IEC 62295

Edition 1.0 2007-11

INTERNATIONAL STANDARD

Multimedia systems – Common communication protocol for inter-connectivity on heterogeneous networks

IEC Central Office
3, rue de Varembé
CH-1211 Geneva 20
Switzerland
Email: inmail@iec.ch
Web: www.iec.ch

About the IEC

The International Electrotechnical Commission (IEC) is the leading global organization that prepares and publishes International Standards for all electrical, electronic and related technologies.

About IEC publications

The technical content of IEC publications is kept under constant review by the IEC. Please make sure that you have the latest edition, a corrigenda or an amendment might have been published.

- Catalogue of IEC publications: www.iec.ch/searchpub

The IEC on-line Catalogue enables you to search by a variety of criteria (reference number, text, technical committee,…). It also gives information on projects, withdrawn and replaced publications.

- IEC Just Published: www.iec.ch/online_news/justpub

Stay up to date on all new IEC publications. Just Published details twice a month all new publications released. Available on-line and also by email.

- Electropedia: www.electropedia.org

The world's leading online dictionary of electronic and electrical terms containing more than 20 000 terms and definitions in English and French, with equivalent terms in additional languages. Also known as the International Electrotechnical Vocabulary online.

- Customer Service Centre: www.iec.ch/webstore/custserv

If you wish to give us your feedback on this publication or need further assistance, please visit the Customer Service Centre FAQ or contact us:

Email: csc@iec.ch
Tel.: +41 22 919 02 11
Fax: +41 22 919 03 00

IEC 62295

Edition 1.0 2007-11

INTERNATIONAL STANDARD

Multimedia systems – Common communication protocol for inter-connectivity on heterogeneous networks

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

PRICE CODE XB

ICS 33.040.40; 33.160; 35.100

ISBN 2-8318-9351-8

CONTENTS

INTERNATIONAL ELECTROTECHNICAL COMMISSION

MULTIMEDIA SYSTEMS – COMMON COMMUNICATION PROTOCOL FOR INTER-CONNECTIVITY ON HETEROGENEOUS NETWORKS

FOREWORD

1) The International Electrotechnical Commission (IEC) is a worldwide organization for standardization comprising all national Electrotechnical committees (IEC National Committees). The object of IEC is to promote international co-operation on all questions concerning standardization in the electrical and electronic fields. To this end and in addition to other activities, IEC publishes International Standards, Technical Specifications, Technical Reports, Publicly Available Specifications (PAS) and Guides (hereafter referred to as "IEC Publication(s)"). Their preparation is entrusted to technical committees; any IEC National Committee interested in the subject dealt with may participate in this preparatory work. International, governmental and non-governmental organizations liaising with the IEC also participate in this preparation. IEC collaborates closely with the International Organization for Standardization (ISO) in accordance with conditions determined by agreement between the two organizations.
2) The formal decisions or agreements of IEC on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested IEC National Committees.
3) IEC Publications have the form of recommendations for international use and are accepted by IEC National Committees in that sense. While all reasonable efforts are made to ensure that the technical content of IEC Publications is accurate, IEC cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.
4) In order to promote international uniformity, IEC National Committees undertake to apply IEC Publications transparently to the maximum extent possible in their national and regional publications. Any divergence between any IEC Publication and the corresponding national or regional publication shall be clearly indicated in the latter.
5) IEC provides no marking procedure to indicate its approval and cannot be rendered responsible for any equipment declared to be in conformity with an IEC Publication.
6) All users should ensure that they have the latest edition of this publication.
7) No liability shall attach to IEC or its directors, employees, servants or agents including individual experts and members of its technical committees and IEC National Committees for any personal injury, property damage or other damage of any nature whatsoever, whether direct or indirect, or for costs (including legal fees) and expenses arising out of the publication, use of, or reliance upon, this IEC Publication or any other IEC Publications.
8) Attention is drawn to the Normative references cited in this publication. Use of the referenced publications is indispensable for the correct application of this publication.
9) Attention is drawn to the possibility that some of the elements of this IEC Publication may be the subject of patent rights. IEC shall not be held responsible for identifying any or all such patent rights.

International Standard IEC 62295 has been prepared by technical area 8: Multimedia home server systems, of IEC technical committee 100: Audio, video and multimedia systems and equipment.

The text of this standard is based on the following documents:

CDV	Report on voting
100/1200/CDV	100/1283/RVC

Full information on the voting for the approval of this standard can be found in the report on voting indicated in the above table.

This publication has been drafted in accordance with the ISO/IEC Directives, Part 2.

The committee has decided that the contents of this publication will remain unchanged until the maintenance result date indicated on the IEC web site under "http://webstore.iec.ch" in the data related to the specific publication. At this date, the publication will be

- reconfirmed,
- withdrawn,
- replaced by a revised edition, or
- amended.

A bilingual version of this publication may be issued at a later date.

INTRODUCTION

Numerous wired and wireless home network technologies of various types have been developed and are in use today. However, since these technologies have been developed for specific functions such as control, A/V and data services, interoperability is not guaranteed among products employing these technologies. Hence, users who wish to implement the home network environment either purchase devices that are based on a single technology for interoperability or install independent, non-compatible networks in their home.

To solve these problems, home network businesses and service providers have taken into account and developed a number of specific technologies in order to allow interoperability among home network technologies. However, most of these technologies are local and offer interoperability between a limited range of devices and give rise to new problems caused by complexity and diversity in technologies of different companies.

In order to incorporate such complex and diverse technologies, there is a need to develop a new convergence technology that can integrate not only current technologies but also those expected to surface in the future.

The needs for the new convergence technology are the following:

– provide interoperability and interconnectivity among heterogeneous networks through a specific convergence layer;
– provide expandability for applications in not only current network technologies, but also new technologies to be developed in the future;
– are applicable in small devices with low processing capabilities by providing protocols such as simple signaling in the convergence layer;
– available at a low cost and simple to implement on a device;
– able to provide diverse home network services (or applications).

MULTIMEDIA SYSTEMS – COMMON COMMUNICATION PROTOCOL FOR INTER-CONNECTIVITY ON HETEROGENEOUS NETWORKS

1 Scope and object

This International Standard specifies the common communication protocol (CCP) layer that is capable of providing interoperability and interconnectivity between heterogeneous network technologies, as well as the basic data transmission scheme between devices linked to heterogeneous networks through the CCP layer. The standard also specifies the packet structure in the CCP layer and the common addressing scheme that can be understood among heterogeneous devices. Furthermore, there are specifications regarding protocols capable of providing diverse home network applications through the CCP layer such as the home network management protocol (HNMP), universal home control protocol (UHCP), home multimedia service protocol (HMSP) and home data service protocol (HDSP).

NOTE HNMP is the overall home network management protocol that detects or registers devices. UHCP controls and monitors devices from remote locations. HMSP is the A/V protocol for home entertainment services. HDSP deals with data and messaging services.

This standard is to be applied to systems with network capabilities and those that constitute home networks such as electronic appliances, A/V components, control devices, network terminals and home servers. Moreover, this standard is applicable to a home network consisting of a single home server.

This International Standard gives

– a definition of the CCP layer,
– a data transmission scheme in the CCP layer,
– a CCP packet structure,
– a CCP addressing scheme,
– a home network management protocol (HNMP),
– a universal home control protocol (UHCP),
– a home data service protocol (HDSP),
– requirements of home multimedia service protocol (HMSP).

A home network provides interoperability and interconnectivity regardless of the appliance manufacturer or the network type so that the user is able to receive desired services at any point in time. However, current home network technologies have independent communication protocol layer structures, as shown in Figure 1, with different addressing schemes, data transmission schemes, data processing methods and data formats.

In order to solve problems associated with interoperability and interconnectivity among heterogeneous network technologies, this standard aims to define the CCP layer as a type of a convergence layer.

Further objectives of this standard include specifying the data transmission method, packet structure and common addressing scheme as well as the signaling protocol for providing home network management, control, A/V and data services.

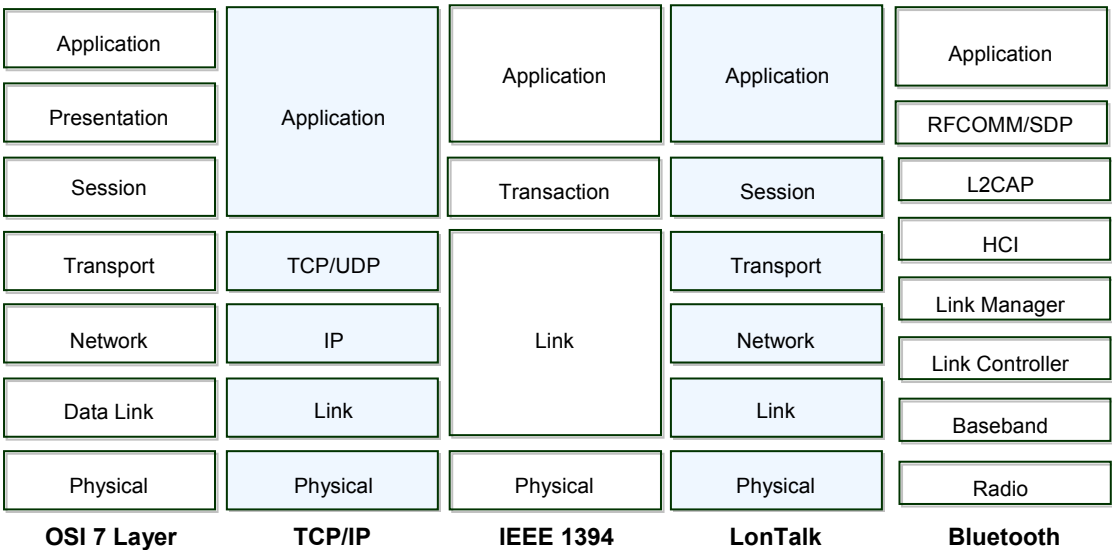


Figure 1 – Communication layer structures of network technologies

2 Normative references

None.

3 Terms, definitions, abbreviations and conventions

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

3.1 Terms and definitions

3.1.1
CCP device
device which has networking capabilities using commercial network technologies to link to a home network; it also supports at least home network management protocol (HMNP) among four protocols provided in the CCP layer and CCP

3.1.2
CCP addressing
common addressing scheme used in the CCP layer which consists of four fields: domain address, cluster address and device ID

NOTE One of the characteristics of CCP addressing is that it has a structure that is understood by application programs embedded in devices linked to heterogeneous networks as well as the device users.

3.1.3
home server interface
HSI
interface module in a home server responsible for connection with a particular network which can process the corresponding physical interface and the communication protocol according to the type of network it is connected to

NOTE A home server interface is shown in Figure 2.

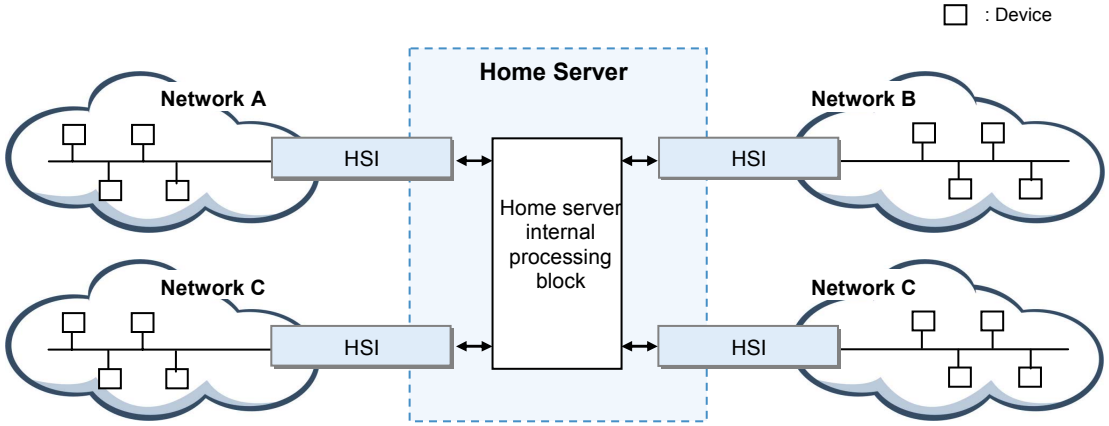


Figure 2 – Server interface

IEC 2073/07

3.1.4
cluster network
collection of devices using an identical physical interface and communication protocol

NOTE 1 As shown in Figure 3, a cluster network is a collection of devices using an identical physical interface and communication protocol. Moreover, a HSI that links a home server to the devices within the cluster network is also part of the cluster network. A cluster address is one of the CCP addressing fields defined in the CCP layer, and it is used to distinguish multiple cluster networks linked to a single home server.

NOTE 2 A HSI that links a home server to the devices within the cluster network is also part of the cluster network.

NOTE 3 A cluster network is shown in Figure 3.

3.1.5
cluster address
one of the CCP addressing fields defined in the CCP layer which is used to distinguish multiple cluster networks linked to a single home server

NOTE A cluster address is shown in Figure 3.

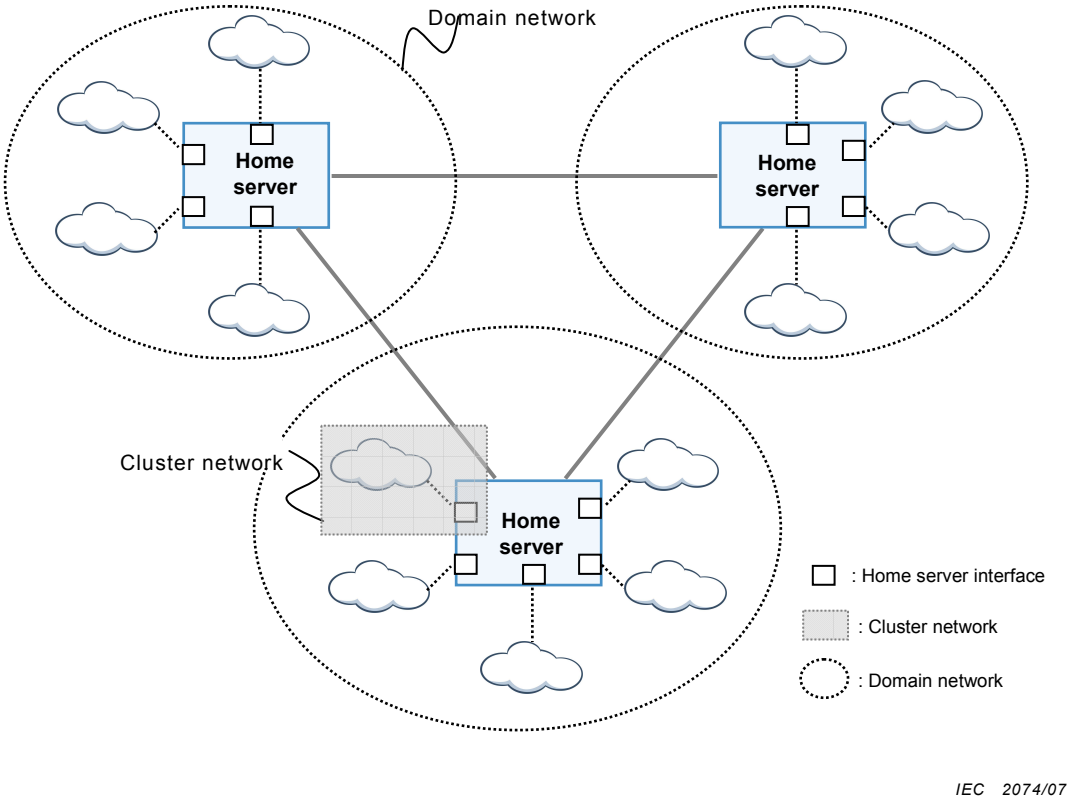


Figure 3 – Cluster and domain network

3.1.6
domain network
collection of devices connected to a single home server

NOTE 1 Devices within the multiple cluster networks linked to the home server and the HSIs managing the cluster network interface constitute the domain network.

NOTE 2 A domain network is shown in Figure 3.

3.1.7
domain address
one of the CCP addressing fields defined in the CCP layer which is used to distinguish multiple domain networks in the home.

NOTE A domain address is shown in Figure 3.

3.1.8
single-HS network
a home network comprising a single home server where there is only one domain network under the single-HS network environment

3.1.9
multi-HS network
a home network with two or more home servers where there are multiple domain networks under the multi-HS network environment

NOTE A multi-HS network is shown in Figure 3.

3.1.10
CCP application program
a CCP application program is positioned at the top of the CCP layer. A CCP application program sends and receives data to/from CCP layer through CCP application programming interface

3.1.11
application program data unit
APDU
the unit of data exchanged among multiple application programs on an equivalent level

3.1.12
CCP application programming interface(API)
the interface between the application program and the CCP layer

3.1.13
lower protocol layers
lower protocol layers refer to all communication protocol layers below the CCP layer, with the exception of the physical layer

NOTE A diagram of lower protocol layers is shown in Figure 5.

3.1.14
lower protocol layers interface
the interface between the APDU and lower protocol layers which are referred to as the lower protocol layers interface

NOTE A diagram of lower protocol layers interface is shown in Figure 5.

3.2 Abbreviations

ACDS	Active Control & Data Service
ACMS	Active Control & Multimedia Service
ACS	Active Control Service
ALLSTAT	All Status
APDU	Application Program Data Unit
API	Application Program Interface
AT	Action Type
ATTR	Attribute
CCP	Common Communication Protocol
CCPADDRVER	CCP Address Version
CCPDEV	CCP Device
CCPH	CCP Header
CCPHDRVER	CCP Header Version
CCPID	CCP Identification
CCPPLEN	CCP Payload Length
CMD	Command
CT	Cast Type
CTRLSTAT	Control Status
DATALEN	Data Length
DEV	Device
DMS	Data & Multimedia Service
DS	Data Service

DSTADDR	Destination Address
ERRCODE	Error Code
FMTS	Final Maximum Transfer Size
FS	Full Service
FSM	Finite State Machine
HDSP	Home Data Service Protocol
HDSPPLEN	HDSP Payload Length
HMSP	Home Multimedia Service Protocol
HNMP	Home Network Management Protocol
HNMPCMD	HNMP Command
HNMPPLEN	HNMP Payload Length
HS	Home Server
HSI	Home Server Interface
MCS	Monitoring Control Service
MON	Monitoring
MS	Multimedia Service
MT	Message Type
MTS	Maximum Transfer Size
MYMTS	My Maximum Transfer Size
PATHLEN	Path Length
PCS	Passive Control Service
PnP	Plug-and-Play
PT	Payload Type
QoS	Quality of Service
REG	Registration
REGSTAT	Registration Status
RES	Response
SEC	Second
SEGNO	Segmentation Number
SEQNO	Sequence Number
SRCADDR	Source Address
STAT	Status
TID	Transaction ID
TT	Traffic Type
UHCP	Universal Home Control Protocol
UHCPPLEN	UHCP Payload Length

3.3 Conventions

UHCP payload content is expressed as italic character as follows:

E.g. <*ITEM*>*argument*</*ITEM*>

4 Requirements

This clause describes the requirements in order to implement the HSI and CCP devices.

4.1 Home server interface requirements

The HSI shall satisfy the following requirements.

4.1.1 Basic requirements for data delivery

– In order to support the device registration process, the HSI shall be able to receive all packets being broadcast to the cluster network from the physical layer and relay them to the CCP layer.

– The HSI shall maintain and manage an address table that contains the CCP addresses of CCP devices in its cluster network as well as the physical addresses used in the cluster network.

– Upon completion of the device registration process, when delivering packets to a particular CCP device within its cluster network, the HSI shall be able to transmit packets from the physical layer using the CCP device's physical address that corresponds to the CCP destination address.

– An HSI shall support the CCP layer as well as the requirements specified in 4.1. HSI can be implemented with the card module-type for the physical interface, communication protocol and the CCP layer.

4.1.2 Functional requirements for HNMP

– After receiving a Device Registration Request (DEV_REG_REQ) packet, the HSI shall be able to send a Device Registration Response (DEV_REG_RES) packet to the CCP device from which the DEV_REG_REQ packet originated.

– The HSI shall be able to store CCP and physical addresses of the CCP device contained in the DEV_REG_REQ packet in the address table.

– The HSI shall be able to allocate a unique Device ID to the CCP device from which the DEV_REG_REQ packet originated. The HSI shall also specify data such as the allocated Device ID, domain address and the HSI's own physical address in DEV_REG_RES packet's HNMP payload prior to transmission.

– The HSI shall be able to notify the fact that a new CCP device has been registered in the home network to all CCP devices within its cluster network as well as all other HSIs in its domain network by transmitting an Add Device (ADD_DEV) packet.

– In order to support CCP device's Plug-and-Play (PnP) function, the HSI shall send on a regular basis an Alive-Check Request (ALV_CHK_REQ) packet to the CCP devices that have completed the device registration process and assess the response packets from them. The HSI shall be able to determine that the non-responsive CCP devices have lost connection with the home network.

– The HSI shall be able to notify the fact that a particular CCP device has been disconnected from the home network to all CCP devices within its cluster network as well as all other HSIs in its domain network by transmitting a Delete Device (DEL_DEV) packet.

4.1.3 Home server interface requirements for unicast, multicast and broadcast

– The HSI shall be able to process all packets received at its physical layer from the physical layer to lower protocol layers before sending them to the CCP layer.

– The HSI shall be able to examine the cluster address in the CCP destination address field of a packet arriving at the CCP layer through lower protocol layers and send the packet to the HSI responsible for the interface of the cluster network that corresponds to the cluster address.

– When a CCP packet arrives from another HSI

  - if the cast type of the received CCP packet is unicast and the CCP destination address is specified, then the HSI receiving the packet shall be able to relay the packet to the particular CCP device within its cluster network,

  - if the cast type of the received CCP packet is multicast, then the HSI receiving the packet shall be able to relay the packet to only the restricted number of the CCP devices within its cluster network in accordance with the content of the multicast,

- if the cast type of the received CCP packet is broadcast, then the HSI receiving the packet shall be able to relay the packet to all CCP devices within its cluster network.

4.2 CCP device requirements

This subclause specifies the requirements for the device registration process needed for CCP devices to perform the plug-and-play function, as well as the requirements for classifying CCP devices according to the CCP functions provided by CCP devices.

4.2.1 Requirements for device registration

– If a CCP device is not aware of the physical address of the HSI of its own cluster network, the HSI's physical address (or network address) shall be obtained through the device registration process.

– For the purpose of device registration, a CCP device shall be able to initially broadcast a device registration request (DEV_REG_REQ) packet to its own cluster network.

– Upon completion of the device registration process, when a CCP device transmits data to a heterogeneous network, it shall send all packets from the physical layer to the HSI of its own cluster network, regardless of the cast type and the CCP destination address.

4.2.2 Requirements for classification of CCP devices

CCP devices refer to devices that support HNMP among the functions provided by the CCP layer and the CCP. CCP devices can be classified according to their processing capabilities and the network technologies being used. This standard shall classify CCP devices into nine categories according to the functions provided by the CCP devices, as shown in Figure 4.

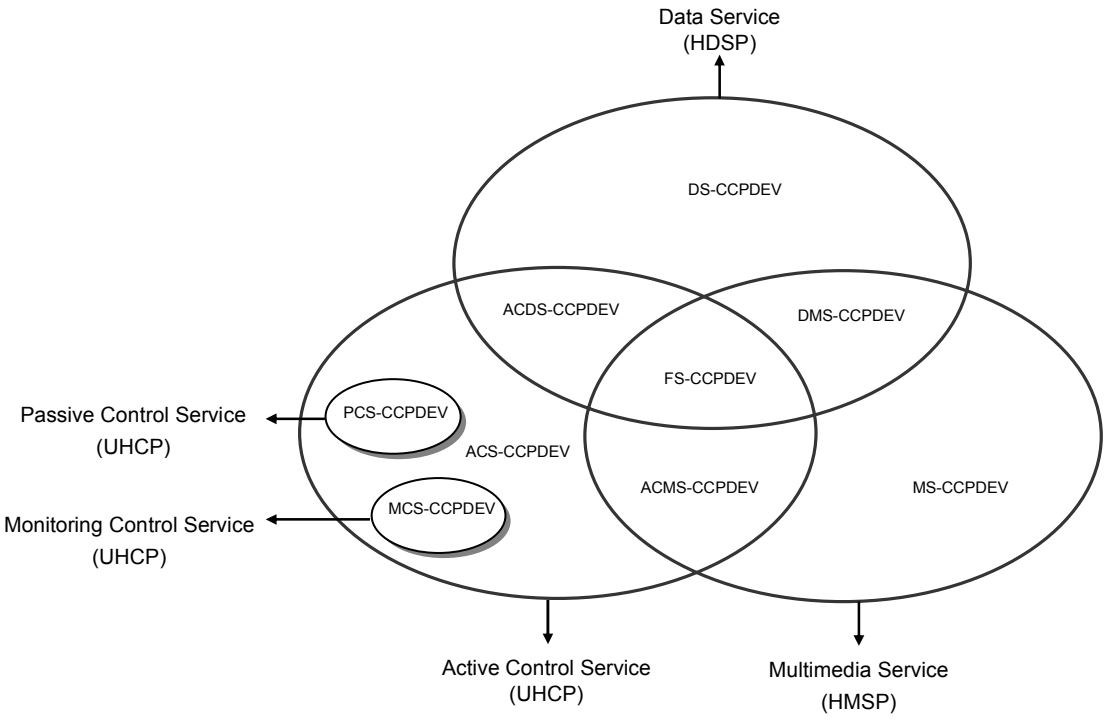


Figure 4 – Classification of CCP devices

– Active Control Service CCP Device (ACS-CCPDEV) is a CCP device that only supports the UHCP function.

– Passive Control Service CCP Device (PCS-CCPDEV) is a CCP device whose UHCP function is carried out by accepting commands from other devices.
– Monitoring Control Service CCP Device (MCS-CCPDEV) is a CCP device that only provides the monitoring capabilities among UHCP functions.
– Data Service CCP Device (DS-CCPDEV) is a CCP device that only provides HDSP.
– Multimedia Service CCP Device (MS-CCPDEV) is a CCP device that only provides the HMSP function.
– Active Control & Data Service CCP Device (ACDS-CCPDEV) is a CCP device that provides UHCP and HDSP functions.
– Active Control & Multimedia Service CCP Device (ACMS-CCPDEV) is a CCP device that provides UHCP and HMSP.
– Data & Multimedia Service CCP Device (DMS-CCPDEV) is a CCP device that provides HDSP and HMSP.
– Full Service CCP Device (FS-CCPDEV) is a CCP device that provides UHCP, HDSP and HMSP functions, and it is also equipped with a display panel for display capabilities.

5 Common communication protocol (CCP) layer

This clause specifies the CCP layer, which allows bilateral communication among heterogeneous networks of various types of application programs.

5.1 CCP layer

The CCP layer provides a common communication channel for devices (or applications executed from the devices) linked to heterogeneous networks. This CCP layer shall be located below all application programs provided by the service providers (or device manufacturers). From OSI 7 layer's perspective, although most network protocols and application programs are similar to the OSI 7 layer structure, they are not clearly divided into seven layers. The location of the CCP layer is variable according to the layer covered by the application program (or the structure of lower protocol layers).

Case 1 in Figure 5 is an example where an application program of a home network device sends data to lower protocol layers through the CCP layer, whereas Case 2 displays an example of an application program of a home network device directly sending data to lower protocol layers without going through the CCP layer.

The application program in Case 1 is transmitting a desired Application Program Data Unit (APDU) to a device connected to a cluster network linked to a heterogeneous network via the home server. On the other hand, the Case 2 example can be used when the application program is to send a desired Application Program Data Unit (APDU) to a device located in the same cluster network using the same network protocol, bypassing the home server. Therefore, in Case 2, all services that have been offered within the same network prior to applying the CCP layer (such as FTP among TCP/IP networks and Telnet service) remain available.

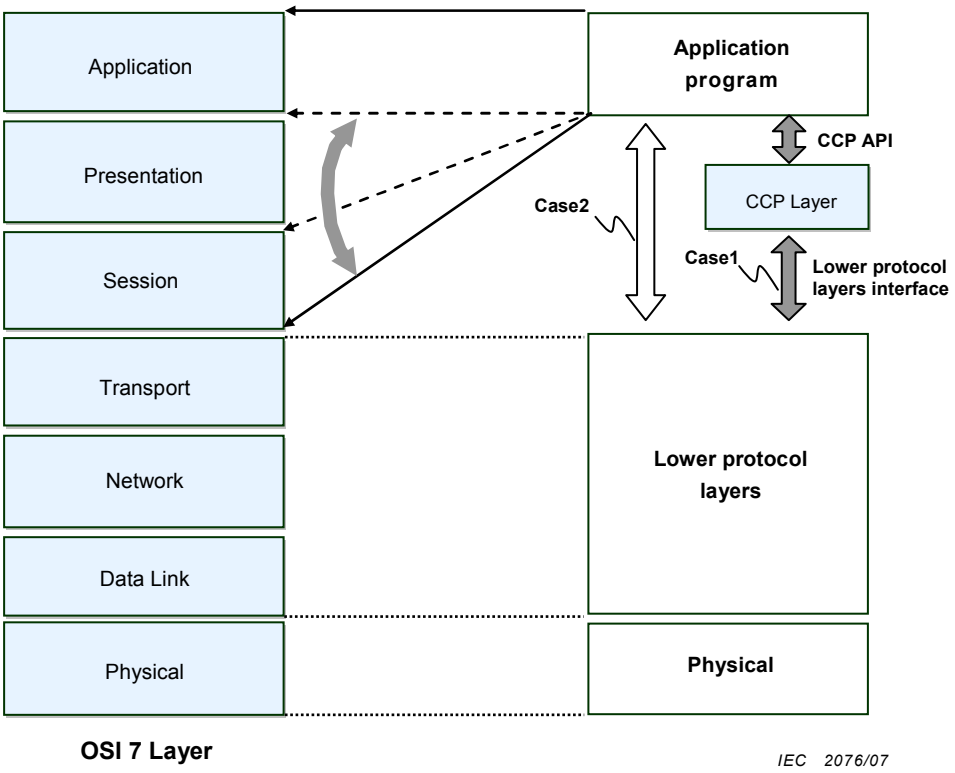


Figure 5 – Definitions of application program, CCP API, lower protocol layers interface, and lower protocol layers

In case of an application program only covering up to the application layer as shown in Figure 6, the CCP layer is located below the application layer. For an application program covering from the application layer to the presentation layer, the CCP layer is located below the presentation layer. For an application program covering from the application layer to the session layer, the CCP layer is located below the session layer.

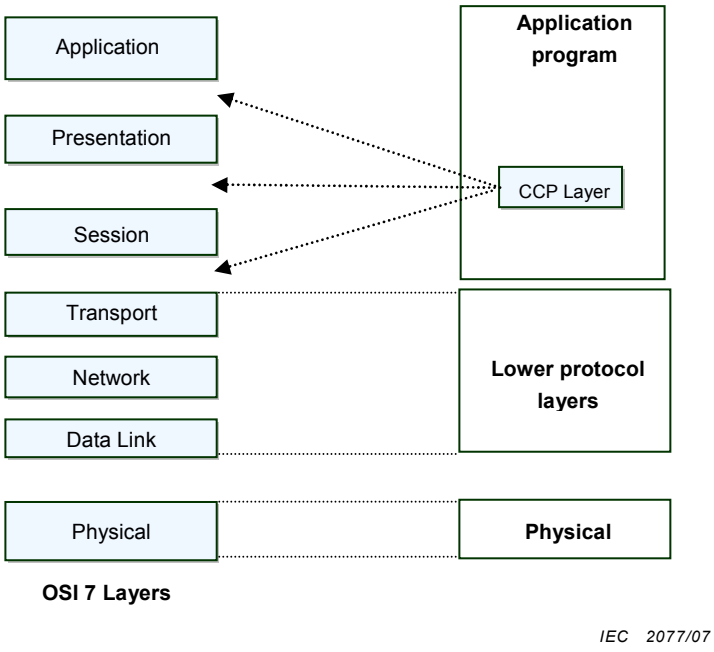


Figure 6 – Location of CCP layer

5.2 Data delivery over heterogeneous networks

This subclause describes the data transmission process using Figure 7. For data communication with a heterogeneous network, an APDU shall go through the CCP layer before being sent to lower protocol layers. For a home network device to transmit an APDU to a heterogeneous network, the application program sends an APDU to the CCP layer through the CCP API, and the CCP layer encapsulates the received APDU with a CCP header and relays it to lower protocol layers.

– DEV_{An}: CCP Device in Network A with Device ID n

– DEV_{Bm}: CCP Device in Network B with Device ID m

– HSI_A, HSI_B: HSI responsible for interface with Network A (or Network B) in Home server.

The data transmission process is as follows.

1) The application program of DEV_{An} sends an APDU to the CCP layer through the CCP API.
2) The CCP layer of DEV_{An} encapsulates the APDU from the application program with a CCP header and sends it to lower protocol layers through the lower protocol layers interface.
3) Lower protocol layers generate a header for each corresponding layer, and the physical layer designates the physical address of HSI_A as the destination address and transmits the packet.
4) The physical layer of HSI_A receives the packet and relays it to lower protocol layers. The Lower Protocol Layers process a header for each corresponding layer and send it to the CCP layer.
5) The CCP layer of HSI_A relays the arrived CCP packet to the home server internal processing block of the home server.
6) The home server internal processing block analyzes the received packet's CCP header information to process the packet and sends it to the HSI(HSI_B) corresponding to the cluster address of the CCP destination address.
7) The CCP layer of HSI_B transmits the received packet to lower protocol layers. When doing so, HSI_B shall search for and retrieve the physical address of DEV_{Bm} corresponding to the CCP destination address from the address table and notify it to the physical layer.
8) The physical layer of HSI_B designates the physical address of DEV_{Bm} as the destination address and transmits the packet.
9) The physical layer of DEV_{Bm} receives the packet and sends it to lower protocol layers, where each header is processed corresponding to each layer and the packet is transmitted to the CCP layer.
10) The CCP layer of DEV_{Bm} delivers the arrived packet to the application program through the CCP API.

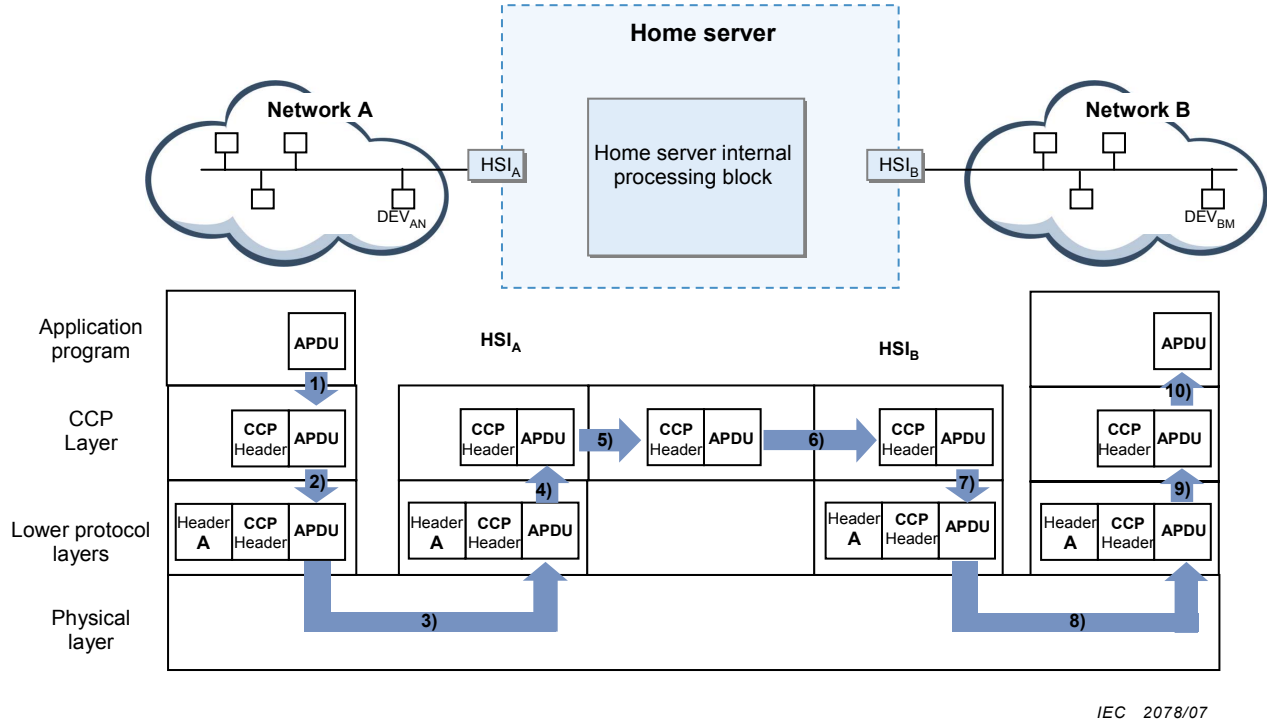


Figure 7 – Example of data transmission over heterogeneous networks using CCP layer

6 CCP addressing

6.1 General

In order to implement intercommunication and interoperability under the home network environment where diverse heterogeneous networks coexist, devices that constitute the home network shall be seen to one another as being linked to a single network regardless of the type of network each device is connected to. This idea can only be materialized if there is an addressing scheme that can be understood between devices of different networks. Furthermore, an addressing scheme that can be understood by humans and application programs will facilitate its use and make implementation of application programs easy.

This clause specifies a new CCP addressing scheme according to the following three functional requirements:

– A new common address structure adequate for the home network environment.

– An addressing mechanism that can be understood by the user as well as the application program.

6.2 An addressing structure to facilitate traffic switching for the home server.Version 0

This subclause describes the CCP address format of version 0. The version field related to the CCP addressing is described in Clause 8. The CCP address format can be changed or extended for future use as other protocols that have their own addressing scheme. Anyway, all CCP devices of CCP address version 0 shall meet the requirements described in 7.2.

In order to satisfy the functional requirements, the CCP address format of version 0 is defined with four bytes, as shown in Figure 8. The CCP address contains three fields: Domain Address, Cluster Address and Device ID.

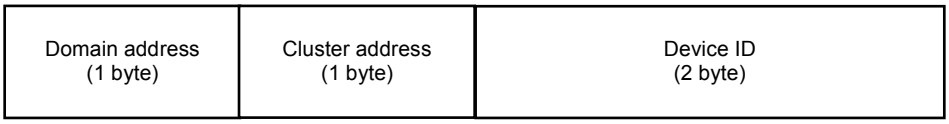


Figure 8 – CCP address format of CCP address by version 0

6.2.1 Domain address

The domain address distinguishes the domain network. When a home network in an office (or home) consists of more than one home server, the domain address is used to differentiate between them. Since one domain network constitutes a single home server, the domain address can be referred to as the home server ID.

The domain address consists of one byte. Therefore, there can be 256 domain addresses (from 0 to 255). Under the single-HS network environment, the domain address of 1 shall be used. Since a device does not know its own domain address before finishing the device registration process described in 9.3, the device shall initially use domain address '0'.

Under the multi-HS network environment where multiple home servers exist, there needs to be a process for allocating a domain address to each home server through the HNMP home server registration procedure. However, this document version does not consider the standards for home server registration under the multi-HS network environment.

6.2.2 Cluster address

The cluster address distinguishes multiple cluster networks linked to a single domain network. Since a single cluster network is linked to a single HSI, the cluster address can be used to switch the packets received within the home server. In order to denote the cluster address as a unique value, the cluster address shall be defined in this standard as the port number of a HSI. Therefore, there can also be 256 cluster addresses (from 0 to 255). Since a device does not know its own cluster address, either before finishing device the registration process described in 9.3, the device shall initially use cluster address '0'.

6.2.3 Device ID field

A device ID field is used to distinguish CCP devices with the same domain and cluster address. Initially, a CCP device is not aware of its device ID. Therefore, each CCP device shall be allocated with its device ID from the HSI through the device registration process, which is one of HNMP functions.

When carrying out the device registration process, the HSI shall be able to assign different device IDs to CCP devices with an identical domain and cluster address.

The device ID is a two-byte field. Therefore, the range of device ID is from 0 to 65,535. Before finishing the device registration process, a device shall initially use device ID '0', and the HSI shall assign device IDs from 1 to 65,535 sequentially, whenever devices register.

7 CCP packet format and fields

7.1 General

This chapter describes the CCP packet structure used in the CCP layer as well as each field used in the CCP packet.

The CCP provides the Home Network Management Protocol (HNMP) for basic home network management services, the Universal Home Control Protocol (UHCP) for device control and

monitoring services, the Home Data Service Protocol (HDSP) for data and messaging services, and the Home Multimedia Service Protocol (HMSP) for multimedia services.

All of the protocols mentioned above (with the exception of UHCP) are signaling protocols based on messages defined in this regulation.

7.2 CCP packet format

Figure 9 shows the CCP packet format of version 0. A CCP packet largely consists of a CCP header and a CCP payload. The CCP header is 28 bytes long, containing CCP Identification, Destination Address, Source Address, Type Field, Reserved Field and CCP Payload Length Field.

The CCP payload comprises the APDUs received from the application program.

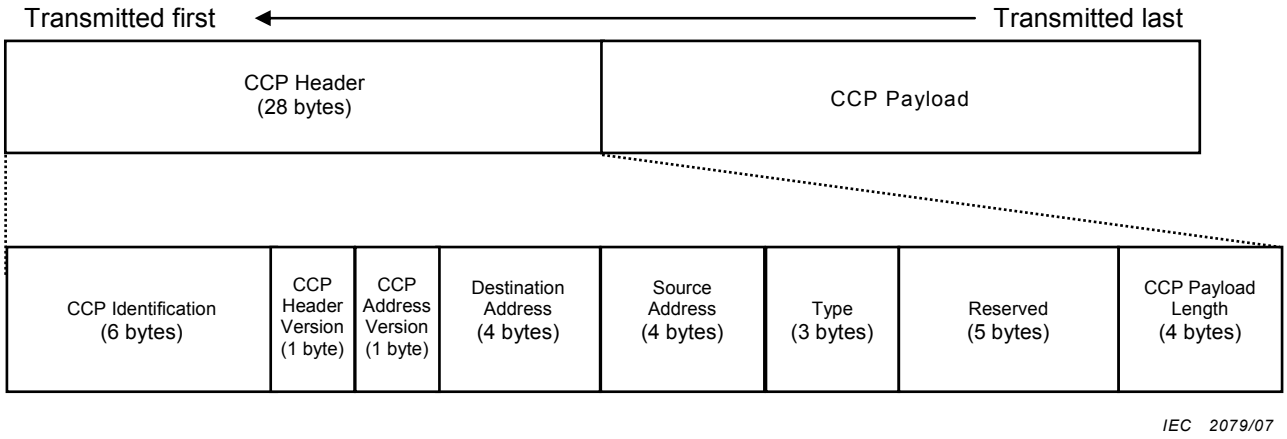


Figure 9 – CCP packet format of CCP header by version 0

7.2.1 CCP identification (CCPID)

CCPID is used to determine whether a packet received at the CCP layer is a CCP packet or not.

If it is a CCP packet, its CCP header is processed and the CCP payload is sent to the application program. If it is not a CCP packet, the packet received from lower protocol layers is transmitted directly to the application program.

Henceforth, the CCP device can communicate with other CCP devices on heterogeneous networks as well as the ones of the same cluster network.

CCPID is six bytes long. The CCPID field is defined as text string of “IECccp”, signifying a CCP header as an ASCII code value (0x494543636370). The CCP layer shall initially check the first four bytes of a packet received from lower protocol layers to assess whether it is a CCP packet.

7.2.2 CCP header version (CCPHDRVER)

Since each field in CCP header can be changed or extended for future use, CCPHDRVER is defined after the CCPID. The length of the CCPHDRVER field is one byte, and it shall be set to 0x00 in case of version 0. This version number will be increased by one if there is any modification or change in the CCP header format and size.

7.2.3 CCP address version (CCPADDRVER)

Since each field in CCP address can be changed or extended for future use, CCPADDRVER is defined after the CCPHDRVER. The length of CCPADDRVER field is one byte, and it shall be set to 0x00 in case of version 0. This version number will also be increased by one if there is any modification or change in the CCP address format and size.

7.2.4 Destination address (DESTADDR)

DADDR indicates the CCP address of the destination CCP device to which the CCP packet is to be delivered. In case that CCPADDRVER is 0, the destination address is four bytes long.

As specified in this clause, a destination address consists of three fields: Domain Address, Cluster Address and Device ID.

7.2.5 Source address (SRCADDR)

SADDR indicates the CCP address of the CCP device transmitting the packet. In case that CCPADDRVER is 0, the source address is four bytes long and also composed of three fields: Domain Address, Cluster Address and Device ID.

7.2.6 Type (TYPE) field

The type fields consist of three fields that indicate the packet delivery method, traffic type and payload type: Cast Type, Traffic Type and Payload Type. The Type fields are three bytes long.

Figure 10 displays the Type fields.

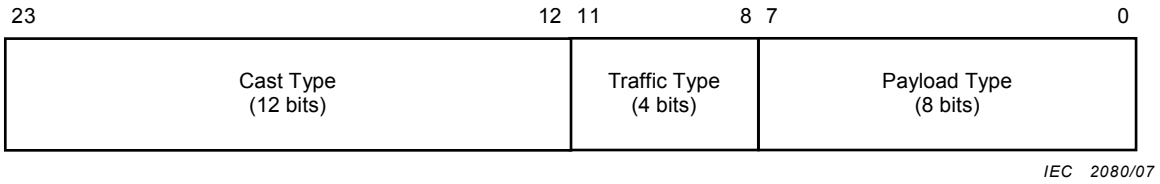


Figure 10 – Type fields

7.2.6.1 Cast type (CT)

The cast type field defines how the source CCP device delivers a packet via the home server. It is a 12-bit field.

As shown in Table 1, this standard provides seven methods: Unicast, Cluster-Multicast, Domain-Multicast, Device-Multicast, Device-Broadcast, HS-Broadcast and Broadcast.

Table 1 – Cast type field

Cast type[11:0]	Notation	Description
0x0XX [a]	Unicast	Used for 1:1 communication between CCP devices.
0x1WW [b]	Domain-Multicast	Used for transmitting packets to HSIs, home servers and all CCP devices within a domain network with the domain address WW.
0x2YY [c]	Cluster-Multicast	Used for transmitting packets to all CCP devices including HSIs within the cluster network with the cluster address YY.
0xF0F	Device-Broadcast	Used when there are multiple domain networks, such as the multi-HS network environment, for transmitting packets only to CCP devices within the domain network.
0xFF0	HS-Broadcast	Used when there are multiple domain networks, such as the multi-HS network environment, for transmitting packets to all home servers and HSIs within the domain network but not to CCP devices.
0xFFF	Broadcast	Used when there are multiple domain networks, such as the multi-HS network environment, for transmitting packets to all HSIs, home servers and CCP devices within the domain network.
Others	reserved	Reserved for future usage.

[a] XX: Don't care
[b] WW: Domain address
[c] YY: Cluster address

7.2.6.2 Traffic type (TT)

The traffic type field is used to specify the data type for the home server to internally process packets when the Quality of Service (QoS) function is to be processed from the home server internal processing block.

The application program may configure the CCP header's traffic type by checking the APDU characteristics. The home server may or may not be designed to support QoS control according to the traffic type field value.

This document does not specify whether the home server should support QoS or the QoS support algorithm.

As shown in Table 2, the traffic type field consists of four bits.

Table 2 – Traffic type field

Traffic type[3:0]	Notation	Description
0000	Don't care	Used when the traffic type does not need to be specified and any QoS processing is acceptable in the home server.
0100	Control Data	Used when the APDU is a signaling message for managing or controlling HNMP, UHCP, HDSP or HMSP.
1000	Non Real-Time Data	Used for general data such as the Internet.
1100	Real-Time Data	Used for real-time data such as audio or video stream.
others	reserved	Reserved for future use.

7.2.6.3 Payload type (PT)

The payload type field indicates the CCP payload type.

There are currently specifications regarding HNMP, UHCP, HDSP and HMSP defined in the CCP. In future, the payload type field can be used in protocols developed for other services using the CCP. As shown in Table 3, the payload type field consists of one byte.

Table 3 – Payload type field

Payload type [7:0]	Notation	Description
0x00	General application	Indicates that the payload type is part of the general application.
0x01	HNMP	Indicates that the payload type is an HNMP-related packet defined in the CCP.
0x02	UHCP	Indicates that the payload type is an UHCP-related packet defined in the CCP.
0x03	HDSP	Indicates that the payload type is an HDSP-related packet defined in the CCP.
0x04	HMSP	Indicates that the payload type is an HMSP-related packet defined in the CCP.
others	reserved	Reserved for future use.

7.2.7 Reserved (RSV) field

The five-byte reserved field is reserved for future use.

7.2.8 CCP payload length (CCPPLEN) field

The four-byte CCP payload length field indicates the byte size of the APDUs in the payload.

7.2.9 CCP payload field

The CCP payload field comprises the APDUs generated by the application program. The CCP payload is written with the signaling messages of the HMSP, HNMP, UHCP and HDSP defined in the CCP.

8 Home network management protocol (HNMP)

8.1 General

The HNMP is used by the CCP to perform home network management functions. Moreover, the HNMP manages home network resources and provides PnP functions for the home server as well as the CCP devices.

The HNMP is a basic packet-oriented signaling protocol provided by the CCP. Simple request and response packets are defined in the HNMP.

All CCP devices under the home network environment consisting of heterogeneous networks shall at least support the device registration function among the HNMP functions. Other HNMP functions can be selectively implemented according to the CCP devices' processing capability and application areas.

8.2 HNMP packet format

An HNMP packet comprises the CCP payload field. When the application program transmits an HNMP packet, it shall set the payload type of CCP header's type fields as 0x01 in order to notify that the CCP payload type is the HNMP packet.

Figure 11 displays the HNMP packet format. An HNMP packet consists of the eight-byte HNMP header and the HNMP payload. The HNMP header contains four fields: the Transaction ID, HNMP Command, Reserved and HNMP Payload Length.

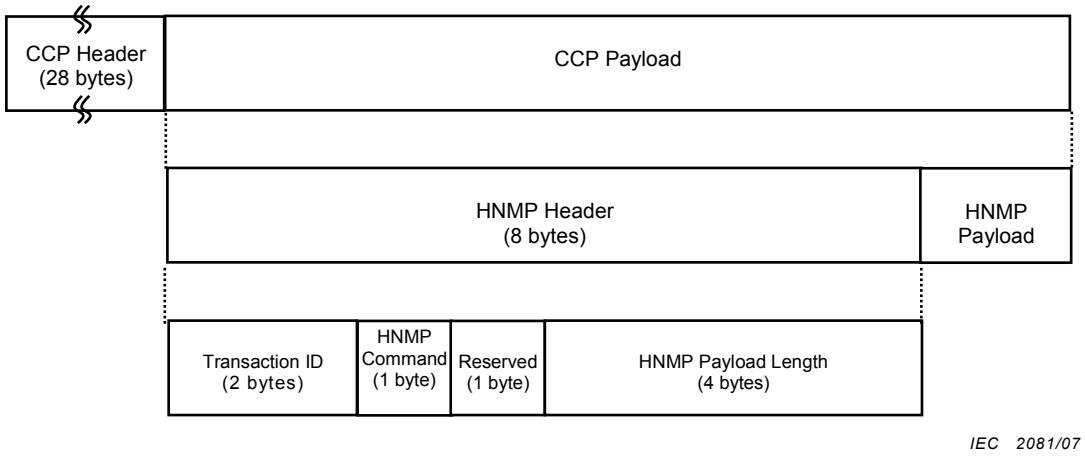


Figure 11 – HNMP packet format

8.2.1 Transaction ID (TID)

When an application program sends and receives multiple request and response packets, the transaction ID allows the application program to find the response packet for the corresponding request packet. The application program can assign random values for the transaction ID. The size of the transaction ID is two bytes.

8.2.2 HNMP command (HNMPCMD)

The HNMP command provides functions related to the home server registration, device registration, device management, alive-check, address information and traffic information. The HNMP command is one-byte long.

8.2.3 Reserved (RSV) field

The one-byte reservation field is reserved for future use.

8.2.4 HNMP payload length (HNMPPLEN) field

The HNMP payload length field indicates the byte size of the HNMP payload. HNMP payload length is a four-byte field.

8.2.5 HNMP payload

An HNMP command may or may not require an HNMP payload according to the HNMP command type.

DEV_REG_REQ, DEV_REG_RES, ADD_DEV, DEL_DEV and DEV_INFO_RES are examples of packets that require an HNMP payload.

8.3 Home server registration

The HNMP provides the home server registration function. Home server registration refers to the process of determining domain addresses among the home servers using the signaling packets of home servers that constitute multiple domain networks under the multi-HS network environment.

This document only considers the single-HS network environment and does not contain detailed information regarding home server registration.

8.4 Device registration

Device registration refers to the process of home server storing a human-readable name of a device and allocating a device ID used in the CCP addressing scheme to a CCP device. When a new CCP device is to be linked to a home network, it shall go through the device registration process. This allows the CCP device to have all CCP address fields such as Domain Address, Cluster Address and Device ID, through which the home server is able to notify other CCP devices that a new CCP device has been connected to the home network.

In order to perform the device registration process, all CCP devices within the home network shall be allocated with a unique address. Therefore, when carrying out the device registration process, the home server shall be able to allocate different device IDs to CCP devices with identical domain and cluster addresses.

Commands related to device registration are Device Registration Request (DEV_REG_REQ) and Device Registration Response (DEV_REG_RES) packets.

Figure 12 displays the packet format, and Figure 13 is an example command sequence related to device registration.

8.4.1 Device registration request (DEV_REG_REQ) packet

Every CCP devices to be registered in a home network shall broadcast a DEV_REG_REQ packet to its cluster network. The broadcast packet is received by the home server HSI and delivered to the CCP layer. At this time, the CCP device shall fill the payload of a DEV_REG_REQ packet with its text-based name which is human-readable, physical address (or network address) size and physical address (or network address). As shown in Figure 12, support protocol field is used to indicate which protocols the CCP device additionally supports among UHCP, HDSP and HMSP. If the CCP device supports UHCP and HDSP, then the UHCP and HDSP fields should be set to 1 s.

The HSI shall notify the CCP device requesting registration with the Domain Address, Cluster Address, a newly allocated Device ID and its physical address (or network address). The HSI shall also register the CCP device's CCP address and physical address (or network address) in the address table for mutual communication between the CCP device and the HSI once the registration process is complete.

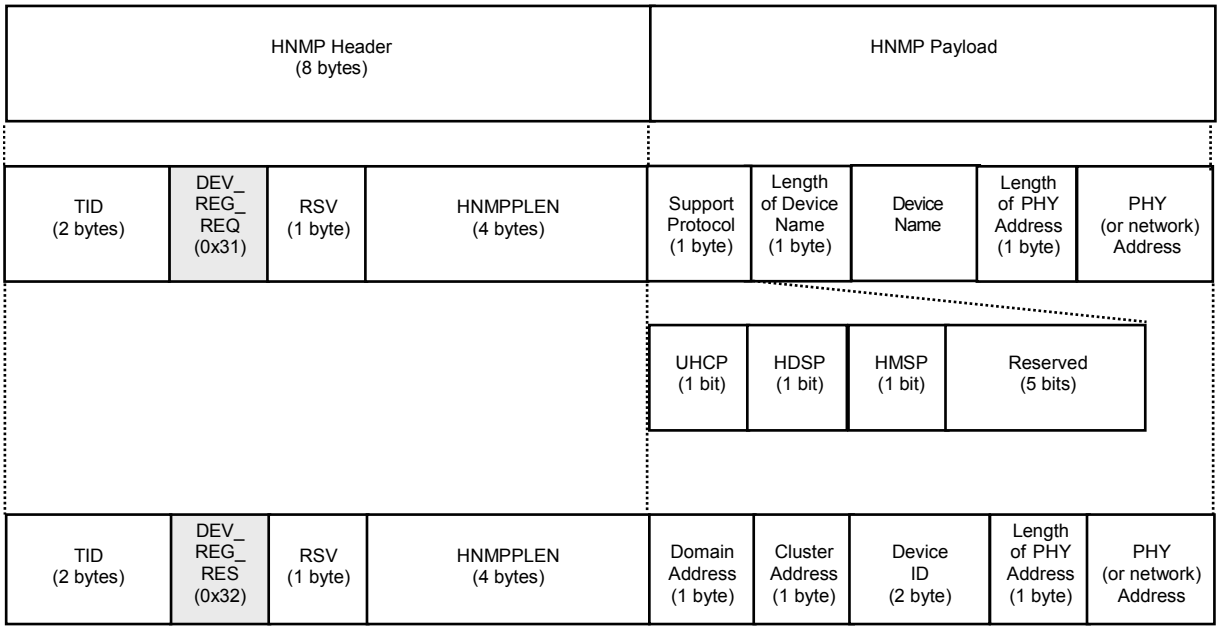
The reason why a device and a HSI exchange their physical addresses (or network addresses) is that both of them actually use physical addresses or network addresses when communicating with each other.

In case that the lower protocol layers of a device have both a network address and a physical address, it is easy to use network address rather than physical address when implementing an interface between the CCP layer and the lower protocol layers. On the other hand, in case the lower protocol layers of a device have only a physical address, there is no way but to use the physical address when interfacing between the CCP layer and the lower protocol layers.

The physical address or network address size is expressed in bytes. The HNMP command field of a DEV_REG_REQ packet is 0x31.

If there is no response for a timeout period after a DEV_REG_REQ packet is sent, the CCP device to be registered performs retries. If there is still no response, the home server is considered not ready and the procedure is repeated after a waiting period.

With respect to the timeout period, the number of retries and waiting periods are not specified in detail in this document because it is related to the communication speed influenced by the processing power of a device and the physical network speed.



IEC 2082/07

Figure 12 – DEV_REG_REQ and DEV_REG_RES packets

8.4.2 Device registration response (DEV_REG_RES) packet

Upon receiving a DEV_REG_REQ packet, the HSI shall return a DEV_REG_RES packet to the DEV_REG_REQ sender device as a response within the timeout period. The HSI shall be able to store the physical address (or network address) of the device included in the DEV_REG_REQ packet payload by mapping it with the CCP address.

Furthermore, the HSI shall write its own domain address, cluster address, the device ID to be allocated to the device, physical address (or network address) size and the physical address (or network address) in the HNMP payload. The physical address (or network address) size is indicated in bytes. In order to send the DEV_REG_RES packet physically, the HSI shall use the CCP device's physical address (or network address) as the destination physical address (or network address).

After receiving the DEV_REG_RES packet, the CCP device configures its own domain address, cluster address and device ID and stores the physical address (or network address) of the HSI responsible for its cluster network. Henceforth, when the CCP device transmits data to another CCP device in a heterogeneous network, it simply has to configure the HSI's physical address (or network address) as the destination physical address (or network address) at the physical (or network) layer. Then, the HSI can receive the data at the physical layer and extracts a CCP packet at the CCP layer. Finally the HSI can forward the packet to another HSI to which the destination CCP device is linked.

The HNMP command field of the DEV_REG_RES packet is 0x32.

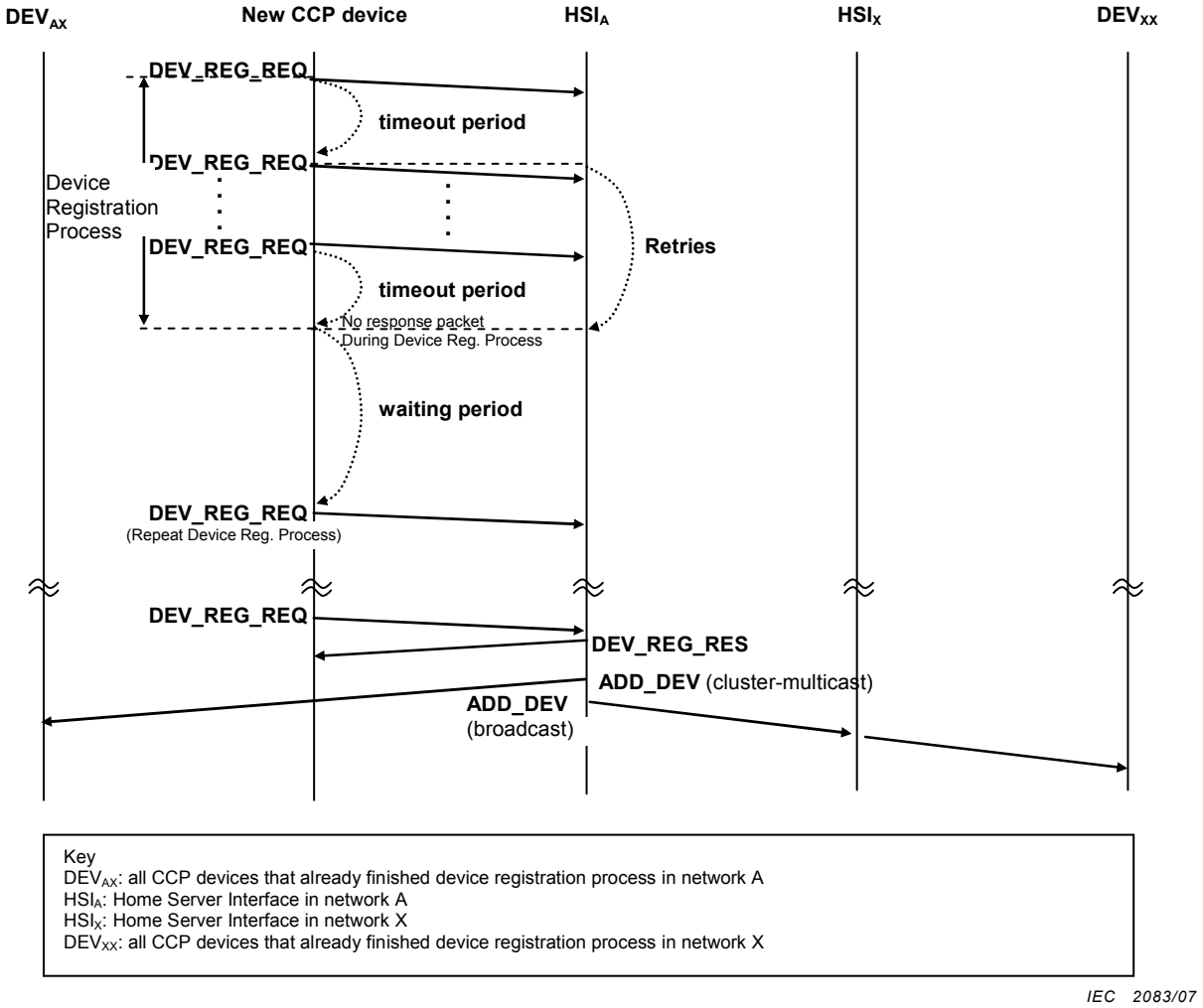


Figure 13 – Example of HNMP command sequence for device registration

8.5 Device management

The HNMP provides HSIs and CCP devices with the address table management function through commands that allow addition and deletion of a device. It can also initialize a particular CCP device through the Device Initialization command.

Requirements for the HSIs (or CCP devices) receiving Add (or Delete) Device packets are as follows:

– The HSI shall be able to deliver the received Add (or Delete) Device packet to another cluster network's HSI.

– The HSI shall be able to deliver the Add (or Delete) Device packet from another HSI to all devices within its own cluster network.

– The HSI (or the CCP device) shall be able to add a new CCP address to the address table that it manages.

– The HSI (or CCP device) receiving a Delete Device packet shall be able to delete the address of the corresponding CCP device from the address table.

– Functions related to device addition and deletion shall be applied only to a limited number of HSIs and CCP devices with sufficient processing capabilities and memory.

The HNMP packet related to device addition, deletion and initialization consists of the Add Device (ADD_DEV), Delete Device (DEL_DEV) and Initialize Device (INI_DEV) packets as shown in Figure 14.

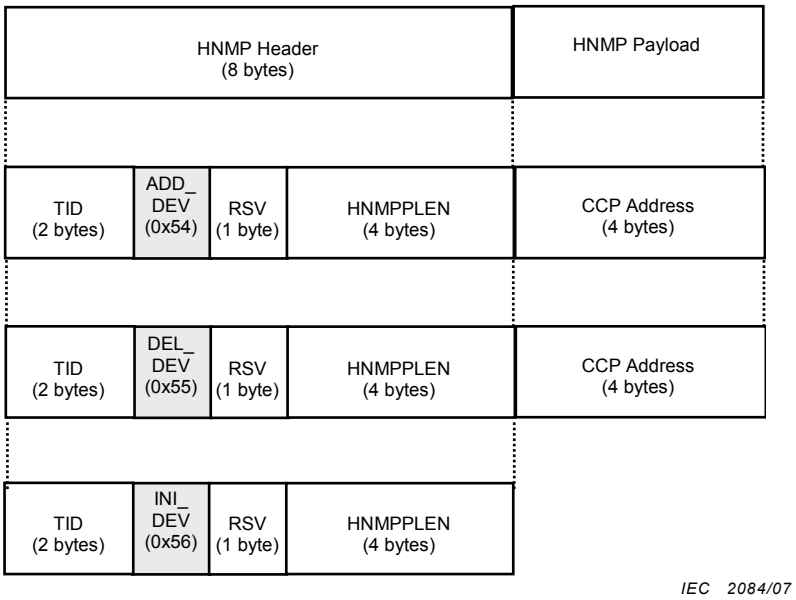


Figure 14 – ADD_DEV, DEL_DEV and INI_DEV packets

A CCP device (or HSI) can assess a particular CCP device's proper operation status and home network connection status by using HNMP's Alive-Check command.

The CCP device receiving an Alive-Check command shall be able to notify the fact that it is still connected to the home network and operating properly. In addition, in order to provide CCP devices' PnP functions, the HSI shall regularly check proper operation of the CCP devices within its cluster network using the Alive-Check command.

HNMP packets related to the Alive-Check command are Alive-Check Request (ALV_CHK_REQ) and Alive-Check Response (ALV_CHK_RES), as shown in Figure 15.

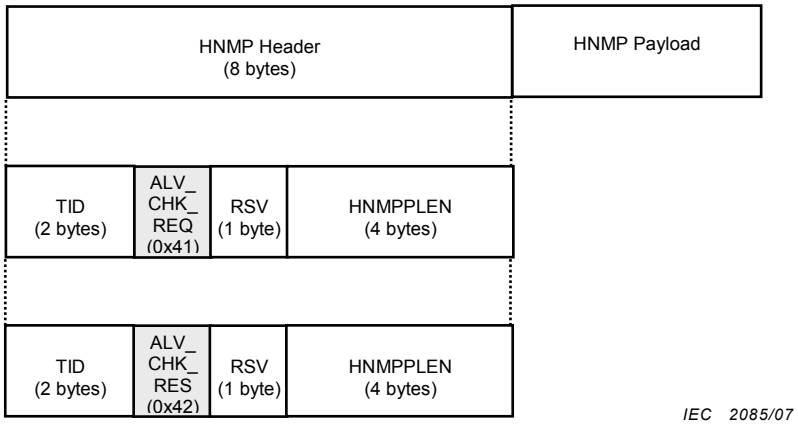


Figure 15 – ALV_CHK_REQ and ALV_CHK_RES packets

Figure 16 displays an example of HNMP packet's command sequence related to device management.

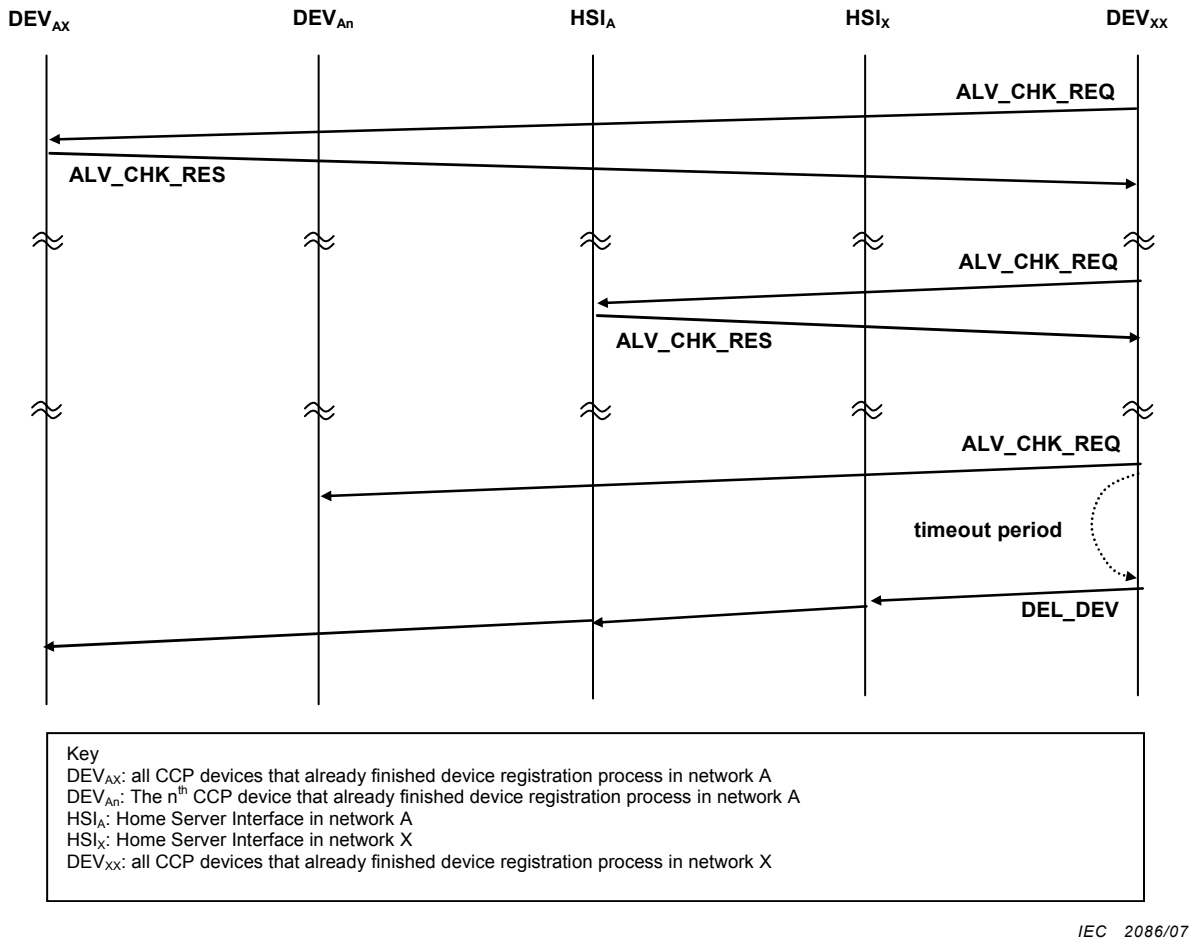


Figure 16 – Example of HNMP command sequence for device management

8.5.1 Add device (ADD_DEV) packet

An ADD_DEV packet is used to notify other CCP devices or HSIs of the fact that a new CCP device is connected to the home network. HSIs or CCP devices receiving the ADD_DEV packet do not have to send a response packet.

Prior to transmission, the four-byte CCP address of the CCP device newly connected to the home network is provided in ADD_DEV packet's HNMP payload. The ADD_DEV packet is usually transmitted with the cast type of CCP header's type fields set as broadcast. A CCP device receiving the ADD_DEV packet can process or discard it according to its address table management capability. Devices with address table management capability and a display panel shall be able to update the address table and indicate on the display panel that a new CCP device has been added. The HNMP command field of the ADD_DEV packet is "0x54".

8.5.2 Delete device (DEL_DEV) packet

A DEL_DEV packet is used to notify other CCP devices and HSIs in the home network of the fact that a particular CCP device has been removed from the home network. For example, a CCP device (or HSI) transmits a DEL_DEV packet if it has assessed that a particular CCP device no longer exists in the home network due to malfunctioning or other reasons. HSIs (or CCP devices) receiving a DEL_DEV packet do not have to send a response packet. Prior to transmission, the four-byte CCP address of the CCP device removed from the home network is provided in DEL_DEV packet's HNMP payload. The DEL_DEV packet is usually transmitted with the cast type of CCP header's type fields set as broadcast. A CCP device receiving the DEL_DEV packet can process or discard it according to its address table management

capability. Devices with address table management capability and a display panel shall be able to update the address table and indicate on the display panel that a particular CCP device has been removed. The HNMP command field of the DEL_DEV packet is 0x55.

8.5.3 Initialize device (INI_DEV) packet

An INI_DEV packet can be sent from a CCP device or a HSI to another CCP device, and the CCP device receiving an INI_DEV packet shall initialize itself. The initialized CCP device does not have to go through another device registration process. In other words, the domain address and the device ID shall be able to maintain their values after initialization. The HNMP command field of the INI_DEV packet is 0x56.

8.5.4 Alive-check request (ALV_CHK_REQ) packet

A CCP device (or home server) can transmit an ALV_CHK_REQ packet to another CCP device (or home server). An ALV_CHK_REQ packet is used to check whether a particular device is connected to the home network and operating properly. A CCP device or home server receiving an ALV_CHK_REQ packet shall return a response packet within two seconds. In order to support devices' PnP functions, a HSI shall regularly send ALV_CHK_REQ packets to devices within its cluster network and receive responses. If a HSI does not receive a response from a CCP device for a timeout period after sending an ALV_CHK_REQ packet, it retries the process *N* times. If there is no response after *N* retries, the corresponding CCP device shall be regarded as having been removed from the home network, and the HSI sends a DEL_DEV packet to other HSIs and all CCP devices within the cluster network. This regulation does not specify the number of retries (*N*). The HNMP command field of the ALV_CHK_REQ packet is 0x41.

8.5.5 Alive-check response (ALV_CHK_RES) packet

A CCP device (or HSI) that receives an ALV_CHK_REQ packet shall return an ALV_CHK_RES packet to the device where the ALV_CHK_REQ packet originated within a timeout period to notify that it is connected to the home network and operating properly. The HNMP command field of the ALV_CHECK_RES packet is 0x42.

8.6 Address and name information of devices

When a CCP device requests a HSI for CCP address and name information for all CCP devices linked to the home network, the HNMP supports the function of notifying the CCP address and name information of all CCP devices. This function can be used by a newly connected CCP device to request the CCP address information of existing CCP devices or a CCP device to update address information of other CCP devices that it manages.

As shown in Figure 17, there are two types of HNMP packets related to the address and name information: Device Information Request (DEV_INFO_REQ) and Device Information Response (DEV_INFO_RES).

A DEV_INFO_REQ packet can be transmitted in one of two ways.

First, if a particular CCP device transmits a DEV_INFO_REQ packet to a HSI as unicast, it can obtain address information of all CCP devices within its own cluster network.

Second, if a particular CCP device transmits a DEV_INFO_REQ packet as HS-broadcast, it can obtain address information of all CCP devices in all cluster networks within the home network.

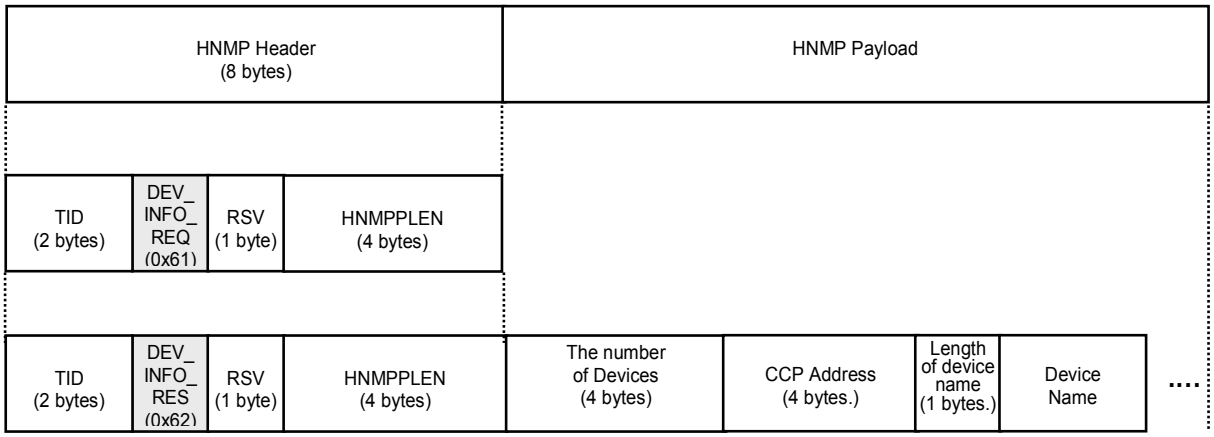


Figure 17 – DEV_INFO_REQ and DEV_INFO_RES packets

Figure 18 displays an example command sequence of the HNMP packet related to the device address and name information.

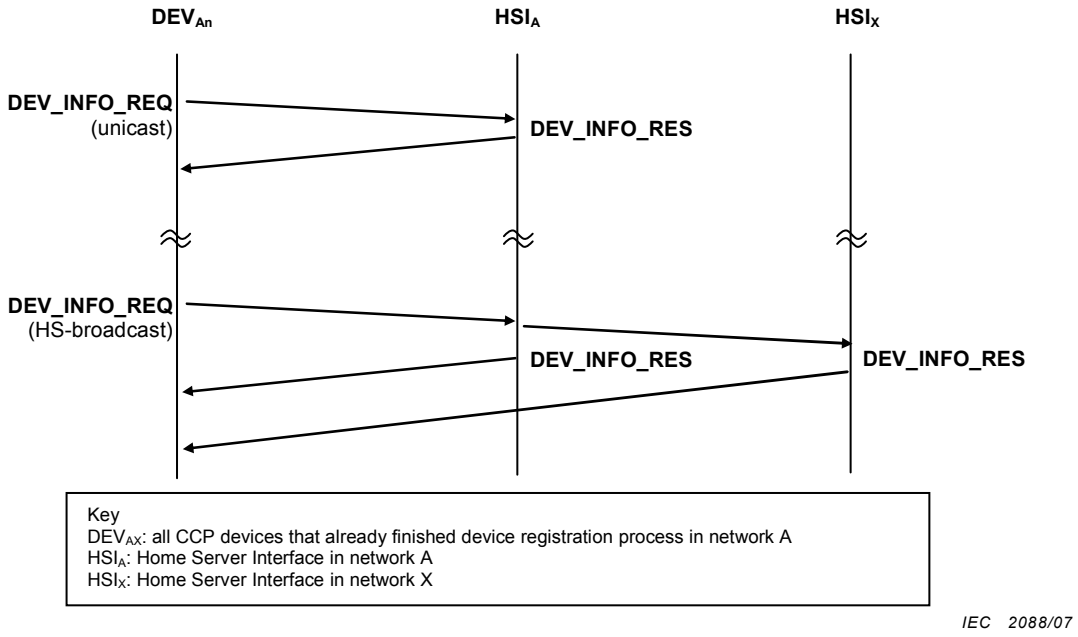


Figure 18 – Example of HNMP command sequence for retrieving device address and name information

8.6.1 Device address and name information request (DEV_INFO_REQ) packet

A DEV_INFO_REQ packet is used by a CCP device with address table management capability to request a HSI for address and name information of all CCP devices linked to the cluster network.

A HSI receiving a DEV_INFO_REQ packet shall write the CCP address and name information of all CCP devices registered in its cluster network in the HNMP payload of the response packet and return the response packet to the CCP device that has sent the request packet. The HNMP command field of the DEV_INFO_REQ packet is 0x61.

8.6.2 Device address and name information response (DEV_INFO_RES) packet

A HSI receiving a DEV_INFO_REQ packet shall return a DEV_INFO_RES packet to the CCP device that has requested the address and name information of CCP devices linked to itself.

In Figure 17, “The number of Devices” field shall be four byte long and shall contain the number of CCP device information listed in the payload of DEV_INFO_RES packet. CCP address, length of device name and device name fields represent information of a specific CCP device. The CCP address field shall be four byte long and contain a CCP address of a specific CCP device. Length of device name field shall be one byte long and contain the length of a device name in byte units. The device name field shall contain the name of a specific CCP device in simple text format. Its size shall be equal to the length of device name field.

A DEV_INFO_RES packet can be transmitted in one of two ways.

First, if a HSI received a DEV_INFO_REQ packet has *N* devices linked to itself, then the HSI makes a DEV_INFO_RES packet of which the number of devices field is set to 1, and it sends it *N* times for each device.

Second, a HSI that received a DEV_INFO_REQ packet makes only one DEV_INFO_RES packet which have address and name information of all CCP devices linked to itself.

The HNMP command field of the DEV_INFO_RES packet is 0x62.

8.7 Other management functions

In addition to the functions specified above, new management packets can be defined and applied when a home network is expanded or new functions become necessary.

9 Universal home control protocol (UHCP)

CCP provides UHCP for device control and monitoring functions. UHCP is a message-based protocol that can provide home network resource controlling and monitoring functions. One of the fundamental protocols provided by CCP, UHCP is used for additional services offered by devices with CCP capabilities.

Since basic home network management is performed by HNMP, not all devices need to be embedded with UHCP. However, if UHCP is implemented according to CCP device processing capability, it can provide additional home network services such as control and monitoring.

9.1 UHCP packet format

Figure 19 displays the UHCP packet format. The UHCP packet is located in the CCP packet payload. In order to indicate that the CCP payload type is a UHCP packet, the UHCP application program shall set the payload type of the CCP header’s type fields as 0x02. An UHCP packet consists of an eight-byte UHCP header and the UHCP payload. The UHCP header contains five fields: Transaction ID, message type, action type, reserved and UHCP payload length.

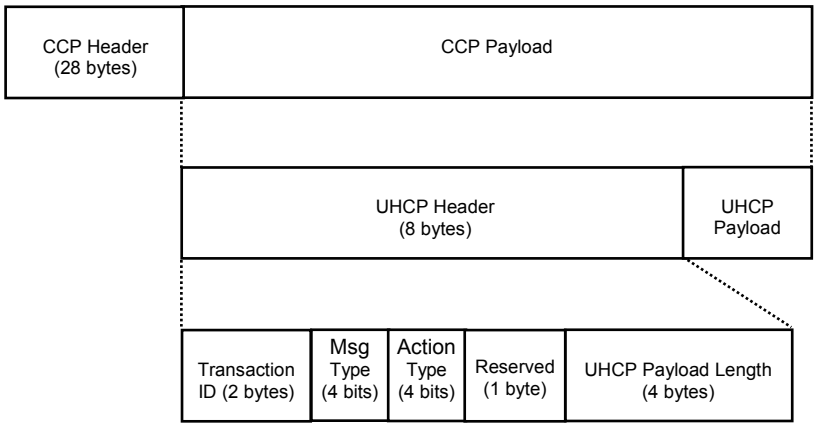


Figure 19 – UHCP packet format

9.1.1 Transaction ID (TID)

When an application program sends and receives multiple request packets and response packets, the transaction ID is used by the application program to find a response packet for a particular request packet. The application program can assign random values for the transaction ID. The transaction ID is a two-byte field.

9.1.2 Message type (MT) and action type (AT)

The message type field is four bits long and it consists of the execution message, query message and notification message. The action type field is four bits long and it indicates detailed actions according to the message type.

Figure 20 displays the message type and action type defined by the UHCP.

UHCP (8 bytes)					UHCP Payload
TID	EXE (0x1)	REG (0x1)	RSV	UHCPPLEN	UHCP Payload
TID	EXE (0x1)	CTRL (0x2)	RSV	UHCPPLEN	UHCP Payload
TID	EXE (0x1)	RES OK (0xE)	RSV	UHCPPLEN	
TID	EXE (0x1)	RES NOK (0xF)	RSV	UHCPPLEN	
TID	QUE (0x2)	REG STAT (0x1)	RSV	UHCPPLEN	
TID	QUE (0x2))	CTRL STAT (0x2)	RSV	UHCPPLEN	
TID	QUE (0x2)	ALL STAT (0x3)	RSV	UHCPPLEN	
TID	QUE (0x2)	RES OK (0xE)	RSV	UHCPPLEN	UHCP Payload
TID	QUE (0x2)	RES NOK (0xF)	RSV	UHCPPLEN	
TID	NTFY (0x3)	N/A (0x0)	RSV	UHCPPLEN	UHCP Payload

IEC 2090/07

Figure 20 – Message type and action type fields of UHCP packet

9.1.3 Reserved (RSV) field

The one-byte reserved field is for future use.

9.1.4 UHCP payload length (UHCPPLEN) field

The UHCP payload length field indicates the byte size of the UHCP payload. The size of the length field is four bytes.

9.1.5 UHCP payload

The UHCP payload may or may not be required according to the message type and the action type.

9.2 Execution messages (EXE)

If the UHCP header's MT field is 0x1, it indicates that the content of the UHCP payload is an execution message. There are four action types according to the execution message: EXE_REG, EXE_CTRL, EXE_RESOK and EXE_RESNOK.

9.2.1 Execution of registration (EXE_REG)

The AT field of an EXE_REG message is 0x1. In order for an UHCP-supporting device to use UHCP functions after completing the HNMP device registration process, it shall first register

its control items and device attributes such as the device type, location, manufacturer and network technology, to the HSI in the domain network. The EXE_REG message is used for registering this kind of information. The device attributes and control items are indicated in the payload.

As shown in Figure 21, a HSI receiving an EXE_REG message shall send a response message within the timeout period. If the device that has sent the EXE_REG message does not receive a response within the timeout period, it carries out retries. If it fails to receive a response after retries of EXE_REG, the home server is regarded as problematic, and the process is repeated after thirty seconds. Since the device must register its attributes through the process in order to perform UHCP functions, if there is no response from the HSI, it continuously repeats the process until registration is complete.

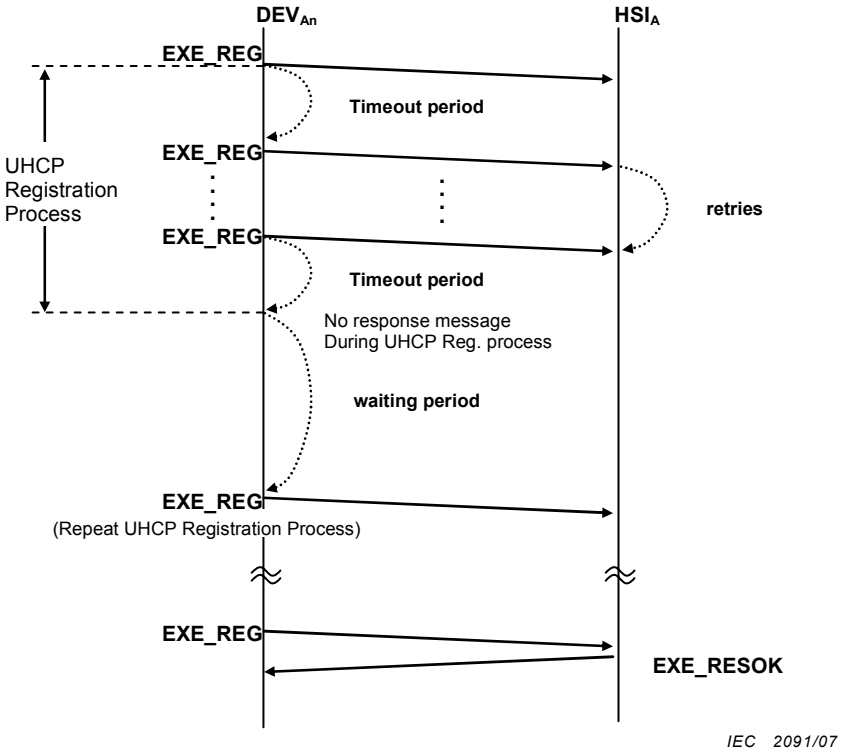


Figure 21 – Example of registration process

9.2.2 Execution of control (EXE_CTRL)

The AT field of an EXE_CTRL message is 0x2. A device receiving an EXE_CTRL packet shall carry out the command according to the payload content and send a corresponding response message.

Figure 22 displays the sequence of an EXE_CTRL message. If the CCP device that has sent an EXE_CTRL message does not receive an EXE_RESOK response message within timeout period, it sends an EXE_RESNOK message to the UHCP application program. Even if the UHCP application program does not receive a response packet for an EXE_CTRL message, it does not attempt a retry. However, an application program using UHCP functions can decide the timeout period and perform a retry according to a user request.

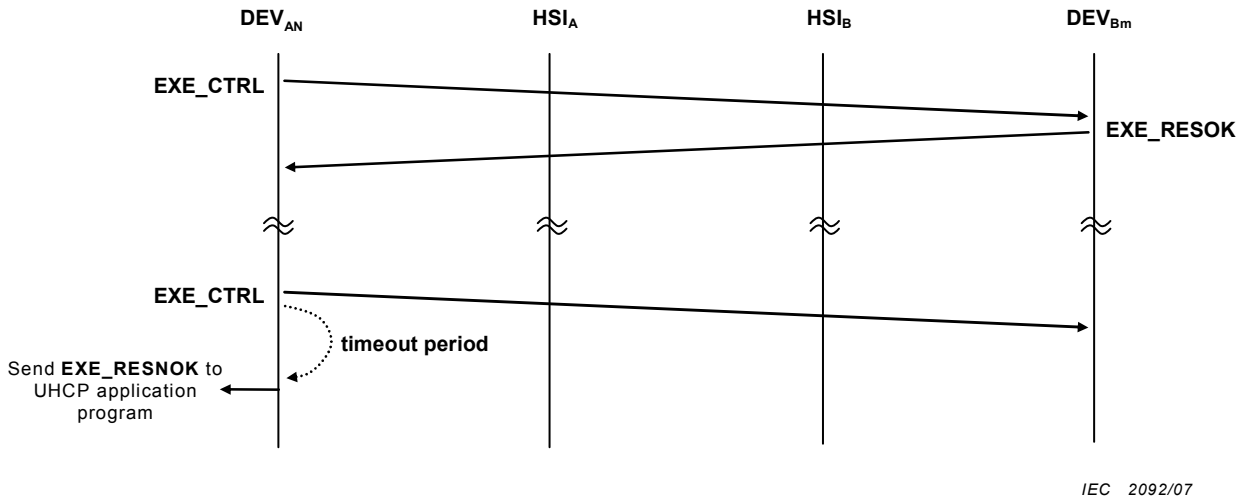


Figure 22 – Example of EXE_CTRL message

9.2.3 Response OK (EXE_RESOK)

The AT field of an EXE_RESOK message is 0xE. A device receiving an EXE_REG or EXE_CTRL packet shall complete the device registration or carry out a command and return an EXE_RESOK message indicating that the task was performed successfully. An EXE_RESOK message does not have an UHCP payload. (Refer to Figure 20)

9.2.4 Response NOK (EXE_RESNOK)

The AT field of an EXE_RESNOK message is 0xF. A device receiving an EXE_REG or EXE_CTRL packet shall return an EXE_RESNOK message to indicate that the registration process or a command has failed. In case of the EXE_REG message, if the device that has sent an EXE_REG message fails to receive an EXE_RESOK or EXE_RESNOK response due to an unstable home server, the device shall be able to send an EXE_RESNOK message to the UHCP application on its own. An EXE_RESNOK message does not have an UHCP payload as shown in Figure 20.

9.3 Query messages (QUE)

If the MT field of an UHCP header is 0x2, it indicates that the UHCP payload content is a query or monitoring message. There are five types of execution messages in the action type according to the query message: QUE_REGSTAT, QUE_CTRLSTAT, QUE_ALL, QUE_RESOK and QUE_RESNOK, as described below.

9.3.1 Query of registration status (QUE_REGSTAT)

The AT field of a QUE_REGSTAT message is 0x1. It is used by a home server or another device to request for a particular device's attributes. A device receiving a QUE_REGSTAT message shall provide its attribute information in QUE_RESOK message's UHCP payload and return it. If there is no response within the timeout period, the device that has sent the QUE_REGSTAT message shall send a QUE_RESNOK message to the UHCP application program. Figure 23 displays an example of a QUE_REGSTAT message. A QUE_REGSTAT message does not have an UHCP payload, as shown in Figure 20.

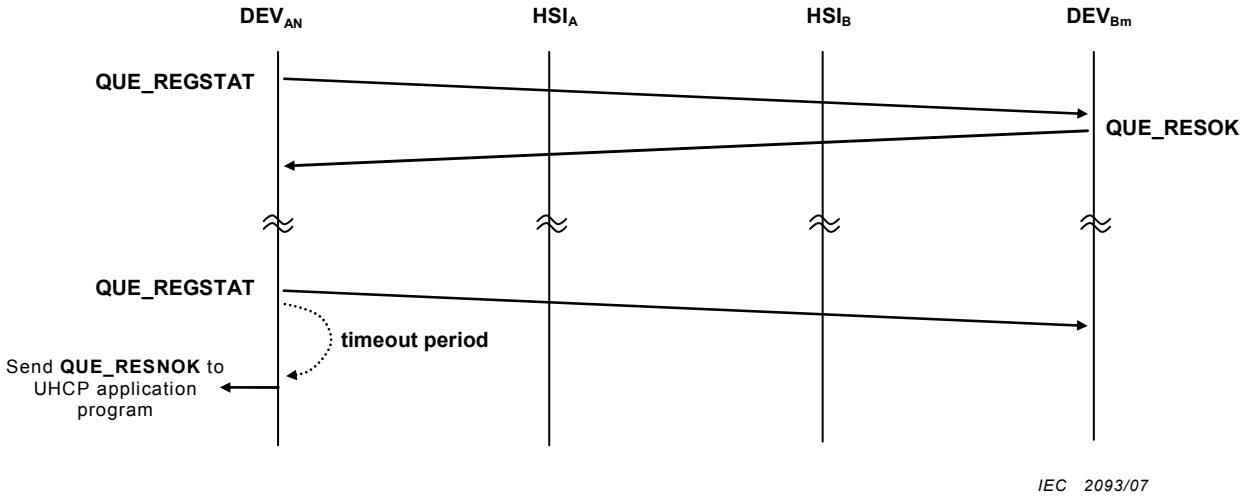


Figure 23 – Example of QUE_REGSTAT message

9.3.2 Query of control status (QUE_CTRLSTAT)

The AT field of a QUE_CTRLSTAT message is 0x2. It is used by a home server or another device to request for information regarding a particular device's command status. A device receiving a QUE_CTRLSTAT message shall provide its command status information in QUE_RESOK message's UHCP payload and return it. If there is no response within timeout period, the device that has sent the QUE_CTRLSTAT message shall send a QUE_RESNOK message to the application program. Figure 24 displays an example of a QUE_CTRLSTAT message. A QUE_CTRLSTAT message does not have an UHCP payload as shown in Figure 20.

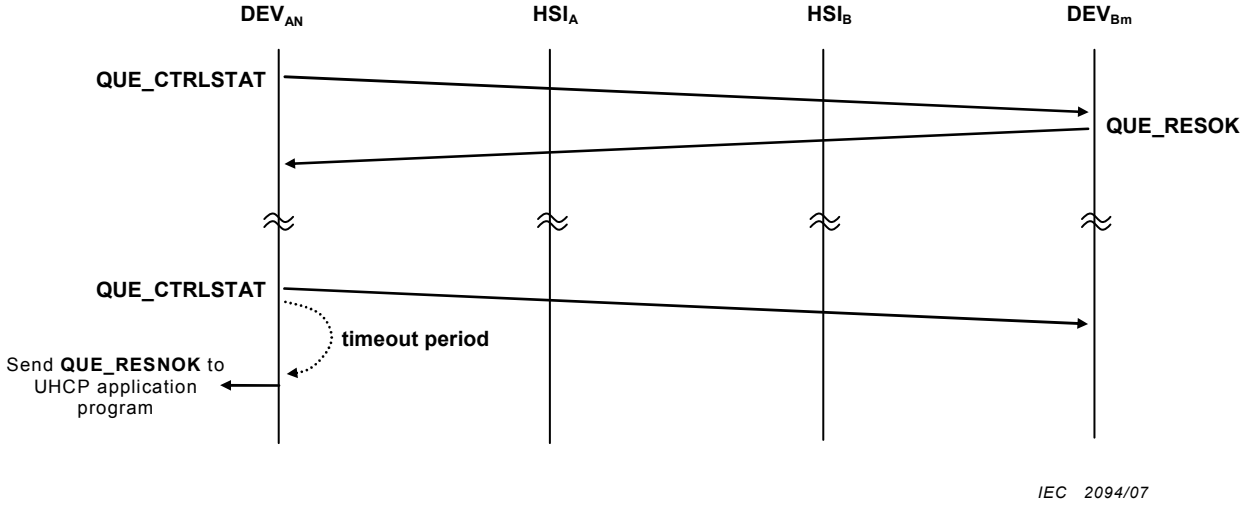


Figure 24 – Example of QUE_CTRLSTAT message

9.3.3 Query of all status (QUE_ALLSTAT)

The AT field of a QUE_ALLSTAT message is 0x3. It is used by a home server or another device to request for a particular device's command status as well as its attributes. A device receiving a QUE_ALLSTAT message shall provide its command status and attribute information in QUE_RESOK message's UHCP payload and return it. If there is no response within a timeout period, the device that has sent the QUE_ALLSTAT message shall send a QUE_RESNOK message to the application program. A QUE_ALLSTAT message does not have an UHCP payload, as shown in Figure 20.

Figure 25 displays an example of a QUE_ALLSTAT message.

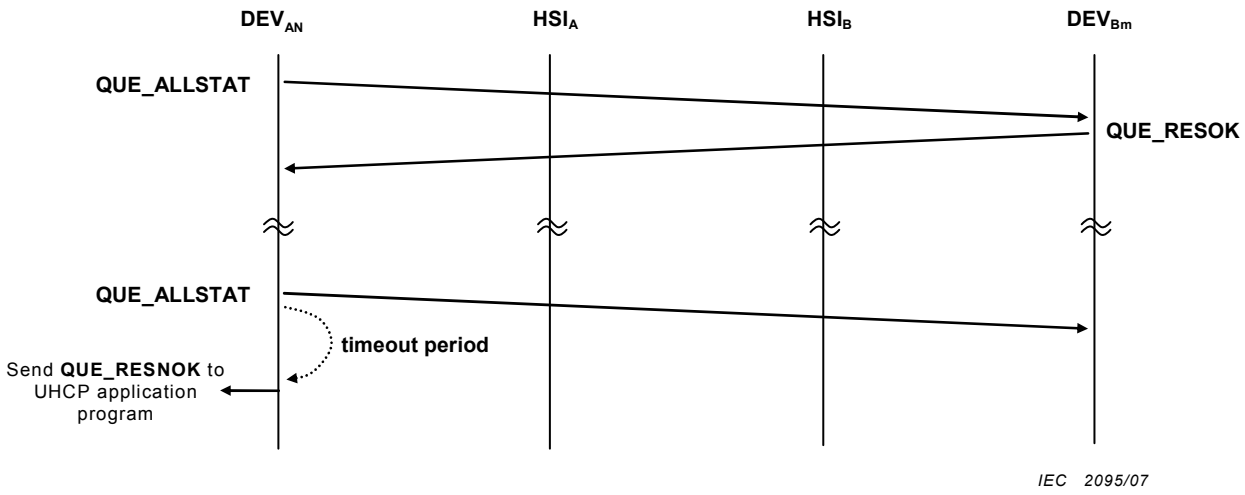


Figure 25 – Example of QUE_ALLSTAT message

9.3.4 Response OK (QUE_RESOK)

The AT field of a QUE_RESOK message is 0xE. It is used by a device receiving a QUE_REGSTAT, QUE_CTRLSTAT or QUE_ALLSTAT message to notify its command status and attributes, information which is written in the UHCP payload for transmission.

9.3.5 Response NOK (QUE_RESNOK)

The AT field of a QUE_RESNOK message is 0xF. If a device receiving a QUE_REGSTAT, QUE_CTRLSTAT or QUE_ALLSTAT message fails to query information, it shall send a QUE_RESNOK message. In addition, if a device that has sent a query message cannot receive a response message due to malfunctioning in the device that has received the query message, the query-requesting device shall be able to send a QUE_RESNOK message to the UHCP application program on its own. A QUE_RESNOK message does not have an UHCP payload. (Refer to Figure 20.)

9.4 Notification messages (NTFY)

If the MT field of an UHCP header is 0x3, it indicates that the UHCP payload content is a notification message. A notification message is used to notify the HSI of abnormal operation such as device malfunctioning or unstable power supply, as well as cases where specific circumstances shall be communicated to the user. An action type is not defined in an NTFY message, and the content of notification is written in the UHCP payload.

9.5 UHCP payload syntax

The syntax used in the UHCP payload is similar to that of markup language and shall meet the following conditions.

9.5.1 Basic syntax for UHCP payload

The content of the UHCP payload shall be created according to the following six fundamental rules.

It shall be written only in text (upper and lower case letters), numbers, “<”, “>” and “/”.

a) Terms indicated in upper case letters are items used in the UHCP payload and they are reserved words. Reserved words shall not be used as arguments.

b) The character "/" shall not be used in an argument.

c) Italicized terms are not reserved words used in the UHCP payload, and they can be specified according to the user's discretion.

d) UHCP payload content is expressed as "<*ITEM*>*argument*</*ITEM*>", which means that the *item* shall have the value of the *argument* or take actions specified by the *argument*.

e) UHCP payload shall begin with a "<UHCP>" and end with a "</UHCP>".

f) Each item within "<>" shall be indicated in upper case letters.

9.5.2 Syntax for UHCP registration

For a device to register its basic attributes and control items, an UHCP registration process is required. The UHCP payload syntax and reserved words for this purpose are specified as below.

a) Indication of registering device attributes and control items shall begin with a <REG> and end with a </REG>.

b) Indication of device attributes shall begin with a <ATTR> and end with a </ATTR>.

c) Device attributes shall consist of four items: name of device, name of vendor, location of device and type of network used by device.

d) Name of the device shall be written as "<DEV> *name_of_device*</DEV>".

e) Name of the vendor who has manufactured the device shall be written as "<VEN>*name_of_vendor*</VEN>".

f) Location of the device shall be written as "<LOC> *location_of_device*</LOC>".

g) Type of network protocol used by the device shall be written as "<NET>*name_of_network_protocol*</NET>".

h) Indication of device's control items shall begin with a <CMD> and end with a </CMD>.

i) One or more control items shall be written between <CMD> and </CMD> in the format of "<COMMAND_TYPE>*current_status*</COMMAND_TYPE>". "<COMMAND_TYPE>" is not a reserved word, and common terms should be used to facilitate understanding of the user.

j) Indication of monitoring items shall start with a <MON> and end with a </MON>.

k) One or more monitoring items shall be written between <MON> and </MON>. A meter shall be written as "<METER>value</METER>", and a sensor as "<SENSOR>value</SENSOR>".

(Example 1) UHCP payload content for UHCP registration of a DVD player:

```
<UHCP>
    <REG>
        <ATTR>
            <DEV>DVD Player</DEV>
            <VEN>vendor</VEN>
            <LOC>Living Room</LOC>
            <NET>IPv4<NET>
        </ATTR>
        <CMD>
            <PLAY>off</PLAY>
            <PAUSE>off</PAUSE>
            <REW>off</REW>
            <FF>off</FF>
        </CMD>
    </REG>
</UHCP>
```

(Example 2) UHCP payload content for UHCP registration of an electric meter:

```
<UHCP>
    <REG>
        <ATTR>
            <DEV>electric meter</DEV>
            <VEN>vendor</VEN>
            <LOC>Living Room</LOC>
            <NET>PLC<NET>
        </ATTR>
        <MON>
            <METER>100kwh</METER>
        </MON>
    </REG>
</UHCP>
```

9.5.3 Syntax for device control

The following reserved words and syntax are specified to express device control commands in the UHCP payload.

a) Shall be used for EXE_CTRL messages.
b) A control command shall begin with a <CTRL> and end with a </CTRL>.
c) One or more control items shall be written between <CMD> and </CMD> in the format of "<*COMMAND_TYPE*>*current_status*</*COMMAND_TYPE*>". "<*COMMAND_TYPE*>" is not a reserved word, and common terms should be used to facilitate understanding of the user.

(Example 1) Payload content of an EXE_CTRL message for controlling a camcorder:

```
<UHCP>
    <CTRL>
        <CMD>
            <PLAY>on</PLAY>
        </CMD>
    </CTRL>
</UHCP>
```

(Example 2) Payload content of an EXE_CTRL message for controlling a dimming light:

```
<UHCP>
    <CTRL>
        <CMD>
            <POWER>on</POWER>
            <DIM_LEVEL>1</DIM_LEVEL>
        </CMD>
    </CTRL>
</UHCP>
```

9.5.4 Syntax for query of controlling and monitoring status

The following reserved words and syntax are specified to express in the UHCP payload the commands for querying a device's UHCP registration status, control status or monitoring information.

a) Shall be used for QUE_RESOK messages that are responses to query messages such as QUE_REGSTAT, QUE_CTRLSTAT and QUE_ALLSTAT.
b) A response to a query message shall begin with a <STAT> and end with a </STAT>.
c) A response to a QUE_REGSTAT message shall begin with a <ATTR> and end with a </ATTR>. Device attributes shall be written between <ATTR> and </ATTR>.
d) Means of indicating device attributes are identical to 10.5.2.
e) A response to a QUE_CTRLSTAT shall begin with a <CMD> and end with a </CMD>. Indication of monitored values, for these cases as a meter or sensor, it shall begin with a <MON> and end with a </MON>. In case of a meter, the monitored value shall be written as "<METER>*value*</METER>" between <MON> and </MON>. In case of a sensor, the sensed value shall be written as "<SENSOR>*value*</SENSOR>" between <MON> and </MON>.
f) Means of indicating a general control status are identical to 10.5.3.

(Example 1) Payload content of a TV set's QUE_RESOK message for a QUE_REGSTAT message:

```
<UHCP>
    <STAT>
        <ATTR>
            <DEV>TV</DEV>
            <VEN>vendor</VEN>
            <LOC>Living Room</LOC>
            <NET>IEEE1394<NET>
        </ATTR>
    </STAT>
</UHCP>
```

(Example 2) Payload content of an electric meter's QUE_RESOK message for a QUE_CTRLSTAT message:

```
<UHCP>
    <STAT>
        <MON>
            <METER>100kwh</METER>
        </MON>
    </STAT>
</UHCP>
```

(Example 3) Payload content of a QUE_RESOK message for a QUE_ALLSTAT message:

```
<UHCP>
    <STAT>
        <ATTR>
            <DEV>TV</DEV>
            <VEN>vendor</VEN>
            <LOC>Living Room</LOC>
            <NET>IEEE1394<NET>
        </ATTR>
        <CMD>
            <POWER>on</POWER>
            <CHANNEL>11</CHANNEL>
            <VOLUME>20</VOLUME>
        </CMD>
    </STAT>
</UHCP>
```

9.5.5 Syntax for notification

a) Automatic notification shall start with a <NOTIFY> and end with a </NOTIFY>.
b) Shall be used for a device's automatic notification (NTFY) and written in the format of "<NOTIFY>*alarm_contents*</NOTIFY>".

(Example) In case of a digital camera:

```
<UHCP>
    <NOTIFY>Low Battery</NOTIFY>
    <NOTIFY>No memory card</NOTIFY>
</UHCP>
```

10 Home data service protocol (HDSP)

CCP provides the HDSP for implementing data services such as file and directory services under the home network environment.

HDSP is a packet-based signaling protocol, one of the fundamental protocols provided by the CCP. HDSP defines simple request/response packets and performs file management and transmission functions within the home network.

All CCP devices that support data services under the home network environment constructed with heterogeneous networks shall support the HDSP.

This document only specifies HDSP's functional requirements.

10.1 Functional requirements of HDSP

Devices and home servers that provide HDSP shall satisfy the following requirements.

10.1.1 Interoperability with CCP

In order to be compatible with CCP, HDSP shall satisfy the following requirements.

– HDSP shall be an application program using the CCP. Namely, commands or action results defined in relation with the HDSP shall be written in the CCP payload.

– The payload type field value in CCP header's type fields shall be 0x3.

10.1.2 File and directory services

HDSP shall provide functions that allow the user to access the files dispersed throughout various devices in the home network. The files shall be readily available to the user regardless of time and place, as if they were stored in a single device.

In order to provide this kind of user-friendliness, HDSP shall satisfy the following requirements.

– HDSP shall provide functions for a particular device to request for information regarding files and directories stored in other devices.

– HDSP shall provide functions for a device receiving a file or directory information request to respond back with its corresponding information.

– HDSP shall provide functions for a particular device to copy, move, delete and change names of files and directories stored in other devices.

– HDSP shall provide functions for a particular device to request transfers of files and directories stored in other devices.

– HDSP shall provide functions for a device receiving a file or directory transfer request to respond back with a corresponding transfer to the requesting device.

– The device requesting a file or directory transfer shall be able to save the incoming file or device in its storage.

10.1.3 Messaging service

HDSP shall provide functions for the devices offering data services within the home network to exchange text messages. In order to do so, HDSP shall satisfy the following requirements.

– HDSP shall provide functions for a particular device to send a real-time message to another device.

– HDSP shall provide a reservation function for a particular device to send a text message to another device at a specified time. In order to do so, the home server shall be able to temporarily store the messages to be exchanged between two devices and transmit them at the specified time.

– A device receiving a message shall be able to display the message on a display screen.

10.2 HDSP packet format

Figure 26 represents the HDSP packet format. The HDSP packet is located in the CCP packet payload. In order to indicate that the CCP payload type is a HDSP packet, the HDSP application program shall set the payload type of the CCP header's option field as 0x03. A HDSP packet consists of an eight-byte HDSP header and the HDSP payload. The HDSP header contains three fields: Transaction ID, HDSP command, and HDSP payload length.

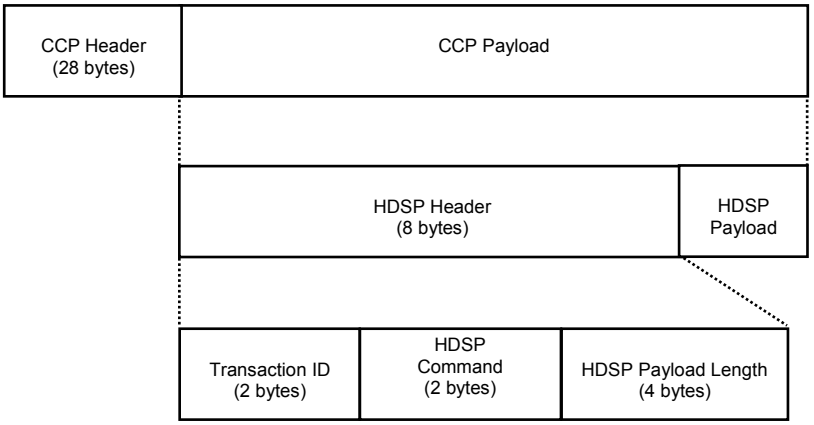


Figure 26 – HDSP packet format

10.2.1 Transaction ID (TID)

When an application program sends and receives multiple request packets and response packets, the transaction ID is used by the application program to find a response packet for a particular request packet. The application program can assign random values for the transaction ID. The transaction ID is a two-byte field.

10.2.2 HDSP command

The HDSP command field is two bytes long. Detailed HDSP commands are shown in Table 4.

Table 4 – HDSP commands

HDSP commands	HEX	Description
DIR_QUE_REQ	0x1011	Request packet for retrieving directory information.
DIR_QUE_RES	0x1012	Response packet corresponding to DIR_QUE_REQ.
DIR_DEL_REQ	0x1021	Request packet for deletion of a directory.
DIR_DEL_RES	0x1022	Response packet corresponding to DIR_DEL_REQ.
DIR_REN_REQ	0x1031	Request packet for renaming of a directory.
DIR_REN_RES	0x1032	Response packet corresponding to DIR_REN_REQ.
DIR_MAKE_REQ	0x1041	Request packet for making of a directory.
DIR_MAKE_RES	0x1042	Response packet corresponding to DIR_MAKE_REQ.
FILE_QUE_REQ	0x2011	Request packet for an inquiring about file information.
FILE_QUE_RES	0x2012	Response packet corresponding to FILE_QUE_REQ.
FILE_DEL_REQ	0x2021	Request packet for deletion of a file.
FILE_DEL_RES	0x2022	Response packet corresponding to FILE_DEL_REQ.
FILE_REN_REQ	0x2031	Request packet for renaming of a file.
FILE_REN_RES	0x2032	Response packet corresponding to FILE_REN_REQ.
FILE_NEGO_REQ	0x2051	Request packet for negotiating for transferring a file.
FILE_NEGO_RES	0x2052	Response packet corresponding to FILE_NEGO_REQ.
FILE_GET_REQ	0x2061	Request packet for copying a file from a remote device.
FILE_GET_RES	0x2062	Response packet corresponding to FILE_GET_REQ.
FILE_PUT_REQ	0x2071	Request packet for copying a file to a remote device.
FILE_PUT_RES	0x2072	Response packet corresponding to FILE_PUT_REQ.
MSG_PUT_REQ	0x3071	Request packet for sending a text message.
MSG_PUT_RES	0x3072	Response packet corresponding to MSG_SEND_REQ.

10.2.3 HDSP payload length (HDSPPLEN) field

The HDSP payload length field indicates the byte size of the HDSP payload. The size of the length field is four bytes.

10.2.4 HDSP payload

The HDSP payload contains information required for a HDSP command.

10.3 Messages for directory services

HDSP messages for directory services are specified in Table 5. Figure 27 shows an example of usage of directory service messages.

Table 5 – Messages for directory services

HDSP header			HDSP payload				
TID	DIR_ QUE_ REQ	HDSPPLEN	PATHLEN (4 bytes)	PATH			
TID	DIR_ QUE_ RES	HDSPPLEN	ERRCODE (4 bytes)	DIRSNO (4 bytes)	FILESNO (4 bytes)	DIRS	FILES
TID	DIR_ DEL_ REQ	HDSPPLEN	PATHLEN (4 bytes)	PATH			
TID	DIR_ DEL_ RES	HDSPPLEN	ERRCODE (4 bytes)				
TID	DIR_ REN_ REQ	HDSPPLEN	PATHLEN1 (4 bytes)	PATHLEN2 (4 bytes)	PATH1	PATH2	
TID	DIR_ REN_ RES	HDSPPLEN	ERRCODE (4 bytes)				
TID	DIR_ MAKE_ REQ	HDSPPLEN	PATHLEN (4 bytes)	PATH			
TID	DIR_ MAKE_ RES	HDSPPLEN	ERRCODE (4 bytes)				

10.3.1 Query request message (DIR_QUE_REQ)

A DIR_QUE_REQ message is used for retrieving directory information of a remote device that also supports HDSP. HDSP payload of a DIR_QUE_REQ message shall be filled with PATHLEN and PATH fields, which are described below.

– PATHLEN: This field shall be four bytes long and shall be filled with byte size of PATH field. When the top directory is to be retrieved, this field shall be set to 0.

– PATH: This field shall be filled with a string of characters representing a full path to a directory of a remote CCP device which also supports HDSP. When the top directory is to be retrieved, this field shall be blanked.

10.3.2 Query response message (DIR_QUE_RES)

A DIR_QUE_RES message is a response of a DIR_QUE_REQ message. HDSP payload of a DIR_QUE_RES message shall be filled with ERRCODE, DIRSNO, FILESNO, DIRS and FILES fields, which are described below.

– ERRCODE: This field shall be four bytes long and shall be filled with an error code specified in 10.6.

– DIRSNO: This field shall be four bytes long and shall be filled with the number of sub-directories in a retrieved directory.

– FILESNO: This field shall be four bytes long and shall be filled with the number of files in a retrieved directory.
– DIRS: This field shall be a field with a string of sub-directories' name in a retrieved directory. Each directory name of this field shall be separated with a horizontal tab (\t) character (0x09).
– FILES: This field shall be a field with a string of files' name in a retrieved directory. Each file name of this field shall be separated with a horizontal tab (\b) c1haracter (0x09).

10.3.3 Deletion request message (DIR_DEL_REQ)

A DIR_QUE_REQ message is used for deleting one directory of a remote device that also supports HDSP. HDSP payload of a DIR_DEL_REQ message shall be filled with PATHLEN and PATH fields, which are described below.

– PATHLEN: This field shall be four bytes long and shall be filled with byte size of PATH field.
– PATH: This field shall be filled with a string of characters representing a full path to a directory that is to be deleted.

10.3.4 Deletion response message (DIR_DEL_RES)

A DIR_DEL_RES message is a response of a DIR_DEL_REQ message. HDSP payload of a DIR_DEL_RES message shall be filled with ERRCODE field, which is described below.

– ERRCODE: This field shall be four bytes long and shall be filled with an error code specified in 10.6.

10.3.5 Renaming request message (DIR_REN_REQ)

A DIR_REN_REQ message is used for renaming a directory of a remote device that also supports HDSP. HDSP payload of a DIR_REN_REQ message shall be filled with PATHLEN1, PATHLEN2, PATH1 and PATH2 fields, which are described below.

– PATHLEN1: This field shall be four bytes long and shall be filled with byte size of PATH1 field.
– PATHLEN2: This field shall be four bytes long and shall be filled with byte size of PATH2 field.
– PATH1: This field shall be filled with a string of characters representing a full path to a directory that is to be renamed.
– PATH2: This field shall be filled with a string of characters representing a full path to a renamed directory.

10.3.6 Renaming response message (DIR_REN_RES)

A DIR_REN_RES message is a response of a DIR_REN_REQ message. HDSP payload of a DIR_REN_RES message shall be filled with ERRCODE field, which is described below.

– ERRCODE: This field shall be four bytes long and shall be filled with an error code specified in 10.6.

10.3.7 Making request message (DIR_MAKE_REQ)

A DIR_MAKE_REQ message is used for making a new directory of a remote device that also supports HDSP. HDSP payload of a DIR_MAKE_REQ message shall be filled with PATHLEN, and PATH fields, which are described below.

– PATHLEN: This field shall be four bytes long and shall be filled with byte size of PATH field.

– PATH: This field shall be filled with a string of characters representing a full path to a directory that is to be newly made.

10.3.8 Making response message (DIR_MAKE_RES)

A DIR_MAKE_RES message is a response of a DIR_MAKE_REQ message. HDSP payload of a DIR_MAKE_RES message shall be filled with ERRCODE field, which is described below.

– ERRCODE: This field shall be four bytes long and shall be filled with an error code specified in 10.6.

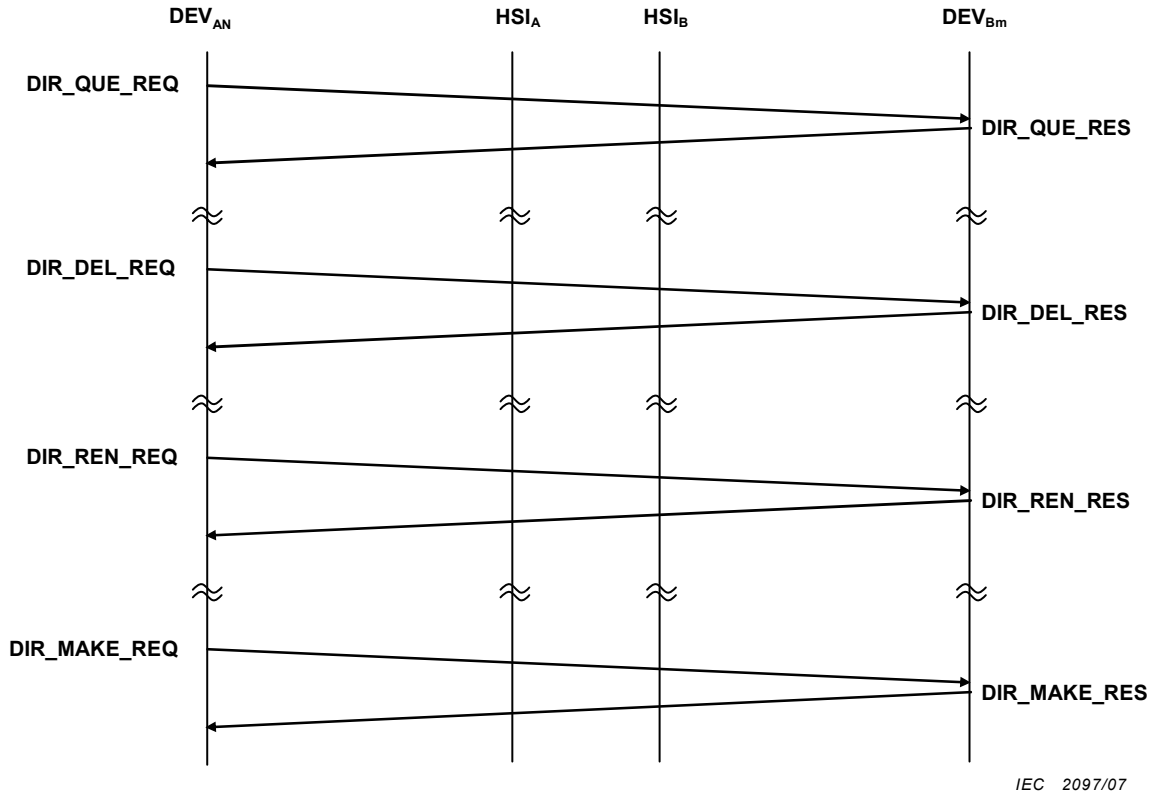


Figure 27 – Example of usage of directory service messages

10.4 Messages for file services

HDSP messages for file services are specified in Table 6. Figure 28 shows an example of usage of file service messages.

Table 6 – Messages for file services

HDSP header			HDSP payload							
TID	FILE_QUE_ REQ	HDSPPLEN	PATHLEN (4 bytes)	PATH						
TID	FILE_QUE_ RES	HDSPPLEN	ERRCODE (4 bytes)	FILESIZE (4 bytes)	YEAR (2 bytes)	MON (1 byte)	DAY (1 byte)	HOUR (1 byte)	MIN (1 byte)	SEC (1 byte)
TID	FILE_DEL_ REQ	HDSPPLEN	PATHLEN (4 bytes)	PATH						
TID	FILE_DEL_ RES	HDSPPLEN	ERRCODE (4 bytes)							
TID	FILE_REN_ REQ	HDSPPLEN	PATHLEN1 (4 bytes)	PATHLEN2 (4 bytes)	PATH1	PATH2				
TID	FILE_REN_ RES	HDSPPLEN	ERRCODE (4 bytes)							
TID	FILE_NEGO_ REQ	HDSPPLEN	MYMTS (4 bytes)							
TID	FILE_NEGO_ RES	HDSPPLEN	ERRCODE (4 bytes)	FMTS (4 bytes)						
TID	FILE_GET_ REQ	HDSPPLEN	FMTS (4 bytes)	SEQNO (4 bytes)	PATHLEN (4 bytes)	PATH				
TID	FILE_GET_ RES	HDSPPLEN	ERRCODE (4 bytes)	SEGNO (4 bytes)	SEQNO (4 bytes)	DATALEN (4 bytes)	DATA			
TID	FILE_PUT_ REQ	HDSPPLEN	SEGNO (4 bytes)	SEQNO (4 bytes)	PATHLEN (4 bytes)	DATALEN (4 bytes)	PATH	DATA		
TID	FILE_PUT_ RES	HDSPPLEN	ERRCODE (4 bytes)							

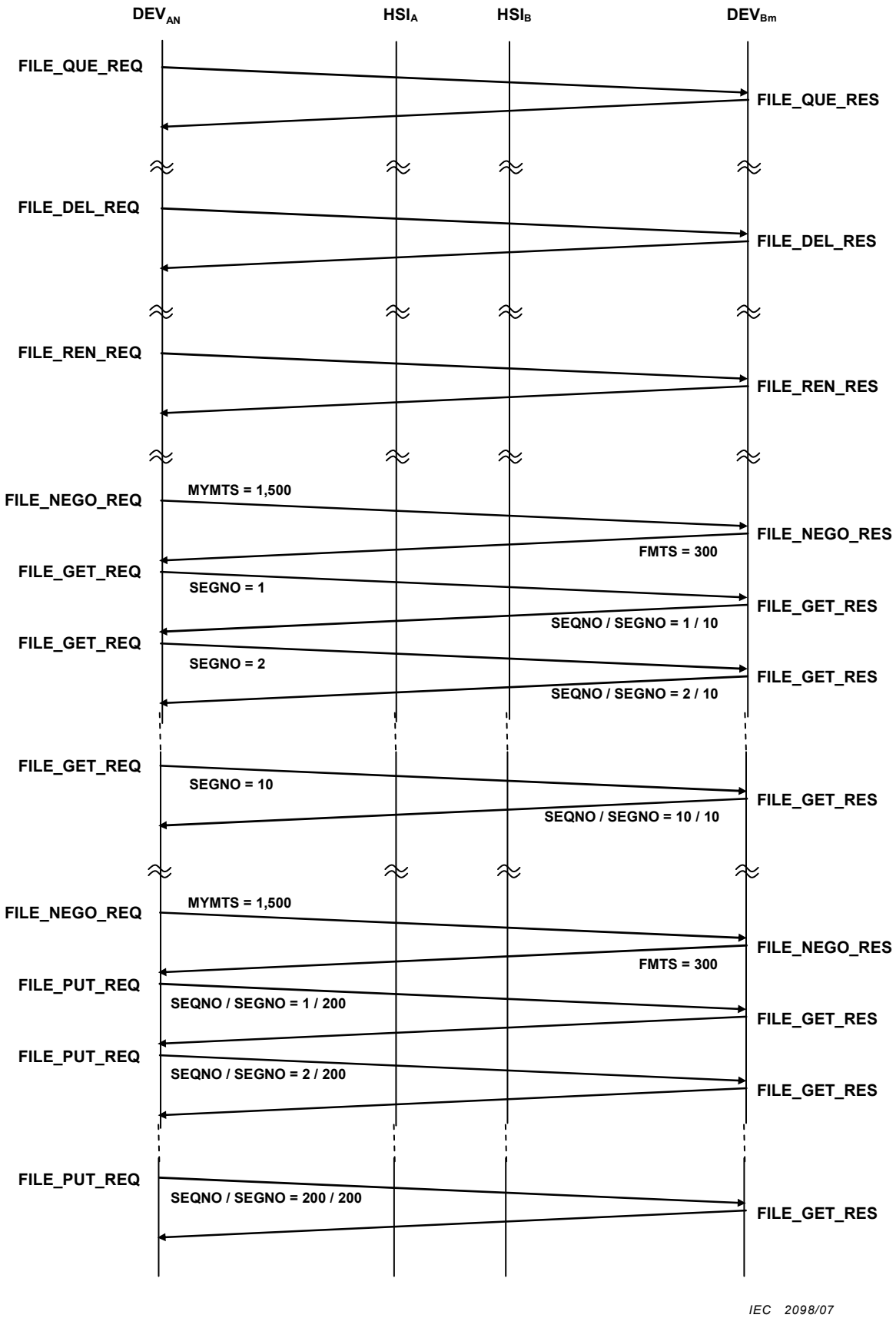


Figure 28 – Example of usage of file service messages

10.4.1 Query request message (FILE_QUE_REQ)

A FILE_QUE_REQ message is used for retrieving information of a file stored in a remote device that also supports HDSP. HDSP payload of a FILE_QUE_REQ message shall be filled with PATHLEN and PATH fields, which are described below.

– PATHLEN: This field shall be four bytes long and shall be filled with byte size of PATH field.

– PATH: This field shall be filled with a string of characters representing a full path to a file that is to be retrieved.

10.4.2 Query response message (FILE_QUE_RES)

A FILE_QUE_RES message is a response of a FILE_QUE_REQ message. HDSP payload of a FILE_QUE_RES message shall be filled with ERRCODE, FILESIZE, YEAR, MON, DAY, HOUR, MIN and SEC fields, which are described below.

– ERRCODE: This field shall be four bytes long and shall be filled with an error code specified in 10.6.

– FILESIZE: This field shall be four bytes long and shall be filled with the size of a retrieved file in byte unit.

– YEAR, MON, DAY, HOUR, MIN and SEC: YEAR field shall be two bytes long. Other fields shall be one byte long. These fields shall represent the latest date and time when a retrieved file was modified.

10.4.3 Deletion request message (FILE_DEL_REQ)

A FILE_DEL_REQ message is used for deleting a file stored in a remote device that also supports HDSP. HDSP payload of a FILE_DEL_REQ message shall be filled with PATHLEN and PATH fields, which are described below.

– PATHLEN: This field shall be four bytes long and shall be filled with byte size of PATH field.

– PATH: This field shall be filled with a string of characters representing a full path to a file that is to be deleted.

10.4.4 Deletion response message (FILE_DEL_RES)

A FILE_DEL_RES message is a response of a FILE_DEL_REQ message. HDSP payload of a FILE_DEL_RES message shall be filled with ERRCODE field, which is described below.

– ERRCODE: This field shall be four bytes long and shall be filled with an error code specified in 10.6.

10.4.5 Renaming request message (FILE_REN_REQ)

A FILE_REN_REQ message is used for renaming a file stored in a remote device that also supports HDSP. HDSP payload of a FILE_REN_REQ message shall be filled with PATHLEN1, PATHLEN2, PATH1, and PATH2 fields, which are described below.

– PATHLEN1: This field shall be four bytes long and shall be filled with byte size of PATH1 field.

– PATHLEN2: This field shall be four bytes long and shall be filled with byte size of PATH2 field.

– PATH1: This field shall be filled with a string of characters representing a full path to a file that is to be renamed.

– PATH2: This field shall be filled with a string of characters representing a full path to a renamed file.

10.4.6 Renaming response message (FILE_REN_RES)

A FILE_REN_RES message is a response of a FILE_REN_REQ message. HDSP payload of a FILE_REN_RES message shall be filled with ERRCODE field, which is described below.

– ERRCODE: This field shall be four bytes long and shall be filled with an error code specified in 10.6.

10.4.7 Negotiation request message (FILE_NEGO_REQ)

A FILE_NEGO_REQ message is used for negotiating maximum transfer size (MTS) with a remote device that also supports HDSP before a device initiates sending (or receiving) a file to (or from) another device. HDSP payload of a FILE_NEGO_REQ message shall be filled with MYMTS field, which is described below.

– MYMTS: This field shall be four bytes long and shall be filled with MTS supported and requested by a device that initiates file transfer. MYMTS shall be represented in byte unit.

10.4.8 Negotiation response message (FILE_NEGO_RES)

A FILE_NEGO_RES message is a response of a FILE_NEGO_REQ message. HDSP payload of a FILE_NEGO_RES message shall be filled with ERRCODE and FMTS, which are described below.

– ERRCODE: This field shall be four bytes long and shall be filled with an error code specified in 10.6.

– FMTS: This field shall be four bytes long and shall be filled with final negotiated MTS. In order to set this FMTS, a device receiving a FILE_NEGO_REQ message shall be able to compare MYMTS in the FILE_NEGO_REQ message with its own MTS and shall also be able to set FMTS to the lower MTS value between them. FMTS shall also be represented in byte unit.

10.4.9 Getting request message (FILE_GET_REQ)

A FILE_GET_REQ message is used for copying a file from a remote device that also supports HDSP. Before sending this message, a device shall initiate negotiation for setting FMTS first. HDSP payload of a FILE_GET_REQ message shall be filled with FMTS, SEQNO, PATHLEN and PATH fields, which are described below.

– FMTS: This field shall be four bytes long and shall be set to the FMTS value in the FILE_NEGO_RES packet sent by a remote device.

– SEQNO: This field shall be four bytes long and shall indicate sequence number counted from 1 to SEGNO which will be set by a remote device.

– PATHLEN: This field shall be four bytes long and shall be filled with byte size of PATH field.

– PATH: This field shall be filled with a string of characters representing a full path to a file that is to be transferred from a remote device.

10.4.10 Getting response message (FILE_GET_RES)

A FILE_FET_RES message is a response of a FILE_GET_REQ message. HDSP payload of a FILE_GET_RES message shall be filled with ERRCODE, SEGNO, SEQNO, DATALEN and DATA which are described below.

– ERRCODE: This field shall be four bytes long and shall be filled with an error code specified in 10.6.

– SEGNO: This field shall be four bytes long and shall be filled with the total number of segments of a file that is to be transferred to the sender of the FILE_GET_REQ message.

– SEQNO: This field shall be four bytes long and shall be set to the SEQNO value of the FILE_GET_REQ message.

– DATALEN: This field shall be four bytes long and shall be set to the size of DATA field in byte unit. This field is usually set to the FMTS value of the FILE_GET_REQ message.
– DATA: This field shall be filled with one segment of a file to be transferred. The size of one segment is usually equal to FMTS or DATALEN value. A sender of this message shall be able to write one segment of a file by using SEQNO and FMTS values.

10.4.11 Putting request message (FILE_PUT_REQ)

A FILE_PUT_REQ message is used for copying a local file to a remote device that also supports HDSP. Before sending this message, a device shall initiate negotiation for setting FMTS first. HDSP payload of a FILE_PUT_REQ message shall be filled with SEGNO, SEQNO, PATHLEN, DATALEN, PATH and DATA fields, which are described below.

– SEGNO: This field shall be four bytes long and shall be filled with the total number of segments of a file that is to be transferred to a remote device.
– SEQNO: This field shall be four bytes long and shall indicate the sequence number counted from 1.
– PATHLEN: This field shall be four bytes long and shall be filled with byte size of PATH field.
– DATALEN: This field shall be four bytes long and shall be set to the size of DATA field in byte unit. This field is usually set to the FMTS value in the FILE_NEGO_RES packet sent by a remote device.
– PATH: This field shall be filled with a string of characters representing a full path to a file stored in a remote device.
– DATA: This field shall be filled with one segment of a file to be transferred. The size of one segment is usually equal to FMTS or DATALEN value. A sender of this message shall be able to write one segment of a file by using SEQNO and FMTS values.

10.4.12 Putting response message (FILE_PUT_RES)

A FILE_PUT_RES message is a response of a FILE_PUT_REQ message. HDSP payload of a FILE_PUT_RES message shall be filled with ERRCODE field, which is described below.

– ERRCODE: This field shall be four bytes long and shall be filled with an error code specified in 10.6.

10.5 Messages for messaging service

HDSP messages for messaging services are specified in Table. 7. Figure 29 shows an example of usage of messaging service messages.

Table 7 – Messages for messaging services

HDSP header			HDSP payload	
TID	MSG_ PUT_ REQ	HDSPPLEN	MSGLEN (4 bytes)	MSG
TID	MSG_ PUT_ RES	HDSPPLEN	ERRCODE (4 bytes)	

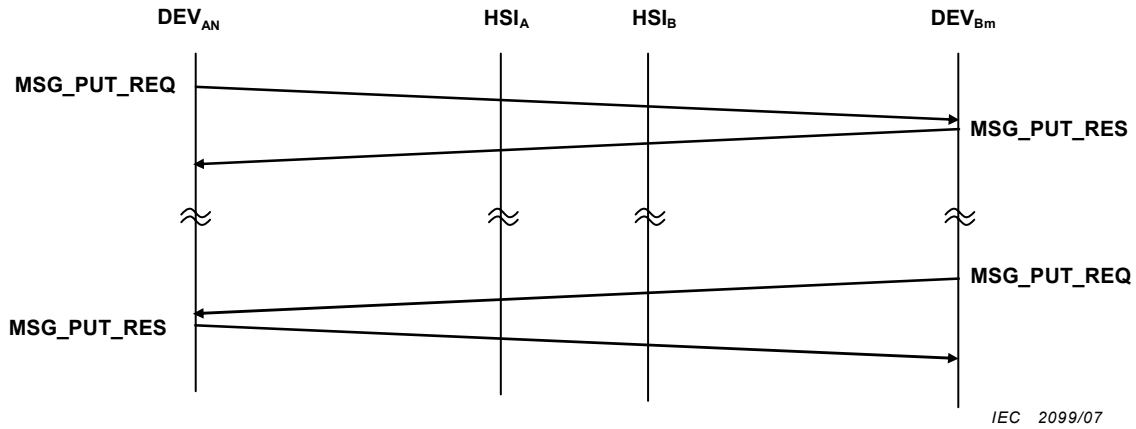


Figure 29 – Example of usage of Messaging service messages

10.5.1 Sending request message (MSG_PUT_REQ)

A MSG_PUT_REQ message is used for sending a simple character-based message to a remote device that also supports HDSP. HDSP payload of a MSG_PUT_REQ message shall be filled with MSGLEN and MSG fields, which are described below.

– MSGLEN: This field shall be four bytes long and shall be filled with byte size of MSG field.

– MSG: This field shall be filled with a character-based message to be sent.

10.5.2 Sending response message (MSG_PUT_RES)

A MSG_PUT_RES message is a response of a MSG_PUT_REQ message. HDSP payload of a MSG_PUT_RES message shall be filled with ERRCODE field, which is described below.

– ERRCODE: This field shall be four bytes long and shall be filled with an error code specified in 10.6. Error codes

10.6 Error codes

Every HDSP response message shall return one of the error codes specified in Table. 8. These error codes are subject to change according to changes or modifications of HDSP messages.

Table 8 – Error codes for HDSP

ERRCODE (4 bytes)	Acronym	Description
0x03000000	HDSPERR_OK	A HDSP request message is processed properly without any error.
0x03010101	HDSPERR_DIR_NOT_FOUND	Directory name of the HDSP request message can not be found.
0X03010102	HDSPERR_DIR_NOT_PERMITTED	It is not permitted to access the directory because of an authority problem.
0X03010103	HDSPERR_DIR_NOT_ACCESSED	The directory is being accessed by another device or internal process.
0X03010204	HDSPERR_DIR_NAME_INVALID	One or more invalid character(s) is(are) used in the directory name of the HDSP request message.
0X03010205	HDSPERR_DIR_NAME_LONG	The directory name of the HDSP request message is too long.
0X03020101	HDSPERR_FILE_NOT_FOUND	The file name of the HDSP request message can not be found.
0X03020102	HDSPERR_FILE_NOT_PERMITTED	It is not permitted to access the file because of an authority problem.
0X03020103	HDSPERR_FILE_NOT_ACCESSED	The file is being accessed by another device or internal process.
0X03020204	HDSPERR_FILE_NAME_INVALID	One or more invalid character(s) is(are) used in the file name of the HDSP request message.
0X03020205	HDSPERR_FILE_NAME_LONG	The file name of the HDSP request message is too long.
0X03020301	HDSPERR_FILE_MTS_CHANGED	The MTS value has been changed.
0X03020404	HDSPERR_FILE_SEGNO_INVALID	The SEGNO value in the HDSP request message is invalid.
0X03020504	HDSPERR_FILE_SEQNO_INVALID	The SEQNO value in the HDSP request message is invalid.
0X03020604	HDSPERR_FILE_DATA_INVALID	The DATA filed in the HDSP request message is invalid.
0X03030706	HDSPERR_RESOURCE_NO_SPACE	The remote device does not have enough space for creating or storing a file or directory.
0X03030807	HDSPERR_RESOURCE_TOO_BUSY	The remote device is too busy to process the HDSP request packet.
OTHERS		Reserved for future use

11 Home multimedia service protocol (HMSP)

CCP provides the HMSP for offering multimedia services such as audio and video features under the home network environment.

HMSP is a packet-based signaling protocol, one of the fundamental protocols provided by the CCP. HMSP defines simple request/response packets and performs tasks such as management, real-time transmission and playing of multimedia resources within the home network.

All CCP devices that support multimedia services under the home network environment constructed with heterogeneous networks shall support the HMSP.

This document only specifies HMSP’s functional requirements.

11.1 Functional requirements of HMSP

Devices and home servers that provide HMSP shall satisfy the following requirements.

11.1.1 Interoperability with CCP

In order to be compatible with CCP, HMSP shall satisfy the following requirements.

– HMSP shall be an application program using the CCP. Namely, commands or action results defined in relation with the HMSP shall be written in the CCP payload.

– The payload type field value in CCP header's type fields shall be 0x4.

11.1.2 Management of multimedia resource

HMSP shall provide functions that allow the user to access the various types of multimedia resources dispersed throughout numerous devices in the home network. The resources shall be readily available to the user regardless of time and place, as if they were stored in a single device.

In order to provide this kind of user-friendliness, HMSP shall satisfy the following requirements.

– HMSP shall provide functions for a device to register its multimedia resources information to the home server.

– HMSP shall provide functions for a particular device to request the home server for multimedia resources information for all devices within the home network.

– HMSP shall provide functions for the home server to respond to a particular device requesting information regarding multimedia resources of all devices within the home network.

NOTE Multimedia resources information includes the name, type, size, stored date and maximum transfer bandwidth of the resources.

11.1.3 Stream and play of multimedia resource

HMSP shall provide functions for streaming various types of multimedia resources stored in numerous devices in the home network as well as playing a multimedia stream being serviced by a particular device specified by the user.

In order to do so, HMSP shall satisfy the following requirements.

– HMSP shall provide functions for a particular device to request the home server for a streaming service of multimedia resources stored in another device. When doing so, the device making the streaming service request shall specify to the home server information such as the names of the multimedia resources, the maximum bandwidth it can handle and the qualities of video and audio desired by the user.

– HMSP shall provide functions for the home server to deliver the received streaming service request to the corresponding device with the multimedia resources.

– HMSP shall allow the device with the multimedia resources to provide the streaming service according to the request from the home server. The device receiving the request shall provide the streaming service only when it is feasible under the considerations of its processing power and the maximum bandwidth of its cluster network.

– If the service is deemed not feasible upon assessment of its processing capability and the maximum bandwidth of its cluster network, the device that has received the streaming service request from the home server shall be able to notify back to the home server with a status report indicating that the requested service cannot be provided. In turn, the home server shall be able to relay the received report to the device that had originally made the streaming service request.

– HMSP shall provide functions for a particular device to request information regarding the multimedia resources being serviced by the home server.

– HMSP shall provide functions for the home server to respond to a request made by a particular device for information regarding the multimedia resources being serviced by the home server.

– HMSP shall provide functions for a particular device to request multicasting of the multimedia resources being serviced.

– HMSP shall provide functions for the home server to accommodate a request made by a particular device for multicasting of the multimedia resources being serviced. However, multicasting shall be provided only when it is feasible upon the considerations of the home server's processing capability and the maximum bandwidth of the network that the requesting device is linked to.

– Upon receiving a request from a particular device to multicast the multimedia resources being serviced, if the home server determines that the multicasting is not feasible due to its processing capability and/or the maximum bandwidth of the network the requesting device is linked to, the home server shall be able to notify the requesting device with an infeasibility report.

– A device receiving multimedia resources shall have capabilities to play the resources.

Annex A
(informative)

FSM of FS-CCPDEV supporting HNMP

A.1 Finite state machine (FSM) for HMSP

Figure A.1 displays the finite state machine (FSM) of an FS-CCPDEV device that supports the HNMP.

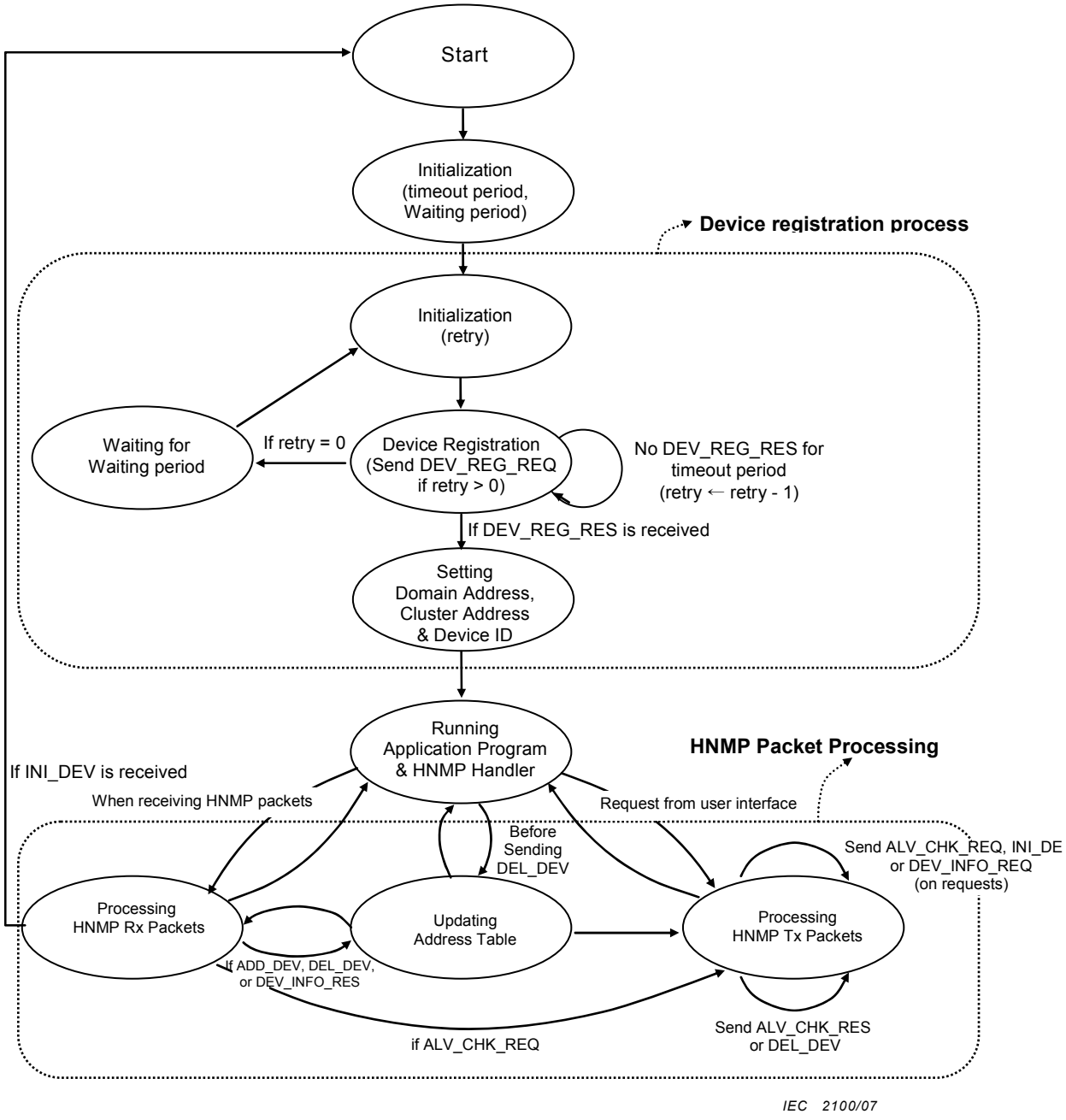


Figure A.1 – FSM of FS-CCPDEV for supporting HNMP

A.2 Start

After the FS-CCPDEV is powered on, it starts the initialization process. If the FS-CCPDEV receives an INI_DEV packet during normal operation, it shall be able to re-initialize itself. A device that has completed the device registration process does not have to register itself again after initialization.

A.3 Initialization

The retry, timeout and waiting variables are initialized. Since these variables are subject to the processing power of a device and the network speed, device manufacturers or developers are recommended to initialize these variables to appropriate constants. If the FS-CCPDEV does not receive a DEV_REG_RES packet within the timeout period, it attempts to send DEV_REG_REQ packets *N* times, where *N* is equal to the retry variables.

A.4 Device registration

Upon completion of initialization, the FS-CCPDEV sends a DEV_REG_REQ packet to the HSI of the cluster network it belongs to, in order to request the domain address, cluster address and its own device ID. When doing so, the FS-CCPDEV writes its name in the DEV_REG_REQ packet payload prior to transmission.

A.5 Waiting

After sending a DEV_REG_REQ packet, the FS-CCPDEV waits for a DEV_REG_RES packet from the HSI. If there is no response within timeout period, the FS-CCPDEV attempts retries. If there is still no response after the retries, the home server is regarded as not being ready, and the device registration process is repeated after a waiting period.

A.6 Setting domain address and device ID

After receiving the DEV_REG_RES packet, the FS-CCPDEV configures domain address, cluster address, device ID and HSI's physical address (or network address).

A.7 Running application program and HNMP handler

Upon successful completion of the device registration process, the FS-CCPDEV simultaneously executes handlers for the application program and HNMP packet processing.

A.8 Processing Rx HNMP packets

If the HNMP handler receives an HNMP packet during application program execution, the HNMP handler processes the received HNMP packet. If the packet is an INI_DEV, device initialization is performed. If the received packet is an ADD_DEV, DEL_DEV or DEV_INFO_RES, the HNMP handler updates its address table. If the packet is an ALV_CHK_REQ, the handler returns an ALV_CHK_RES packet.

A.9 Processing Tx HNMP packets

The user shall be able to send an ALV_CHK_REQ or INI_DEV to another device within the home network through the user interface. In addition, the user shall be able to send a DEV_INFO_REQ to the HSI to request for address information of the home network devices. If an ALV_CHK_REQ is sent according to the user request and an ALV_CHK_RES packet is

not returned, the process is retried *N* times, where *N* is equal to the retry variable. If an ALV_CHK_RES packet still fails to be returned, a DEL_DEV shall be sent as a broadcast. Moreover, when an ALV_CHK_REQ is received, an ALV_CHK_RES packet shall be returned as a response within a timeout period.

A.10 Updating address table

When the FS-CCPDEV receives an ADD_DEV, DEL_DEV or DEV_INFO_RES packet, it shall be able to update the address table it manages.

Annex B
(informative)

FSM of FS-CCPDEV for supporting UHCP

B.1 Finite state machine (FSM) for UHCP

Figure B.1 displays the finite state machine (FSM) of an FS-CCPDEV with UHCP functions implemented.

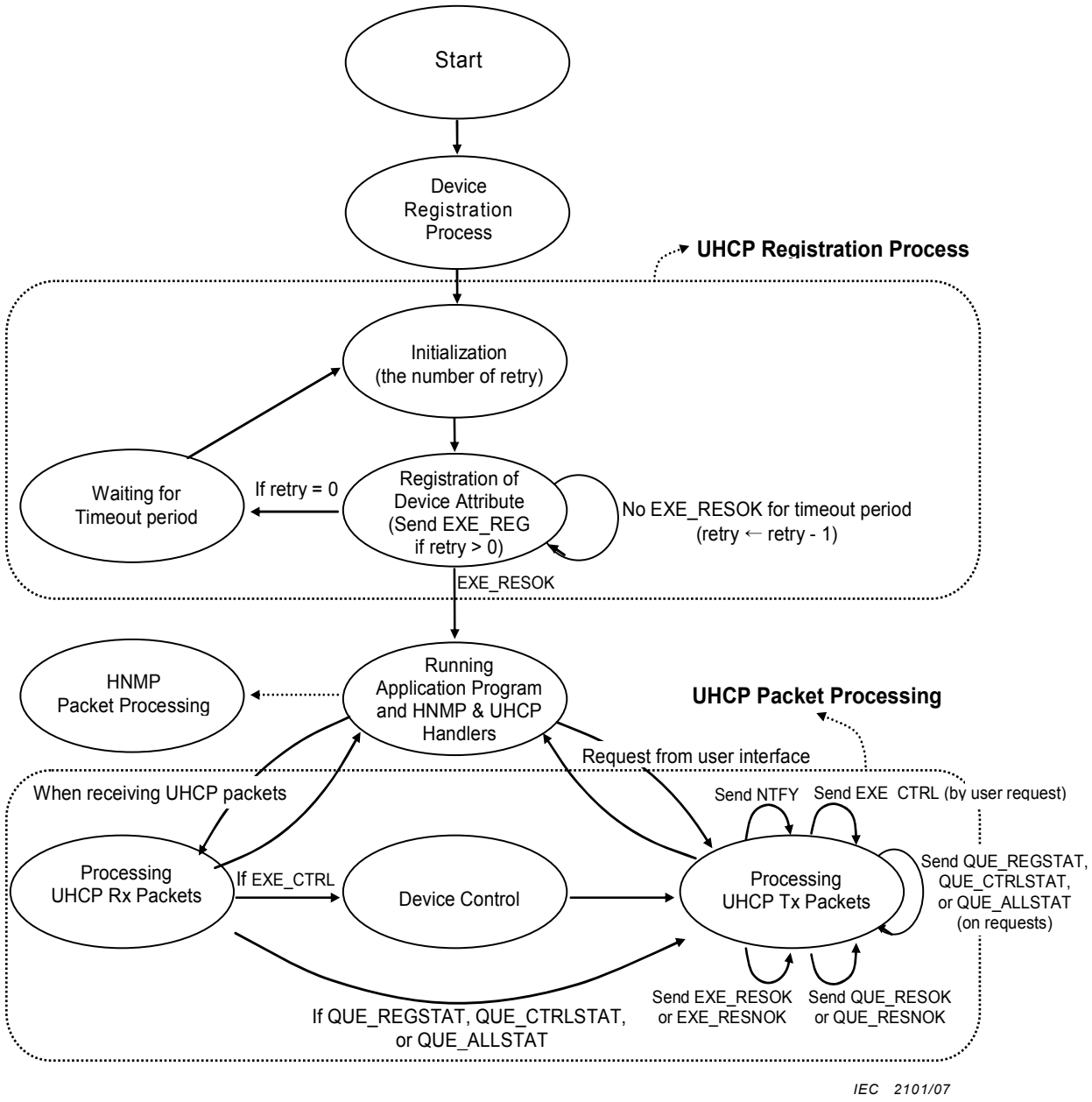


Figure B.1 – FSM of FS-CCPDEV for supporting UHCP

B.2 Start

Refer to Annex A.1.

B.3 Device registration process

Refer to Annex A from A.2 to A.6.

B.4 Initialization

The retry variable is initialized. If the FS-CCPDEV does not receive an EXE_RESOK packet within the timeout period, it attempts retries *N* times, where *N* is equal to the retry variable.

B.5 Registration of device attribute

Upon completion of initialization, the FS-CCPDEV sends an EXE_REG packet to the HSI of the home server, in order to register its device attributes and control or monitor items in the home server.

B.6 Waiting

If the FS-CCPDEV does not receive an EXE_RESOK response packet within the timeout period, the FS-CCPDEV attempts retries. If there is still no response after retries, the home server is regarded as not being ready, and the device attribute registration process is repeated after a waiting period.

B.7 Running application program and HNMP and UHCP handlers

Upon successful completion of the device attribute registration process, the FS-CCPDEV simultaneously executes handlers for the application program, HNMP packet processing and UHCP message processing.

B.8 Processing UHCP Rx packets

If the UHCP handler receives an UHCP packet during application program execution, the UHCP handler processes the received UHCP packet. If the packet is an EXE_CTRL, the device itself is controlled according to the control command. If the received packet is a query request command such as QUE_REGSTAT, QUE_CTRLSTAT or QUE_ALLSTAT, the HNMP handler writes its current registration status and control status in the QUE_RESOK payload and returns the packet. If the requested query is not feasible, the handler returns a QUE_RESNOK packet.

B.9 Processing UHCP Tx packets

The user shall be able to send control request commands such as EXE_CTRL and query request commands such as QUE_REGSTAT, QUE_CTRLSTAT or QUE_ALLSTAT to another device within the home network through the user interface. On the other hand, if the FS-CCPDEV receives a control request command, it shall be able to return an EXE_RESOK or EXE_RESNOK packet. Furthermore, the FS-CCPDEV shall be able to notify its abnormal status with the NTFY packet.

B.10 Device control

When the FS-CCPDEV's UHCP handler receives an EXE_CTRL packet, it shall be able to control itself according to the content of the EXE_CRTL packet payload.

———————

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

3, rue de Varembé
P.O. Box 131
CH-1211 Geneva 20
Switzerland

Tel: + 41 22 919 02 11
Fax: + 41 22 919 03 00
info@iec.ch
www.iec.ch