depict a zero-to-more relationship.

Figure 4: Legend of class diagrams

Figure 4 presents an overview of the top layer in the MHEG-5 class hierarchy. The next part of this clause introduces the concepts defined by ISO/IEC DIS 13522-5 [4] by explaining the classes shown in this figure and their subclasses.

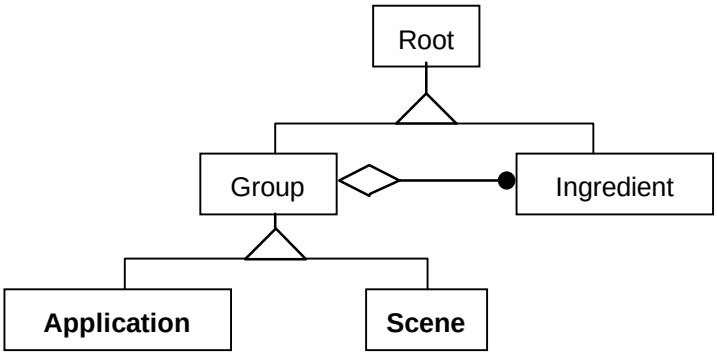


Figure 5: Diagram of the top of the class hierarchy

4.4.1.1 Root

This is the abstract base class for all other MHEG-5 classes[2)]. Its main functionality is to provide semantics for generic MHEG-5 behaviour (activation, deactivation, preparation, destruction), and to provide a mechanism for object identification.

4.4.1.2 Group

This is an abstract base class for the Application and Scene classes. Its main functionality is that of allowing the grouping of objects of other classes for exchange between the MHEG-5 engine and other entities (similar to a "set" in standard object-oriented terminology). The objects that are grouped by this class are objects of the class Ingredient. Each Ingredient is always contained in exactly one Group. Objects within a Group may be referenced from objects in other Groups under certain conditions (see below).

4.4.1.3 Application

Objects of the Application class group objects of the Ingredient class. The Application class also has the semantic constraint that only one Application object may be active at once, and that no other objects may be active unless an Application object is active.

An idle MHEG-5 engine starts an application by preparing and activating the corresponding Application object. When the Application object becomes active, it automatically runs an OnStartUp action, which can be used to run the first Scene object of the application. Since (exactly) one Application object is active whenever another object is active, the Ingredients contained in the Application object are visible and available to other objects that are active simultaneously. More specifically, any Ingredients contained in an Application object are available to all Scenes that are active simultaneously. This can be used to describe application-wide behaviour.

4.4.1.4 Scene

Objects of the Scene class group objects of the Ingredient class. The purpose of the Scene class is to allow for spatially and/or temporally co-ordinated presentation. Only one Scene object may be active at a time within an MHEG-5 engine.

The Scene class provides the special action TransitionTo, which makes it possible to perform a graphical transition between two scenes. Objects of the Scene class can also "share" the objects that they contain with other scenes; this allows for uninterrupted presentation of those objects over scene transitions. Finally, the Scene class provides information about the co-ordinate system to be used for visual presentation.

4.4.1.5 Ingredient

The Ingredient class is an abstract base class for the Link, Procedure, Palette, Font, CursorShape, Variable and Presentable classes. The subclasses of the Ingredient class are presented in the following

---

2) With the exception of the abstract mix-in classes and of the Action class.

figure. The main functionality of the Ingredient class is to specify the generic behaviour of objects that can be part of a Scene or an Application object.

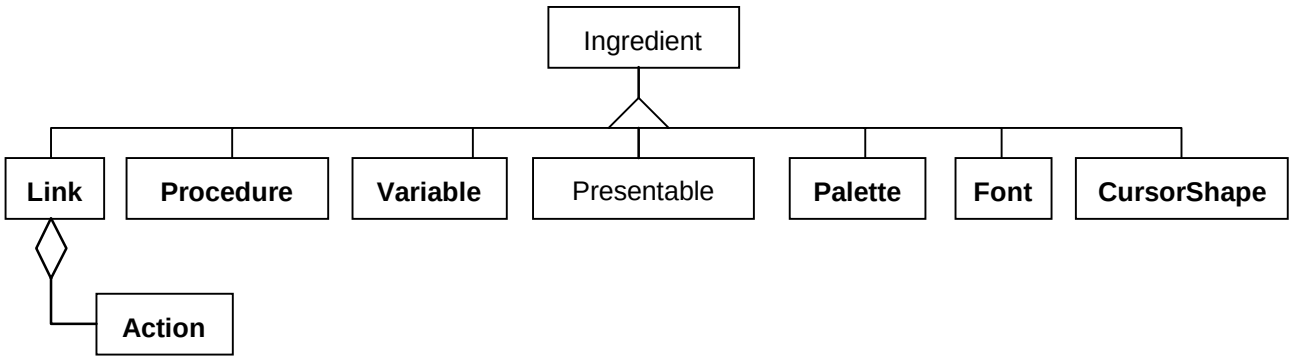


Figure 6: Diagram of subclasses of the Ingredient class.

4.4.1.6 Link

Link objects are used to express the behaviour of MHEG-5 applications. A Link object consists of a condition and an Action object. When the condition part evaluates to True, the Link is said to "fire"; this leads to the running of the associated Action object. The condition contains three parts: an event code (identifying which event has to be reacted on), a reference to the object from which the event should emanate, and a value that specifies the required value of the event parameter. In other words, a Link fires only if it is active and only if the right event is generated by the right object and contains the right event parameter. Active Links should be part of either an active Scene or an active Application.

4.4.1.7 Action

An Action object has the functionality of executing, in synchronous sequence, a series of "elementary actions" as the result of a Link firing. An elementary action consists of the object to which the action is to be "targeted" and a list of values representing the parameters of the action. In fact, targeting an elementary action to an object corresponds to calling a method of an object in any ordinary object-oriented programming language.

The Action class does not inherit from any other MHEG-5 class. Specifically, it does not inherit from Root, which means that Action objects cannot be addressed as individual entities.

4.4.1.8 Procedure

The Procedure class provides the functionality of calling a piece of procedural code from within the MHEG-5 context and exchanging parameters with it.

Three types of procedural calls can be made:

- to a piece of code that is specific to the device on which the MHEG-5 engine is running. This is called a "custom" procedure; it can be used, for instance, to call device-specific runtime libraries;

- to a piece of code that is located on a device different to the one where the MHEG-5 engine is running. This is called a "remote" procedure; it can be used, for instance, to implement a remote procedure call, where the actual body of the procedure is located at the server in a client-server system;

- to a piece of code which is exchanged as a part of an MHEG-5 object. This is called a "script" procedure. The purpose is to allow for the exchange of pieces of procedural code, and for calling those pieces of code.

  NOTE: The term "script" does not imply that the procedural code shall be written in a traditional scripting language.

4.4.1.9 Palette, font, and cursor shape

The Palette class provides the possibility to encapsulate the encoded representation of a Colour Look-Up Table (CLUT). The function of a CLUT is to translate a colour index to a true colour value. A Palette may be used, for instance, with bitmaps to specify the colours in which the bitmap is to be rendered.

Similarly, the Font class allows applications to encapsulate the encoded representation of a font. A Font object, when associated with a Text object, is used to render the text of that object.

The CursorShape class, finally, allows applications to encapsulate the encoded representation of the bitmaps, mask, and other data needed to render a free-moving cursor. The free-moving cursor shape can be set and reset using a method of the Scene class.

For Fonts, Palettes and CursorShapes, the actual representation of the objects is not specified by ISO/IEC DIS 13522-5 [4]. However, an application area where fonts and/or CLUTs and/or free-moving cursors are necessary features of applications can specify their encoding and semantics.

4.4.1.10 Variable

The Variable class provides the possibility to store and retrieve values. Objects of the Variable class are MHEG-5 objects to which the SetVariable and TestVariable actions can be targeted. Possible uses of Variables include parameter passing to and from Procedure calls, storage of the state of other MHEG-5 objects, passing indirect parameter values to actions and address indirection, i.e. as pointers to MHEG-5 objects.

4.4.1.11 Presentable

The Presentable class is an abstract base class for the Audio, Visible, TokenGroup, TemplateGroup, and Stream MHEG-5 classes. The subclasses of the Presentable class are shown in Figure 7.

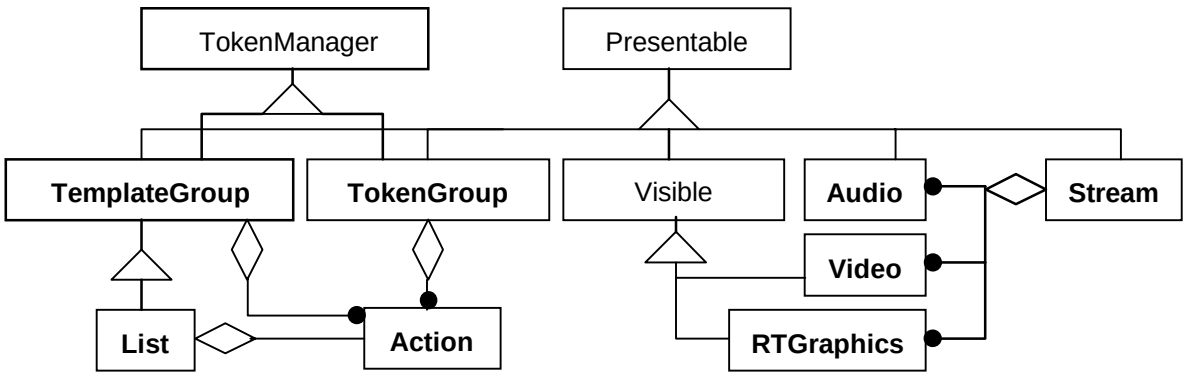


Figure 7: Diagram of subclasses of the Presentable class.

Objects of the Presentable class represent information that can be directly seen or heard by the user. An important functionality of the Presentable class is to allow for the (optional) handling of the actual encoded representation of the content data. This can be done either by "inclusion" or by "reference". In the former case, the content data is actually transmitted as part of the Presentable object itself; in the latter case, the Presentable object merely provides an external reference to the data.

4.4.1.12 Token group

The TokenGroup class provides the facility to navigate a logical token among a set of Visible objects. This structure can be used, for example, to manage the navigation of a "focus" among a set of buttons or other elements of a Scene. Some sets of actions may also be attached to the objects and executed on demand on the visible object that has the token. The latter feature provides a compact way of expressing behaviour when an element of the group gets or looses the focus.

4.4.1.13 Template group and list

The TemplateGroup and the List classes provide tools to navigate among a set of items that share the same structure and the same behaviour. These data structures might be used to implement a menu for selection, a group of checkboxes, a carrousel of bitmaps, a fill-in form, a scrollable list of items, etc.

The TemplateGroup class is composed of six major elements:

- a Template data structure that defines a model for the *Items;*

- a set of *Items*, that are instantiations of the Template, created by filling in content data;

- a set of CellPositions, that are positions in the Scene co-ordinate system. Each visible *Item* is displayed at one CellPosition;

- a single token that is distributed among the cells and can be used to implement for instance a focus behaviour;

  a number of Action objects that are automatically executed when an item enters or leaves a specific state (item at cell holding token, item selected).

In addition, the List class provides facilities to add and remove items to/from the group, as well as scrolling functions.

4.4.1.14 Stream

The Stream class defines a multiplex of continuous media for synchronization in time. Audio, Video and RTGraphics objects might be elementary streams of a Stream multiplex: they are intended to be presented at the same time to the user. This structure can be used, for instance, to present video and audio synchronously and to switch from one audio channel to another. The content data of the Stream object is a reference to a real multiplex containing the elementary streams and some additional data for synchronization. During the rendering process, the Stream player generates time-based events and marker-based events that might be used by the MHEG-5 application to trigger some Links.

4.4.1.15 Audio

The Audio class implements a sequence of audio data that can be used as an elementary stream of a Stream multiplex or directly as any Ingredient.

4.4.1.16 Interactible

The Interactible class is an abstract mix-in class inherited by the MHEG-5 classes HyperText, EntryField, Slider, and Button. Its main functionality is that of allowing the user to interact with objects of its sub-classes. These interactions enable the user to change the status and/or appearance of the objects, for instance by entering text in an EntryField object. When interaction is taking place, a certain class of events, called the "UserInput events" are not visible to Link objects, since those events are assumed to be used for the user interaction. The interaction can be aborted by targeting a certain action at the Interactible object.

Another functionality of the Interactible class is the ability to generate events associated with free-moving cursors (CursorEnter, CursorLeave).

4.4.1.17 Visible

The Visible class implements the functionality associated with rendering pieces of visual material at some location on the display screen. The subclasses of the Visible class are presented in Figure 8.

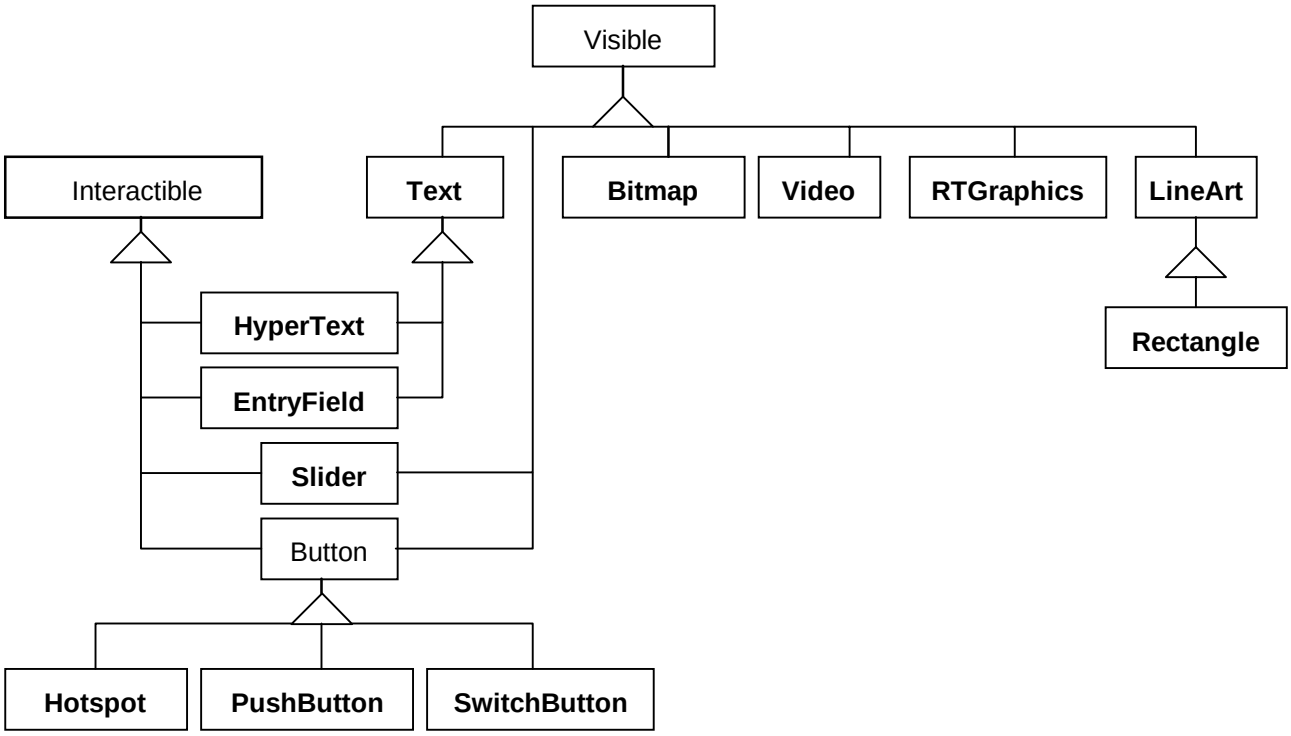


Figure 8: Diagram of subclasses of the Visible class.

The subclasses of the Visible class are described below:

- LineArt:
  an object of the LineArt class represents a graphical object with vectorial representation. It can be used, for instance, to present polyline objects, ellipses, bezier curves, etc.;

- Rectangle;

- Bitmap;

- Video;

- RTGraphics:
  an object of the RTGraphics (Real Time Graphics) class represents a stream of graphics objects that are displayed using autonomous placement and synchronization. The RTGraphics stream can be used in conjunction with video and audio, for instance, to create a subtitling application;

- Text:
  text objects represent text strings. A Text object may be associated with a Font object, which describes the font in which the text should be rendered (in case no font object is given, the default font is assumed). Text has two subclasses: HyperText and EntryField, which implement different types of interactive text;

- Slider:
  a Slider is an Interactible used to let the user set linearly a position within a certain range (given by a minimum and maximum value);

- Button:
  The Button class has the three subclasses: PushButton, SwitchButton, and Hotspot. Buttons are rectangular areas on the screen with which the user can interact. Each of the three types of buttons has a specific event-generating behaviour attached to it. PushButtons and SwitchButtons are associated with a text item, which represents the text to be displayed in the middle of the button.

4.4.2 MPEG

MPEG defines generic standards for coding of moving pictures and associated audio. It provides specifications for audio compression, video compression, audio-video-data multiplexing and transport mechanisms for those signals. The specifications can be used either for residential telecommunication, business telecommunications, computer, professional and entertainment applications.

Implementations of MPEG can be found inside either a work station, a PC, a videophone terminal, Karaoke equipment or a set top box. MPEG is a generic set of standards. To implement MPEG in a given environment, MPEG standards need to be profiled and parameterized respectively. The best known profiling activities based on the MPEG-2 specifications are, on European level: Digital Audio Broadcasting (DAB) and Digital Video Broadcasting (DVB), and on international level: DAVIC.

4.4.2.1 The MPEG standards family

MPEG-2 is the second step in the MPEG activity. It was preceded by MPEG-1 and MPEG-4 is currently under development.

4.4.2.2 MPEG-1

MPEG-1 was designed to operate at Compact Disc (CD) speeds (1,5 Mbit/s) in an almost error-free environment. MPEG-1 is able to process progressive scan sequences only. In progressive scan all the lines of one picture are captured at the same time; this is the case e.g.: for a movie or for a computer monitor. On the contrary, in an interlaced scan, all the even lines are first captured and then all the odd lines. The interlaced scan is used for TV applications. Nevertheless adaption techniques are defined to use MPEG-1 at higher bit rates, in a non error free environment and with sequences that were originally interlaced. In a Compact Disc-Interactive (CD-I) application the bit rate is shared as follows:

- audio: 0,256 Mbit/s;
- video: 1,150 Mbit/s;
- systems: 0,094 Mbit/s.

The system parts of the standard release the multiplexing of audio, video and data streams.

4.4.2.3 MPEG-2

NOTE: The relevant MPEG-2 reference document for the purpose of this ETR is ISO/IEC DIS 13818-6 [6].

From the very beginning, MPEG-2 targeted TV-like applications. That means that MPEG-2 should be able to cope with environments where errors are likely; interlaced signals and higher bit rates are also covered. MPEG-2 is very flexible but as an indication, it may be assumed that signals are encoded in the range from 2 to 8 Mbits/s depending on the sequence complexity and the target quality. High Definition Television (HDTV) signals are encoded at a range from 15 to 20 Mbits/s.

The following subclauses give more details about the different parts of MPEG-2 specifications.

4.4.2.3.1 MPEG-2 Systems

This part of the standard provides for the multiplexing and accurate synchronization of multiple, audio video and data streams for correct presentation at the decoder.

The basic multiplexing approach is as follows. After encoding, the video and audio compressed elementary streams are packetized to produce Packetized Elementary Stream (PES) packets. PES packets are of variable and relatively long length. This length is not specified by the MPEG standard but normally limited to 64 Kbytes. Each PES may be protected with an optional Cyclic Redundancy Check (CRC). Those packets are then inserted into transport packets in one of the following forms:

1) the Transport Stream (TS);
2) the Program Stream (PS).

Both stream definitions are packet oriented and may be considered as transport layer entities.

PS utilizes variable-length packets and is intended for error-free environments. The packets are generally large. PS combines one or more streams of PES packets which have a common time base.

TS is especially designed for transmission in errored conditions and includes features for enhanced error resiliency and packet loss detection. The packets have a fixed length of 188 bytes. Many programmes, each with many components, may be combined in a TS.

PS and TS are designed for different applications. It is possible and reasonable to convert from one to another; however, one is not a subset of the other.

4.4.2.3.2 MPEG-2 Video

The MPEG-2 Video standard provides for the digital encoding of moving video at various data rates from about 1,5 Mbit/s to in excess of 60 Mbit/s, enabling a common technology to be adopted for many applications ranging from home entertainment quality video up to professional HDTV.

Because of the conflicting requirements of random access and highly efficient compression three main picture types have been identified. Intracoded frames (I-pictures) are coded without reference to other pictures. They provide access points to the coded sequence where the decoding process can start but they are encoded only with moderate compression. Predictive coded frames (P-pictures) are coded more efficiently using motion compensated prediction from a past I-picture or P-picture and they are generally used as a reference for further prediction. Bi-directional-predictive coded pictures (B-pictures) provide the highest degree of compression but require both past and future reference pictures for motion compensation. B-pictures are never used as reference for prediction. The organization of the three picture types in a sequence is very flexible. The choice is left to the encoder and depends on the application requirements.

4.4.2.3.3 MPEG-2 Audio

The MPEG-2 Audio standard specifies the coded representation of multichannel and multilingual high quality audio for broadcasting, transmission and storage media and the method for decoding multichannel and multilingual high quality audio signals. The input of the encoder and output of the decoder are compatible with existing Pulse Code Modulation (PCM) standards.

The main strategies for this multichannel coding scheme are subband filtering, perceptual model of the ear, common bitpool for channels, joint stereo coding, common masking threshold and dynamic crosstalk of the surround sound signal.

Transparent audio quality where no degradation is perceived is achieved at a bit rate of about 100 kbits/s per monophonic channel. The MPEG-2 audio standard also provides improved quality of mono and conventional stereo signals for bit rates at or below 64 kbits/s per channel

4.4.2.3.4 MPEG-2 DSM-CC

As DSM-CC is a key technology in DAVIC systems itself, it is introduced in greater detail in subclause 4.4.3.

4.4.2.4 MPEG-4

MPEG-4 is an emerging coding standard that supports new ways of access, manipulation and communication of digital audio-visual data. Among the new functionality to be addressed by MPEG-4 the following should be mentioned:

- improved coding efficiency;
- content-based scalability. This allows the access of only some parts of a bitstream depending on the required received image content;
- content-based manipulation;
- bitstream editing;
- robustness in error-prone environments;

- content based multimedia data access tools;

- coding of multiple concurrent data streams;

- coding of hybrid natural and synthetic data;

- improved temporal random accessibility;

- etc.

An important role in MPEG-4 will be assigned to a virtual machine standard providing software portability across different platforms. As DAVIC specifications and the ISO MHEG group will also select a virtual machine for their respective environment, it is envisaged to agree on one single standard for virtual machines.

The MPEG-4 specification completion is scheduled for November 1998.

4.4.3 DSM-CC

NOTE: The relevant DSM-CC reference document for the purpose of this ETR is ISO/IEC DIS 13818-6 [6]. The CORBA and IDL reference documents essential for that subclause are OMG CORBA 2.0 [7] and ISO/IEC 14750-1 [8] respectively.

The Digital Storage Media Command and Control (DSM-CC) specification is a set of protocols intended to provide the control functions and operations specific to managing ISO/IEC 11172 (MPEG-1, see annex A) and ISO/IEC 13818-1 [5] (MPEG-2) bitstreams. DSM-CC has been selected by DAVIC as end-to-end protocol for a DAVIC compliant system. According to the "one functionality one tool" principle of DAVIC, DSM-CC is the only DAVIC compliant end-to-end protocol.

DSM-CC provides functionality on different levels. It supports application oriented functions as well as network oriented functions. Therefore within the two major parts of the DSM-CC document (User-to-User and User-to-Network) a number of functional entity types within the DSM-CC specifications can be identified. The functional entity types defined are:

- Configuration (UN Config);
- Session Management;
- Resource Management;
- Compatibility Assurance;
- Download;
- Service Gateway Management;
- Stream Service Control;
- File Service Control;
- Database Service Control;
- Event Management;
- Stream Timelines.

The Configuration, Session Management and Resource Management are included in the DSM-CC User-to-Network part. The purpose of the Configuration feature is to provide the DSM-CC specific and network specific parameters to the Users (Client and Server) which are needed for communication with the Session and Resource Manager (SRM) and the initiation of the first (default) User-to-User Session. The Session and Resource Management features provide management of finite resources needed for User-to-User Sessions.

The Compatibility Assurance and Download features are included in the DSM-CC User-to-User part because they involve signalling between two Users (Client and Server). However, these two features are different from the other User-to-User features in that the underlying transport or delivery mechanism of the required messages is different. Compatibility provides a method for a Client to communicate its capabilities to the Network and/or Server so that the compatibility of applications can be assured. Download is a general purpose large data file transfer protocol which has limited requirements on its underlying transport. The Download uses a simple message based model under the assumption that it may be needed to download a more complex Remote Procedure Call (RPC) protocol stack for the use of

the other User-to-User functions. DSM-CC defines a method for flow controlled download as well as a non-flow controlled (broadcast) method.

The Session Gateway Management, Service Gateway Management, Stream Service Control, File Service Control, Database Service Control, and Event Management features of DSM-CC are also included in the DSM-CC User-to-User part. These features are all implemented using an object paradigm. All of the signalling is described on the User side by interfaces provided by Interface Description Language (IDL) definitions. The Over-the-Network signalling (where needed) is implemented using Remote Procedure Calls (RPC). The IDL and RPC techniques provide a higher level of abstraction than used by the Compatibility Assurance, Download, and User-to-Network features. For this reason these object based features have different requirements for their underlying transport.

The Stream Service Control function requires the ability to make absolute timeline position references to MPEG-2 streams.

4.4.3.1 DSM-CC User-to-User

DSM-CC User to User is implemented using an object paradigm. All of the signalling is described on the User side by interfaces provided by Interface Description Language (IDL) definitions. The Over-the-Network signalling (where needed) is implemented using Remote Procedure Calls (RPC).

When an application invokes a DSM-CC operation the IDL-compliant implementation of the DSM-CC library on the terminal handles the calls. The library decides whether the requested operation can be provided locally or if a request has to be made over a network to perform the operation. If a request has to be made, then the DSM-CC library compiles the operation call down to an RPC operation call conforming to the Object Management Group (OMG) Universal Networked Object (UNO) specification.

UNO specifies the protocol syntax for RPC primitives on a TCP/IP protocol stack. The part of the UNO specifications that is used for that purpose is the Internet Inter-Object Request Broker (ORB) Internet Inter-ORB Protocol (IIOP). An RPC protocol mainly consists of a request and response primitive. The data carried within this RPC protocol are encoded using the Common Data Representation (CDR) specification which is defined in the UNO specification as well.

4.4.3.2 Information flows

Although DSM-CC is a protocol that can be used between two users in a symmetric way neither a terminal nor a server is likely to implement the entire set of DSM-CC User-to-User functions. A terminal will only implement the subset of functions that are relevant to a terminal, a server will only implement service specific functions.

EXAMPLE: A terminal is likely to implement Digital Storage Media (DSM) FileRead to read files on the server. A server is not likely to implement DSM FileRead to read files on the terminal.

A standard DAVIC compliant STU will only implement DSM FileRead and maybe DSM FileWrite. It will not be able to handle DSM FileRead or DSM FileWrite requests from a server. Hence a server cannot initiate a transfer of files using DSM-CC User-to-User commands.

MHEG-5 applications consist of MHEG objects that can be stored on server. The objects can be stored in different physical files. Objects contained in one physical file can reference objects in another physical file. An MHEG application starts with the execution of the first object. If during the execution of an object, another object which is in another physical file on the terminal is referenced, a DSM FileRead primitive is issued to retrieve the physical file containing the referenced object. So it is always the application running on the terminal that requests a new file. The server never sends a file without a request from a terminal.

This information flow is not compatible with Videotex VEMMI. In VEMMI the server can send information without having received a terminal request. Information is decoded and executed by the terminal as it is received. The application which runs on the server can send information at any time on an established connection.

Subclause 7.1 shows how this information flow problem can be solved for the purpose of interworking of MHEG-5 applications with a VEMMI service Gateway.

4.4.3.3 DSM-CC User-to-User system logical entities

- **End-user applications** that control the viewing of digitally stored media and response to human user input. In the interworking scenario, the end user application is represented by MHEG-5 objects that result from a translation of VEMMI objects. The MHEG-5 objects are dynamically created by the Interworking unit (VEMMI service gateway);

This subclause maps the logical entities defined in the DSM-CC User-to-User on the implementation of a DAVIC system that supports interworking with a VEMMI system using a VEMMI-service gateway. The system logical entities of DSM-CC User-to-User are:

- **DSM library** which mediates between the end-user application and the remote service interface, providing a simplified function call interface to the end-user application, and an RPC interface to the remote service. The MHEG objects that were created by the VEMMI gateway contain references to objects not present at the terminal. These references will trigger operation calls to the DSM library. The DSM library then triggers the RPC request for the referenced object;

- **Services** which contain application objects such as directories, streams and other data. A Service is a logical entity in the system that provides function(s) and interface(s) in support of one or more applications. Services are accessible through the Directory interface. Services may be located and distributed in any manner on heterogeneous server hardware platforms. The VEMMI gateway is a service in the sense of DSM-CC. An intelligent gateway server acts as a VEMMI terminal for the VEMMI system and provides a DSM-CC directory for the DAVIC-System. The File objects of the DSM-CC Directory are created dynamically and the gateway server provides also files on which the terminal has write access in order to be able to retrieve user inputs;

- A **Service Gateway** which provides an interface for browsing and discovering services, authentication and authorization of end-users, registration of services and end-users, and resolution of the connection between clients and services. The Service Gateway is a specialized form of service. The VEMMI Gateway is registered with the general Service Gateway. If the user selects the VEMMI gateway in the service selection application of the Service Gateway the Service Gateway launches the VEMMI Gateway service. The Service Gateway acts as an information broker;

- **Directory** objects and associated interface, which may be configured as a service, providing name space and browsing of objects to applications. The VEMMI Gateway creates a Directory providing a namespace for the dynamically translated VEMMI objects and for files containing user inputs;

- **Stream** objects and associated interface, which may be bound to a Directory, providing ISO/IEC 13818-1 [5] continuous media streams to clients at their request. VEMMI for the time being does not support streams;

- **File** objects and associated interface, which may be bound to a Directory, providing data storage and retrieval to applications. VEMMI objects are translated to MHEG-5 objects that are put together in different files following naming conventions and rules laid out in this document. The VEMMI Gateway also creates files that can be written by the terminals in order to provide a way to catch user inputs;

- **View** objects and associated interface, which may be configured as a service, providing directory sort and filter operations and relational data access to applications. View objects are not used for the purpose of interworking;

- Any **Custom object** that is application-specific in function and interface, that invokes operations over and above what is specified by the DSM primitives. Custom objects are specified in IDL and configured into a service. The service with augmented interfaces is defined and registered with the Service Gateway. Custom objects are not used for the purpose of interworking.

4.4.3.4 Required interfaces for the purpose of interworking with VEMMI

For the purpose of the interworking the following interfaces are required by a STU and a VEMMI Gateway respectively. A STU is likely to offer more interfaces (e.g.: Stream interface) than the ones listed in this subclause. However, this subclause only lists those that are necessary for the purpose of interworking.

The Core interfaces represent the minimum requirement for a DSM-CC Service Complex.

- **Base** Interface:
  This interface provides commonly used operations: close, destroy and IsA. It is an abstract interface, meaning it is included (inherited) by other interfaces;

- **Access** Interface:
  This interface provides commonly used attributes for size, history (version and date), lock status and permissions. It is also an abstract interface, included (inherited) by other interfaces;

- **Directory** Interface:
  This interface provides a CORBA name service interface plus operations to access objects and object data through depth and breadth-first path traversal;

- **Stream** Interface:
  This interface enables a client to interactively control MPEG continuous media streams;

- **File** Interface:
  This interface enables a client to access data within an object which is a sequence of bytes;

- **ServiceGateway** Interface:
  This interface provides a directory of services and enables a client to attach to a service domain.

Minimum Compliancy Set for the purpose of interworking:

Table 1: Minimum compliancy set for the purpose of interworking

		Reader **Consumer Client**	**Writer** **VEMMI Gateway**
1	Base close	x	x
2	Base destroy		
3	Base isA		
4	Directory list	x	x
5	Directory resolve	x	x
6	Directory bind		x
7	Directory bind_context		x
8	Directory rebind		x
9	Directory rebind_context		x
10	Directory unbind		x
11	Directory new_context		
12	Directory bind_new_context		
13	Directory destroy		
14	Directory open	x	x
15	Directory close	x	x
16	Directory get	x	x
17	Directory put		x
18	CosNaming: BindingIterator next_one	x	x
19	CosNaming: BindingIterator next_one	x	x
20	CosNaming: BindingIterator destroy		
21	Stream resume		

(continued)

Table 2 (concluded): Minimum compliancy set for the purpose of interworking

		Reader Consumer Client	Writer VEMMI Gateway
22	Stream pause		
23	Stream status		
24	Stream reset		
25	Stream play		
26	Stream jump		
27	File read	x	x
28	File write	(note)	x
29	ServiceGateway attach	x	x
30	ServiceGateway detach	x	x
31	ServiceGateway suspend		
32	ServiceGateway resume		
NOTE: The FileWrite interface is not needed for interworking with VEMMI applications. A method of reporting user inputs from an STU to a server is described in clause 9.			

5 Videotex systems and VEMMI

A Videotex service is an interactive service, which enables, through appropriate access by standardized procedures, users of Videotex terminal equipment (dedicated terminals or microcomputer based software) to communicate with data bases and other computer based applications via telecommunication networks.

A set of protocols are defined in Videotex, the most up-to-date one being VEMMI (see subclause 5.2.1).

Videotex represents actually some 10 millions of users world-wide and the number of Videotex users is still growing.

5.1 Videotex information flows

For the purpose of this document it is important to differentiate different information flow scenarios that may occur in a Videotex system. The following scenarios are identified:

- the request-response scenario;
- the indication scenario.

5.1.1 Request-Response scenario

In most traditional Videotex systems information is always sent from a server to a terminal as a response to a request. This request may be one of the following:

- a request for a Videotex page in a page based system;
- a link offered by the system followed by the user;
- data entry in a Videotex input field sent to the Videotex application (in this case the request is performed at the application level).

Videotex applications falling in the above category never need to send information (objects) without the user requesting for it.

If the information flow of a Videotex application fits with the above requirements it can be mapped on to DSM-CC User-User RPC mechanisms. The execution model of a DAVIC compliant STB supports an information pull. The DSM-CC User-to-User protocol is based on an RPC mechanism.

This requirement is fulfilled for VEMMI page-based applications.

5.1.2 Indication scenario

This is the more general scenario where a Videotex or VEMMI host send information without invitation from the terminal. Ways to solve problems arising in this case are introduced in subclause 7.1.2.

5.2 Key Technologies

The following key technology can be identified in a VEMMI based information retrieval service.

5.2.1 VEMMI

NOTE: The relevant VEMMI reference document for the purpose of this ETR is ETS 300 709 [3].

VEMMI is an object oriented client server protocol allowing the implementation of multimedia applications distributed between a terminal and a server. VEMMI is especially well suited for a microcomputer platform because it takes benefit of the local terminal capabilities (local storage, operating system and other terminal platform, graphical user interface). The advantage of VEMMI lies in the fact that it is highly interactive. A number of input controls are defined and local actions defined with the objects can be used to put part of the application logic in the terminal.

The following subclauses present the objects and components defined in VEMMI. The Objects and components can be used in a distributed (mainly server-driven) application over a telecommunication network (Internet, Videotex, etc.). The VEMMI standards provides also the protocol elements that are necessary for a server to manage and drive the application (DisplayObject, StoreObject permanently on the terminal, etc.).

VEMMI objects offer a basic choice of dialogue elements needed by VEMMI applications.

Following VEMMI objects are defined in this ETS:

- Application Bar;
- Button Bar;
- Pop-Up Menu;
- Dialogue Box;
- Message Box;
- Operative object;
- Bitmap resource object;
- Videotex resource object;
- Text resource object;
- Font resource object;
- Metacode object;
- Sound object;
- Video object;
- Multimedia resource object.

A VEMMI application can be designed using any of the defined objects. No particular VEMMI object or component is mandatory within a VEMMI application. All VEMMI objects may be multiple within a VEMMI application, except the Application Bar. Within an object, all components may be multiple except certain restrictions applying to specific components.

5.2.1.1 The application bar

The Application Bar allows the user to make a choice between the different VEMMI application parts and sub-application parts offered by the selected VEMMI application. When used, the Application Bar is unique and located either on the top (horizontal Bar) or the left side (vertical Bar) of the Defined Display Area (DDA).

The Application Bar is subdivided into three different logical groups of Menu Choice components. These groups differ in their behaviour and functionality. The three different groups are named:

- Bar;
- Pull-Down Menu;

- Cascading Menu.

The Bar is a horizontal or vertical list of Menu Choice components which represents the different parts of the VEMMI application.

The Pull-Down Menus are vertical lists of Menu Choice components which are associated with the same Menu Choice component of the Bar. The Pull-Down Menus represent the different sub-application parts of the VEMMI application.

The Cascading Menus are vertical lists of Menu Choice components which are associated with the same Menu Choice component of the Pull-Down Menu. The Cascading Menus represent the different sub application parts of the VEMMI application.

5.2.1.2 The button bar

The Button Bar permits a choice among a set of alternatives, at a given time, during the execution of the VEMMI application. Each choice is represented by a Button. The Button Bar may be located anywhere in the DDA. The Button Bar can be horizontal or vertical.

The Button Bar is composed of a series of components, named Buttons. The purpose of the Buttons is to trigger a local action to be immediately performed.

5.2.1.3 The pop-up menu

The Pop-Up Menu offers appropriate choices and sub-choices for a given VEMMI element in its current context. The Pop-Up Menu may be located anywhere in the DDA.

The Pop-Up Menu is subdivided into two different logical groups of Menu Choice components. These groups differ in their behaviour and functionality. The two different groups are named:

- Primary Pop-Up Menu;
- Cascading Menu.

The Primary Pop-Up Menu is a vertical list of Menu Choice components which offers appropriate choices for a given VEMMI element in its current context.

The Cascading Menu is a vertical list of Menu Choice components associated with the same Menu Choice component of the Primary Pop-Up Menu. The Cascading Menu offers appropriate sub-choices for a given VEMMI element in its current context.

5.2.1.4 The dialogue box

The Dialogue Box is the object where the main interaction between the user and the VEMMI application takes place. To enable this interaction, a set of components is defined. These components can be classified as presentation or dialogue components.

Presentation components are inaccessible; their purpose is only to present the different dialogue components coherently and attractively.

Dialogue components permit the interaction between the user and the VEMMI application.

Five presentation components are defined:

- the Separator;
- the Frame;
- the Text Presentation Area;
- the Text component;
- the Graphic Output Area;

Ten dialogue components are defined:

- the Push Button;
- the Text Input Field;
- the Check Box;
- the Radio Button;
- the List Box;
- the Combination Box;
- the Slider;
- the Sensitive Area;
- the Sensitive Text;
- the Multimedia Area.

The separator component

A Separator is a horizontal or vertical solid line. Its goal is to visually separate different dialogue components within the Dialogue Box.

The frame component

A Frame is a presentation element to visually separate a particular area of the Dialogue Box and its different components.

The text presentation area component

The Text Presentation Area is a rectangular area in which text data is displayed. The goals of this component are:

- to present text information;
- to title or to label dialogue components;
- to present results from the VEMMI application execution.

The text component

The purpose of this component is to split large text data into units (text components) and to define the necessary structural information (concatenation of the text components) in order to be displayed in a text area.

The graphic output area component

The purpose of this component is to present graphical data to the user. Several graphical data encoding formats are supported.

The sensitive text component

The purpose of this component is to define the activation and validation operations for sensitive text strings.

The push button component

The purpose of the Push Button is to trigger a local action to be immediately performed.

The text input field component

The purpose of the Text Input Field is to collect text data, entered by the user. It is a rectangular area composed of a text label associated with an input area.

The check box component

The purpose of the Check Box is to enter and to display an independent user choice. A Check Box keeps the value marked or unmarked independently of any other Check Boxes.

The radio button component

The purpose of the Radio Button is to enter and to display a user choice. The Radio Button permits a single choice among several possibilities offered in a Radio Button group. The activation of one Radio Button leads to the deactivation of the other Radio Buttons belonging to the same Radio Button group.

The list box component

The purpose of the List Box is to offer a single or multiple choice among a list of text items. The list is, generally, not entirely visible to the user, so different controls are offered to scroll the list up and down.

The combination box component

The purpose of the Combination Box is to combine the functionality of a single choice List Box with the functionality of a Text Input Field. It contains a list of text items the user can scroll through to complete the Text Input Field. A parameter of the Combination Box specifies whether the Text Input Field content can be edited or not. If the Text Input Field content can be edited, the user can type text directly into the Text Input Field.

A variation of the Combination Box is a Drop Down Combination Box. It is composed of a Combination Box and a Push Button. Only the Text Input Field and the Push Button are displayed until the user selects the associated Push Button. The validation of the Push Button causes the display of the associated List Box.

The slider component

The slider offers the selection of an analogue value by moving a adjustable marker on a slide bar between a minimum and a maximum value. The intervals are set by the application.

The sensitive area component

The purpose of the Sensitive Area in the Dialogue Box is to offer a selection area associated with a Graphic output Area.

The multimedia area component

The purpose of this component is to present multimedia information to the user and to allow integration of the Internet World Wide Web.

5.2.1.5 Operative object

With this object an application references a program which will be linked to the VEMMI application. This object type provides a method to extend the capabilities of an application during runtime.

5.2.1.6 Bitmap resource object

A bitmap resource object contains either the bitmap definition itself or only a reference to a file with the bitmap definition. The object is referenced via the Bitmap Identification Number (BIN).

5.2.1.7 Videotex resource object

A Videotex object contains either the Videotex content itself or only a reference to a file with the Videotex content. The object is referenced via the Videotex Identification Number (VIN).

5.2.1.8 Text resource object

This object defines text content as a resource which can be referenced via the Text Identification Number (TIN). It contains either the text content itself or a reference to a file with the text content.

5.2.1.9 Font resource object

This object combines a set of text attributes in on font resource which can be referenced via the Font Identification Number (FIN).

5.2.1.10 Metacode object

The metacode object contains VEMMI commands. This object provides an easy way to avoid unnecessary dialogue steps with the host application.

5.2.1.11 The message box

The purpose of the Message Box is among others to display information, not requested by a user but sent by the VEMMI application in response to an unexpected event, or when something undesirable might occur.

5.2.1.12 Sound object

The purpose of this object is to link files with sound/audio content to the application.

5.2.1.13 Video object

The purpose of this object is to link files with moving video content to the application.

5.2.1.14 Multimedia resource object

A Multimedia object contains a reference to a file with multimedia content. The object is referenced via the Multimedia Identification Number (MIN).

6 Difference between VEMMI and MHEG-5

The main difference between VEMMI and MHEG-5 is that in MHEG-5 the multimedia applications are expressed and exchanged between a server and a terminal using MHEG-5 objects. The MHEG-5 application runs on terminal. In VEMMI the application itself can not be exchanged. Only the VEMMI objects that should be presented to the user as a result of an application process are transmitted to a terminal. The applications run on a server.

MHEG-5 is based on the scene concepts. Objects presented at a given time during the application on the user screen form an MHEG-5 scene. In an MHEG-5 application there are no objects outside scenes. An MHEG-5 application is a sequence of scenes; switching from one scene to another is done using the TransitionTo action. After a TransitionTo objects from the previous scene are not available to the MHEG 5 engine anymore. Only entire scenes or applications can be requested from an MHEG 5 application server. Individual objects can not be exchanged. Consequently all objects that are intended to be displayed together shall be known at the time of the scene creation. It is not possible to add objects to an existing scene while it is running on the terminal.

VEMMI is not based on the scene concept. VEMMI applications are based on individual objects that can be created on the fly and transmitted to the terminal. The behaviour of a VEMMI terminal is to process and consequently to add objects when they are received from a VEMMI host. The terminal does not need to know the objects that are going to be displayed together in advance.

This leads to incompatibility in the execution model of VEMMI and MHEG-5 applications. However, clause 9 shows how interworking is still possible.

7 Access to VEMMI applications from a DAVIC compliant platform

To access VEMMI applications from a DAVIC compliant platform according to the general DAVIC system architecture the following two basic options are available:

1) the use of a VEMMI Service Gateway (VSG) on the DSM-CC protocol stack;

2) the use of a downloaded VEMMI terminal on an appropriate protocol stack.

Both solutions are discussed in the following subclause. In subclause 7.4 the advantages and disadvantages of both basic solutions are discussed and appropriate recommendations are made.

7.1 VEMMI Service Gateway on the DSM-CC protocol stack

The approach presented in this subclause is intended to allow STUs in a Video on Demand (VoD) DAVIC 1.0 environment to access applications on a VEMMI platform without the introduction of additional specific functionality.

The STUs are assumed to cope only with the minimum functionality required by DAVIC 1.0 specifications Profile 2 (backchannel available). The implementation of the DAVIC high level Application Programming Interface (API) (MHEG5 engine) has either been downloaded previous to the start of the VEMMI service or is resident on the terminal.

No assumption is made on the actual underlying network technology or protocol stack except the requirements already stated in DAVIC 1.0. The only protocol that will be used for the purpose of this mapping is the User-to-User part of DSM-CC.

The entity that performs the actual mapping is located at Reference Point A9 of the DAVIC System Reference Model. The mapping is implemented as a Service Gateway offered either by the Service Provider or by the Application Provider. The Service Gateway physically connects the VEMMI service provider with the VoD service and performs the mapping from VEMMI protocol elements to MHEG-5 objects that can be handled directly by the implementation of the high level API on the STU. To Transfer the MHEG-5 object the DSM-CC protocol is used similar to a regular VoD application scenario.

The approach consist of two fundamental steps:

1) mapping of VEMMI service elements to MHEG-5 objects;

2) defining and execution model for the VEMMI service gateway and make sure that the execution model of the STU and the protocol stack used supports applications with a VEMMI like information flow.

Clause 9 of this document gives an example for such a mapping between VEMMI objects and MHEG-5 objects. A VEMMI Service Gateway (VSG) should implement rules like the ones defined in clause 9 for the entire VEMMI-syntax with in order to allow a generic mapping between VEMMI and MHEG-5 applications.

7.1.1 Mapping of VEMMI service elements to MHEG-5 objects

An example for such a mapping is given in clause 9.

The following subclauses introduce several possibilities to support VEMMI on a DSM-CC protocol stack.

7.1.2 Execution models and protocol stacks

To run a VEMMI application a symmetrical protocol stack is required. Both entities communicating via this protocol stack require the capability to send and receive data at any time during a given session.

This requirement leads to some difficulties in a DAVIC system because the protocol stack there is based on an RPC mechanism as described in subclause 4.4.3. The following methods can be used to implement VEMMI on an RPC-protocol stack:

- the Polling method;
- the PassThrough method;
- the Download method;
- VEMMI in page-based mode.

7.1.2.1 Polling method

This subclause describes a polling method that can be used in the implementation of a Service Gateway to overcome the difficulties induced by the RPC mechanisms.

The VSG should, when creating MHEG-5 objects from VEMMI objects, always create additional specific MHEG-5 objects that perform, when executed, the polling for new information. This can be implemented through the use of the MHEG-5 remote procedure object that allows an MHEG-5 application on a terminal to start a procedure on a remote server with a given set of input and output parameters.

This remote procedure, that has to be implemented on the VSG, can check if new information is available and return a boolean attribute to the terminal that it can use to trigger a TransitionTo action that results in a DSM File Read.

The IDL interface description that is required by the MHEG-5 remote procedure object is a mandatory part of the DAVIC 1.0 specification. It is further explained in clause 9.

This method allows an MHEG-5 terminal to inquire in a DAVIC compliant way if new information is available. If the result of the inquiry is positive the terminal can request the download of the new information in the traditional way.

7.1.2.2 PassThrough method

This subclause describes a method to implement a VEMMI-like information flow on a DSM-CC avoiding the RPC-method of DSM-CC User-to-User. Instead of having a bi-directional data path between two users (Server-Terminal) the DSM-CC User to Network PassThrough messages are used. This enables a bi-directional symmetrical link between the VSG and the STU.

The main disadvantage of this solution is that the VEMMI information would have to be handled by the network. Each PassThrough request from the terminal would have to be translated by the network in a PassThrough indication (and of course similar the other way round). Furthermore it is unclear if the execution model of the STU could associate data received that way with the MHEG execution entity.

7.1.2.3 Download method

The DSM-CC User-to-User download enables the server to recreate a strongly typed file system on the terminal. This can be used by a VSG to push information to the terminal (to download files that represent VEMMI objects).

7.1.2.4 VEMMI in page-based mode

In page based VEMMI mode VEMMI objects are always stored in Videotex files (frames) and never transmitted without being explicitly requested by the terminal. This scenario is fully compatible with the DSM-CC scenario where files are always received on request only. A VSG dealing only with page based applications need not implement specific behaviour in order to cope with the DSM-CC protocol stack. It simply needs to provide the object conversion.

7.2 Downloading the VEMMI terminal and running it on an appropriate protocol stack

This subclause describes the access of VEMMI applications from a DAVIC platform by downloading executable software representing the VEMMI client. The downloading procedure is described in the DSM-CC standard.

In order to run the VEMMI client on the STU an appropriate protocol stack is needed. The following protocols are appropriate for VEMMI:

- ETS 300 072 (see annex A) Videotex presentation layer protocol;
- ETS 300 223 (see annex A) and ETS 300 709 [3] Syntax based Videotex protocols;
- TCP-IP.

As TCP-IP is already used within DAVIC 1.0 only VEMMI on top of TCP-IP is investigated for the purpose of this document.

A DAVIC 1.0 STU does not have full featured TCP-IP access allowing it to run Internet client software such as WWW, FTP, Telnet or VEMMI. The following subclauses investigate possibilities of providing access to applications on TCP-IP networks. These TCP-IP networks can be private networks without interconnection to other networks or they can be connected to the Internet. Hence the provided solutions are generic for every application that can run above TCP-IP (on the Internet). However, examples are always referring to VEMMI rather than to other services like the WWW.

To provide such full featured TCP-IP (Internet) access several ways could be envisaged:

1) a proxy server providing Internet access;
2) protocol conversion to DSM-CC;
3) TCP-IP using ISDN links;
4) TCP-IP using MPEG-2 Transport Stream;
5) TCP-IP over ATM networks:
   a) LAN emulation;
   b) Multiprotocol Over ATM (MPOA);
   c) Classical TCP-IP over ATM.
6) access to Internet services using OMG UNO.

The following subclauses introduce all possibilities in detail.

7.2.1 A proxy server providing Internet access

Internet access is achieved via Internet Access Gateway (IAG). All information is carried in an S2 channel. An S1 channel is NOT used. Since the default RPC, OMG UNO, is selected in the DAVIC 1.0 S2 channel, an Internet session starts in this case with protocol switching.

The system configuration is shown in figure 9. In this scenario, an STU acts as a host machine on Internet via a proxy server that supports the VEMMI protocol (possibly amongst other Internet protocols). A corresponding client software (a VEMMI terminal) is installed on the STU.

A service gateway navigates a user to an IAG, which is a type of application server and works as a proxy server. Since DAVIC 1.0 STU speaks IIOP at default, IIOP is used to interact with the service gateway. To establish and terminate an Internet session the DSM-CC User-to-User primitive set could be extended. The suggested additional primitives are **DSM_ext InternetService Open** and **DSM_ext InternetService Close**.

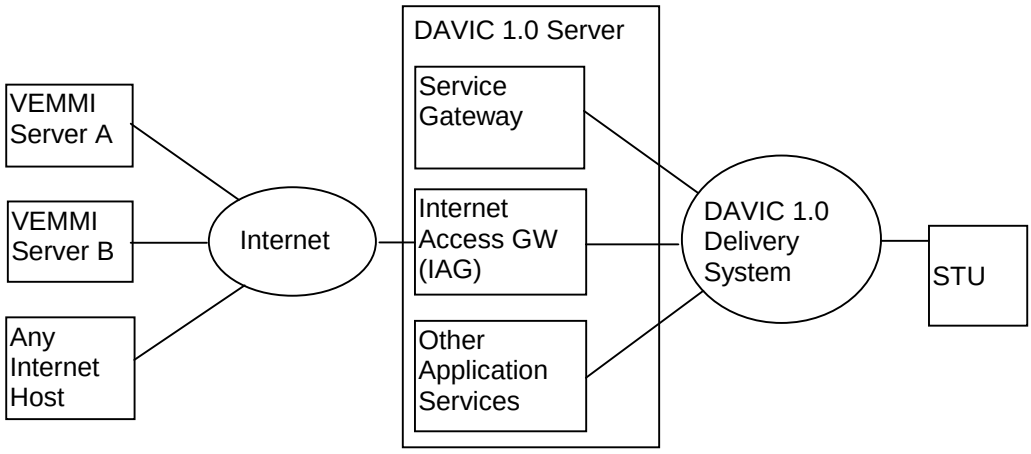


Figure 9: System configuration

InternetService Open is used to find IAG's IP address and its proxy's port number. The semantics of **aPathType** and **rPathSpec** are the same as **DSM Directory Open.**

```
module DSM_ext {
    interface InternetService {
        void Open (
            in PathType aPathType,
            in PathSpec rPathSpec,
            out IPaddress adrs,     //IP address of IAG
            out PortNo port     //Port for proxy
        )
        raises (OPEN_LIMIT, NO_AUTH, NotFound,CannotProceed, InvalidName);
    };
};
```

After **InternetService Open** has successfully responded, the STU talks to the IAG's proxy without using IIOP. The protocol stack at this stage is shown in Figure 10. Although Figure 10 shows only the VEMMI protocol, many other ones supported by the proxy could be used

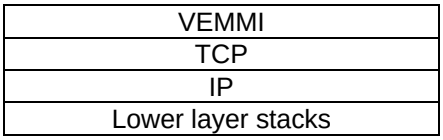


Figure 10: Protocol stacks between an IAG and an STU (scenario 1)

InternetService Close is used to indicate that the STU does not access the Internet any more.

```
module DSM_ext {
    interface InternetService {
        void Close();
    };
};
```

7.2.2 Protocol conversion to DSM-CC

Internet access is achieved via IAG. All information is carried on an S2 channel. An S1 channel is NOT used. Since the default RPC, OMG UNO, is selected in the DAVIC 1.0 S2 channel, an Internet session is terminated at an IAG.

The overall system is the same as in figure 9. The difference is that an STU is NOT a host machine on the Internet. Internet savvy protocols such as Hypertext Transfer Protocol (HTTP), File Transfer Protocol (FTP) and VEMMI are terminated in an IAG. **DSM File Write** is used to carry messages and files in both directions. Figure 11 shows the protocol stack between an IAG and an STU.

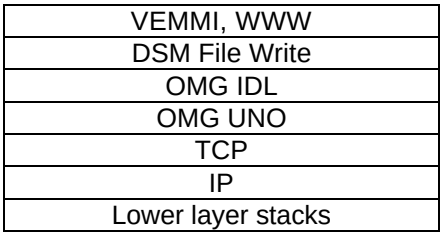


Figure 11: Protocol stacks between an IAG and an STU

A walk-through for VEMMI access is illustrated as an example (see Figure 12).

1) A service gateway navigates a user to an IAG, which is an application server;

2) An STU-side application program (VEMMI-client) sends VEMMI data to the IAG-side using **DSM File Write;**

3) The VEMMI software on the IAG interacts with a VEMMI server, using VEMMI-protocol;

4) VEMMI server sends a VEMMI object to the VEMMI software at the IAG-side. The VEMMI software sends the object to the VEMMI-client on the terminal using **DSM File Write**;

5) After the file transfer is completed, the VEMMI-client at the terminal interprets and executes the objects.

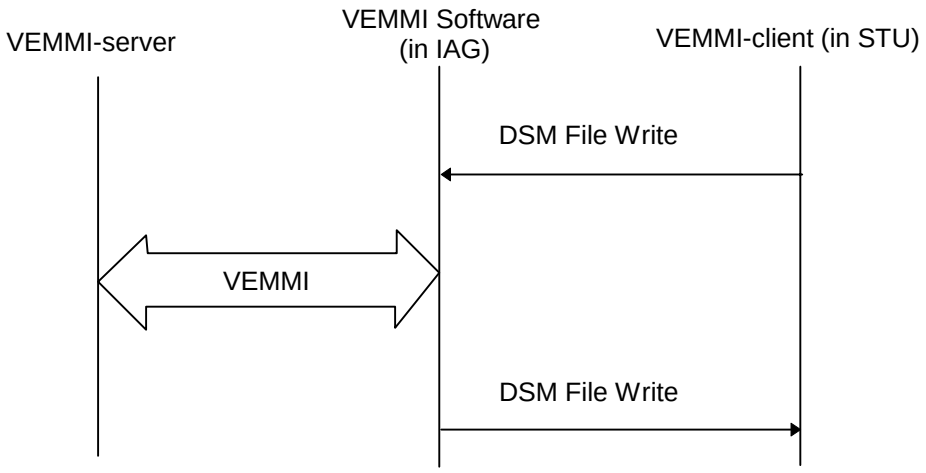


Figure 12: Walk-through of VEMMI access

7.2.3 Internet access using ISDN links.

This subclause proposes a protocol stack which allows flexible use of a standard ISDN connection. The solution uses a standard modified version of the Point-to-Point protocol (PPP) - Multilink PPP (MP), which can flexibly use the resources provided by ISDN connections. This protocol is already gaining acceptance for inverse multiplexing between networks.

The solution has low impact on the DAVIC system model and uses the same A9 interface as currently specified in DAVIC 1.0.

7.2.3.1 Requirements

DAVIC 1.0 specifies a system for the delivery of mainly video-oriented applications to a client. The DAVIC system allows the end-user access to one server at a time. VEMMI access over the Internet could use a far more flexible approach to client access to servers, allowing many semi-permanent logical links to be set up between the client and a number of servers.

Bandwidth on demand

It is important that the DAVIC system offers economical use of the network in order to achieve wide adoption by network and service providers (who have to buy the equipment) and service consumers (who have to pay bills for using the network and services).

Symmetric access

A wide range of applications other than VEMMI can be accessed using the Internet. In order not to preclude any of these applications, any candidate should offer a symmetrical interface to allow transmission at the same rate as received signals.

Common server access technology

To reduce the impact of providing access to VEMMI applications on the Internet in the DAVIC system, the new application should adopt a common A9 interface (delivery system to server). The A4 interface has both near and long term recommendations, and this proposal is suitable for both these alternatives.

7.2.3.2 System description

The proposed system maps into the DAVIC system reference model by treating the Internet gateway as the service provider, the set top as the service consumer and any of the existing delivery system options.

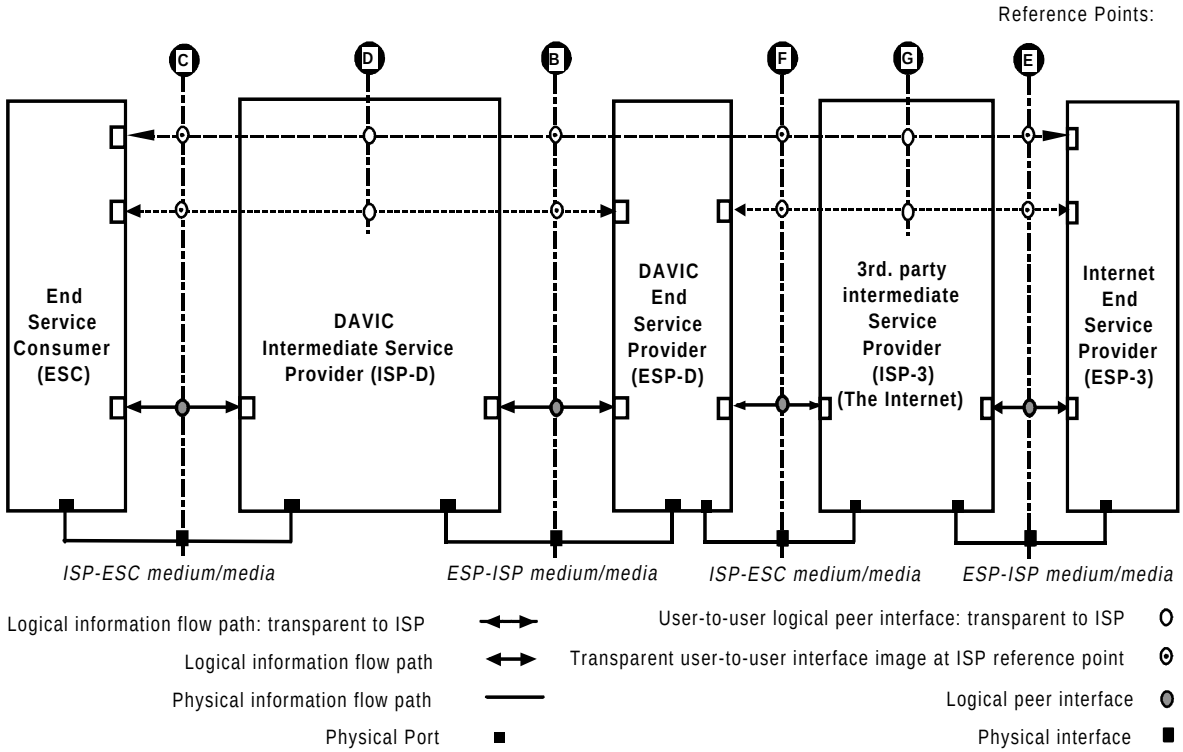


Figure 13: Reference points and interfaces for access to VEMMI applications on the Internet

7.2.3.3 Impact on the DAVIC delivery system

A new bi-directional stream (S1*) for the end-to-end service should be used. This should use the same ISDN Interface as the S2 stream. Use of Multilink Protocol beneath the network layer in the protocol stack allows flexible use of the available bandwidth.

7.2.3.4 Impact on the server

A stream service entity should provide access to the Internet.

The server should provide access to a DNS for IP packets. This will permit the user to obtain the associated ISDN address for the Internet ESP.

7.2.3.5 Impact on the STB

There will be a new bi-directional stream (S1*) terminating at the STB from the VEMMI service provider (third party service provider). Whilst the service using the third party network is in progress, this will "replace" the DAVIC S1 stream.

The existing application control (S2) stream should remain between the DAVIC server (ESP) and the STB (ESC).

7.2.3.6 VEMMI service provider (third party ESP) & third party ISP

The VEMMI service Provider is a terminal on the Internet implementing a VEMMI server. The third party Internet Service Provider (ISP) is the Internet itself. For quick implementation of Internet access it is proposed that the Internet should not be affected by DAVIC proposals. The third party ESP and third party ISP have only been added to the DAVIC system model to aid understanding and complete the model of service delivery.

7.2.3.7 The Service

The proposed system uses Multilink Point-to-Point Protocol (MP) to create a single logical link across the ISDN-Basic Rate Interface (BRI) between the VEMMI server and the STB. MP is able to utilize bandwidth between two end-systems as it is required. It does this by only using the individual links as necessary, for

example using one B-channel until a threshold is reached and only then utilizing a second B-channel. ISDN is particularly well suited to this way of working, as it has fast setup and cleardown of channels. The B-channels should only be used if they are not already in use.

MP negotiates configuration options between the end points in the same way as conventional PPP. A multilink option message is sent during the option negotiation. This indicates that a terminal is willing to combine multiple connections as a single pipe. The terminals can configure the link so that connections can be cleared down when not in use, leaving the line free at the STB as often as possible. This would allow a second or third DAVIC stream to use the same ISDN address(es).

The MP termination recombines and sequences the received packets from one or more physical links. The reconstructed packets are then presented to the network layer entity in the protocol stack for processing. Although many network layers can work with MP, IP carrying TCP is expected to be used for transport over ISDN B-channels.

7.2.3.8 Scenario

The scenario of a service in progress is as follows:

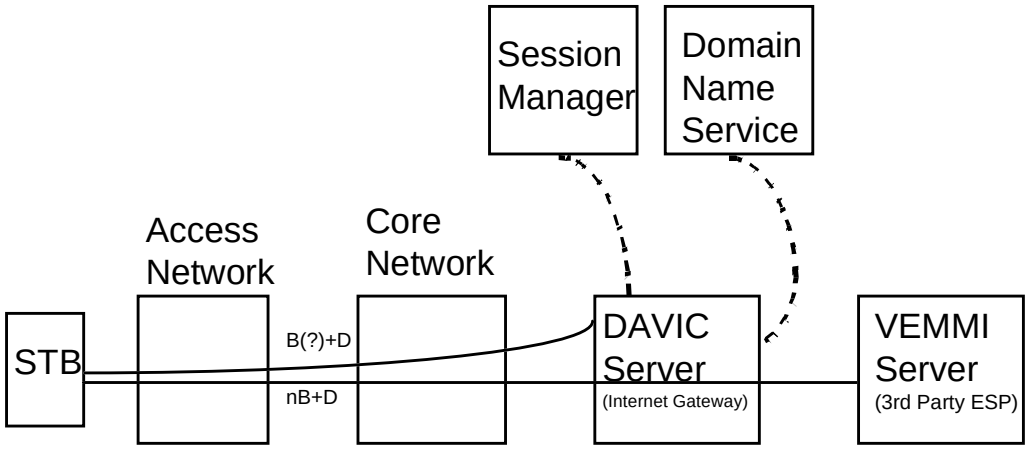


Figure 14: Scenario for internet access

When the STB is turned on, a session control relationship is established, using dial up of the session manager where the address has been read from a smart card.

A MP link is set up with this server: A connection is set up between the server and Set Top box and the link control and network control negotiation is resolved.

S2 links are resolved using MP and used to transmit navigation screens to the user. If the data waiting to be sent requires the use of a larger amount of bandwidth than can be offered by the first B-channel, then an additional B-channel connection can be set up between the two end-points. Once the buffered data falls beneath a negotiated threshold, the second B-channel can be cleared down. The user can respond using packets which are sent to the third party ISP.

Following the browsing phase of service selection, the user selects Internet access, and the session control controls the transfer of the session (if necessary) to a new server - one which can provide an Internet gateway service.

Flexible allocation of channels using MP enables shared channels to be available to different DAVIC streams.

7.2.3.9 Relevant Protocol Stacks

VEMMI (or other Internet applications)
TCP
IP
MP
I.430/1

a) Downstream

VEMMI (or other Internet applications)
TCP
IP
MP
I.430/1

b) Upstream

Figure 15: A1 Protocol stack for S1* information flow for ISDN networks

7.2.4 Internet access using MPEG2 encapsulation

This subclause specifies methods which allow the user in a DAVIC compatible system to get access to VEMMI applications on the Internet or on a similar public or private network using IP protocol. This subclause specifies three access methods which provide different grades of service.

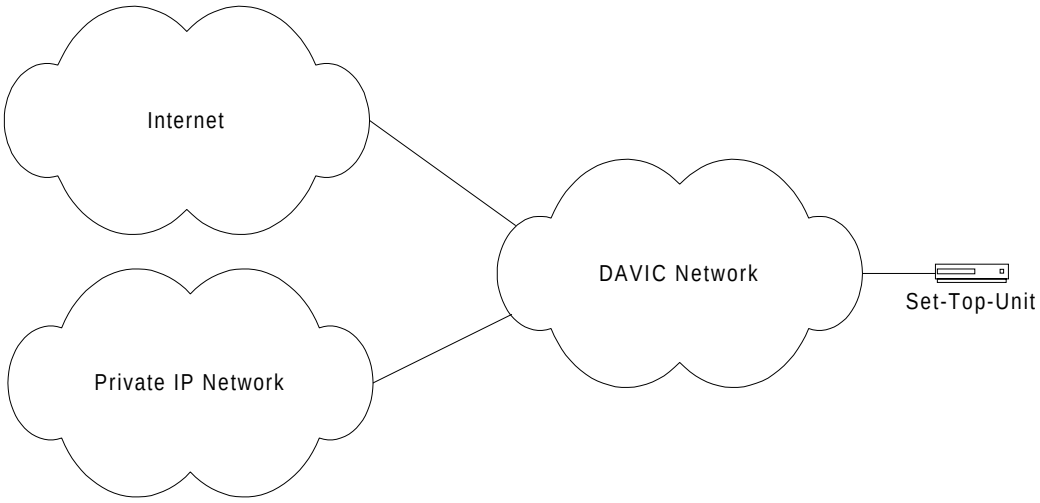


Figure 16: General model

Internet access is considered as a service which is provided by a server. The service provider system is the router which routes the protocol packets between the Internet and the DAVIC network in this model. The service is accessed using the same DSM-CC session set-up sequence as for other services in the DAVIC reference model. IP protocol is carried through the return channel and/or encapsulated in an MPEG-2 transport stream.

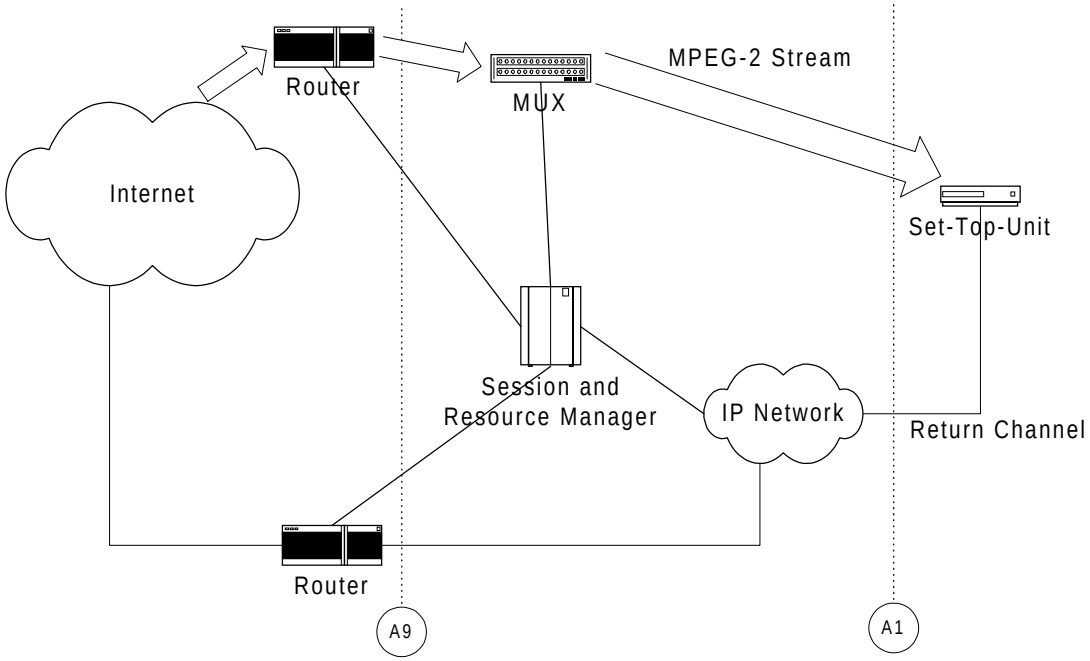


Figure 17: Network configuration

Figure 17 shows the network configuration. The routers act as service provider systems and connect the set-top-unit to the Internet. Return channel is assumed to use IP protocol as in current DAVIC model, but it is not automatically connected to the Internet; the router between the DAVIC network and the Internet provides this connection as a service. Figure 17 illustrates the DAVIC A9 interface between the service provider system and the core network and the A1 interface between the set-top-unit and the access network.

The session and resource manager is an entity which manages the sessions and network resources as in the DSM-CC model. The MUX represents the equipment between the core network and the access network; it may be used to multiplex the data stream together with MPEG-2 video streams. If one whole physical channel in the access network is dedicated for transporting data, multiplexing is not needed and the function is to deliver the stream from the core network to the access network.

7.2.4.1 Reference models for IP access

DSM-CC reference model

In the DSM-CC reference model, the server and the client communicate with the Session and Resource Manager (SRM) using DSM-CC User-to-Network protocol. In the Internet access service, the router is considered as the server and it communicates with the SRM using User-to-Network protocol.

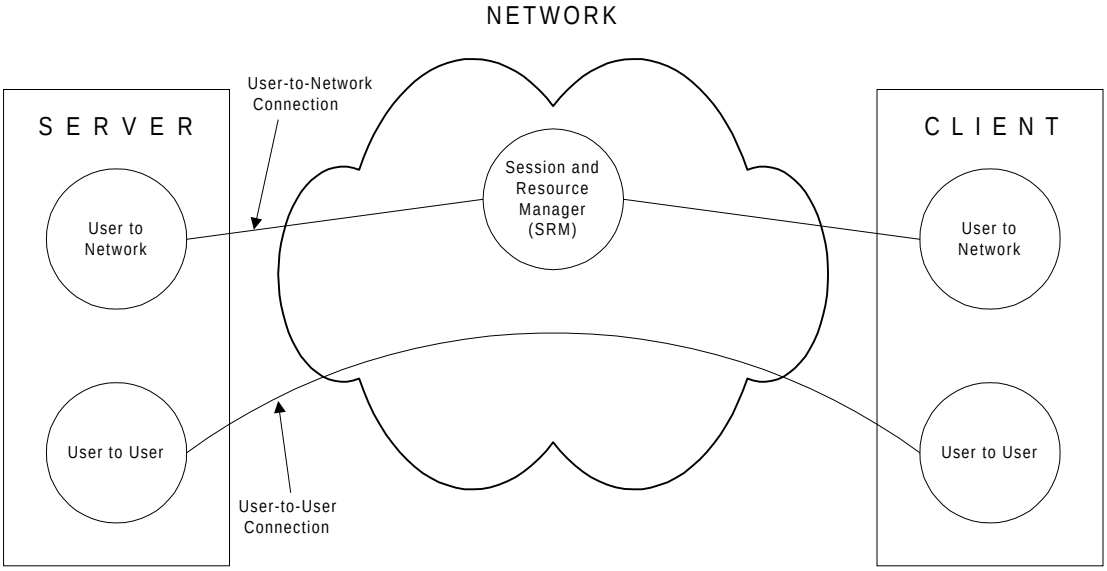


Figure 18 DSM-CC reference model

DAVIC reference model

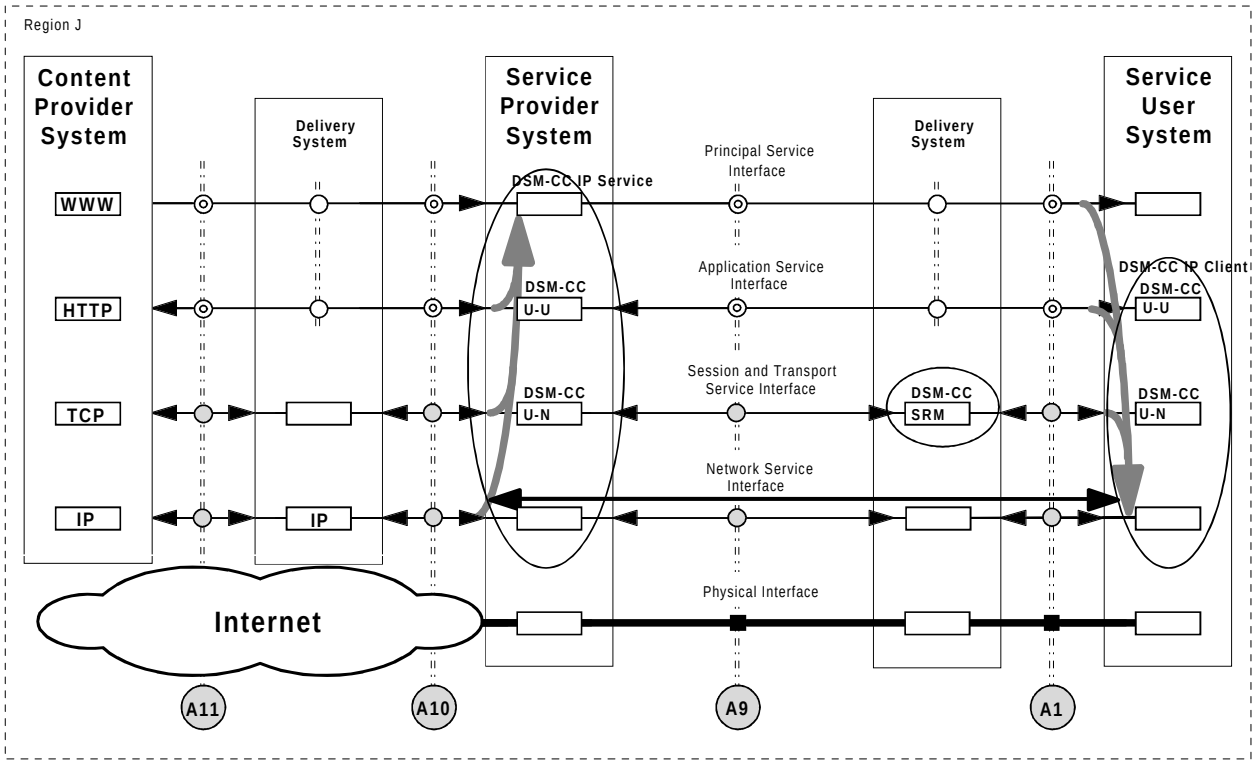


Figure 19: Internet access service in DAVIC model

Internet access service can also be mapped to the DAVIC reference model. The router is considered as a service provider system which provides access to the Internet. Internet servers are then the content provider systems; for example a VEMMI server may be a content provider system.

Encapsulating IP protocol to an MPEG-2 stream means in the DAVIC model that the whole protocol stack from the IP level upwards is lifted in the service provider system to the principal service layer (S1) and then dropped back to respective layers at the service user system.

Using the return channel for Internet access means just passing the network service layer IP protocol through the service provider system.

DSM-CC User-to-Network protocol and the SRM are on the Session and in the Internet access case, the A11 and A10 interfaces are in the Internet and are not compatible with the DAVIC interfaces.

7.2.4.2 IP protocol support

IP protocol encapsulation in MPEG-2 transport stream

The IP protocol encapsulation is performed as specified in the DSM-CC specification. LLC/SNAP multiprotocol encapsulation is used to encapsulate IP packets into the DSM-CC sections of private tables of the MPEG-2 transport stream. The private tables are divided to 188 byte transport stream packets. The PID (Packet ID) is used for addressing within the stream.

The router has to know the stream and PID corresponding to the IP address to be able to route the IP packets. The service user system receives that stream and uses the PID to extract the packets from the stream.

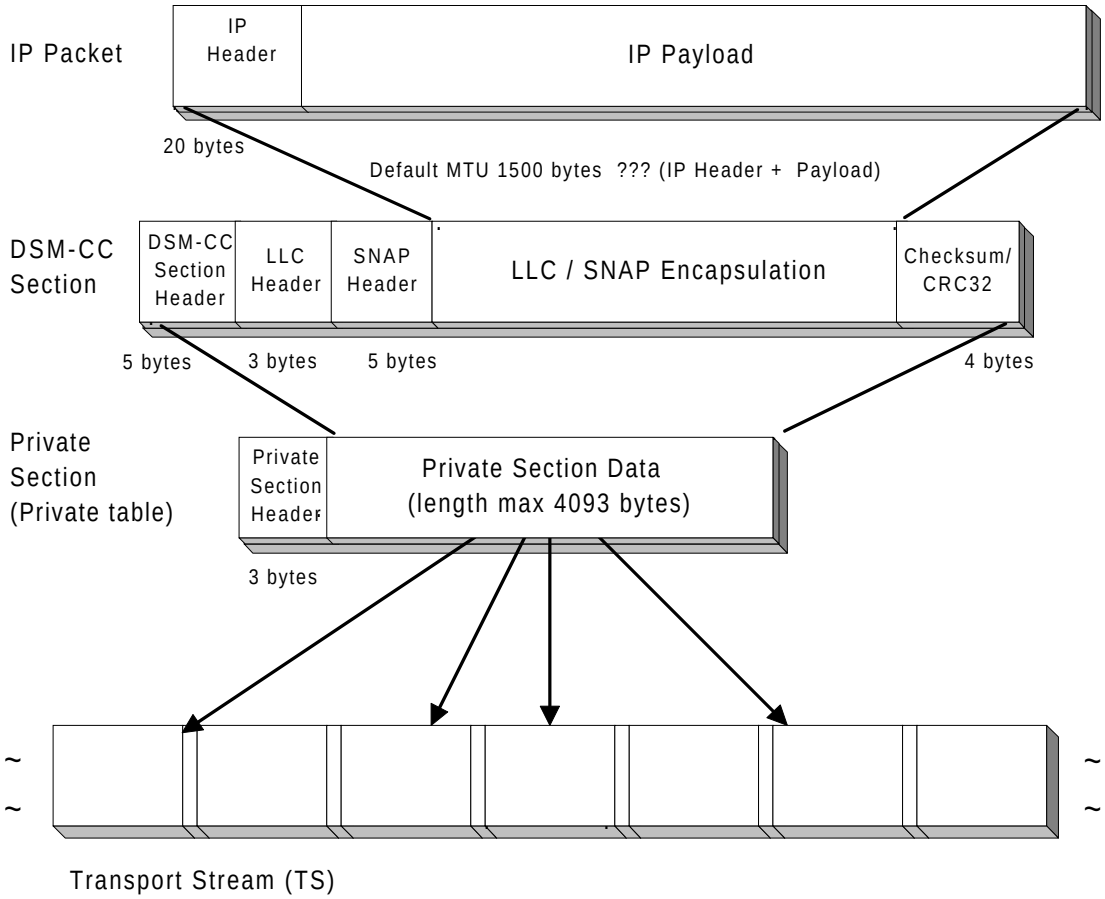


Figure 20: IP encapsulation in MPEG-2 stream

Set-top-unit protocol stacks

If the client wants to use an Internet access service which is based on IP protocol encapsulation in MPEG-2 stream, it needs such receiver and MPEG-2 demultiplexer that it is capable of receiving the private tables from the stream. In addition to that it needs software for extracting the IP packets from the private tables and an IP protocol stack which is capable of receiving packets from several ports. The IP stack should also be able to route all upstream packets to the return channel since the MPEG-2 stream is unidirectional.

If the client uses only the return channel for accessing the Internet, no additional features are required for the set-top-unit, since the IP protocol stack is already required by the DAVIC model.

7.2.4.3 Accessing the service

The client software has to start the Internet access service by using DSM-CC messages before using Internet connections.

This start-up sequence is an example of starting a service which uses MPEG-2 stream to transport the IP packets. Services using the return channel are otherwise similar except the allocation of MPEG-2 stream PID and possibly the allocation of the IP address (if already allocated) is not needed.

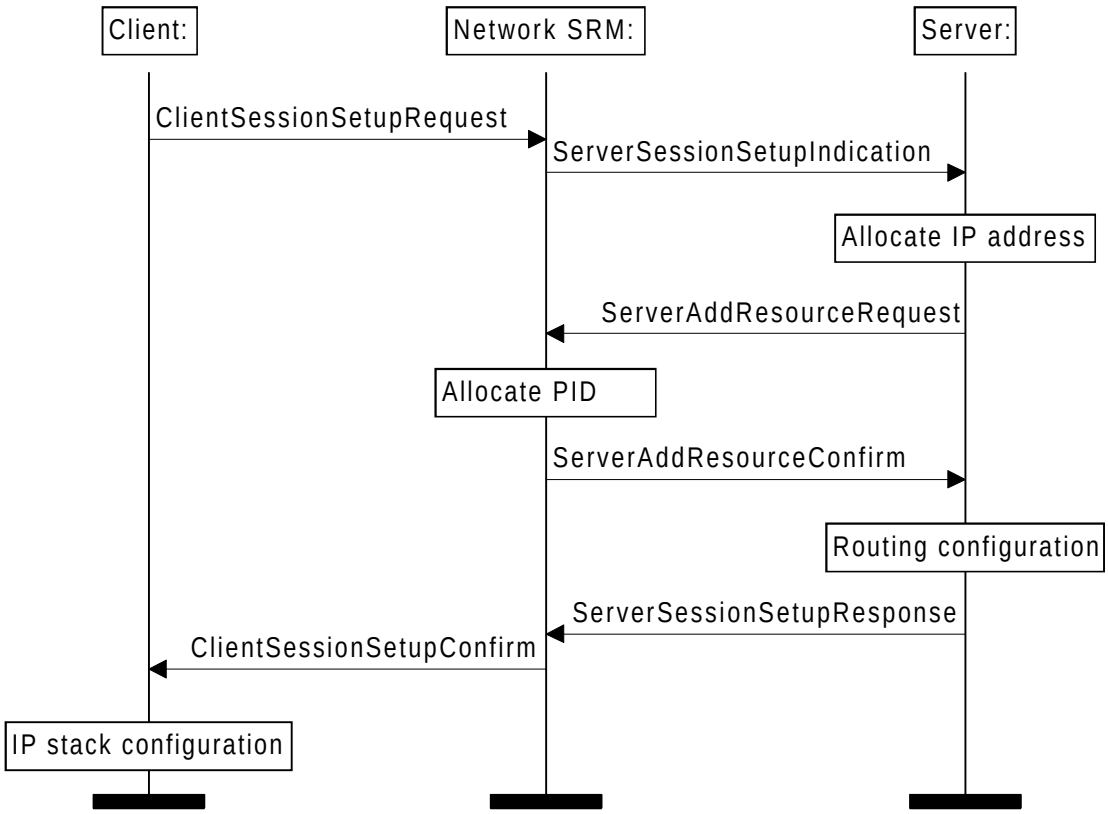


Figure 21: Session set-up sequence

The DSM-CC client software first sends a DSM-CC ClientSessionSetupRequest message to the SRM. The SRM forwards it to the server (router) as a ServerSessionSetupIndication message.

The server has an IP address space from which it dynamically allocates an IP address for the client. The IP routeing in the Internet has to be configured so that packets to those addresses are routed to this router.

The server sends a ServerAddResourceRequest with the IP address descriptors to the SRM and requests the SRM to allocate MPEG-2 PIDs for those IP addresses. The SRM responds with a ServerAddResourceConfirm with the MPEG-2 Program descriptor containing the PID values.

Then the server configures the routeing tables and accepts the session with a ServerSessionSetupResponse. The SRM sends a ClientSessionSetupConfirm containing the resource descriptors to the client. The client configures the IP stack and is ready to use the Internet applications.

The session is closed using DSM-CC session release command sequence.

7.2.4.4 Service types

Two types of Internet access services are specified. The types have different grade of service and the application can choose which type of service it requires.

Bi-directional IP access using return channel

This service uses the return channel to provide access to the Internet. The return channel has to use IP protocol as in current DAVIC model, however the lower layer protocols and the physical medium may be any suitable technology for this purpose. The IP network in the DAVIC network is connected to the Internet with a router which is considered as a server providing Internet access service.

This service provides a typical Internet access service which can be used to access VEMMI applications. The capacity of IP service depends on the type and the bandwidth of the return channel.

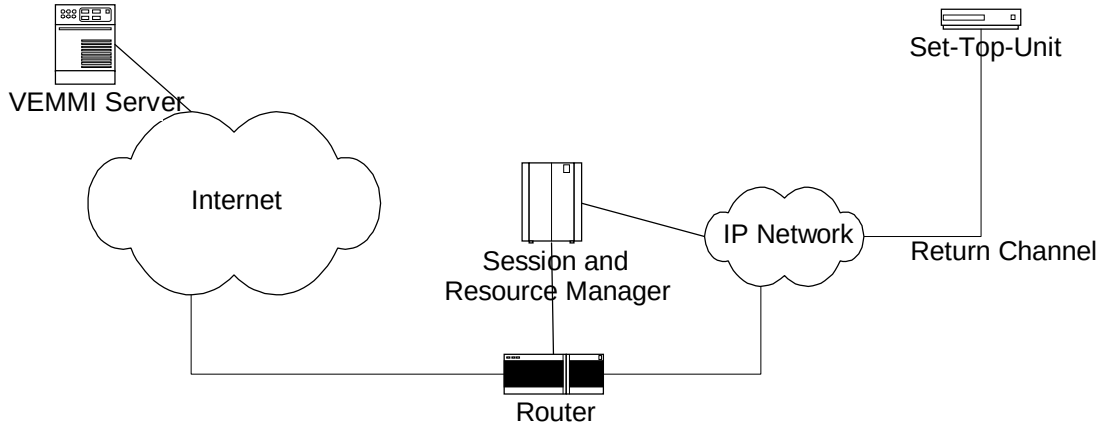


Figure 22: Bi-directional IP access using return channel

Asymmetric IP access using return channel and MPEG-2 transport stream

The asymmetric IP access to the Internet is provided by using both the MPEG-2 transport stream and the return channel.

All packets downstream from the Internet to the set-top-unit are routed to the MPEG-2 transport stream and the packets upstream from the set-top-unit to the Internet are routed through the return channel.

If the return channel has a relatively low bandwidth, this type of service can provide broadband downstream and a narrower band upstream. This suites well the typical case that clients send relatively few packets to the network, but they receive lots of packets. This is suitable for VEMMI applications because they require only limited bandwidth for the upstream channel.

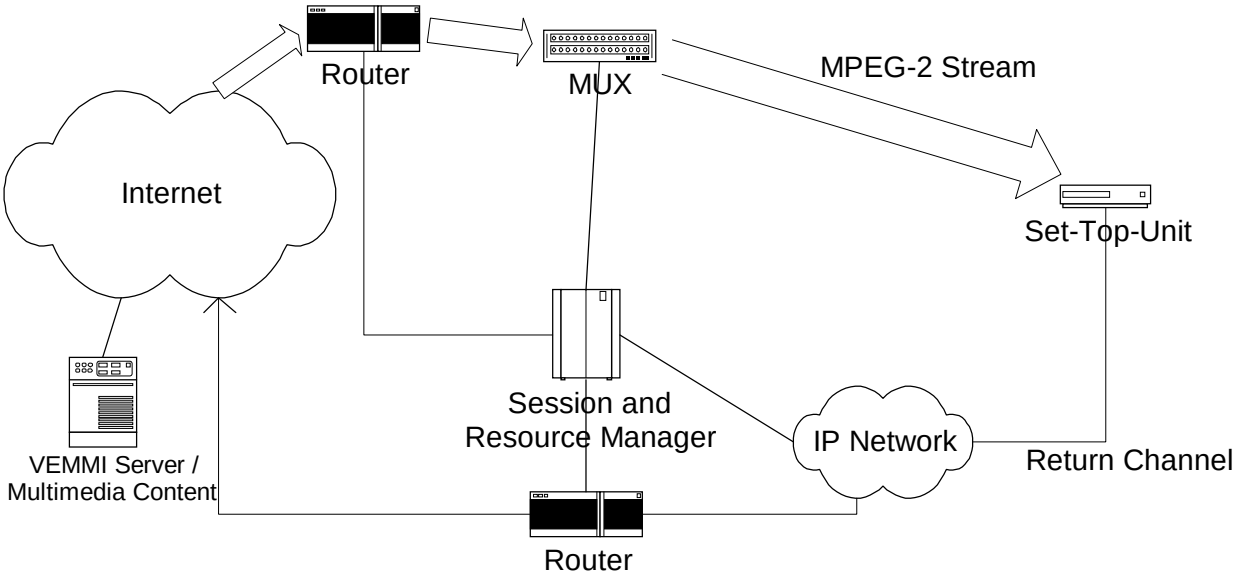


Figure 23: Asymmetric IP access using return channel and MPEG-2 stream

7.2.5 Internet access via ATM

To support current and future IP based applications over public ATM networks, several solutions for carrying IP traffic over public ATM networks are discussed in various standardization bodies and fora.

The Internet is based on a connectionless network layer protocol called IP (Internet Protocol). In its header, each IP packet contains the entire source and destination address, which is currently a sequence of four bytes. Packets are forwarded by IP routers through examining their destination address and retrieving the corresponding next hop from a routeing table.

ATM networks, on the other hand, are inherently connection-oriented and do not support connectionless traffic immediately. A virtual connection shall first be established between two ATM end systems before any data can be exchanged. In addition, public ATM networks are based on CCITT Recommendation E.164 (see annex A) addresses that cannot be derived directly from IP addresses. Therefore, ATM end systems require additional functions to carry IP traffic over an ATM network. The basic principle underlying all variants described below is shown in figure 24:

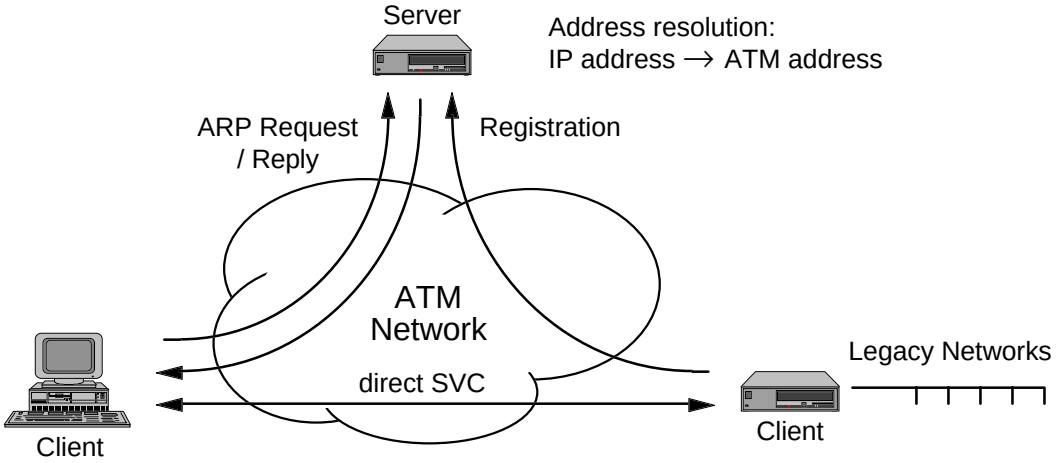


Figure 24: Mapping connectionless IP traffic to a connection-oriented ATM network

All concepts discussed in the literature or in standardization bodies and fora are overlay models in the sense that they do not require any changes to the underlying ATM network infrastructure. End systems and servers are connected to the ATM network via standardized user-network interfaces, and all additional functions required to carry connectionless traffic are transparent to the ATM network. Hence,

network operators will be able to carry IP traffic simply by adding a small number of specialized servers to their existing ATM network infrastructure.

An address resolution server (ARP) is the central component in all the variants for carrying IP traffic over an ATM network. Clients are able to query the server via an address resolution protocol to obtain the mapping between an IP address (or any other non-ATM address) and the corresponding ATM address of the destination end system or host. Once the ATM address is known, the client is able to open a direct virtual connection to its destination. This connection is maintained as long as there are packets to be sent to the corresponding destination

Configuration of the address resolution server can be either manually or automatic. Clients are able to register their addresses automatically, allowing the server to know exactly which clients are reachable at any point in time.

The following subsections describe three different approaches that can be used to carry IP traffic over an ATM network. All concepts are supported by international standardization activities, but with a varying level of maturity. While the specifications for LAN Emulation (ATM Forum) and Classical IP over ATM (Internet Engineering Task Force) are complete and first commercial implementations are becoming available, Multiprotocol over ATM (MPOA) is currently under discussion in the ATM Forum.

7.2.5.1 LAN Emulation

LAN Emulation as specified by the ATM Forum provides the same functionality as a shared medium local area network, but uses an ATM-based transport. LAN Emulation replaces the medium access control (MAC) layer of Ethernet (802.3) or Token Ring (802.5) network interface cards, and is completely transparent to higher layer protocols. Thus, existing protocol stacks (including IP) and applications can be used without any modification.

LAN Emulation requires at least two different servers to be present in the network: a LAN Emulation server (LES) that is used for address registration and resolution, as well as a broadcast and unknown server (BUS). Each client is automatically registered at the LES with its MAC level address and its ATM address. The BUS is responsible for the distribution of broadcast and multicast packets. In order to overcome the delay resulting from setting up a connection to a new destination, initial unicast packets can also be relayed via the BUS until a direct ATM virtual connection has been established.

Both server functions are potential performance bottlenecks that limit the number of clients communicating directly via LAN Emulation. In particular, the amount of broadcast traffic that shall be processed by each end system becomes prohibitive in a large network. Although multiple LAN Emulation subnetworks can be installed on the same ATM network, they can only be interconnected via MAC-level bridges or routers. Although LAN Emulation may be a practical solution for IP subnetworks of limited size, it is clearly not suitable for a large-scale ATM-based Internet.

7.2.5.2 Classical IP over ATM

The Internet Engineering Task Force (IETF) has produced a set of specifications for the transport of IP packets over an ATM network without the overhead of an unnecessary MAC layer. In the literature, this approach is commonly referred to as Classical IP over ATM, because it maintains the classical concept of IP subnetworks. Any network technology (including ATM, Frame Relay, Ethernet, etc.) can be used within an IP subnetwork, but different subnetworks can only be interconnected through IP routers.

Classical IP over ATM as described in RFC 1577 is based on the same concepts as LAN Emulation. The main differences are related to addressing (IP addresses instead of MAC-level addresses) and encapsulation of IP packets. RFC 1483 specifies LLC/SNAP encapsulation as the default way of mapping IP packets into ATM AAL5 protocol data units.

In its most simple form, Classical IP over ATM can be based on permanent virtual circuits between ATM end systems or routers. Switched virtual circuits (RFC 1755), on the other hand, are certainly the preferred way of carrying IP traffic over an ATM network. The mechanisms for setting up connections are the same as in the case of LAN Emulation, but with more stringent requirements for connection setup delays. This is due to the lack of a BUS server that could relay IP packets while the connection setup is still in progress.

The limitations of Classical IP over ATM are mainly related to the address resolution server which is not able to handle an extremely large number (several thousand) of clients. Although multiple servers can be

installed in an ATM network, clients connected to different ARP servers cannot communicate directly because the ARP servers are independent and cannot exchange information. An attempt to overcome this problem is the Next Hop Resolution Protocol (NHRP) that extends address resolution beyond the boundary of a single IP subnetwork. However, the definition of NHRP is still in progress, and the protocol may become obsolete when the ATM Forum specification for Multiprotocol over ATM is available.

7.2.5.3 Multiprotocol over ATM (MPOA)

The ATM Forum is currently defining a new specification called Multiprotocol over ATM (MPOA) that is intended to significantly extend the capabilities for carrying multiple higher layer protocols (IP, OSI, etc.) over an ATM network. MPOA should combine the advantages of LAN Emulation, Classical IP over ATM and NHRP into a single architecture. The MPOA working group also considers an integrated approach to IP and PNNI routeing. A first version of MPOA is expected to be available during 1996.

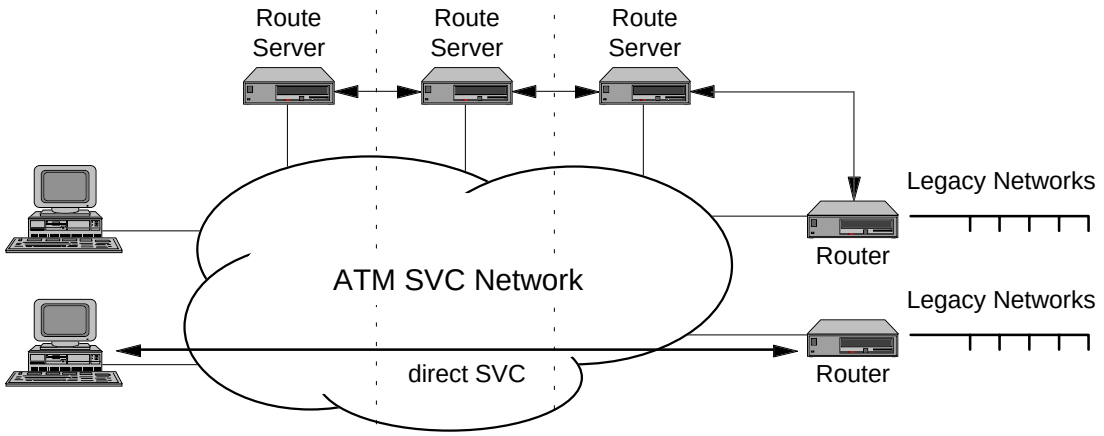


Figure 25: Multiprotocol over ATM (MPOA)

As shown in figure 17, MPOA introduces the concept of route servers that combine the functionality of an address resolution server and a next hop server. Each route server is responsible for a number of local clients to which it provides the address resolution service (note that MPOA supports multiple network layer protocols, not just IP). In addition, route servers participate in network layer routeing protocols such as OSPF, and they are able to forward routeing information using the Route Distribution Protocol (RDP).

MPOA supports direct connections between end systems and hosts attached to different route servers. It also resolves the network layer address of a non-ATM host to the ATM address of the router which is closest to the destination. Due to its distributed architecture, MPOA should therefore be scaleable to very large networks carrying IP traffic.

7.2.6 Access to Internet services using OMG UNO

The solution space for Internet access divides into the following domains:

- the Public Interface;
- the Private Interface.

The domains are introduced in the following subclauses.

7.2.6.1 Public Interface

The first domain is the space for *public* interfaces. A service publishes its interface. A client interacts with the service through a stub, which marshals the signature. The signature becomes the payload of a known protocol stack.

This architecture is the foundation of multimedia consortia design. The Interactive Multimedia Association, Digital Storage Media Command & Control, and Digital Audio Video Council all leverage the architecture. In all three cases a service specifies its interface with the Interface Definition Language of the Object Management Group. The question then shifts to the protocol stack below the interface.

This solution adopts the Universal Network Object specification of the Object Management Group. The stack is the basis of the service design (known as usr-usr) of the Digital Storage Media Command & Control initiative. It is also the recommendation of Digital Audio Visual Council for client/service stacks. (The second CFP included a request for a RPC stack. It was the joint decision of the Server Technology Group and the Settop Technology Group to recommend the Universal Network Object protocol. The plenary confirmed the decision at the Melbourne meeting.)

The specification first provides concrete rules (known as the Common Data Representation) about how to encode the signature as the payload. It also defines a session protocol which realizes remote procedure call semantics. The message set includes Request, Reply, and Exception functions. It also includes functions which allow both a client and a service to migrate. If the client invokes a request, it can receive a response which confirms, that while the service exists, its location is elsewhere. The protocol on the client side then forwards the request to the new location. (If the service is proactive, it will inform the client of the location migration before the request. There is a message for this purpose.) The client can also alert the service of location migration. The protocol anticipates nomadic clients, service migration, and failure recovery. It also anticipates the Digital Storage Media Command & Control session transfer concept, since the service object references, as well as the session gateway, can migrate.

The protocol reflects certain application requirements. There is a context field which can contain data which supplements the data of the object reference. The field can return, for example, values for download connections and stream connections, which have no object references. The protocol also provides a Principal field, which can contain certificates. This anticipates authentication.

The protocol requires that the stack below it release certain semantics. The specification defines the default stack as TCP over IP. The specification also describes how to publish an object reference. The structure of the object reference is a list of protocol stacks (with protocol state) which the service supports. The implication is that, if a client detects a stack it supports, it can select the native stack in preference to the default stack. The specification also describes the concept of (but not interface for) protocol bridges. The protocol bridge translates stacks to interface networks which otherwise could not interoperate.

7.2.6.2 Private interface

The second domain of the solution space relates to "private" interfaces. The premise is that there is a client front-end which exchanges protocol with a service back-end. (The front-end could be the result of download.) The signature found in the payload, in this case, can be private. It is still necessary, however, for the client front-end and service back-end to adopt known protocols below the application specific signature.

The recommendation is HTTP. It provides the mechanism for the front-end and back-end to exchange data and code. It can also leverage the stack of the public interfaces, since both build on TCP and IP.

The companion files describe in detail the Object Management Group stack and Internet Engineering Task Force stacks.

7.3 Constraints in a broadcast environment

Interworking between a DAVIC compliant broadcast system and VEMMI applications is only possible for page-based VEMMI applications. In this case all VEMMI objects belonging to the application have constantly to be broadcasted or cyclic retransmission has to be used.

7.4 Conclusions

Considering that

1) the main aim when implementing interworking is not to offer a given technology on the terminal but to provide a service to the user (a user does not care if the VEMMI-protocol is used; only about accessing the respective application);

2) the only way to provide interworking without changing or amending the DAVIC 1.0 specifications is the use of a VSG;

3) the high level API was optimized to run on a STU facing minimal resource constraints;

4) VEMMI was optimized for a PC platform;

5) content datatypes used by the high level API were optimized for a STU facing minimal resource constraints;

6) content datatypes used by VEMMI were optimized for a PC platform.

The VSG is the recommended approach to interworking. The use of a VSG also allows service providers to offer billing, accounting and access control functions that are essential for a commercial system.

However, if VEMMI applications do not run on the STU and STUs will be used as high speed access units for a PC (by connecting the PC to the STU access network), then it is more advisable to install the VEMMI client software directly on the PC and not use a VSG for interworking.

8 Access to MHEG-5 applications from a Videotex platform

8.1 Downloading the MHEG terminal and the protocol stack

If the terminal used to access the VEMMI platform is a PC then interworking is possible by emulating a DAVIC STU. A PC wanting to access a DAVIC needs the implementation of the high level API (a MHEG-5 engine), a DAVIC compliant protocol stack and a connection to a DAVIC compliant NIU.

If the terminal used to access the VEMMI platform is a dedicated terminal the solution described in subclause 8.2 should be used.

8.2 Service gateway

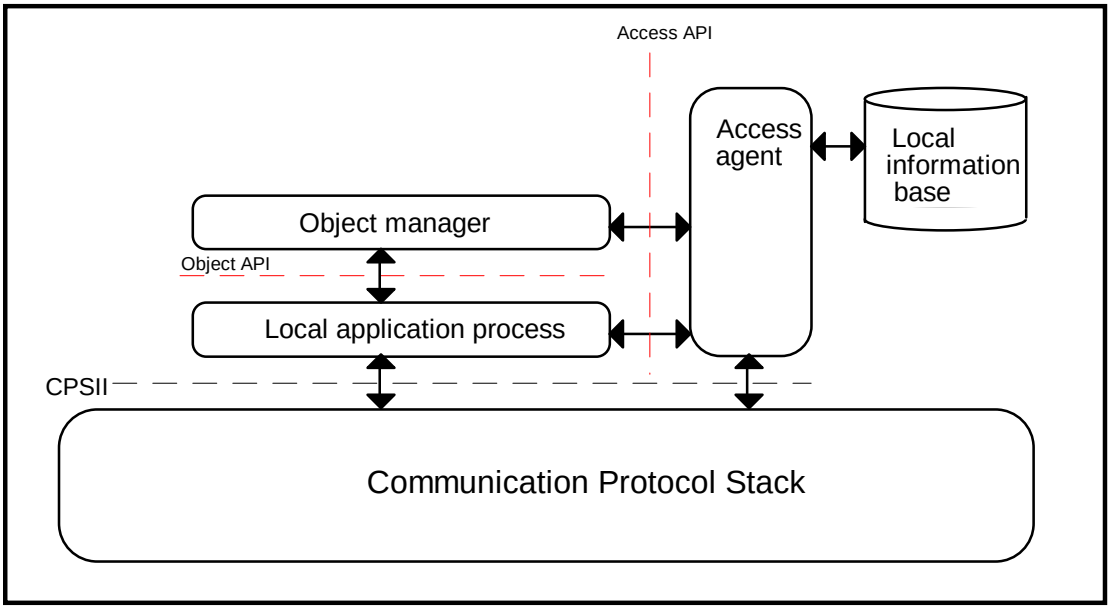


Figure 26: Server function

Figure 26 shows the reference model of a server function as defined in ETR 173 [2]. The server function is assumed to have an object handler that is capable of handling multimedia objects. In a VEMMI environment this object handler is a VEMMI server that creates and manipulates VEMMI objects.

For the purpose of interworking with a MHEG-5 application the object handler should be a full featured MHEG-5 engine such as would be used on a terminal. The MHEG-5 engine is capable of handling MHEG-5 objects i.e. parsing, executing action and using the presentation API to interact with the user.

The presentation API used by an MHEG-5 engine is not standardized. It depends on the operating system and on the environment that was used to implement the engine. The presentation API supports functions like display bitmap etc.

If a MHEG-5 terminal engine is used as a VEMMI server it should use the VEMMI protocol instead of the presentation API to interact with the user. Instead of calling a local API primitive that will cause the display of an object on the screen, the corresponding VEMMI service primitive should be used.

The MHEG-5 terminal engine that acts as a VEMMI server can handle MHEG-5 objects and process the application logic of these objects. When it comes to user interaction the MHEG-5 engine uses the VEMMI protocol.

The application in that case is not executed at the terminal as a regular MHEG-5 application would be but on the VEMMI server. All VEMMI compliant terminals can connect to the VEMMI server (MHEG-5 terminal engine) and use the application represented by MHEG-5 objects.

8.3 Conclusions

The favoured solution for access of MHEG-5 applications from a VEMMI terminal is the solution described in subclause 8.1. It is generic and does not require any specific changes at the terminal and VEMMI access network.

9 Possible architecture of a VSG

The architecture of a VSG could be as shown in the following figure:

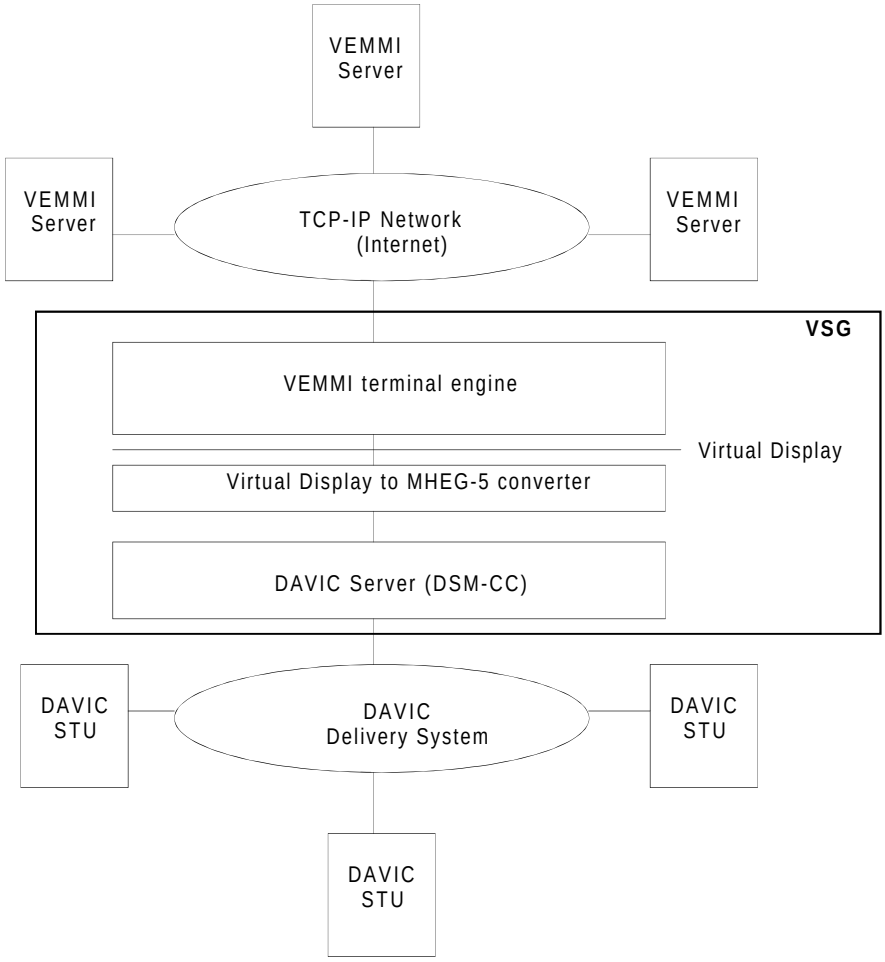


Figure 27: Possible architecture of a VSG

DAVIC terminals are connected to a DAVIC server via a DAVIC compliant delivery system. The DAVIC server implements the DSM-CC interfaces relevant to a server. On the server a gateway application is running. The application consists of two major functional units. A full featured VEMMI terminal and a Virtual Display to MHEG-5 converter. The VEMMI terminal is connected to a TCP-IP network (possibly the Internet) that hosts VEMMI servers.

Using this configuration the DAVIC STUs can remote control the VEMMI terminal running on the VSG. Whenever the VEMMI terminal normally would have to display something it translates the function call to the appropriate MHEG-5 objects.

When these objects are executed by the DAVIC STU every user interaction on executable is reported to the VEMMI terminal on the VSG. If due to the interaction the "virtual" display of the VEMMI terminal is updated a new set of MEHG-5 objects are created and the terminal is asked to request their download.

9.1 The virtual display concept

On every VEMMI terminal platform the VEMMI implementation makes use of a platform specific interface to display the VEMMI objects on the display device. These interfaces can be on different levels according to the present environment. The virtual display concept introduced in this document is on an intermediate level between the VEMMI object definition and the local presentation API.

Instead of calling a specific Presentation API-primitive to display a given VEMMI object the VEMMI implementation on the VSG can convert the information to MHEG-5 objects and store them in a DSM-CC directory structure. Every change of the content of this virtual display should trigger the creation of a MHEG-5 scene object including all objects currently displayed on the virtual display.

All VEMMI objects that are received from the VEMMI server (e.g.: metacode etc.) only show effect if they change the display.

All VEMMI elements that have an associated local action should be translated in a way that a link object is created with the associated action to run a remote procedure. The parameters of this remote procedure call allow identification of the trigger event and the source MHEG-5 object.

On the VSG the data received with the RPC can be used to simulate the user interaction for the VEMMI engine. As a result the VEMMI implementation on the VSG could perform the VEMMI local action that is associated to the respective component. A returned parameter of the RPC can be used in the terminal to fire a link for a TransitionTo action that will cause the terminal to request for a new Scene.

9.2 The RPC mechanism

In part five of the DAVIC specifications the following IDL interface is defined.

```
module DAV{
    typedef struct {
        string  groupidentifier;
        long    objectnumber;
    } obref;
    typedef struct {
        string  pubref;
        string  sysref;
    } contref;
    typedef sequence<boolean>bools;
    typedef sequence<long>longs;
    typedef sequence<string>strings;
    typedef sequence<obref> obrefs;
    typedef sequence<contref> contrefs;
    typedef struct {
        bools b;
        longs l;
        strings s;
        obrefs o;
        contrefs c;
    } pars;
    interface MHEG {
        void call (
            in string   procedurename;
            in pars     inpars;
            out pars    outpars;
        )
    }
)
```

This interface can be used to start a remote procedure and to exchange parameters with it. A MHEG-5 runSynchronous or runAsynchronous action targeted to a remote procedure object will trigger an RPC request according to the above specification.

The interworking application can define such a remote procedure object. To each interactable where a VEMMI local action is defined, a link object is targeted. The link object triggers a run action with a set of parameters identifying the object that triggered the action and the actual trigger condition.

The run action is mapped on an RPC call according to the IDL definition above. The RPC implementation on the server can now use the information from the call to simulate the corresponding user interaction on the VEMMI implementation on the VSG.

An Out-Parameter of the RPC can be set if the user interaction changed the display. This parameter can be used by the terminal to determine whether it should execute a TransitionTo another Scene. This Scene in the meantime has been prepared by the VSG (VEMMI display translated to MHEG-5 objects).

Annex A (informative): Bibliography

- ETS 300 072 (1990): "Terminal Equipment (TE); Videotex presentation layer protocol; Videotex presentation layer data syntax".

- ETS 300 223 (1993): "Terminal Equipment (TE); Syntax-based Videotex; Common end-to-end protocols".

- ETS 300 777-2 (1996): "Terminal Equipment (TE); End-to-end protocols for multimedia information retrieval services; Part 2: Use of Digital Storage Media Command and Control (DSM-CC) for basic multimedia applications".

- ISO/IEC 11172 (MPEG-1) (1993): "Information technology - coding of moving pictures and associated audio for digital storage media up to about 1,5 Mbit/s".

- CCITT Recommendation E.164 (1991): "Numbering plan for the ISDN era".

History

Document history	
October 1996	First Edition

ISBN 2-7437-0822-0
Dépôt légal : Octobre 1996