

**DRAFT**  
**Common control interface**  
**for networked digital audio and video products –**  
**Part 5-2: Transmission over networks – Signalling**

## Foreword

*[EdNote] To be added; see 5.2.1.3 and 6.1.3 in Directives Part 2*

## 0 Introduction

### 0.1 Structure of the family of standards

IEC 62379 specifies the Common Control Interface, a protocol for managing networked audiovisual equipment. It is intended to include the following Parts:

1. General
2. Audio
3. Video
4. Data
5. Transmission over networks

Part 1 specifies aspects which are common to all equipment, and includes an introduction to the Common Control Interface.

Parts 2 to 4 specify control of internal functions specific to equipment carrying particular types of live media. Part 4 refers to time-critical data such as commands to automation equipment, but not to packet data such as the control messages themselves.

Part 5 specifies control of transmission of these media over each individual network technology. It includes network specific management interfaces along with network specific control elements that integrate into the control framework.

Within part 5, sub-part 1 specifies management of aspects which are common to all network technologies. Sub-parts 3 onwards specify management of aspects which are particular to individual networking technologies.

Sub-part 2 (this document) specifies protocols which can be used between networking equipment to enable the setting up of calls which are routed across different networking technologies.

### 0.2 Support for “**Ffuture Nnetworks**” (FN)

#### 0.2.1 Rationale

Most current networks use IP (v4 or v6), but there is a growing recognition that this form of packet switching, which is appropriate for the irregular and unpredictable traffic generated by IT applications, has significant shortcomings for the “streamed” media which form an increasing proportion of the traffic carried on the Internet. For instance, ~~ITU-T Q21/13 and ISO/IEC JTC1/SC6 are both studying~~ developing “Future Networks” (FN) technology, which aims to meet the requirements of these and future services and overcome the deficiencies of the current IP based networks. ~~F~~Nuture networks, unlike “next generation” networks, ~~are~~is not

constrained by a requirement to use the same routing techniques as the current Internet. [This issue is also addressed in ITU-T Y.3001.](#)

When ~~these networks are~~FN is deployed ~~they~~it will need to co-exist and interwork with the previous technologies, for instance via gateways. To give the best performance, applications will need to be able to take advantage of improvements in Quality of Service (QoS) parameters such as latency and jitter, which are likely to be significantly lower on ~~FN~~the new networks, and of new services which ~~they~~it may provide.

Some likely features of ~~the new networks~~FN, and the way in which the protocols specified in this Standard support them, are described below.

### 0.2.2 Separation of route-finding from data-forwarding

~~Future networks are~~FN is likely to support circuit switching, in which routing decisions (which can be computationally intensive) are made by software before transmission of the data begins, and data forwarding (which, once the routing decisions have been made, is very simple) is handled by dedicated hardware. This reduces complexity in routers, and therefore also reduces the amount of power they consume. It also allows much more information about the transmission to be exchanged than could reasonably be accommodated in packet headers, and potentially allows the network to choose to switch some flows in the optical domain.

IP networks also separate discovery of the destination address, though not of the route to that address, from forwarding of the data, with protocols such as DNS and SIP. Also, MPLS can provide predefined routing within parts of an IP network.

The signalling protocols specified in this Standard provide for discovery of the route before transmission of the data begins, as in the call connection procedures that are used on circuit-switched networks, but the definition of a route (or rather, of a link: see 0.3 and 0.5) is broad enough that this can also apply to the discovery of the destination address on a IP network.

### 0.2.3 Mobility and resilience

The signalling protocols allow alternative routes to be discovered, and data flow to be switched between routes. This enables both failover when a link fails in a static network and handover when a mobile device moves from one cell or point of attachment to another.

The level of resilience can be chosen to suit the application, for instance in the case of failover the new route may be found when the failure occurs, or a backup route may be chosen and kept in reserve until needed, or multiple copies of the data may be sent by different routes.

### 0.2.4 Addressing

No assumptions are made about the form of addressing used by the underlying network. The signalling messages do not have a fixed format (or fixed length field) for an address, so a very wide range of addresses can be used, including URLs. [These addresses are independent of the addresses used for routing within the network, which will be dependent on the routing technology. Thus processes such as DNS lookup take place entirely within the network, and are not visible to end-systems, which therefore do not need to be aware of any low-level addressing other than that used on the link or subnetwork to which they are connected.](#)

As further detailed in 0.4, an address may include an *identifier* for a piece of equipment or a service, or a *locator* which shows where to find it, or both. There may be more than one locator, and the locators are then applied sequentially, in the same way that with a phone number containing a country code and area code, the call is routed first to the country and then to the area within that country.

### 0.2.5 Negotiation and reporting of QoS parameters and data format

Provision is made for the application, or the source of the data stream, to specify the bandwidth required. This information can then be used to reserve capacity on the links traversed by the data, if this is supported by the underlying network.

The signalling messages report the end-to-end QoS to be expected. This allows an application to know, for example, whether to expect the incoming data packets to arrive at regular or irregular intervals, and allocate buffer space appropriately, to minimise latency in the first case and dropped packets in the second.

The signalling messages also describe other features of the data, such as encoding and encapsulation, so they contain all the information the recipient will need to decode the data. This can be used in a variety of ways, including negotiating the best compromise between quality of compressed ~~material~~media and capacity available in the network, also to support transcoding in gateways. The way in which formats and encapsulations are represented is fully extensible, providing a globally-unique coding not only for standard formats and protocols but also for those that are manufacturer- or application-specific or experimental.

Thus some elements of the signalling messages (e.g. QoS parameters) are intended for the network, while others (e.g. encoding and encapsulation) are intended for the destination application. The network is expected to pass the latter through without interpreting them; for instance, it should not make assumptions about the service required based on the type of media being conveyed, as happens when “deep packet inspection” is used in IP networks.

*[EdNote] We maybe ought to go further, e.g. have an Information Element (see 5.3.2 and 5.6) that is a wrapper for all the information that the network is not to interpret, or maybe a separate wrapper for each layer in the OSI model.*

### 0.3 Structure of the network

A network is composed of end equipment, switches, and links.-

~~End equipment units convey media flows between media ports (such as analogue or digital audio or video connectors) and network ports, the ports being part of the unit. media ports may be physical interfaces, such as analogue or digital audio or video connectors, or internal sources or sinks of media flows, such as processing elements or memory. Switches similarly convey media data between network ports.~~

A link connects network ports on different units together. It may be a dedicated point-to-point link, such as Ethernet over Cat5e cabling or a leased line connection over a telecommunications network, but it may also be a subnetwork (switched or shared-media) that connects more than two ports, or a larger network such as a switched telecommunications network or the current Internet.

Switches convey media (and other) data between network ports. A switch contains two “planes”: the data plane, which forwards incoming data to the appropriate output port (or ports, in the case of multicast), and the control plane, which processes the signalling messages specified in clause 5.

The route followed by the media data from the source to the destination may need to traverse more than one kind of network. For instance, audio being transmitted from a studio in one location to a studio in another location will probably travel over the local infrastructure in both studios and a wide area network that links them; these three networks may well all use different technologies.

This sub-part of Part 5 specifies protocols which allow calls to be connected across heterogeneous networks which may be modelled in any level of detail, from individual

switches and point-to-point links to a single “cloud” with no internal structure visible, even if they use very different addressing schemes.

Different networking technologies also vary widely in the quality of service they are able to provide, from circuit-switched networks offering fixed latency and guaranteed delivery, through managed packet networks which can provide a certain level of assurance, to “best effort” packet networks (such as the Internet) which do not offer any guarantees at all.

#### **0.4 Addressing**

Addresses may perform one or both of two functions: location and identification. An identifier selects a particular object such as a physical unit, a service, or a piece of data. A locator shows where the required object is to be found.

Fixed-line telephone numbers are locators. They include a country code and an area code, which help the system to route the call to a specific telephone line. The call is answered by whoever happens to be near the phone at the time, and the caller then has to ask for the person they wish to speak to by name (which is, of course, an identifier).

The 48-bit MAC addresses used in Ethernet and other IEEE802 networks (apart from group addresses and locally-administered addresses) are identifiers which uniquely identify a particular interface but do not contain any information as to where on the network the interface is to be found.

The 20-byte NSAP addresses used in ATM networks include a locator (the prefix) and an identifier (the ESI) in separate fields.

In IPv4 addresses the subnet part is a locator and the host number is an identifier, although (unlike MAC addresses) the relationship between this identifier and a particular piece of equipment is often not permanent, and Network Address Translation means that a piece of equipment may appear to have different addresses in different parts of the network and several different pieces of equipment may appear to have the same address.

The principal form of unique identifier for physical equipment used in this standard is the IEEE 64-bit extended unique identifier (EUI-64, see <http://standards.ieee.org/regauth/oui/tutorials/EUI64.html>). An EUI-64 for any piece of equipment that has an interface which has a MAC address can be generated by inserting FFFE<sub>16</sub> into the middle of the MAC address. Identifiers that are not unique may also be used, for instance the name of a service may be used as the identifier for any piece of equipment that offers that service.

For calls within a subnetwork, only the identifier is required.

An address may consist of a series of addresses such that the call is routed to the first address, then from there to the next, etc. Each address is interpreted in the context of the equipment or location specified by the previous address, which thus acts as a locator. For instance, the address for an audio call may consist of the address of a piece of audio equipment followed by the identifier of a particular audio port on that equipment. The address of a piece of data may consist of its identifier (perhaps an IEC UUID or a SMPTE UMID) preceded by the address of a server (or group of servers) on which a copy is to be found. The address of a gateway (or even of a specific port on a gateway or switch) may be used as the locator for equipment accessed through that gateway; the address should, however, not be a unique identifier for the unit, because that would prevent calls being (re-)routed if the specified unit failed.

#### **0.5 Calls, routes, paths, and flows**

A call is defined in IEC62379-1 as conveying information either from a source unit to one or more destination units or (in both directions) between two units.

A path is a contiguous set of links along which the information might be conveyed from one unit to another. We use the term “path” when describing the process of discovering how to get from one unit to another across the network. A path may branch, in which case the different branches are different attempts at finding a way through; the eventual data flow will only be along one of the branches.

Once a set of links leading from one unit to the other has been discovered, it is called a “route”. A route may also branch, but only as required to deliver multicast data to several different destinations.

A route exists only in the control plane. It does not carry data, apart from signalling messages which are copied from one link to another (often being modified in the process) by software running in a computer which controls the unit through which they pass.

A flow is a single stream of information which is conveyed along a route.

A flow exists in the forwarding data plane. In the case of a circuit switched network, data units are copied from one link to another by the switching fabric with no intervention from the control plane software.

A call may carry several different flows. For instance, a connection between two studios may carry programme audio, talkback, and signalling (e.g. “on air” light) data as separate flows. Compressed data may be partitioned into a flow carrying a base layer and one or more flows carrying enhancement layers that improve the quality or add, for instance, surround sound channels; all destinations will receive the base layer, but the enhancement layer(s) would not be received by destinations that were not able to use the extra data, or did not have high enough bandwidth connections.

A call that needs high reliability (such as one that is part of the programme chain of a live broadcast, or in a safety-critical public address system) may have more than one route. For the highest reliability, all flows will be sent on two or more routes and the destination unit will use one copy and discard the other(s). Alternatively, a backup route might only carry the base layer, or it might be set up with no flows at all, with the flows only being connected in the event of failure of the main route.

Subclause 4.3 defines a structure for identifying calls, routes, and flows. This structure can be larger than would be the case if (like, for instance, ATM VCIs, or IP addresses and port numbers) it was transmitted with the data. Call identification consists of a globally-unique identification of the “owner”, which is normally either the originator of the call or the source of the data, and the “call reference” which is a 32-bit handle chosen by the owner. Thus the call identifier is unique across the whole network. We use 32 bits so that the size of a call reference is unlikely to be a limitation on the number of calls that a unit can make, and the time before a call reference value is re-used can be made adequately long.

We use 7 bits for the “route reference”, a handle on the route which a call follows. Most calls will only have one or two routes, but there will be cases where a call needs to set up new backup routes when existing routes fail and, again, the size of the field allows adequate time to elapse before handle values are re-used. Also, there is no particular benefit in shaving a small number of bits off this field.

The network is expected to ensure as far as possible that different routes for the same call do not pass through the same piece of equipment, which would then be a single point of failure.

The “flow reference” is a handle on the flow within the call, and is 24 bits. Again, most calls will only have a small number of flows, but this field has been made large enough that a call can carry a “tunnel” through which a large number of flows are routed (similar to an ATM VPC).

The flow reference is independent of the route, so flows that carry the same data by different routes can be easily identified. Where the same material is transmitted by different end units (for instance, where an audio input to the network is duplicated), the same call identifier (but, of course, different route references) may be used by the two units. If the two streams are identical (which can only be the case with digital inputs), they will use the same flow reference, but otherwise (for instance where the same analogue signal is digitised separately in each unit) they may need to use different flow references.

*[EdNote] Need a way to link the two units so that they both use the same call reference (which will be owned by one of them); the two routes will be requested separately by the destination which must use the same (temporary, see 0.6.3) call reference so that the network knows they are to be kept separate.*

Note that a route has an existence independent of the flows that follow it. A route may be established without any flows, in which case signalling messages can still be passed along it. ~~In a circuit-switched network, a flow will be set up exists in the switching fabric data plane,~~ whereas ~~the~~ route ~~will exist~~s only in software records ~~held by the switches in the control plane~~. Thus there are two distinct phases in connecting a flow: establishing the route, and setting up the switching fabric so that the data will be forwarded along it.

Also note that a flow passes through the same set of links and switches as the route it follows. This is different from the situation with SIP (RFC 3261), in which the media flow will in many cases not pass through the SIP server(s) that connected it. See also 6.1.

## 0.6 Route finding

### 0.6.1 General

The process of establishing a route begins with the caller creating a new route identifier and sending a "FindRoute request" signalling message on one or more of its network ports. Each recipient checks whether it is the called party, in which case it responds to the request, or is directly connected to it, in which case it sends the request on to it; otherwise it either rejects the request or sends it on via one or more of its network ports to units which are in some sense "closer" to the called party.

This standard does not specify how a switch chooses on which ports to forward a FindRoute request; the mechanism will usually be network-specific. Normally a switch will forward the request on just one port rather than flooding it to all ports. However, if a loop is formed it can easily be detected, by comparing the route identifier in an incoming request with those of requests that have already been forwarded, and this will occur before any data begin to flow.

As the request progresses through the network, it builds up a path which may have many tentacles reaching out towards the called party in different directions. When the request reaches the called party, a "FindRoute response" message is sent back along the path, and as it makes its way back towards the caller, the message accumulates "route metric" information which includes a measure of the "cost" of the path in terms of congestion and number of hops, and more literally the cost of the call if it passes over a public network. Each such response message is an offer to connect the route by the path over which it has travelled; the calling party chooses one offer and sends a "FindRoute confirmation" message which also causes all the other paths to be cleared down.

A ClearDown request may be issued at any time by any of the parties involved in the call; if issued by a switch it propagates in both directions. In the case of a multicast, in the direction towards the source it stops when it reaches a branching point.

A ClearDown received in reply to a FindRoute request is a negative response which may be a "refusal" or a "backtrack". A refusal indicates that the called party has been found but is not willing to accept the call, or that the called party's location has been found and the called party is not there; any resources reserved for the call are relinquished as the refusal

propagates back towards the caller. A backtrack indicates that the called party's location was not reachable via the path the request followed (maybe because the required resources are not available along that path, or because a link has been lost and the routing tables have not yet been updated to reflect the new topology), or that the called party is a service which was not available at the location the request reached but may be available at a different location; a switch receiving a backtrack may send out a further request along a different path, or if all possible paths have been tried pass the negative response back towards the caller.

If the call is charged for, charging begins when the confirmation is received by the service provider; inclusion of the new flow in routing tables etc may be delayed until the confirmation has been received, to prevent the user consuming the media without paying. Also, a call may be disconnected if no positive confirmation has been received within a specified time (which should be long enough for a user to reply to a "do you want to pay for this call?" alert).

The ClearDown request may be used as a negative FindRoute confirmation, for instance if the user does not wish to pay for the call or has got a better offer via another path.

### 0.6.2 Additional options

A switch may treat a positive response with a reduced capacity or high cost as a backtrack and try another path.

In any circumstances where a switch has forwarded a FindRoute request along more than one path, it propagates the first positive response, and any subsequent response that is a significant improvement on it, towards the caller; all such responses except the last include an "interim offer" indication which contains the switch's (EUI-64) identifier and a serial number. If the last response is not one that it would forward to the caller, it repeats the last response without the "interim offer" indication. A response message received by the caller may contain "interim offer" indications from several switches; one with no "interim offer" indications is final. A negative response is always final.

In the case where more than one response has been sent, a positive confirmation must be forwarded along the correct onward path and ClearDown requests sent on the other paths. The caller must include in the confirmation message all the "interim offer" indications from the response message, so that the correct path can be chosen at each switch.

*[EdNote 1] Procedures for the case where a switch receives requests for different routes of the same call are tbd; we need to ensure that the routes follow different paths as far as possible, but it is not clear how that would be done if both routes are routed simultaneously (and if they are not, how we stop the first route closing off options for routing the second route); also, we must not reject a call merely because there are parts of the path they have to share, though the caller needs to be told that the routes are not completely separated.*

*[EdNote 2] WeFN will also need some kind of "fast connect" so we're it's not slower than TCP in the case where a session is being set up to a destination that is already in the routing tables. That isn't mentioned in this Introduction, which concentrates on the service for media flows; see also 6.7 We've said that setting up a route takes no more effort than routing a TCP-SYN packet, which is not true if the destination is already in the routing tables, which it will be for the whole route if the caller has recently had a session with the same host, and will be for most of the route to popular destinations in any case.*

### 0.6.3 Call to receive a multicast

The request message may include an upper limit on the capacity required and also a preferred and a minimum capacity. For instance, in the case of linear PCM audio, it might request 16-bit 96kHz but be prepared to accept 24-bit if that is what the source is outputting, or 48kHz if the network is busy. Usually the source will have advertised the formats it supports, so the caller can be reasonably sure what the options are. As the request message propagates through the network, link capacity may be reserved based on the parameters in

the message (reserving the maximum at this stage, if available, else adjusting the message to reflect what is available).

The identifier for a multicast flow is owned by the source. The FindRoute messages use a temporary, "local", flow identifier owned by the caller; the response and confirmation messages include both identifiers.

Multicast calls include a specification (by the source) of the action to be taken by a switch that finds it is already carrying the flow requested by the caller. This action may be (a) to connect the caller to the flow without propagating the request on further, (b) to connect the caller and also inform the source that it has done so, or (c) to forward the request to the source. Case (c) will be appropriate for "private" flows where connection of the caller requires approval from the source. Except for case (c), switches must store some of the "route metric" information, including the latency, in case they later want to copy the flow to other callers; for parameters such as latency the value for the whole flow starting from the source, and maybe including processing delays upstream of the sending interface, is required, while for those that measure congestion etc the FindRoute response accumulates the value for the part of the flow downstream of the branching point.

### **0.7 Message format**

The messages used to implement the protocol are structured in a similar way to those in ITU-T Q.2931, with a fixed-format header followed by information elements in tag-length-value (TLV) form. As with a number of protocols and file formats that use TLV (including the ASN.1 coding used by SNMP), information elements may be nested within other information elements.

The TLV format makes it easy for the software in end equipment and switches to parse the messages, extracting the information they need and ignoring information which they do not need. The recipient of a text format, such as that used in SIP, must scan the whole message to find the newline characters, remove white space, and recognise keywords whether they are in upper case, lower case, or a combination of the two. This adds unnecessary complexity which may not be particularly onerous for PCs etc but can be more so for the embedded code in interface units.

The coding also provides various extension mechanisms that allow manufacturer-specific and application-specific information to be carried transparently and identified unambiguously.

### **0.8 Media coding and encapsulation**

To minimise the amount of transcoding required when transmitting audio and video over heterogeneous networks, and to increase the likelihood that equipment designed for different applications will interoperate successfully, this standard defines formats that can be used with any link technology.

The format for pulse-code modulated (PCM) audio is based on that specified in IEC 62365, which is in turn based on that specified in IEC 60958-4. Whereas IEC 62365 uses the fixed 48-octet cell size of ATM networks, the specification in 7.3 allows for variable package sizes. It supports networks in which it is efficient to send packages containing one sample per channel (to minimise latency) which, if the number of channels is small, as in mono or stereo, will be smaller than ATM cells. It also supports those with large per-package overheads (such as RTP/UDP/IP) for which larger packages should be sent.

## Common Control Interface – Part 5-2: Transmission over networks – Signalling

### 1 Scope

This International Standard specifies protocols which can be used between networking equipment to enable the setting up of calls which are routed across different networking technologies.

It also specifies encapsulation of digital media within those calls.

### 2 Normative references

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

*[EdNote] tbd; I guess we need to list 62379-1 and 62379-5-1, also ASN.1 (for Table 1), AES53 (for 7.3.2), and IEC62365 (for 7.3.3); what else? Also need an informative reference to Q.2931 and ISO TR 29181.*

### 3 Terms and definitions

For the purposes of this document, the terms and definitions given in IEC 62379-1 and IEC 62379-5-1 and the following terms and definitions apply.

*[EdNote 1] tbd; note that the above wording imports all the definitions etc in Parts 1 and 5-1.*

*[EdNote 2] I have added some definitions copied from 29181-3; I'm not sure whether we could import them in the same way as the 62379-1 and -5-1 definitions, considering it is a TR and thus not normative.*

*[EdNote 3] I have changed "foreground flow" to "synchronous flow" and "background flow" to "asynchronous flow" to align the text with the language of 29181-3. I have kept the names "foreground" and "background" for the services that carry them. Maybe this is a useful distinction; maybe it would be better to use "synchronous" and "asynchronous" everywhere.*

#### Data unit

A sequence of one or more octets which is conveyed across the network as a single unit.

#### Flow

A sequence of data units.

#### Asynchronous flow

A flow consisting of data units for which the time of arrival at the destination is unimportant.

#### Synchronous flow

A flow for which the network delay experienced by data units is required to be within specified limits. The size of the units, and also the number of units to be transmitted per unit time, may be fixed or may be variable with a defined upper limit.

**Connectionless data unit**

A data unit which is not part of a flow.

**Network delay**

Time from submission of a data unit to the network by the sender to its delivery to the recipient.

**Network element**

A piece of equipment which takes part in the process of conveying data and implements this Standard; includes switches, gateways, and interfaces, but not equipment internal to legacy networks.

**Adjacent**

Two network elements are adjacent if data can be sent from one to the other without passing through any other network element.

**End equipment**

Equipment that is connected to the network and produces or consumes data units.

**background**

best-effort point-to-point service carrying asynchronous flows

**call**

[NB must align with Part 1/5-1, also TR29181-3]

**call identifier**

The first 12 octets of a flow identifier (see 4.3)

**foreground**

fixed bit-rate one-to-many service carrying synchronous flows.

**link**

The means by which data units are conveyed between adjacent network elements.

**path**

A sequence of links

**route**

...

**route identifier**

The first 13 octets of a flow identifier (see 4.3), excluding the least significant bit of the 13th octet

**Abbreviations**

**3.1.1**

**information element**  
**IE**

**4 Identification**

**4.1 Byte order**

All multiple-octet quantities shall be coded with the most significant octet first.

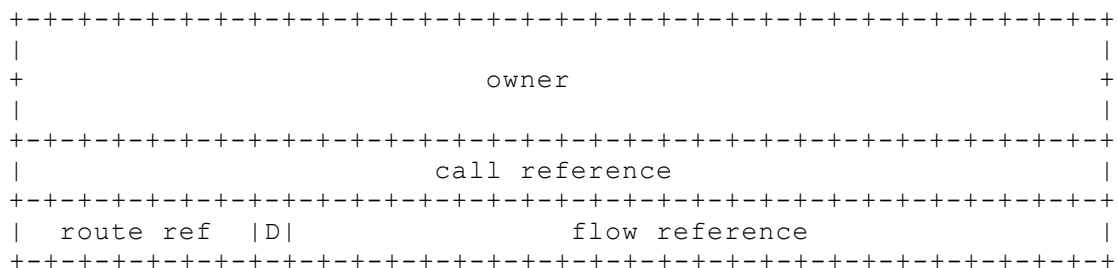
**4.2 Unit identification**

Each switch or end equipment unit shall have assigned to it a globally unique EUI-64.

NOTE The EUI-64 may be derived from the MAC address of one of the unit’s interfaces by concatenating the first three octets of the MAC address, hexadecimal FFFE, and the last three octets of the MAC address.

**4.3 Flow identifiers**

Each flow shall be identified by a 16-octet value formatted as shown in Figure 1.



**Figure 1: structure of flow identifier**

The first 8 octets shall contain an EUI-64 identifying the unit which created the flow identifier. This unit is referred to in this standard as the “owner” of the flow identifier.

The next 4 octets shall contain a nonzero reference number, chosen by the owner, identifying the call of which the flow is a part. Call references should be chosen in a way that maximises the time from when a call ceases to exist until its reference is re-used for a new call.

The next octet shall contain, in its most significant 7 bits, a nonzero reference number, chosen by the owner, selecting the route followed by the flow; and, in its least significant bit, a “direction” indicator which shall be 0 for a flow in the direction away from the owner and 1 for a flow in the direction towards it.

The remaining 3 octets shall contain a nonzero reference number, chosen by the owner, identifying the flow within the call. Where a pair of flows in opposite directions carries a two-way protocol such as TCP, the same flow reference value shall be used for each of them.

NOTE 1 The call reference distinguishes between different calls with the same owner. A call may be connected over several different routes (for instance, to give resilience in the event of failure) and the route reference distinguishes between them. The flow reference distinguishes between different flows within a call; for instance, a single call may carry video, programme audio, and talkback, and each of these streams will have its own flow reference. The flow reference is orthogonal to the route reference, so flows carrying the same content over different routes have the same flow reference, and flows carrying different content must have different flow references even if they do not follow the same route. Different routes may carry different sets of flows, for instance a backup route might only carry the most important flows, or different kinds of flow may have different backup routes.

Zero in the route reference field shall indicate all routes for the call. Except where specified otherwise (e.g. in 5.6.3), zero in the "direction" bit and the flow reference field shall indicate all flows for the route(s), and zero in the flow reference with 1 in the "direction" bit is reserved.

NOTE 2 These identifiers are the same at all points in the system and there is just one set of call reference values for each owner. This contrasts with call references in Q.2931 where a call has a different reference on every link, and a switch can use the same call reference for different calls as long as they are on different links. See also 0.5.

In the case of a unicast call, the owner shall be the caller. In the case of a multicast, the owner shall be the sender or, in the case where for resilience the same content is transmitted by more than one unit, one of the senders; a caller wishing to join a multicast shall use a temporary call identifier (which it owns) until the identifier of the multicast (which may need to be created by the sender) has been discovered.

#### 4.4 Address format

An address shall be represented as an octet string formatted as specified below.

NOTE It is thus a valid `TAddress` value (unless it is more than 255 octets long).

The `TDomain` value identifying an address in the form specified in this subclause shall be

{ iso(1) standard(0) iec62379 network(5) signalling(2) address\_format(2) }

The first octet shall contain a "type" code; the format and interpretation of the remainder of the address shall depend on the type and on the length of the octet string, as specified in Table 1.

**Table 1: address type codes**

type	remainder of octet string
0	second octet contains <i>n</i> , third to ( <i>n</i> +2)th inclusive are an address (which shall not be of type 0) which acts as the locator, remainder is another address (the "local" address), which is to be interpreted at the specified location (see note 1)
1	second octet contains <i>n</i> , third to ( <i>n</i> +2)th inclusive are an OID, remainder is a value (see notes 2,3,4)
2	second octet contains <i>n</i> , third to ( <i>n</i> +2)th inclusive are an OID, remainder is a value (see notes 2,3,5)
3	second octet contains <i>n</i> , third to ( <i>n</i> +2)th inclusive are a TDomain value (which is an OID), remainder is an address within that domain (see notes 2,6)
4	IPv4 address if 4 octets, subnet address + subnet mask if 8 octets
5	EUI-64 identifier
6	reserved for IPv6
7	URL
8	UDP, TCP, etc, port number coded in 2 octets as a 16-bit unsigned integer
9	IEC62379 block identifier
10	service name coded as a UTF8 string
11	NetBIOS unique name
12	NetBIOS group name
13	NSAP address (20 octets) or prefix

type	remainder of octet string
14	E.164 address
15-255	reserved

NOTE 1 In the case of type 0, the locator may be the address of a gateway, or some other identification of a subnetwork (such as an NSAP prefix or an IPv4 subnetwork address). It may also identify end equipment. The locator is not allowed to be a type 0 address so that the sequence of locations will be “flat” rather than nested.

NOTE 2 The OID in types 1-3 is coded in the same way as in ASN.1 Basic Encoding Rules (BER), including the compressed form for the first two arcs, but omitting the tag and length. These address types should only be used for address formats that cannot be supported by any of the other types. An IPv4 address, for instance, should use type 4 although it could also be expressed as a type 3 address or, if the target unit’s MIB includes its IP address, as a type 1. An IPv4 address with a port number should be type 0, with a type 4 locator and type 8 local address.

NOTE 3 The value which follows the OID in types 1-2 is coded using ASN.1 Basic Encoding Rules (BER), including the tag and length.

NOTE 4 Type 1 identifies a unit in whose MIB the object identified by the OID (such as unitName or unitLocation) has the specified value. In the case of scalar objects, the OID may omit the final zero arc. Units are not required to support all objects in their MIBs that could be used in this context, and switches are not required to remember any but the most obviously useful for the units connected to them. Objects in IEC62379 should be used in preference to those in other MIBs, for instance unitLocation (1.0.62379.1.1.1.2) rather than sysLocation (1.3.6.1.2.1.1.5).

NOTE 5 Type 2 identifies a port or other resource within a unit, by searching the unit’s MIB, usually for a columnar object such as aPortName to select an audio port. The OID omits the index arcs. As with type 1, units are not required to support all objects in their MIBs that could be used in this context, so a management terminal which has discovered the port name (e.g. from a status broadcast) should use type 9 instead.

NOTE 6 For types 3 onwards, the remainder of the value is an octet string containing the address in the appropriate format. Some codes are only valid in certain contexts, for instance an IEC62379 block identifier would not be valid on a subnetwork and a NetBIOS name would only be valid within a subnetwork that supported NetBIOS protocols.

## 5 Message format

### 5.1 General

All signalling messages shall consist of an octet string with the format shown in Figure 2.



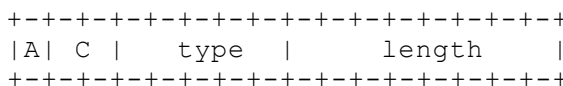
**Figure 2: signalling message format**

NOTE The header is specified in 5.2 and the variable part in 5.3. The format of the fixed part depends on the message type, and is specified in 5.5.

Encapsulation of the octet string for transmission on a link (see 6.1) depends on the technology used for the link, and is not specified in this standard. It should not place a limitation on the maximum length of a message.

### 5.2 Header

The header of a signalling message shall consist of two octets formatted as shown in Figure 3.



**Figure 3: signalling message header**

The most significant bit of the first octet shall be 1 for an acknowledgement, 0 otherwise. The next two bits shall code the message class as specified in Table 2. The least significant five bits shall code the message type as specified in Table 3.

The second octet shall contain the number of octets in the fixed part.

**Table 2: message class**

code	message class
00	request
01	response
10	confirmation
11	completion

**Table 3: message type**

code	message type
0-7	reserved for link management messages
8	FindRoute
9	ClearDown
10	AddFlow
11	NetworkData
12	UserDataEndToEndData
13	ConnectionlessData
14-15	reserved
16-19	reserved for link-specific resource management messages
20-31	reserved

NOTE Message types 0-7 and 16-19 may be used for carrying messages that are specific to a particular networking technology, on links that use that technology. Use of such message types should be negotiated when the link is established. Message types 14-15 and 20-31 may be specified in future versions of this standard, or in related standards, including for exchanging network topology information, and should not be used for other purposes.

## 5.3 Variable part

### 5.3.1 General

The variable part of a signalling message shall contain zero or more Information Elements (IEs), as specified in 5.3.2. The first IE shall begin in the octet immediately following the fixed part of the message, and each subsequent IE shall begin in the octet immediately following the IE that precedes it.

If a variable part includes more than one IE of a particular type, all the IEs of that type shall be adjacent.

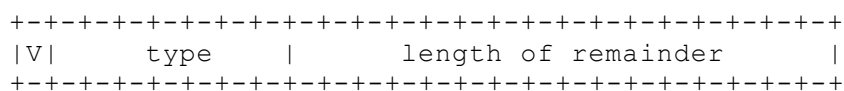
NOTE 1 This provision allows the recipient of a message to build a table showing the location in the message of each IE type, to simplify processing of the message.

If the encapsulation of the message does not specify the number of octets it occupies, or the number of octets is greater than the length of the message, then the octet immediately following the last IE (or the fixed part if there are no IEs) shall contain zero and any further octets shall be ignored by the recipient of the message.

NOTE 2 A zero in the IE type field is therefore interpreted as "end of message".

### 5.3.2 Information Element format

Each IE shall begin with a 3-octet header with the format shown in Figure 4



**Figure 4: Information Element header**

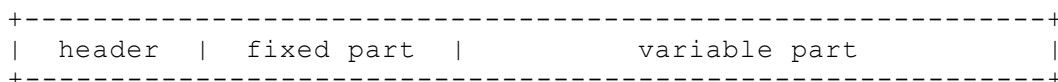
The most significant bit of the first octet shall be 1 if the IE includes a variable part, 0 otherwise.

The least significant 7 bits of the first octet shall contain the IE type, coded as specified in 5.5.

The next two octets shall contain the number of further octets in the IE.

In the case of an IE with no variable part, the remaining octets shall contain the fixed part.

In the case of an IE with a variable part, the fourth octet shall contain the number of octets in the fixed part. If this number is nonzero, the fixed part shall begin in the fifth octet and the variable part shall begin in the octet immediately following the fixed part, as shown in Figure 5. Otherwise, the variable part shall begin in the fifth octet.



**Figure 5: Information Element with nonempty fixed part and variable part**

NOTE 1 The format of the fixed part (in either case) depends on the type, and is specified in 5.5.

NOTE 2 If there is a variable part, the fixed part cannot be longer than 255 octets, whereas if there is no variable part the format allows the fixed part to be up to 65535 octets.

The format of the variable part (if present) shall be the same as the format of the variable part of a signalling message.

In this standard, IEs in the variable part of an IE are referred to as being “contained” or “nested” within the latter IE.

### 5.4 Data formats

Within the fixed part (of a message or an IE), the following conventions as to how various values will be represented shall apply unless the specification explicitly states otherwise:

- ◆ OID: coded in the same way as in ASN.1 Basic Encoding Rules (BER), including the compressed form for the first two arcs, but omitting the tag and length
- ◆ integer: most significant octet first; defined by the context as either signed or unsigned
- ◆ octet string: verbatim
- ◆ address: octet string, as specified in 4.4

NOTE The length of a value is fixed by the context; often it will be in an unsigned integer immediately preceding the value, or deduced from the length of the fixed part.

## 5.5 Contents of fixed and variable parts

### 5.5.1 FindRoute messages

The fixed part of a FindRoute message shall consist of 13 octets containing the route identifier for the route, with the least significant bit of the 13th octet coded as zero.

In messages that are not acknowledgements, the variable part shall contain IEs as specified in 6.2 for each message class.

In an acknowledgement message, the variable part shall contain a copy of each type 22 or 27 IE that was in the message being acknowledged. There should be no other IEs.

### 5.5.2 ClearDown messages

The fixed part of a ClearDown request message shall consist of 3 octets containing a serial number chosen by the sender. The recipient shall not attribute any significance to the serial number. The variable part shall contain IEs as specified in 6.3 for each message class.

The fixed part of the acknowledgement to a ClearDown request message shall consist of 3 octets containing the serial number that was in the message being acknowledged. The variable part should be empty.

NOTE The only purpose served by the serial number is to show which message is being acknowledged. The recipient cannot, for instance, assume that a gap in serial number values means that a message has been lost.

Other classes (i.e. response, confirmation, and completion) shall not be used with the ClearDown message type.

### 5.5.3 AddFlow messages

The fixed part of a AddFlow message shall consist of 13 octets containing the route identifier for the route, with the least significant bit of the 13th octet coded as zero. The variable part shall contain IEs as specified in 6.4 for each message class.

### 5.5.4 NetworkData and [UserDataEndToEndData](#) messages

The fixed part of a NetworkData or [UserDataEndToEndData](#) message shall consist of 13 octets containing the route identifier for the route, with the least significant bit of the 13th octet coded as zero. The variable part shall contain IEs as specified in 6.5.

### 5.5.5 ConnectionlessData messages

The fixed part of a ConnectionlessData request message shall be empty (i.e. of zero length). The variable part shall contain IEs as specified in 6.6.

Other classes (i.e. response, confirmation, and completion) shall not be used with the [ClearDownConnectionlessData](#) message type.

## 5.6 Information Element types

### 5.6.1 Coding of “type” field

The coding of the least significant 7 bits of the first octet of an IE shall be as specified in Table 4.

**Table 4: Information Element types**

type	interpretation	refer to
0	reserved for "end of variable part" indication	5.3.1
1, 2	reserved for address registration	Note
3	called address	5.6.2
4	flow descriptor	5.6.3
5	data format or protocol	5.6.4
6	start time	5.6.5
7	end time	5.6.6
8	call importance	5.6.7
9	service (or programme) name	5.6.8
10	source name	5.6.9
11	destination name	5.6.10
12	privilege level	5.6.11
13	password	5.6.12
14	charge for call	5.6.13
15	calling address	5.6.14
16	route metric	5.6.15
17	foreground service parameters	5.6.16
18	background service parameters	5.6.17
19	foreground link-specific resource allocation	5.6.18
20	background link-specific resource allocation	5.6.18
21	end-to-end delay	5.6.19
22	route identifier of multicast	5.6.20
23	cause	5.6.21
24	route to be cleared down	5.6.22
25	alternatives	5.6.23
26	group	5.6.24
27	interim offer	5.6.25
28	path MTU	5.6.26
29	number of destinations	5.6.27
30	destination selection	5.6.28
31	user data	5.6.29
32-63	reserved	
64-71	reserved for link-specific resource management	5.6.30
72-127	reserved	

The coding and semantics of the fixed and (if present) variable parts are specified below.

NOTE IE types 1 and 2 are intended for use with messages (not specified in this standard) which allow end systems to inform the network of identifiers by which they can be addressed. See also Note to Table 3.

### 5.6.2 Called address

The fixed part of a type 3 IE shall contain the address of the called party, in the form specified in 4.4. There shall be no variable part.

### 5.6.3 Flow descriptor

```

+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|F|  reserved |D|                               flow reference                               |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

**Figure 6: fixed part of flow descriptor IE**

The fixed part of a type 4 IE shall consist of four octets as illustrated in Figure 6. The most significant bit of the first octet shall be 1 for a synchronous (foreground) flow and 0 for an asynchronous (background) flow. The least significant bit of the first octet shall be 1 for a flow in the direction towards the owner and 0 for a flow in the direction away from the owner. The remaining bits in the first octet shall be coded as zero by the sender and ignored by the recipient.

The other three octets shall contain a flow reference as specified in 4.3, except that zero shall indicate that the flow reference is unspecified.

NOTE Unlike the case where zero indicates “all flows”, the “direction” flag is still valid when the flow reference is coded as zero.

There may be a variable part which may contain IE types 5-13 and 17-21, either directly or contained (directly or in type 26 IEs) in one or more type 25 IEs.

The direction and flow reference shall be combined with a route identifier specified elsewhere in the message, to form the flow identifier for the flow to which the IEs in the variable part, and the flag in the most significant bit of the first octet, apply.

### 5.6.4 Data format or protocol

The fixed part of a type 5 IE shall consist of an OID representing a property of the call, route, or flow to which it applies. There shall be no variable part.

EXAMPLE The type 4 IE for a flow carrying audio data may carry a type 5 IE with an OID descended from `audioSignalFormat` (specified in IEC62379-2) to describe the audio signal, and another type 5 IE which describes the encapsulation on the network.

### 5.6.5 Start time

The fixed part of a type 6 IE shall consist of 9 octets coded with a `DateTime` value (as specified in IEC62379-1) showing the UTC date and time by which the route or flow is to be connected. There shall be no variable part.

### 5.6.6 End time

The fixed part of a type 7 IE shall consist of 9 octets coded with a `DateTime` value (as specified in IEC62379-1) which is the UTC date and time after which the resources reserved for the route or flow may be released. There shall be no variable part.

### 5.6.7 Call importance

The fixed part of a type 8 IE shall consist of 1 octet coded with the importance followed by 1 octet coded with the reconnection priority. There shall be no variable part.

NOTE Importance and reconnection priority are specified in IEC62379-5-1.

### 5.6.8 Service (or programme) name

The fixed part of a type 9 IE shall consist of the service (or programme) name coded as a UTF8 string. There shall be no variable part.

NOTE Service name is specified in IEC62379-5-1.

### 5.6.9 Source name

The fixed part of a type 10 IE shall consist of the source name coded as a UTF8 string. There shall be no variable part.

NOTE Source name is specified in IEC62379-5-1.

### 5.6.10 Destination name

The fixed part of a type 11 IE shall consist of the destination name coded as a UTF8 string. There shall be no variable part.

NOTE Destination name is specified in IEC62379-5-1.

### 5.6.11 Privilege level

The fixed part of a type 12 IE shall consist of 1 octet coded with the privilege level. There shall be no variable part.

NOTE Privilege level is specified in IEC62379-1.

### 5.6.12 Password

The fixed part of a type 13 IE shall contain a password. There shall be no variable part.

*[EdNote] The format and how we stop snooping on this are tbd; should include an indication of the scope, e.g. for access to the network or to the destination equipment or service, and may also include user name.*

### 5.6.13 Charge for call

The fixed part of a type 14 IE shall either be of zero length or consist of 8 octets containing two 32-bit unsigned integers, followed by an OID which indicates how the integers are to be interpreted. There shall be no variable part.

A zero length fixed part shall indicate that there is no charge for the call.

EXAMPLE An OID might be defined which indicates that the first number is the number of pence to be charged for connecting the call and the second is the charge for call duration in thousandths of a penny per second.

*[EdNote] We ought to allow more than one IE of this type, e.g. one for the call charge and another to pay for the content, in which case we maybe need some nested IEs with billing information – I'm assuming the ISP/telco will charge the full amount to the user and then pass on appropriate amounts to the rights holder etc, as they currently do with 09 numbers in UK. The IE ought to contain some kind of assurance for the rights holder that that will happen. We also ought to support a model where the responder pays something, as with 0800 numbers and click-through adverts.*

### 5.6.14 Calling address

The fixed part of a type 15 IE shall contain the address of the calling party, in the form specified in 4.4. There shall be no variable part.

*[EdNote] We should maybe add an octet at the front, indicating how trustworthy this address is. ATM had a field which recorded whether the address had been supplied by the caller or the network, and if by the caller whether the network had checked it, and if so whether it appeared to be correct. Here we might have flags which a switch could set to show that the FindRoute request was received from a part of the network that it considered untrustworthy, or that it did not think the given address correctly identified the caller. Or maybe we should*

define an IE which could go in the variable part to indicate any issues a switch had with the address, including an identification of the switch and a code indicating what the problem is.

### 5.6.15 Route metric

The fixed part of a type 16 IE shall contain **[tbd]**. There shall be no variable part.

*[EdNote] The format needs to include a status coded as:*

*0 = accumulating value in this direction, in which case the value is the accumulated value up to and including the link over which the message is transmitted,*

*1 = as 0 but additionally indicating that the result must be reported to the sender,*

*2 = reporting the total value for the opposite direction.*

*Must also include number of hops, and proportion of available space required on each link (and/or on smallest capacity / most congested link), maybe also an indication of the QoS likely to be achieved.*

### 5.6.16 Foreground service parameters

The fixed part of a type 17 IE shall consist of eight octets. The first four shall be coded with the maximum number of payload octets in a data unit and the other four with the maximum number of data units per second. There shall be no variable part.

NOTE 1 These are the “package size” and “package rate” parameters reported in the list of sources specified in IEC62379-5-1.

NOTE 2 These parameters refer to the data units that the application will transmit on a flow. The maximum size of data unit supported by the route is reported in the type 28 IE.

The value for the maximum number of data units per second shall be the maximum that could be required, including an allowance for inaccuracy of the source frequency.

EXAMPLE Audio sampled at 48kHz ± 10ppm requires 47999.52 to 48000.48 data units per second (assuming one data unit per sample) so 48001 must be requested. The network will then round this up to a capacity that it can allocate, which may be different on different links traversed by the route.

### 5.6.17 Background service parameters

The fixed part of a type 18 IE shall consist of four, eight, twelve, or sixteen octets containing one to four unsigned 32-bit integers. The first ~~four~~ integer shall be ~~coded with the average packet size, the next four with the averagemean~~ number of ~~packets~~ data units to be ~~transmitted~~ per second. The second shall be the mean size of the data units, in octets, and the last third shall be four with the burst size specified as follows:- Assuming a model where packets are written to a buffer pool at the offered rate and transmitted on from that buffer pool at the specified average rate, the burst size shall be the maximum number of packets in the buffer pool. The fourth shall be an estimate of the total number of data units to be transmitted.

In each case zero shall indicate “unspecified” and FFFFFFFF<sub>16</sub> shall indicate “more than FFFFFFFE<sub>16</sub>”. Where there are less than four integers, the remaining values shall be interpreted to be unspecified.

There shall be no variable part.

NOTE The background service parameters are intended to give an estimate of the load that the flow will place on the network and on the recipient, but (unlike the foreground case) not form a basis for QoS guarantees. The parameters for the direction towards the sender of a message indicate the maximum load expected from incoming traffic.



A type 24 IE with a variable part shall indicate clearing down of the flows indicated by the type 4 IEs contained in it, but retention of the route even if it then does not carry any flows.

A type 24 IE with no variable part shall indicate clearing down of the route and all flows carried on it.

### 5.6.23 Alternatives

The fixed part of a type 25 IE shall be of zero length. The variable part may contain any number of IEs of one type which may be 3-14 or 17-18 or 26.

A type 25 IE shall indicate alternatives for the information in the IEs in the variable part, which shall occur in decreasing order of preference.

### 5.6.24 Group

The fixed part of a type 26 IE shall be of zero length. The variable part may contain IEs of types 3-14 and/or 17-18. The type 26 IE shall only occur inside a type 25 IE; it shall indicate that the IEs contained in it form a single alternative.

EXAMPLE A call requesting a high-quality video signal but prepared to accept a more heavily-compressed format might include a type 25 IE containing two type 26 IEs, one for each format, each containing a type 5 describing the format and a type 17 showing the throughput required to convey it. Note that the network should only look at the type 17, and not the type 5, when deciding whether the call can be connected (see 0.2.5).

### 5.6.25 Interim offer

The fixed part of a type 27 IE shall consist of 10 octets of which the first eight shall contain an EUI-64 identifying a switch and the remaining two shall contain a serial number, chosen by the switch identified by the first eight octets, which identifies an “offer” uniquely within the context of the switch and the call for which the offer is made. There shall be no variable part.

A FindRoute response message shall be an “interim offer” if it contains at least one type 27 IE, a “final offer” if it does not contain any type 27 IEs.

### 5.6.26 Path MTU

The fixed part of a type 28 IE shall consist of either one or two “packet size records” as specified in this subclause. There shall be no variable part.

If there is just one packet size record, it shall apply to data units on all flows. If there are two packet size records, the first shall apply to synchronous flows and the second to asynchronous flows.

A packet size record shall consist of three 32-bit unsigned integers as follows:

- a) the first shall be the maximum number of octets in a data unit, such that a data unit larger than indicated will be either fragmented or discarded;
- b) the second shall be the minimum number of octets in a nonempty data unit, such that any data unit with fewer octets will have padding added; and
- c) the third shall be the number of overhead octets, i.e. the difference between the second number and the number of octets required to transmit a data unit of that size.

In the case of a single link, the values shall reflect the technology used for that link.

EXAMPLE 1 Sending an  $n$ -octet UDP datagram over Ethernet using tagged packets with Ethertype 0800<sub>16</sub> requires 12 octets inter-packet gap, 8 octets preamble, 18 octets Ethernet header, 20 octets IP header, 8 octets UDP header,  $n$  octets data, and 4 octets FCS. The first number is  $1500 - (20 + 8) = 1472$ , the second is  $64 - (18 + 20 + 8 + 4) = 14$ , and the third is  $12 + 8 + 18 + 20 + 8 + 4 = 70$ .

EXAMPLE 2 When sending packet data over ATM using AAL5, in which a whole number of cells each with 5 octets header and 48 octets payload is used, the last 8 octets of the last cell of a packet being taken up with the AAL5 trailer, the first number is 65535, the second is  $48 - 8 = 40$ , and the third is  $5 + 8 = 13$ . The segmentation and reassembly process is not considered to be fragmentation, partly because it is invisible to the destination application but also because it would be unhelpful to set the maximum size of a data unit as low as 40 octets.

EXAMPLE 3 For a technology in which the packet header varies from 1 octet for a data unit of size 0 to 15 octets to 3 octets for a data unit of size 256 to 4095 octets, and there is no inter-packet gap, the first number is 4095, the second is 1, and the third is 1.

In the case of a route which traverses multiple links, the values shall be derived from the records for all the links, the first number being the smallest of the first numbers, the second being the largest of the second numbers, and the third being the largest of the third numbers.

EXAMPLE 4 For a route that traverses all three of the technologies in Examples 1-3, the first number would be 1472 (the smallest of 1472, 65535, and 4095), the second 40 (the largest of 14, 40, and 1), and the third 70 (the largest of 70, 13, and 1).

NOTE 1 The first number shows the largest packetdata unit size supported by the route.

NOTE 2 The second and third numbers provide an indication of the smallest packetdata unit size that can be transmitted efficiently. Together with the information in the type 21 IE, they allow an application to choose an appropriate packet size for a live media flow, balancing the increase in efficiency when sending larger packets against the increase in latency waiting for enough data to fill the larger packet.

NOTE 3 The specification of the third number allows for cases, as in Examples 2 and 3, where the number of overhead octets is different for different data unit sizes. The record does not attempt to model details of the encoding, so does not show that in Example 2 the next smallest size above 40 octets is 88 octets, nor that in example 4 the second and third numbers do not refer to the same technology. It also does not report the overhead for a zero-length packet, which may not be a good indication of the overhead incurred by small non-empty packets.

*[EdNote] I think this IE should also record the maximum bandwidth available, in octets per second including overheads (because this is likely to be independent of the packet size). What values are needed? Perhaps (a) total link capacity, (b) capacity that can be allocated to synchronous flows (which is usually around 70-90% of the total link capacity), (c) capacity allocated to synchronous flows, and (d) estimated capacity used by asynchronous flows; with (c) and (d) being instantaneous values at the time the value is written.*

### 5.6.27 Number of destinations

The fixed part of a type 29 IE shall consist of a single unsigned integer. There shall be no variable part.

NOTE The length is not specified, and there is no requirement to use the minimum number of octets. In many cases one or two octets will be enough, but for some applications (live broadcast of a sports event, for instance) a larger number will be needed, and the sender may choose to always use four octets. There may be applications (for instance, distributing global information in ubiquitous sensor networks) in which even four octets is not enough.

### 5.6.28 Selector for an individual destination

A type 30 IE selects a destination of a multicast and is in the context of a network element on the route of that multicast. The context may be the destination, or any network element through which the route passes on its way to the destination.

The fixed part shall be coded with a “selector” as specified below. There shall be no variable part.

An “arc” shall be coded as one octet containing  $n$ , followed by  $n$  octets containing a value which identifies a network element adjacent to the network element which is its context. The coding of this identification is private to the network element which is its context, but shall remain valid for as long as the link between the two network elements is operational.

NOTE 1 The coding of the arc must therefore remain valid after the route has been torn down.

[EdNote] Would it be better for an “arc” to be the EUI-64 of the network element? This fixes the size as 8 octets per arc, whereas for the regime described above it would be maybe 2 or 3.

for a port number and 5 for an IPv4 address, but 17 for an IPv6 address and maybe more for other forms of identification that might be used. Where point-to-point links are used, a switch will know the EUI-64 of each adjacent network element, but where a “link” is, say, a fairly large Ethernet network it might not.

In the context of the destination, the selector shall be of zero length.

In the context of a network element through which the route passes on its way to the destination, the selector shall consist of an arc which identifies the next network element through which the route passes on its way to the destination, followed by the selector in the context of that network element.

NOTE 2 The selector therefore specifies the path to the destination as a sequence of arcs and is valid even if the route is not set up. It thus contains an arc for each switch through which the route passes, including those at which it does not branch.

NOTE 3 A source may store the selector for each destination if the application requires it, but switches do not need to remember anything about individual destinations.

### 5.6.29 User data

The fixed part of a type 31 IE shall contain a data unit. There shall be no variable part.

*[EdNote\_1] This is intended for (1) the payload of an UserDataEndToEndData or ConnectionlessData message, and (2) data for higher-layer protocols, such as TCP SYN packets, that can usefully be carried in FindRoute messages.*

*[EdNote\_2] We need to ensure that TCP (or equivalent) sessions don't take longer to set up than with IP. If the destination is **not** already in the routing tables, the SYN packet will take about as long as a FindRoute message to arrive, and we don't want to take the extra time for the FindRoute response to return and the SYN packet then to be sent on the data plane. If the destination **is** already in the routing tables, IP will deliver the SYN packet more quickly, so we also need a way of accelerating the processing of FindRoute and ConnectionlessData messages in this case; it might simply be hardware acceleration for some of the control plane functions.*

### 5.6.30 Link-specific resource management

The format and interpretation of IE types 64 to 71 inclusive shall be specific to the networking technology used on the link over which the message passes.

NOTE These IE types are intended for management of low-level resources on the link, including flow labels (or virtual circuit numbers), port numbers, etc, and allocation of bandwidth. The format and interpretation are therefore not specified in this standard.

## 6 Protocols

### 6.1 General

The unit transmitting a signalling message shall be referred to as the “sender”, and the message shall be sent to a specific other unit which shall be referred to as the “recipient”. In the case of a message transmitted on a point-to-point link, the recipient shall be the unit at the opposite end of the link from the sender. Otherwise, the recipient shall be a specific unit connected to the same link and selected by an address, channel number, etc, as appropriate for the technology used on the link.

NOTE 1 A signalling message arriving at a network port is always processed locally. In a switched network it is never forwarded to another port by the routing fabric.

The recipient of a valid signalling message other than an acknowledgement shall send a “reply” to its sender. Where no reply is specified in this clause 6, the reply shall be an

acknowledgement message with the same class and type as the message that was received. Where a reply is specified, an acknowledgement may be sent in addition to the reply but is not required unless there will be a delay before the specified reply can be sent. The sender shall treat the acknowledgement or reply as an indication that the message has been received and will be processed. A unit shall not send an acknowledgement if it is possible that the message could be lost (for instance, through lack of buffer space) while awaiting processing in the protocol stack.

The recipient of an invalid signalling message shall silently ignore it.

The recipient of a signalling message where the context indicates that the message is a repetition should reply with an acknowledgement but otherwise ignore the message.

EXAMPLE 1 An incoming FindRoute request for a route of which the unit already has a record can be assumed to be a repetition if it comes from the same sender as the original FindRoute request; if it comes from a different sender it is not a repetition but indicates that two different paths from the caller have converged at the unit.

EXAMPLE 2 An incoming ClearDown for a set of routes and/or flows that do not exist may be assumed to be a repetition, the routes etc having been cleared down when the original was received.

EXAMPLE 3 If an incoming FindRoute request is a repetition of one to which a FindRoute response was sent in reply, an acknowledgement is sent. Although the repetition suggests the response message may have been lost, the response is not repeated at this stage. It is left to the normal timeout mechanism to repeat the response later if no reply to it is received.

The sender of a signalling message for which no acknowledgement or reply is received after a timeout shall repeat the message. After a longer timeout, or after a specified number of repetitions, the sender shall abandon the attempt to send the message and take appropriate action. The length of the timeout(s), number of repetitions, and action to be taken on abandoning the attempt to send the message depend on the technology used for the link and are therefore not specified in this standard.

NOTE 2 "Appropriate action" for a FindRoute will usually be to proceed as if a ClearDown had been received. "Appropriate action" for a ClearDown will usually be to complete the disconnection of the route or flow unilaterally. The failure to receive a reply might be treated as an indication that the link, or the intended recipient unit, has failed; however, implementers should also take into account the possibility that an error in the message, or in the other unit's interpretation of it, has caused it to be ignored.

## 6.2 Establishing a route

### 6.2.1 Connection of flows

The procedures specified in this subclause 6.2 may be used to establish a route on which there are no flows (it may still carry [UserDataEndToEndData](#) messages, and may have flows added later as specified in 6.4), or flows may be connected at the same time as the route is established. In the latter case, synchronous flows are connected by messages passing in the same direction as the flow, while asynchronous flows are connected by messages passing in the opposite direction. A type 19 or 20 IE shall be present in the variable part of the type 4 IE that describes a flow if, and only if, the flow is connected by the message.

NOTE 1 Synchronous flows require information (such as a layer 2 multicast address, and traffic parameters) from the source. Asynchronous flows require information (such as a layer 2 unicast address) from the destination. In each case, if information is required from the opposite end this can be provided in the preceding message in the sequence.

NOTE 2 A type 19 or 20 IE of zero length may be used if the technology used on the link does not require any of the information the IE would carry.

A route may be established using FindRoute request and response messages, or may also require a confirmation message. Flows may be connected by any of these messages, or by a completion message.

NOTE 3 Synchronous flows away from the caller and asynchronous flows towards the caller may be connected by the request message but it will often be more efficient for them to be connected by the confirmation message as this avoids setting up flows on branches that do not reach the destination. Synchronous flows towards the caller

and asynchronous flows away from the caller may be connected by the response message but in some circumstances (as when paid-for media should not be sent to the caller until the confirmation message has been received) will be connected by a completion message.

*[EdNote] We ~~will~~ might need to make exceptions to this for TGFN's "fast connect" mechanism for TCP-like sessions, or it might use a completely different mechanism; the request should ideally carry the outgoing SYN packet and also connect the flows in both directions, so that the SYN packet in the reverse direction can use the data plane connection. In the case of a label-switched layer 2, the caller needs to know the channel identifier for the incoming flow in time to receive the incoming SYN packet, so we probably need to specify that the response is sent instead of the ack, without waiting for it to propagate from the far end; the caller doesn't know whether the call is actually successful until it receives either the SYN packet or a ClearDown.*

The process of connecting or disconnecting a flow is specific to the technology used for each link, and is additional to the actions specified by this standard.

### 6.2.2 Request message

To establish a route, an end unit, which is referred to here as the "caller", shall send one or more FindRoute requests. If more than one is sent, they shall all be sent to different recipients. The route is then "pending" until further action is taken as specified in 6.2.4.3.

The message shall ~~include a type 15 (calling address) IE. It shall also~~ include sufficient information to identify the called party.

NOTE 1 This will usually be a type 3 (called address) IE but can, for instance, also be type 9 (programme name) or 10 (source name).

The message should include a type 15 (calling address) IE.

NOTE 2 The calling address may be omitted if the caller's identity is not to be disclosed to the called party, to preserve the caller's privacy. In the case of a point-to-point bidirectional call, a server can simply send replies on the call on which the request was received, without needing to know the client's address. (This is, of course, not the case on IP networks.) The server might restrict the access it provides to anonymous callers, or to reject all such calls. A destination can join a multicast without the source needing to know its address; again, the source can choose to prevent anonymous destinations joining the call (see 6.2.3.1).

The message shall include a type 4 (flow descriptor) IE for each flow that is to be connected on the route, except where the existence of the flow is implied by other parts of the message and the caller has nothing to say about it, for instance where the call is to "take" a source in whatever format the called party chooses to transmit. The type 4 IE may be at the top level (directly in the variable part of the message) or inside a type 25 (see below).

NOTE 3 The flow reference may be zero, to indicate that it should be chosen by the called party, unless the called party requires explicit flow reference values.

The IEs permitted or required inside a type 4 IE depend on the flags in the fixed part. For a synchronous flow, type 17 may be included. For an asynchronous flow, type 18 may be included. Also, either kind of flow may have types 5-13 IEs in the same way as the route (see below). For flows towards the caller (which is also the owner of the route identifier) types 17 and 18 specify the maximum values it can support; for flows in the opposite direction they specify parameters of the data stream it intends to transmit, and if type 18 is absent for an asynchronous flow a network-dependent default shall be assumed. Inclusion of types 19 and 20 shall be according to the requirements of the technology used for the link. Type 21 shall be included if, and only if, the flow is a synchronous flow in the direction away from the caller.

One or more type 5 IEs may be included at the top level (i.e. directly inside the variable part of the message) and/or contained in any of the type 4 IEs. A type 5 IE applies to the route if it is at the top level, to the flow otherwise. Where there are multiple type 5 IEs, each specifies a different aspect of the data format, encapsulation, protocol, etc.

| NOTE 43 Although this specification does not require any type 5 IEs to be included, the called party may refuse the call if the data format etc are not adequately specified.

One type 6 (start time) IE may be included at the top level, where it applies to all flows, or inside some or all of the type 4 IEs. If present, the flow(s) will not be set up in the switching fabric until shortly before the indicated time.

| NOTE 54 This allows capacity to be reserved ahead of time for scheduled events such as live outside broadcasts.

One type 7 (end time) may be included at the top level (where it applies to all flows) or inside some or all of the type 4 IEs. If present, it indicates that after the indicated time the flow can be torn down if any of the resources it uses are required for other calls, and if all flows have been torn down the route may be torn down also.

| NOTE 65 This may allow a flow to use resources which have been reserved for a flow that is scheduled to be connected later.

IE types 8-13 may be included both at the top level, where they apply to the path, and inside a type 4, where they apply to the flow.

One type 14 IE may be included, and if present shows the maximum the caller is willing to pay. If absent, the caller shall be given the opportunity to accept or reject any charge specified in the response message.

One type 16 (route metric) IE, with status 0 or 1, may be included.

One type 30 (destination selection) should be included if, and only if, the call is to join a multicast for which the approval of the source is, or might be, required.

| NOTE 76 The type 30, if included, will be empty (see 5.6.28).

Where the caller can accept several alternatives for the value conveyed by an IE of a particular type, for instance to receive at a lower quality if there is not enough bandwidth to support a higher quality signal, it may replace the IE with a type 25 IE which contains several IEs of the relevant type, one for each alternative. In the case of a value conveyed by more than one IE (for instance, multiple type 5 IEs specifying different aspects of the format plus a type 17 IE specifying the resources required to carry that format), the IEs specifying each alternative may be collected together in a type 26 IE, and the type 26 IEs collected into a type 25 IE. There may be more than one type 25 IE, in which case they are separate sets of alternatives to be applied orthogonally (for instance, one for the called address and another for the data format and flow descriptors). When the message is passed on (see 6.2.3), any alternatives that cannot be supported (for instance, needing more capacity than is available on the link) shall be omitted.

### **6.2.3 Action on receiving a FindRoute request**

#### **6.2.3.1 Joining an existing multicast in a switch**

A switch receiving a FindRoute request for a multicast which already passes through it shall take action depending on the “openness” of the multicast (see 5.6.20).

If the addition of a new destination requires the approval of the source, the switch shall pass the request (modified as required, e.g. with type 16 (route metric) and 30 (destination selector) IEs updated) on to the next network element in the direction towards the source, except that if the request does not contain a type 30 IE the switch shall not forward it but shall instead send a ClearDown in reply, as specified in 6.3.

Otherwise, the switch shall act as the responder as specified in 6.2.3.3.

If the source requires to be informed of the total number of destinations, the switch shall also send a NetworkData request containing a type 29 IE (see ) to the source not less than [tbc – 5?] seconds after the flow is connected. There should be an interval of at least [tbc – 2?] seconds between any two such messages for the same route.

*[EdNote 1] This leaves unspecified what you do if there aren't any flows; maybe it doesn't matter, as the destination won't be receiving any content.*

*[EdNote 2] Numbers to be confirmed in three places here and one place in each of 6.5.2 and 6.5.4.*

NOTE The message to the source will usually not be sent until the confirmation message arrives. Where multiple changes occur for the same multicast within a short space of time, the number of messages sent should be limited to one every [tbc – 2?] seconds.

### 6.2.3.2 Other cases in a switch

A switch receiving a FindRoute request which is not for a multicast that already passes though it shall

store any information that will be needed for processing subsequent messages, and pass on a copy (modified as required, e.g. with the type 16 (route metric) IE updated, and with an extra arc added to the type 30 IE if present) to one or more recipients, except that if there are no suitable recipients

it shall send a ClearDown in reply, as specified in 6.3.

NOTE 1 Subclause 6.1 requires an acknowledgement to be sent unless the ClearDown request is sent immediately.

NOTE 2 The information needing to be stored will include the sender of the incoming message, the recipient(s) of the outgoing message(s), and in each case the class of the most recent message (at this stage a request) and in the case of a response whether the offer was interim or final (see 5.6.25).

NOTE 3 The mechanism for deciding whether to pass the message on, and if so to which recipient(s), is not specified by this standard. It may use network-specific legacy protocols such as DNS or SIP. A new standard for exchanging network topology information may be developed using some of the message types which are shown as “reserved” in Table 3.

### 6.2.3.3 Action by end units

If an end unit receiving a FindRoute request can accept the call, it shall send a FindRoute response in reply. The unit is then referred to as the “responder”. Otherwise it shall send a ClearDown in reply, as specified in 6.3.

The FindRoute response message shall contain the same IEs as in the request (though they may be in a different order), with the following exceptions.

- a) For any type 25 IE, one alternative shall be chosen and shall replace the type 25 IE. In general, any tentative data (e.g. flow references coded as zero) shall be replaced with actual data.
- b) Type 14 IEs shall show the actual charge that will be levied when the confirmation is received; absence of a type 14 IE in the response message shall show that the call is free.
- c) Any type 16 IE with status 0 shall be removed. Any type 16 IE with status 1 shall have its status changed to 2. A type 16 IE with status 0 or 1 may be added.
- d) Any type 4 IE that connected its flow shall be removed. Within a type 4 IE, a type 17 or 18 IE shall indicate the actual values to be used; these may differ from the values in the request message. A type 19 (for foreground) or 20 (for background) IE shall be added if, and only if, the flow is to be connected by the response message.

NOTE 1 In the case of a synchronous flow that is connected by the response message, the requirement to add a type 19 IE applies even if the flow is already being transmitted on the link traversed by the message.

- e) Additional type 4 IEs may be included, for instance for a flow which is implied by a type 5 or 9 or 10 IE but for which no type 4 IE was included in the request message.
- f) If the call was to "take" a multicast, a type 22 IE shall be added, showing the path identifier and openness assigned to the multicast by its source. There is no correlation between the path selectors in the two identifiers, but the flow reference values shall correspond. If necessary, the flow reference values in type 4 IEs shall be changed to those used in the call indicated by the type 22 IE. (For instance, it may be clear from embedded type 5 IEs which flow is which, or there may only be one flow.) However, the "direction" flag in the low bit of the first octet of the type 4 IE shall still be coded as appropriate for the path in the fixed part of the message (so set to 1 for a flow towards the caller).
- g) Any type 30 IE in the request shall be omitted, except where the responder is the source of a multicast that requires approval.

NOTE 2 Where the responder is the source of a multicast that requires approval, switches on the part of the route that is already carrying the multicast can use the type 30 IE to route the response message and therefore do not need to remember the caller's route identifier.

- h) Any IEs of type 3, 5-13, or 15 at the top level that are unchanged from the request message should be omitted, likewise IEs of type 5-13 inside type 4 IEs.

NOTE 3 It is assumed that anything that requires the information from these IEs will have collected it from the request message.

#### **6.2.4 Action on receiving on a FindRoute response**

##### **6.2.4.1 Messages including a type 30 IE**

When a unit other than the caller receives a valid FindRoute response message which includes a type 30 IE, if the first arc identifies a network element to which the multicast identified in the type 22 IE is already being transmitted, it shall remove the first arc from the type 30 IE and pass the message on to that network element, modifying it as specified in 6.2.4.2.

NOTE 4 Removing the first arc changes the context to the recipient.

Otherwise it shall remove the type 30 IE and process the message as specified in 6.2.4.2.

##### **6.2.4.2 Messages not including a type 30 IE**

When a unit other than the caller receives a valid FindRoute response message which does not include a type 30 IE, if the unit has no record of the route, or has already sent a final offer (a FindRoute response message without a type 27 IE) for the route, or has received a FindRoute confirmation message for the route, it shall send a ClearDown request in reply.

Otherwise, it shall pass the message on towards the caller, modified as specified below.

NOTE 1 Subclause 6.1 specifies that an acknowledgement may (but need not) be sent in the former case, and must be sent in the latter case.

If the message is an interim offer, or there are other recipients of the request message from which a final offer (or a ClearDown request) has not yet been received, and at least one interim offer for the route has already been sent towards the caller, the unit may store the message for a limited time (the limit not being specified in this standard) in the expectation that other offers will be received, and at any time before the expiry of the time limit choose one offer from all those that have been received, passing that offer on towards the caller and replying with a ClearDown request to the others.

NOTE 2 The first offer is passed on towards the caller without delay; this minimises the time to make the connection in the case where the caller will accept the first offer it receives.

The message that is passed on shall be modified as follows.

- a) If the request was passed to more than one recipient, and either the response is an interim offer or there is at least one other recipient from which a final offer has not yet been received, a type 27 IE shall be added containing the unit's EUI-64 and a serial number from which the unit can identify the sender of the response message. Otherwise, no type 27 IE shall be added and the unit shall remember the sender as the sender of the final offer.

NOTE 3 This information may be required when the confirmation message arrives, see 6.2.5.

- b) IEs such as type 16 (route metric) (except where the status is 2) and type 21 (end-to-end delay) shall be updated to add the values for the link over which the message is to be sent.
- c) A type 14 IE may be added.

*[EdNote] See note on 5.6.13*

- d) Link-specific IEs such as types 19 and 20 may need to be modified to apply to the link over which the message is sent.

### 6.2.4.3 Action of the caller

When a unit receives a valid FindRoute response message for a route for which it is the caller it shall proceed as follows:

- ◆ If the route is no longer pending, or for any reason the offer is unacceptable, it shall reject the offer by replying with a ClearDown request.

NOTE 1 Reasons for rejecting an offer could include unacceptable parameters such as a data format which the unit cannot process, or an offer which is in some sense "better" (e.g. shorter route) having already been received.

- ◆ Otherwise, if the message is an interim offer, or there are other recipients of the request message from which a final offer (or a ClearDown request) has not yet been received, the unit may store it for later processing (when the final offer arrives, or after a timeout).
- ◆ Otherwise, if the message does not contain a type 14 IE with a nonempty fixed part, nor any flows that are not connected by the response message, nor a type 16 IE with status 1, and no previous interim offers have been received, then establishment of the route is complete, the route is no longer pending, and no further action is required other than sending an acknowledgement and any actions specified elsewhere than in this standard.

NOTE 2 Such actions would include link-specific actions required to connect a flow.

- ◆ Otherwise, a FindRoute confirmation as specified below shall be sent in reply, possibly after awaiting the user's agreement that the offer is acceptable.

NOTE 3 User confirmation is particularly relevant in the case where there is a type 14 IE with a nonempty fixed part. The mechanism for requesting user confirmation is application-specific and outside the scope of this standard.

NOTE 4 Subclause 6.1 requires an acknowledgement to be sent in all cases where a reply is not sent immediately.

A stored interim offer may be processed at any time, either to reject it (for instance because a better offer has been received) or to send a confirmation. When a confirmation of an offer is sent, all other offers awaiting processing should be rejected.

When a confirmation of an offer has been sent, the route is no longer pending.

The FindRoute confirmation message shall contain the same IEs as in the response (though they may be in a different order), with the following exceptions.

- a) All IEs except types 4, 14, 16, 22, and 27 shall be removed.
- b) Any type 16 IE with status 0 or 2 shall be removed. Any type 16 IE with status 1 shall have its status changed to 2. A type 16 IE with status 0 or 1 may be added.
- c) Any type 4 IE for a flow that was connected by the response message shall be removed.

- d) Within a type 4 IE, IEs of types 5-13 that are unchanged from the response message should be omitted. Types 19 and 21 (for foreground) or type 20 (for background) shall be added if, and only if, the flow is in the correct direction to be connected by this message.

NOTE 5: Whereas the responder can choose whether to connect a flow by the response message or wait until the completion message, the caller has no discretion in the case of the confirmation message because it is the last opportunity to connect the flow.

- e) Additional type 4 IEs may be included, for instance for a flow which it would be inappropriate to connect before the route has been chosen. Types 19 and 21 (for foreground) or type 20 (for background) shall be included if, and only if, the flow is in the correct direction to be connected by this message.
- f) If there is a type 14 showing a nonzero charge in the response message, there shall be an identical type 14 in the confirmation message, showing that the caller is willing to pay the indicated charge. Otherwise no charge shall be levied. If there is no type 14 IE in the response message, or there is one which shows that the call is free, then there shall be no type 14 IE in the confirmation message.

*[EdNote] See note on 5.6.13*

### 6.2.5 Passing on a FindRoute confirmation

When a unit other than the responder receives a valid FindRoute confirmation message, it shall proceed as follows.

- ◆ If the unit has no record of the route, it shall send a ClearDown request in reply.
- ◆ Otherwise, it shall pass the message on towards the responder, as specified below, except that if it cannot identify the unit to which the message should be passed it shall clear the route down as specified in 6.3.

NOTE Subclause 6.1 requires an acknowledgement to be sent in reply if the message is passed on.

If the request was passed to exactly one recipient, the response shall be passed to that same recipient. The response should not contain a type 27 IE with the unit's EUI-64.

NOTE In this case it is expected that the unit will not have added a type 27 IE to the response message. It may check for a type 27 IE with its own EUI-64 in the response message, and take appropriate action if one is found, or it may simply pass the message on without checking it.

If the request was passed to more than one recipient, then if the confirmation message contains a type 27 IE with the unit's EUI-64, the IE shall be removed from the message and used to identify the recipient to which the message is passed. Otherwise the message shall be passed to the sender of the response message that was passed on as a final offer. In either case, the unit shall send a ClearDown request to all the other recipients of the FindRoute request for the route that have not already been sent a ClearDown.

### 6.2.6 Action of the responder on receiving a FindRoute confirmation

When a unit receives a valid FindRoute confirmation message for a route for which it is the responder, if the message does not contain any flows that are not connected by the confirmation message, nor a type 16 IE with status 1, no further action is required other than sending an acknowledgement and any actions specified elsewhere than in this standard. Otherwise, it shall send in reply a FindRoute completion message as specified below.

NOTE 1 Subclause 6.1 requires an acknowledgement to be sent if a completion message is not sent.

The FindRoute completion message shall contain the same IEs as in the confirmation (though they may be in a different order), with the following exceptions.

- a) All IEs at the top level except types 4, 16, 22, and 27 shall be removed.
- b) Any type 16 IE with status 0 or 2 shall be removed. Any type 16 IE with status 1 shall have its status changed to 2. A type 16 IE with status 0 may be added.

- c) Any type 4 IE for a flow that was connected by the confirmation message shall be removed.
- d) Within a type 4 IE, IEs of type 5-13 that are unchanged from the confirmation message should be removed. Types 19 and 21 (for foreground) or type 20 (for background) shall be added.

NOTE 2 We assume the flow is in the correct direction to be connected by this message, because 6.2.4.3 requires any flows in the other direction to be connected by the confirmation message.

- e) Additional type 4 IEs may be included, but only for flows that are connected by this message. Types 19 and 21 (for foreground) or type 20 (for background) shall therefore be included.

### 6.2.7 FindRoute completion message

When a unit other than the caller receives a valid FindRoute completion message, it shall proceed as follows.

- ◆ If the unit has no record of the route, it shall send a ClearDown request in reply.
- ◆ Otherwise, it shall pass the message on towards the caller, modified as specified below.

NOTE 1 Subclause 6.1 requires that an acknowledgement be sent in the latter case.

The message that is passed on shall be modified as follows.

- a) IEs such as type 16 (route metric) and type 21 (end-to-end delay) shall be updated to add the values for the link over which the message is to be sent.
- b) Link-specific IEs such as types 19 and 20 may need to be modified to apply to the link over which the message is sent.

When the caller receives a valid FindRoute completion message, it shall take no action other than sending an acknowledgement and any actions specified elsewhere than in this standard.

NOTE 2 Such actions would include link-specific actions required to connect a flow.

## 6.3 Disconnection of routes and flows

### 6.3.1 General

Whereas the protocol for setting up a route involves messages passing along the whole length of the path between the caller and the responder, disconnection proceeds independently on each link. This makes the process more robust, simplifies clearing down a route from the middle (as must be done when a link breaks), and allows switches to reroute calls around a broken link without tearing down and rebuilding the whole route. It also simplifies the situation where a branch of a multicast is being pruned back from a destination that has ceased to receive it at the same time as another destination in the same part of the network is joining it.

When a route is cleared down, all flows that follow the route are also cleared down. Other routes for the same call are not affected; to clear down a call, each route must be cleared down separately.

Alternatively, an individual flow may be cleared down along a route, leaving other flows in place. Again, to remove a flow from a call that has several routes the flow must be cleared down separately for each route.

Clearing down of a flow or route may be initiated by any of the units through which it passes.

NOTE A single message can request clearing down of multiple flows and/or routes. This is useful in the event of a broken link.

### 6.3.2 ClearDown request message

A unit shall request clearing down of one or more flows and/or routes across a link by sending a ClearDown request message.

The variable part of a ClearDown request message shall contain one or more type 24 IEs. A type 24 IE with no type 4 IEs contained within it shall request clearing down of the indicated route and removal of all flows connected on it. A type 24 IE containing at least one type 4 IE shall request removal of the flow(s) specified by the type 4 IE(s), leaving the route in place even if all its flows have been removed.

In the case of a multicast for which a FindRoute response has been received by the sender of the ClearDown, the type 24 IE shall show the source's route identifier, not the caller's.

NOTE Between sending a FindRoute response and receiving the acknowledgement, a unit must be prepared to receive a ClearDown with either identifier and must interpret the "direction" bit in a type 4 IE in the context of the identifier in the type 24 IE in which it is contained.

The message should not contain any other IEs, either at the outer level or inside other IEs, except type 23.

There may be one or more type 23 (cause) IEs at the top level and/or in the variable part of each type 24 or type 4 IE. A type 23 IE at the top level applies to all flows and/or routes in the message. A type 23 IE in the variable part of a type 24 IE applies to all flows being removed from that route. If there is no type 23 IE applying to a flow, the cause shall be assumed to be "normal call clearing".

### 6.3.3 Action on receiving a ClearDown request message

A unit receiving a ClearDown request message shall send an acknowledgement in reply, and shall take whatever action is necessary to remove flows and clear down routes as specified in the message, including issuing a ClearDown request for other links traversed by the route as appropriate.

NOTE 1 This action is specific to the technology used for the network and the link. Removal of a flow may require the use of other messages, for instance of types 16-19.

NOTE 2 If part of a unicast route has been cleared down, or if, when part of a multicast has been cleared down, the remaining part has no source, or no destinations, it will usually be appropriate to clear the route down completely. A similar consideration applies to individual flows. However, in some circumstances, or for some applications, a different action may be preferable, for instance when the cause is a broken link to attempt to reroute around the break instead of clearing the whole route down.

The recipient of the message shall ignore any IE which specifies a flow or route which it does not believe is present on the link, or which it is already in the process of removing.

NOTE 3 This covers the case where the message is a repetition. It also avoids problems in the event that clearing down of a route is requested from both ends at the same time.

## 6.4 Adding new flows to an existing route

*[EdNote] tbd. Format will be as FindRoute, but the flows are added to an existing route. IE types 3 and 15 are optional, and if present must match the existing endpoints, though they do not need to be the same as the addresses used in the original FindRoute.*

## 6.5 Information messages related to a route

### 6.5.1 General

NetworkData and [UserDataEndToEndData](#) messages shall be used to convey along a route information that is not part of a flow.

A unit receiving an n UserDataEndToEndData message shall send an acknowledgement for it (see 6.1) and also proceed as follows:

- a) If the unit has no record of the route, it shall send a ClearDown request for the route in reply (in addition to the acknowledgement).
- b) Otherwise, if the unit is an endpoint of the route it shall process the IEs it contains and may send in reply an n UserDataEndToEndData message of the subsequent class (e.g. a response in reply to a request).
- c) Otherwise, it shall pass the message on to the next unit along the route.

A unit receiving a NetworkData message shall proceed as for an n UserDataEndToEndData message except that in case (c) it shall take any action specified elsewhere in this subclause 6.5, and also make appropriate changes to any IEs of types 16, 21, 28, and 30 that are present, before passing the message on.

### 6.5.2 Notification of number of destinations

A switch shall generate a NetworkData request message containing a single type 29 IE to inform the source of a change in the number of destinations as specified in 6.2.3.1.

Before passing on a NetworkData request message containing a type 29 IE a switch shall update its own record of the number of destinations and add to the number in the IE the number of destinations reached via neighbours other than the one from which the message was received.

The message shall not be passed on until *[tbc – ??]* seconds have elapsed since the previous NetworkData request message (if any) was sent for the same route. If further such messages for the same route are received during that time, only one shall be passed on.

*[EdNote] The way this would be implemented is as follows, but it might be rather over-the-top to describe it in detail. For each route with openness 01 there is state consisting of (1) whether there is an unreported change (2) whether a message has been sent recently. Sending a message clears (1), sets (2), and starts a timeout which when it expires clears (2) if (1) is clear else sends another message. An incoming message, also a local changes such as ~~the one~~ described in 6.2.3.1, sends a message if (2) is clear, sets (1) else.*

### 6.5.3 Changing service parameters

*[EdNote] This subclause will describe the cross-layer negotiation for existing routes, e.g. when the network asks the application to reduce the bandwidth used (which the application might do by compressing the content more), or the application asks if it can increase the bandwidth.*

### 6.5.4 Out-of-band data

An application may use an n UserDataEndToEndData message to send along a route information that is not part of any flow. The message shall include a type 31 IE holding the data unit to be conveyed, and may include a type 5 IE which indicates how it should be interpreted.

A message from the source of a multicast may include a type 30 IE, in which case it shall be delivered only to the destination selected; otherwise it shall be delivered to all destinations.

An application shall not send a EndToEndData request message if less than *[tbc – ??]* seconds have elapsed since the previous EndToEndData request message (if any) was sent for the same route.

NOTE EndToEndData is intended for occasional protocol messages between application entities in the endpoints. It is not intended to be used instead of connecting an asynchronous flow. See also 6.7.*[EdNote] This raises a fairly*

~~fundamental issue regarding the architecture of FN. I can imagine two scenarios: (1) two planes, as in the current draft of ISO 29181-3, and (2) three planes, with a third, “mezzanine”, plane between the other two which provides some hardware acceleration to the processing of signalling messages (including routing UserData messages without needing any software intervention) and maybe also routes IP datagrams. Scenario (1) requires UserData and ConnectionlessData messages to be used only in situations that will occur very rarely. In scenario (2) we can encourage their use for functions that in IP networks use RTCP, ICMP, etc; the mezzanine plane might also provide a way of achieving fast connect for TCP-like sessions (see EdNote at end of).~~

~~[EdNote] In any case, we need a mechanism to set limits on how many of these messages can be sent, to prevent them overloading the processor (or the mezzanine plane) in a switch and denying service (intentionally or unintentionally) to other functions such as call connection. Maybe we specify there shall always be a response, and a message shall not be sent until the previous one has been acknowledged; or maybe we can treat anything that sends too many of them as “jabbering” and take the link to it down. We need to be careful not to treat repetitions (e.g. when an acknowledgement is lost) as illegal.~~

## 6.6 Connectionless data messages

### 6.6.1 Request

If an application requires to send an isolated data unit to a remote application, it may use a ConnectionlessData request message, containing the following IEs:

- ◆ type 3 shall be included, specifying the destination
- ◆ types 10, 12, 13, and/or 15 may be included, to identify the sender to the destination application
- ◆ type 31 should be included; if omitted, the data unit is assumed to be of zero length
- ◆ type 5 may be included, to indicate how the message should be interpreted
- ◆ type 30 (in the context of the sender of the message) may be included; a reply shall be sent if, and only if, this IE is present

An application shall not send a ConnectionlessData request message if less than [tbc – ??] seconds have elapsed since the previous ConnectionlessData request message (if any) was sent.

NOTE ConnectionlessData is intended for occasional isolated messages; it is not intended as a replacement for Internet Protocol. See also 6.7.

*[EdNote\_1] Should there also be a payment mechanism, analogous to paying for text messages, using some variation on type 14? Or do we just say that anything that needs paying for should use a connection?*

*[EdNote\_2] When there is a type 30, maybe there should also be some kind of serial number, so the reply can be matched up with the original request.*

*[EdNote\_3] Unlike setting up connections, there isn't a way to detect loops in the path, so we also need a “time to live” field (as in IPv6), unless we use the size of the type 30 IE as a measure of the length of the path.*

### 6.6.2 Action on receiving request

If the recipient of a ConnectionlessData request message is its destination, it shall process the message. If, and only if, the message includes a type 30 IE, it shall send a ConnectionlessData response message including a copy of the type 30 IE from the request message and any of the IEs that may appear in a ConnectionlessData request message except type 3.

A switch receiving a ConnectionlessData request message shall forward it to an adjacent network element, first updating the type 30 IE (if present).

*[EdNote] It may also need to update “time to live” (see above).*

NOTE As with a FindRoute request (see ), the mechanism for deciding to which adjacent network element to forward it is not specified by this standard. Unlike the FindRoute request, only one copy of the message is forwarded.

### 6.6.3 Action on receiving response

The recipient of a ConnectionlessData response message shall examine the type 30 IE and if it is nonempty remove the first arc from the type 30 IE and forward the message to the unit identified by it.

If the type 30 IE is empty, the recipient shall assume it is the destination for the message.

## 6.7 “Fast connect” protocol

*[EdNote 1] FN will need an efficient way to connect asynchronous flows for the kind of transactions that occur with HTTP where a TCP session is connected, carries about a dozen packets, and is then terminated. Although the session begins with an exchange of SYN packets, the necessary routing information will typically already be in the data plane caches, so the SYN packets can be routed without intervention from the control plane and the exchange takes less time than setting up an asynchronous flow. The details of this facility are not yet decided; it might use a simplified version of the FindRoute request message which can be processed without software intervention provided the route to the destination is in a cache, or additional routing functionality in the data plane, or some other mechanism. This subclause will describe any signalling or other messages needed to support it.*

*[EdNote 2] It is not yet clear whether the service defined in this subclause will be suitable for applications that currently use UDP. Those that currently use RTP over UDP should, of course, use synchronous flows instead*

*[EdNote 3] The Notes to 6.5.4 and 6.6.1 refer to this subclause, in the expectation that it will define a mechanism which can be used for a service which can be used in the same way as IP. They may need modification.*

## 7 Media formats

### 7.1 Identification

Encapsulation formats specified in this clause 7 shall be identified by object identifiers rooted at the following location in the MIB tree:

```
mediaEncapsulation OBJECT IDENTIFIER ::= { iso(1) standard(0)
    iec62379 network(5) signalling(2) encapsulation(3) }
```

Where the definitions make reference to parameters, these parameters shall be interpreted as positive integer values which shall be appended to the object identifiers as additional arcs, in the order in which the parameters are listed.

NOTE: These OIDs are appropriate for use in type 5 IEs. In the case of PCM audio (for example), it will be appropriate to have two type 5 IEs, one containing an OID from 4.1.2 of IEC 62379-2 specifying how the audio is encoded and another containing an OID from this clause specifying how the encoded audio is encapsulated for transport across the network.

## 7.2 Packet data

### 7.2.1 General packet data

The object identifiers specified in this subclause may be used to identify a flow that consists of a stream of packets without specifying anything other than the size and frequency of the packets.

```
fixedPackage OBJECT IDENTIFIER ::=
    { mediaEncapsulation fixed(1) }

variablePackage OBJECT IDENTIFIER ::=
    { mediaEncapsulation variable(2) }
```

A stream of fixed-size packets transmitted at regular intervals may be identified by an object identifier consisting of `fixedPackage` with two parameters, of which the first is the number of data octets in each packet and the second is the number of packets per second.

A stream of variable-size packets transmitted at regular intervals may be identified by an object identifier consisting of `variablePackage` with two parameters, of which the first is the maximum number of data octets in each packet and the second is the number of packets per second.

Packets that are not transmitted at regular intervals may be identified by an object identifier as above but omitting the second parameter.

### 7.2.2 IEEE formats.

*[EdNote\_1] AVB (see [http://en.wikipedia.org/wiki/Audio\\_Video\\_Bridging](http://en.wikipedia.org/wiki/Audio_Video_Bridging)) is likely to be widely used in edge networks, so there will be a requirement to convey digital media in data units similar to the IEEE 802 packets used for AVB, which use a format based on IEC 61883 which is fixed at 32 bits per sample with the audio data at the low end (in IEC 62365 it is at the high end) and format information (which in our case would be redundant) along with the AES3 flags in the high byte. The packets have a fairly large header (typically 48 octets plus the Ethernet MAC and PHY encapsulation) and therefore ~~presumably~~ carry data for several sampling periods when the number of channels is small.*

*[EdNote\_2] It may also be useful to define formats for carrying IEEE 802 packets more generally, e.g. for VPNs.*

## 7.3 Pulse-code modulated audio

### 7.3.1 Rationale

The format specified in this subclause 7.3, which is developed from the format specified in IEC 62365, is appropriate for carrying uncompressed audio over networks where the overheads per data unit are small and low latency can be guaranteed; in these circumstances it can support applications that need latencies lower than can be achieved with legacy packet networks.

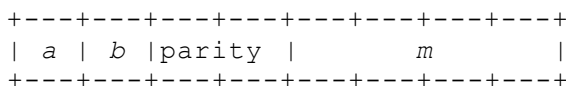
The sequencing octet (see 7.3.2) provides, in only 8 bits, detection of missing or duplicated data units and a sample-accurate timestamp which allows different streams to be aligned.

The subframe format includes two optional four-bit fields. One (see 4.1.3 of IEC 62365) allows IEC 60958 audio to be carried transparently, and the other (see 4.1.4 of IEC 62365) provides data protection appropriate to PCM audio.

*[EdNote] It would be comparatively simple to design an audio interface that supports both the format defined here and AVB.*

### 7.3.2 Sequencing octet

An octet formatted as shown in Figure 8 should accompany the audio sample data for each sampling instant.

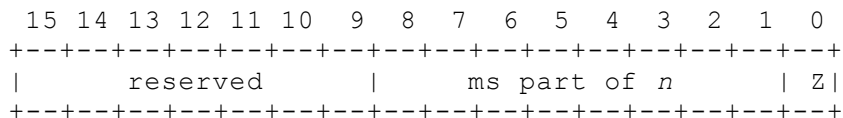


**Figure 8: sequencing octet**

In the following description:

- ◆ *n* is the sample number modulo 3072, in the range 0 to 3071 inclusive, encoded as a 12-bit binary number;
- ◆ *k* is the least significant 6 bits of *n*;
- ◆ *a* is bit *k* of the "long" string specified below;
- ◆ *m* is the least significant 4 bits of *n*; and
- ◆ *b* is bit *m* of the "short" string specified below.

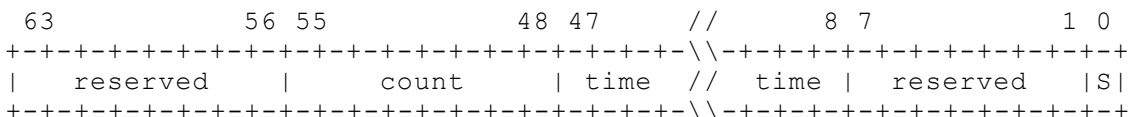
Starting at the most significant end of the sequencing octet, the first bit shall contain *a*, the second shall contain *b*, the third shall be such that there are an odd number of ones in the first three bits, the fourth shall be such that the total number of ones in the octet is odd, and the remaining four shall code the value of *m*.



**Figure 9: "short" bit string value**

The "short" string shall consist of 16 bits numbered from 0 to 15 as shown in Figure 9. Bits 1 to 8 inclusive shall carry the most significant 8 bits of *n* (bit 8 being the most significant bit). Bit 0 shall be coded as 1 if bits 1 to 8 are all zero, 0 otherwise. Bits 9-15 are reserved and should be coded as zero on sending and ignored on reception.

NOTE 1 It follows from the above definitions that *b*=1 and *m*=0 if, and only if, *n*=0.



**Figure 10: "long" bit string value**

The "long" string shall consist of 64 bits numbered from 0 to 63 as shown in Figure 10, and shall be updated at the same time that *n* changes from 3071 to 0. Additionally, bit 0 shall be set to 0 whenever *n* changes to a nonzero multiple of 64. The value when transmission commences shall be the same as if transmission had been continuing for the preceding two seconds.

NOTE 2 Using the description below, the value is not defined until the first update, and not fully defined until a further update at which the number of seconds is incremented. The above provision ensures that the sequencing

octets seen by the recipient of a new multicast are well-defined and are the same as in the case where it has joined a multicast which other units were already receiving.

The value assigned when the string is updated shall be as follows. In the description, the “new” value is the value assigned and the “old” value is the value immediately before the update. The new value for bits 8 to 47 inclusive shall be the number of seconds that have elapsed since the sample-numbering epoch specified in AES53 (bit 47 being the most significant). If the new value in bits 8 to 47 is the same as the old, the new value in bit 0 shall be zero and the new value in bits 48 to 55 inclusive shall be one more (modulo 256) than the old value, interpreted as an integer with bit 55 the most significant bit. If the new value in bits 8 to 47 is not the same as the old, the new value in bit 0 shall be 1 and the new value in bits 48 to 55 inclusive shall be zero. Bits 1-7 and 56-63 shall be reserved and should be coded as zero on sending and ignored on reception.

NOTE 3 Bit 0 indicates “new second”; the first sample of a new second always has  $a=1$  and  $n=0$  and its “sequencing” octet is coded as 0xE0. This value in the sequencing octet does not occur at any other time.

NOTE 4 Assuming a sampling rate of more than 3072 Hz, the new value in bits 8-47 when bit 0 is a 1 is 1 more than the old value. The value in bits 48-55 will not wrap round unless the sampling rate is more than 780kHz.

NOTE 5 AES53 specifies that the number of seconds increments immediately before a sample whose number is a multiple of 3072, so the value in bits 8-47 always corresponds to the number of seconds as specified by AES53. As noted in A.3 of AES53, the number of samples in any one “second” will always be to a multiple of 3072, but the long-term average will be equal to the sampling rate.

### 7.3.3 Subframe format

The subframe format shall be as specified in 4.1 of IEC 62365, except that

- a) if the sequencing octet specified in 7.3.2 is present and the protocol overhead specified in 4.1.4 of IEC 62365 is included in the subframe, then the sequencing bit in the protocol overhead shall be replaced by a flag which is 1 if the subframe is valid, 0 if it should be ignored; and
- b) the subframe is not required to be a whole number of octets.

### 7.3.4 Frame format

A frame shall consist of either

- d) a sequencing octet followed by zero or more subframes, all of the same format, or
- e) one or more subframes, all of the same format.

NOTE 1 Frames with a sequencing octet and no subframes may be used to convey synchronisation information.

For each flow, the number of subframes, the format of the subframes, and whether the sequencing octet is included shall be the same in every frame.

There should be one subframe per audio channel, carrying samples taken at the same instant, with the  $c$ th subframe in a frame carrying the sample for channel number  $c$ .

NOTE 2 In some circumstances there will not be a one-to-one relationship between audio channels and subframes; see 7.3.6.5 and 7.3.6.6.

Option (a) above should be used in preference to option (b).

NOTE 3 Inclusion of sequencing information makes synchronisation of different signals easier and more reliable. The sequencing word specified in 4.1.4.1 of IEC 62365 requires at least 8, and preferably 12, subframes per data unit, whereas on a network that supports smaller data units there may only be one or two. The sequencing octet specified in 7.3.2 carries more information than the IEC 62365 sequencing word, and is independent of the number of subframes per data unit, so provides a convenient way of conveying end-to-end sequencing information. However, for some transports it may be impractical to include it. Where part of a route is unable to convey the sequencing information, the FindRoute messages will be adjusted to show that it is absent or unreliable.

### 7.3.5 Transport

Each data unit shall contain one or more frames. Each frame shall be contained within a single data unit.

NOTE 1 The number of frames is chosen by the application; it should take into account the limits on data unit size signalled by the type 28 IE (see ). The size of a frame is fixed for each flow, so the number of frames in a data unit can be deduced from the size of the data unit.

NOTE 2 Splitting a frame across several data units is specifically excluded. If the number of audio channels is such that a frame would be larger than the maximum data unit size, the channels should be divided into groups with each group being transmitted in a separate flow, and the groups being resynchronised by the receiving application using the information in the sequencing octet. Each flow should be routed separately (i.e. have a different route identifier) in case the total bandwidth required exceeds the capacity of an individual link.

EXAMPLE A bundle of 500 audio channels, sampled at 96kHz with 32 bits per subframe, is to be sent over a 1Gb/s Ethernet network. The frame size is thus 2001 octets (including the sequencing octet), which is more than the standard Ethernet packet size, and the total bandwidth required (excluding overheads) is 1536Mb/s, which is more than can be carried by a single link. Splitting the channels into two groups of 250 reduces the frame size to 1001 octets and allows the network to route the two groups over separate links. However, the bandwidth required for each group is 768Mb/s, so if less than 77% of the capacity of a link can be allocated to synchronous flows smaller groups must be used, and at least three will be needed.

### 7.3.6 Signalling of format

#### 7.3.6.1 General

Encapsulation of pulse-code modulated audio as specified in shall be identified by an object identifier consisting of `pcmAudioEncap` with the five parameters specified in 7.3.6.2 to 7.3.6.6 inclusive.

```
pcmAudioEncap OBJECT IDENTIFIER ::= { mediaEncapsulation pcm(3) }
```

#### 7.3.6.2 Synchronisation information

The first parameter shall be a value of the following type:

```
SynchronisationInfo ::= INTEGER {
    absent          (0),
    sequencingOctet (1),
    iec62365       (2)
} (absent .. iec62365)
```

`absent` shall indicate that no synchronisation information is included in the data stream

`sequencingOctet` shall indicate that the first octet of each frame is a sequencing octet as specified in 7.3.2.

`iec62365` shall indicate that synchronisation information is embedded in the subframes as specified in 4.1.4.1 of IEC 62365.

#### 7.3.6.3 Non-audio fields in subframe

The second parameter shall be a value of the following type:

```
additionalFields ::= INTEGER {
    absent          (0),
    ancillary      (1),
    overhead       (2),
    both           (3)
} (absent .. both)
```

`absent` shall indicate that each subframe consists only of an audio sample word

`ancillary` shall indicate that each subframe consists of an audio sample word followed by four bits containing ancillary data as specified in 4.1.3 of IEC 62365.

`overhead` shall indicate that each subframe consists of an audio sample word followed by four bits containing protocol overhead as specified in 4.1.4 of IEC 62365 as modified by 7.3.3.

`both` shall indicate that each subframe consists of an audio sample word followed by four bits containing ancillary data as specified in 4.1.3 of IEC 62365 and four further bits containing protocol overhead as specified in 4.1.4 of IEC 62365 as modified by 7.3.3.

#### **7.3.6.4 Audio sample word length**

The third parameter shall be the number of bits of audio sample data in each subframe.

NOTE This number may be greater than the bit depth signalled in the audio signal format, in which case IEC 62365 (like IEC 60958-4) requires the value to be padded with zeroes at the least significant end.

#### **7.3.6.5 Subframes per frame**

The fourth parameter shall be the number of subframes in each frame,

NOTE For PCM audio data, this should be the same as the number of audio channels signalled in the audio signal format. See also note 2 to 7.3.6.6.

#### **7.3.6.6 Frame rate**

The fifth parameter shall be the nominal number of frames per second, rounded up to an integer.

NOTE 1 For PCM audio data, this should be the same as the sampling frequency signalled in the audio signal format. Where a PCM audio channel is used to carry non-PCM data, as in IEC 61937, the frame rate may be unrelated to the audio sampling frequency.

NOTE 2 Where the audio source and destination use double sampling frequency mode on IEC 60958 or AES10, a frame rate of half the sampling frequency, and a number of subframes which is twice the number of audio channels, may be used in order to avoid translating the channel status information, although this practice is to be discouraged as it prevents interworking with equipment using conventional encoding at the higher frequency.

### **7.4**

*[EdNote] We need similar subclauses for other traffics, such as (a) the codecs in clause 3 of EBU Tech 3326 and/or generalised RTP payloads (b) video in the formats commonly used in studios (c) maybe MPEG-1/2 Transport (d) any others?*