INTERNATIONAL TELECOMMUNICATION UNION

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

G.8110/Y.1370

(01/2005)

SERIES G: TRANSMISSION SYSTEMS AND MEDIA, DIGITAL SYSTEMS AND NETWORKS

Digital networks – Design objectives for digital networks

MPLS Layer Network Architecture

CAUTION !

PREPUBLISHED RECOMMENDATION

This prepublication is an unedited version of a recently approved Recommendation. It will be replaced by the published version after editing. Therefore, there will be differences between this prepublication and the published version.

NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

Compliance with this Recommendation is voluntary. However, the Recommendation may contain certain mandatory provisions (to ensure e.g. interoperability or applicability) and compliance with the Recommendation is achieved when all of these mandatory provisions are met. The words "shall" or some other obligatory language such as "must" and the negative equivalents are used to express requirements. The use of such words does not suggest that compliance with the Recommendation is required of any party.

INTELLECTUAL PROPERTY RIGHTS

ITU draws attention to the possibility that the practice or implementation of this Recommendation may involve the use of a claimed Intellectual Property Right. ITU takes no position concerning the evidence, validity or applicability of claimed Intellectual Property Rights, whether asserted by ITU members or others outside of the Recommendation development process.

As of the date of approval of this Recommendation, ITU [had/had not] received notice of intellectual property, protected by patents, which may be required to implement this Recommendation. However, implementors are cautioned that this may not represent the latest information and are therefore strongly urged to consult the TSB patent database.

ITU-T Recommendation G.8110/Y.1370

MPLS Layer Network Architecture

Summary

This Recommendation describes the functional architecture of MPLS networks using the modelling methodology described in ITU-T Recommendations G.805 and G.809. The MPLS network functionality is described from a network level viewpoint, taking into account MPLS network layering, definition of characteristic information, client/server associations, networking topology and layer network functionality. The functional architecture of the server networks used by the MPLS network is not within the scope of this Recommendation. Such architectures are described in other ITU-T Recommendations or IETF RFC's.

This Recommendation is based on IETF RFC's 3031, 3032, 3270 and 3443.

CONTENTS

ITU-T Recommendation G.8110/Y.1370

MPLS Layer Network Architecture

1 Scope

This Recommendation describes the functional architecture of MPLS bearer plane networks using the modelling methodology described in Recommendations G.805 and G.809. The MPLS network functionality is described from a network level viewpoint, taking into account an MPLS network layered structure, client characteristic information, client/server associations, networking topology, and layer network functionality providing MPLS signal transmission, multiplexing, supervision, performance and survivability.

The basis for this first version of the Recommendation is the MPLS specification in IETF RFC 3031, RFC 3032, RFC 3270 and RFC 3443.

MPLS OAM as specified in Recommendations Y.1711, Y.1712 and Y.1713 is not described in this version. It will be added along with other MPLS OAM under development in the IETF in the next version of this Recommendation.

2 References

The following ITU-T Recommendations and other references contain provisions, which, through reference in this text, constitute provisions of this Recommendation. At the time of publication, the editions indicated were valid. All Recommendations and other references are subject to revision; users of this Recommendation are therefore encouraged to investigate the possibility of applying the most recent edition of the Recommendations and other references listed below. A list of the currently valid ITU-T Recommendations is regularly published.

The reference to a document within this Recommendation does not give it, as a stand-alone document, the status of a Recommendation.

- ITU-T G.805 (2001), Generic functional architecture of transport networks.
- ITU-T G.809 (2003), Functional architecture of connectionless layer networks.
- ITU-T Y.1711 (2004), Operation and maintenance mechanism for MPLS networks.
- ITU-T Y.1712 (2004), OAM functionality for ATM-MPLS interworking.
- ITU-T Y.1713 (2004), Misbranching detection for MPLS networks.
- RFC 3031 (2001), Multiprotocol label switching architecture.
- RFC 3032 (2001), MPLS label stack encoding.
- RFC 3270 (2002), Multi-Protocol Label Switching (MPLS) support of Differentiated Services.
- RFC 3443 (2003), Time To Live (TTL) processing in Multi-Protocol Label Switching (MPLS) networks.

3 Definitions

This Recommendation uses terms defined in G.805:

3.1 access point

3.2 adapted information

3.3 characteristic information

3.4 client/server relationship

3.5 connection

3.6 connection point

3.7 layer network

3.8 link

3.9 link connection

3.10 matrix

3.11 network

3.12 network connection

3.13 port

3.14 reference point

3.15 subnetwork

3.16 subnetwork connection

3.17 termination connection point

3.18 trail

3.19 trail termination

3.20 transport

3.21 transport entity

3.22 transport processing function

3.23 unidirectional connection

3.24 unidirectional trail

This Recommendation uses terms defined in G.809:

3.25 access point

3.26 adapted information

3.27 characteristic information

3.28 client/server relationship

3.29 connectionless trail

3.30 flow

3.31 flow domain

3.32 flow domain flow

3.33 flow point

3.34 flow point pool

3.35 flow termination

3.36 flow termination sink

3.37 flow termination source

3.38 layer network

3.39 link flow

3.40 network

3.41 network flow

3.42 port

3.43 reference point

3.44 traffic unit

3.45 transport

3.46 transport entity

3.47 transport processing function

3.48 termination flow point

This Recommendation uses the following terms defined in RFC 3031:

3.49 Forwarding Equivalence Class

3.50 label

3.51 label merging

3.52 labelled packet

3.53 label stack

3.54 label swap

3.55 label swapping

3.56 label switched hop

3.57 label switched path

3.58 MPLS label stack

This Recommendation uses the following terms defined in RFC 3032:

3.59 Bottom of Stack

3.60 Time To Live

3.61 Experimental Use

3.62 Label value

3.63 IPv4 Explicit Null Label

3.64 Router Alert Label

3.65 IPv6 Explicit Null Label

3.66 Implicit Null

This Recommendation uses the following terms defined in RFC 3270:

3.67 Per Hop Behaviour

3.68 EXP inferred PHB scheduling class LSP

3.69 Label inferred PHB scheduling class LSP

This Recommendation defines the following

3.70 Z layer: A sublayer for modelling Penultimate Hop Popping. A Z sublayer is a flow based sublayer. The Z layer flow domain is a matrix level flow domain. A Z network flow is always of the form link flow-flow domain flow-link flow.

4 Abbreviations

This Recommendation uses the following abbreviations:

AG	Access Group
AI	Adapted Information
AP	Access Point
BA	Behaviour Aggregate
CI	Characteristic Information
CP	Connection Point
DSCP	Diff-Serv Code Point
E-LSP	EXP-Inferred-PSC LSP
EXP	Experimental Use
FDF	Flow Domain Flow
FEC	Forwarding Equivalence Class
FP	Flow Point
FPP	Flow Point Pool
FT	Flow Termination
FTP	Flow Termination Point
LF	Link Flow
LSP	Label Switched Path
L-LSP	Label-Only-Inferred PSC LSP
MPLS	Multi-Protocol Label Switching
NF	Network Flow
OA	Ordered Aggregate
PHB	Per Hop Behaviour
PHP	Penultimate Hop Pop
PSC	PHB Scheduling Class
S	Bottom of Stack
TCP	Termination Connection Point
TFP	Termination Flow Point
TFPP	Termination Flow Point Pool

TTL Time-To-Live

5 Conventions

The diagrammatic convention for connection-oriented layer networks described in this Recommendation is that of Recommendation G.805.

The diagrammatic convention for connectionless layer networks described in this Recommendation is that of Recommendation G.809 with the exception of the colouring of atomic function and port symbols.

All transport entities within this Recommendation are unidirectional.

6 Transport functional architecture of MPLS networks

6.1 General

The functional architecture of MPLS transport networks is described using the modelling techniques defined in Recommendations G.805 and G.809. MPLS networks may exhibit both connection-oriented and connectionless behaviour, the functional architecture based on the flow model for G.809 and the connection model for G.805 collectively is sufficient to model the MPLS architecture. The specific aspects regarding the characteristic information, client/server associations, the topology and partitioning of MPLS transport networks are provided in this Recommendation. This Recommendation uses the terminology, functional architecture and diagrammatic conventions defined in Recommendations G.805 and G.809.

The description of the MPLS architecture is organised as follows:

- MPLS functional architecture based on MPLS specific headers (MPLS shim header)
  - The functional architecture of MPLS networks that support flow properties, e.g. multipoint-to-point flows in the form of multipoint-to-point LSP trees in a single layer network, are described in section 7 using the flow model of G.809.
  - The functional architecture of MPLS networks that exhibit connection oriented behaviour are described in section 8 using the connection model of G.805.
  - MPLS hierarchies may be described using a G.805 model or a G.809 model. In addition an MPLS hierarchy may require both a G.805 and a G.809 based description for different levels in the hierarchy. MPLS hierarchies are described in section 9
- MPLS functional architecture based on MPLS label encapsulation within the header of another technology.
  - This is not considered any further in this version of the Recommendation.

The use of multicast is left for further study.

6.2 MPLS Network layered structure

One layer network is defined in the MPLS transport network architecture:

- MPLS Layer Network.

The MPLS layer network is a path layer network. The MPLS layer network characteristic information can be transported through MPLS links supported by trails in other path layer networks (e.g. Ethernet MAC layer network, SDH VC-n, OTH ODUk).

6.2.1 MPLS Adapted Information

The MPLS layer network adapted information is a (non-) continuous flow of MPLS_AI traffic units. The MPLS_AI traffic unit consists of an MPLS_AI header containing the S field of the MPLS shim header and an MPLS payload field. The MPLS payload field carries adapted client information.

6.2.2 MPLS Characteristic Information

The MPLS layer network characteristic information is a (non-) continuous flow of MPLS_CI traffic units.

The MPLS_CI traffic unit consists of an MPLS_AI traffic unit extended with an MPLS_CI header containing the TTL field of the MPLS shim header.

Note - The MPLS 20-bit Label and 3-bit EXP are considered part of the MPLS header (RFC 3031). In the layer network model, both are associated with the MPLS link, not with the MPLS characteristic information.

When the client layer network of MPLS is itself MPLS the payload information includes the MPLS_CI traffic unit extended with a 3-bit EXP field and a 20-bit Label from the MPLS shim header. In this case the payload is equivalent to a labelled packet in RFC 3031. The information structures are shown in Figure 1 along with the relationship to label stack entries.

The MPLS_CI traffic unit is transported over an MPLS Link within a link specific frame or packet, of which the generic format is depicted in Figure 2.

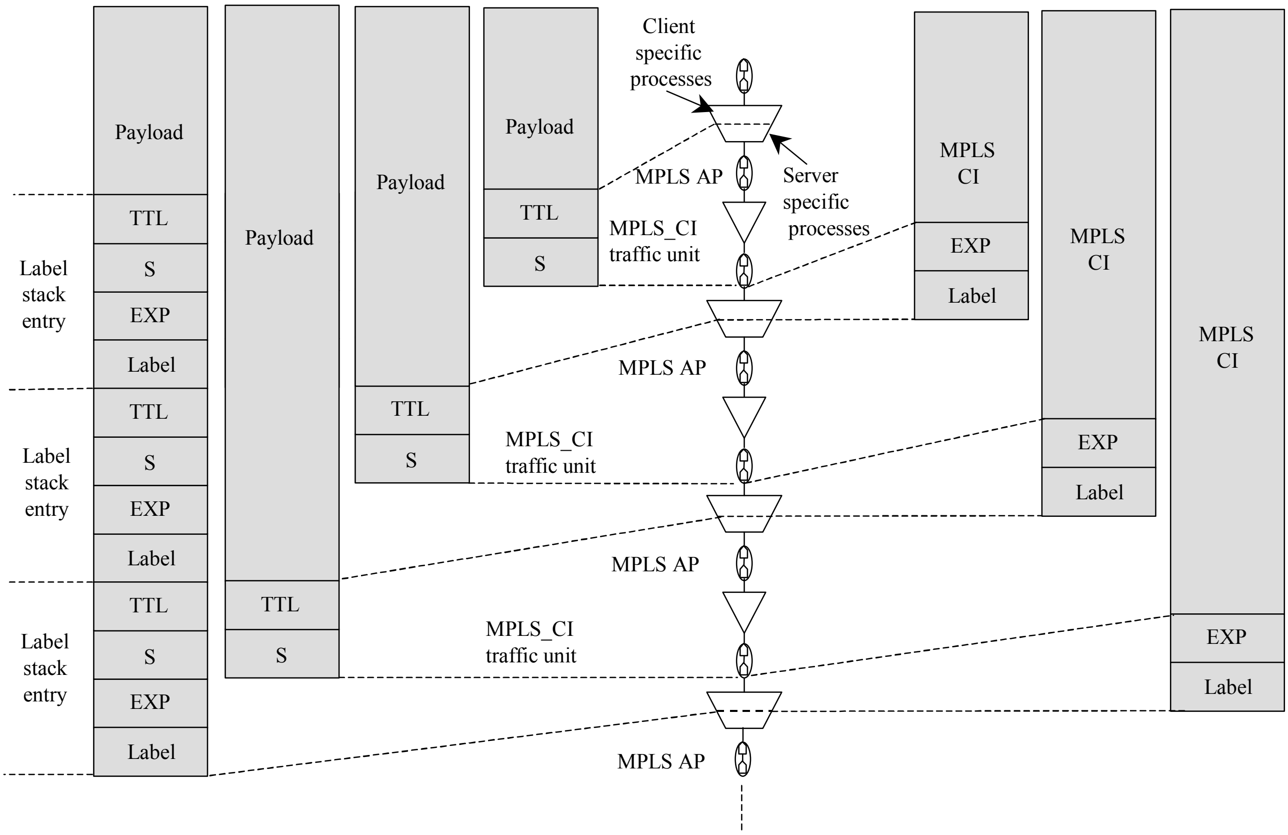
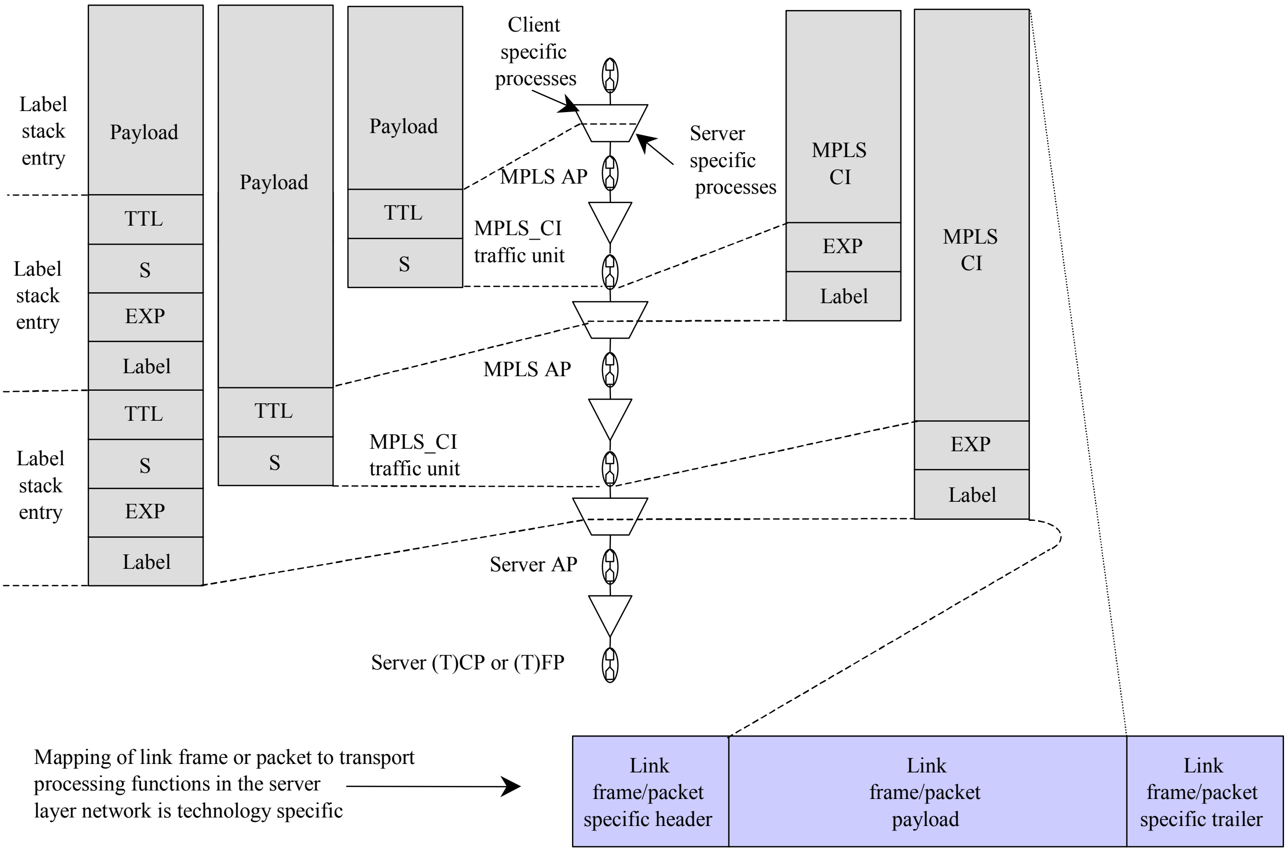


Figure 1/G.8110/Y.1370 – Example of recursive behaviour of MPLS Characteristic Information (MPLS_CI) traffic unit in an MPLS label stack


Label stack entry
Payload
TTL
S
EXP
Label
Label stack entry
TTL
S
EXP
Label
Label stack entry
Payload
TTL
S
Payload
TTL
S
Client specific processes
Server specific processes
MPLS AP
MPLS_CI traffic unit
MPLS AP
MPLS_CI traffic unit
Server AP
Server (T)CP or (T)FP
MPLS CI
EXP
Label
MPLS CI
EXP
Label
Mapping of link frame or packet to transport processing functions in the server layer network is technology specific
Link frame/packet specific header
Link frame/packet payload
Link frame/packet specific trailer

(Note that the presence, or otherwise, of a link frame/packet specific trailer is technology specific)

Figure 2/G.8110/Y.1370 – MPLS Characteristic Information (MPLS_CI) traffic unit format and its relationship to other information entities including the relationship with encapsulating link frames/packets.

7 MPLS shim header functional architecture description based on G.809

7.1 MPLS Layer Network

The MPLS layer network provides the transport of adapted information through an MPLS connectionless trail between MPLS access points. The MPLS layer network characteristic information is transported over an MPLS network flow between MPLS termination flow points.

The MPLS layer network contains the following transport processing functions, transport entities and topological components (see Figure 3):

- MPLS connectionless trail
- MPLS flow termination source (MPLS_FT_So)
- MPLS flow termination sink (MPLS_FT_Sk)
- MPLS network flow (NF)

- MPLS link flow (LF)
- MPLS flow domain flow (FDF)
- MPLS flow domain (FD)
- MPLS link

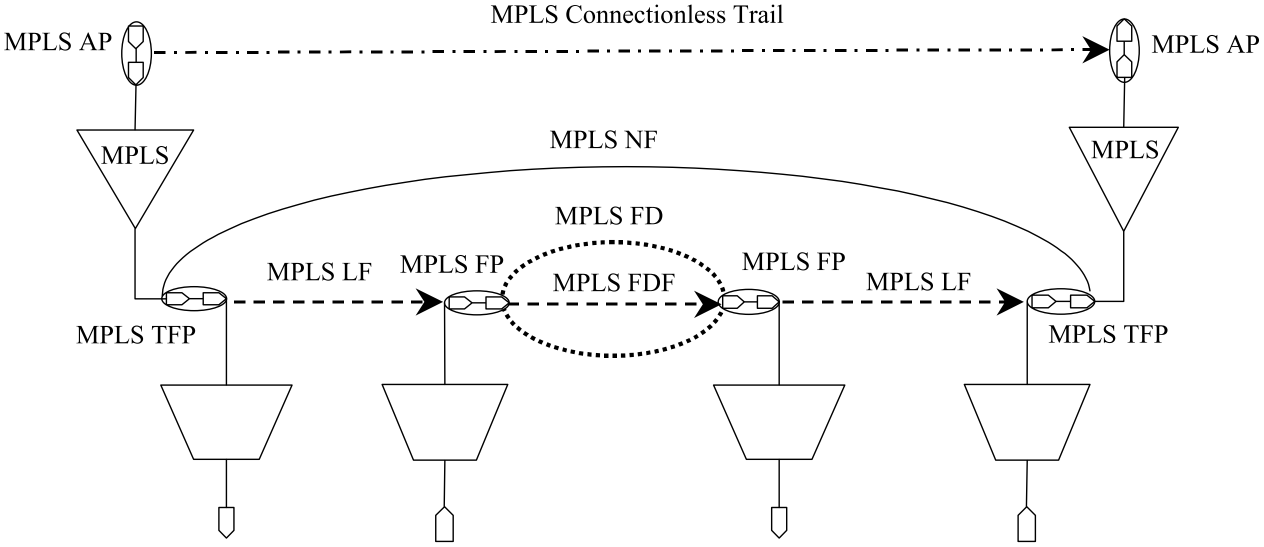
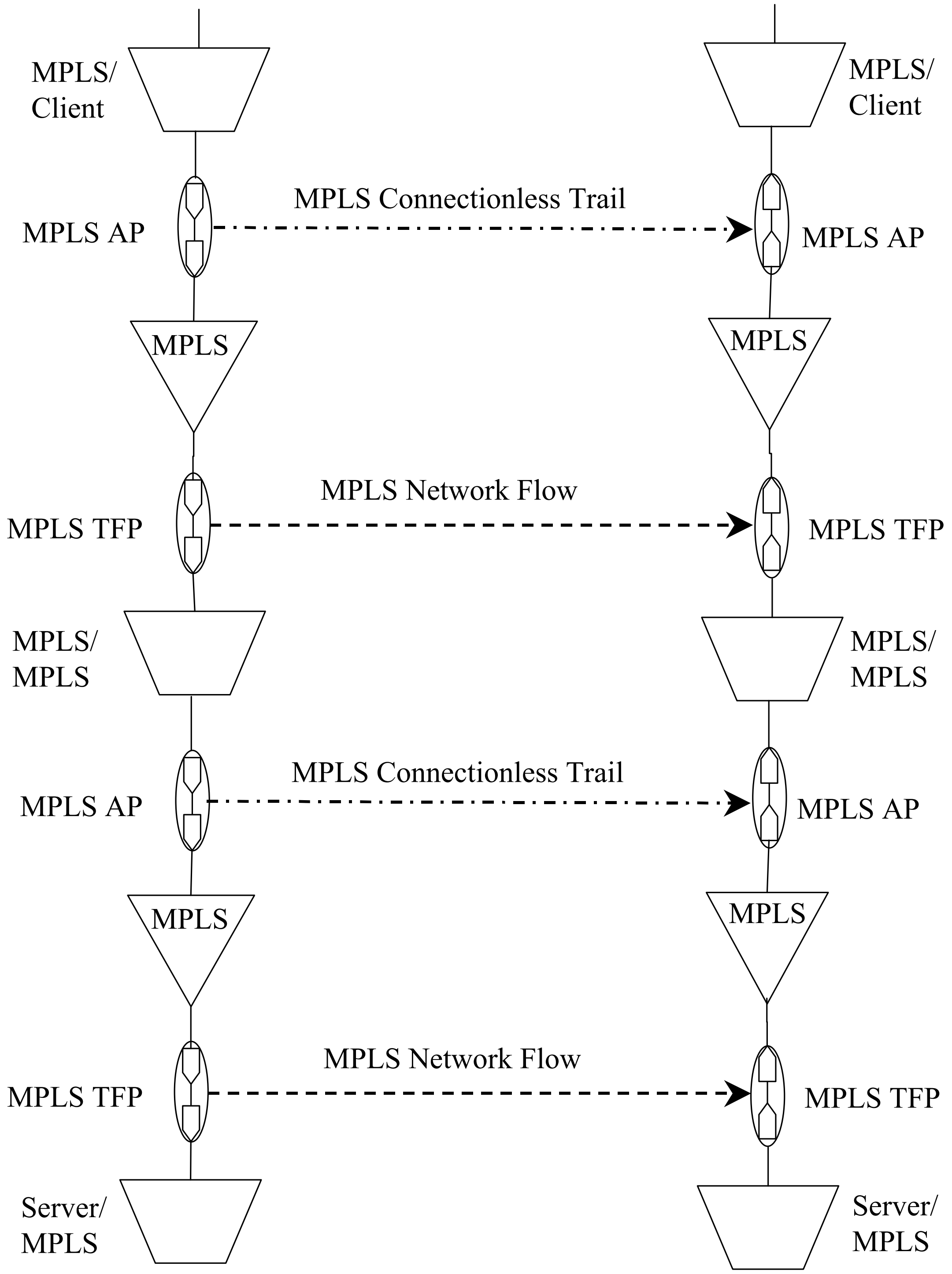


Figure 3/G.8110/Y.1370 – MPLS layer network example

The MPLS layer network may be employed recursively to describe an MPLS hierarchy, implemented as a label stack. This is described by the use of sub-layering. A transport network based on MPLS can be decomposed into a number of independent transport sublayer networks with a client/server association between adjacent sublayer networks. An example of MPLS sublayers and their structure and the adaptation functions is shown in Figure 4. This convention is used in this Recommendation.

The label stack is related to the MPLS sublayers in such a way that the bottom of the stack is associated with the MPLS sublayer at the top of the diagram (where the client is not MPLS), whilst the top of the stack is associated with the MPLS sublayer at the bottom of the diagram.

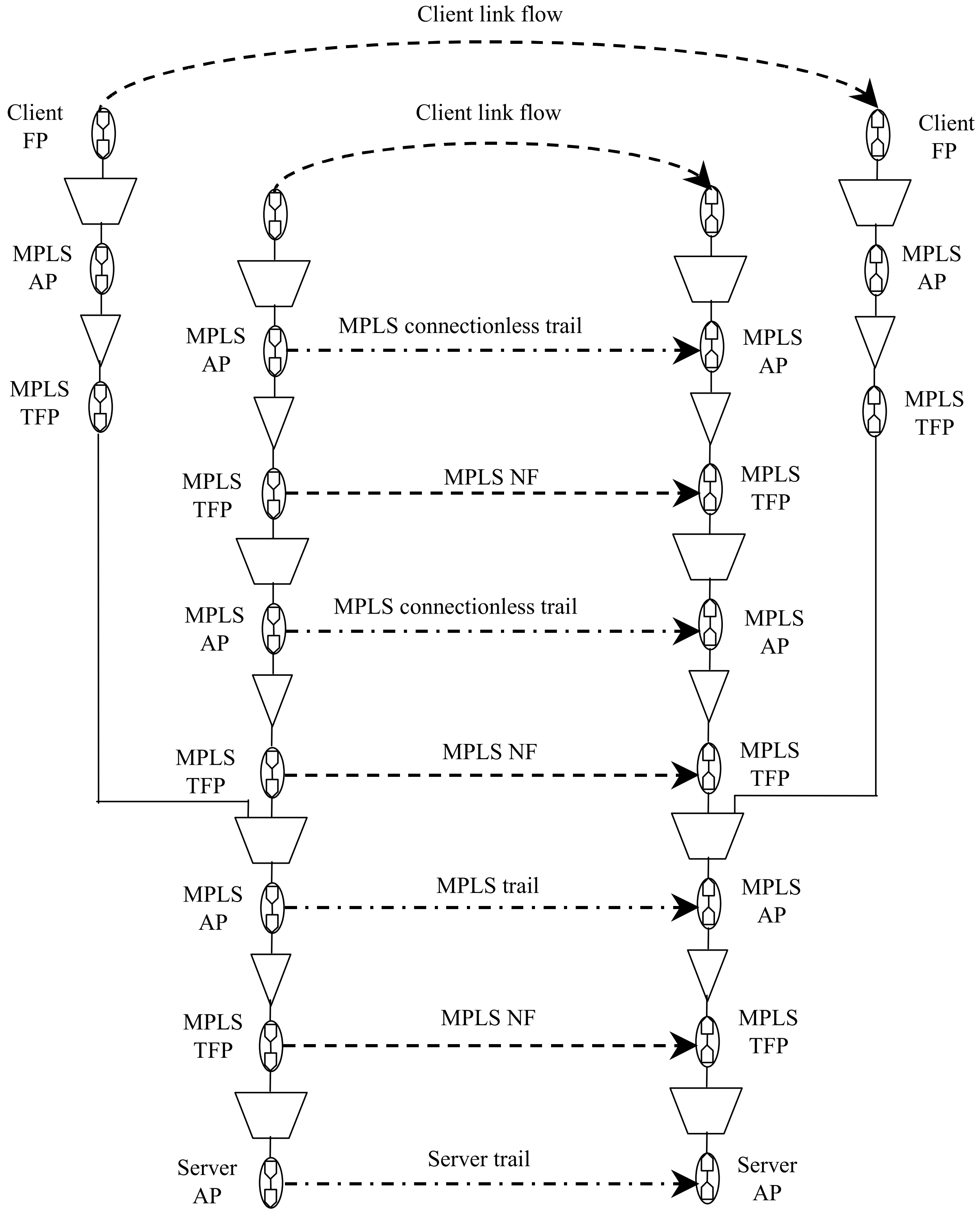


Two levels are shown in this example. Additional sublayers can be added as required. The bottom of the stack is at the top.

Figure 4/G.8110/Y.1370 – Example of MPLS hierarchy illustrated using sub-layering

Two levels are shown in this example. Additional sublayers can be added as required. The bottom of the stack is at the top.

MPLS allows for the creation of an arbitrary depth of sublayers, or label stacks. An example is shown in Figure 5.



The outermost client flow is supported by an MPLS hierarchy with a stack depth of two, whilst the inner client flow is supported by a MPLS stack of depth three. As such, other than the bottom of the stack an MPLS sublayer has no designated depth.

Figure 5/G.8110/Y.1370 - Example of MPLS stack depths.

7.1.1 MPLS Topological Components

The MPLS topological components are:

- MPLS layer network
- MPLS Flow Domain
- MPLS Link
- MPLS Access Group

The MPLS layer network is partitioned into a number of MPLS Flow Domains interconnected by MPLS links.

7.1.1.1 MPLS Layer Network

The MPLS layer network is defined by the complete set of MPLS access groups that may be associated for the purpose of transferring information. The information transferred is characteristic of the MPLS layer network and is termed MPLS characteristic information. The associations of the MPLS flow terminations (that form a connectionless trail) in the MPLS layer network are defined on a per traffic unit basis, which is the MPLS_CI traffic unit. The topology of the MPLS layer network is described by MPLS access groups, MPLS flow domains and the MPLS flow point pool links between them. The structures within the MPLS layer network and its server and client layer networks are described by the components below.

7.1.1.2 MPLS Flow Domain

An MPLS flow domain is defined by the set of MPLS flow points that are available for the purpose of transferring information. MPLS_CI traffic unit transfers, across the MPLS flow domain, that correspond to a particular association between ingress and egress MPLS flow points, need not be present at all times. In general, MPLS flow domains may be partitioned into smaller flow domains interconnected by MPLS flow point pool links. The matrix is a special case of an MPLS flow domain that cannot be further partitioned. Unless otherwise explicitly stated the description of flow domains in this Recommendation is at the matrix level.

7.1.1.3 MPLS Flow Point Pool Link

An MPLS flow point pool link consists of a subset of the MPLS flow points at the edge of one MPLS flow domain or MPLS access group that are associated with a corresponding subset of MPLS flow points at the edge of another MPLS flow domain or MPLS access group for the purpose of transferring MPLS characteristic information. The MPLS flow point pool link (FPP link) represents the topological relationship and available capacity between a pair of MPLS flow domains, or an MPLS flow domain and an MPLS access group, or a pair of MPLS access groups.

Multiple MPLS flow point pool links may exist between any given MPLS flow domain and MPLS access group or pair of MPLS flow domains or MPLS access groups. MPLS flow point pool links are established at the timescale of the MPLS server layer network.

7.1.1.4 MPLS Access Group

An MPLS access group is a group of co-located MPLS flow termination functions that are connected to the same MPLS flow domain or MPLS flow point pool link.

7.1.2 MPLS Transport Entities

The MPLS transport entities are:

- MPLS Link Flow

- MPLS Flow Domain Flow
- MPLS Network Flow
- MPLS connectionless trail

7.1.3 MPLS Transport Processing Functions

The MPLS transport processing functions are:

- MPLS flow termination function
- MPLS to client layer network adaptation functions

7.1.3.1 MPLS Flow Termination

The MPLS_FT_So function inserts the 8-bit TTL field in the MPLS_CI traffic unit. The MPLS_CI traffic unit is output via the MPLS TFP.

The MPLS_FT_Sk terminates and processes the 8-bit TTL field as described in section 13.2.

Note that a flow termination is associated with one LSP instance.

7.1.3.2 MPLS to client layer network adaptation functions

The MPLS/client adaptation functions are described in section 10.

7.1.4 MPLS Reference Points

The MPLS reference points (Figure 3) are:

- MPLS Access Point (AP)
- MPLS Termination Flow Point (TFP)
- MPLS Flow Point (FP)
- MPLS Flow Point Pool (FPP)
- MPLS Termination Flow Point Pool (TFPP)

7.1.4.1 MPLS Access Point

An MPLS Access Point (MPLS AP) represents the binding between an MPLS Flow Termination function and one or more MPLS/client, or MPLS/MPLS adaptation functions.

7.1.4.2 MPLS Termination Flow Point

An MPLS Termination Flow Point, (MPLS TFP), connects an MPLS Flow Termination (MPLS FT) function with an MPLS Link.

7.1.4.3 MPLS Flow Point

An MPLS Link connects to an MPLS Flow Domain or another MPLS Link via an MPLS Flow Point. This flow point is provided through the Server/MPLS, or MPLS/MPLS adaptation function.

7.1.4.4 MPLS Flow Point Pool

A group of MPLS Flow Points is referred to as MPLS Flow Point Pool (FPP). An FPP has the same properties as its flow points.

7.1.4.5 MPLS Termination Flow Point Pool

A group of MPLS Termination Flow Points is referred to as MPLS Termination Flow Point Pool (TFPP). A TFPP has the same properties as its termination flow points.

7.2 MPLS Layer Network Partitioning

MPLS layer network partitioning can be represented in a single layer network by means of a geometric translation of the MPLS hierarchy. This is illustrated in Figure 6. Figure 6(a) illustrates an example of an MPLS hierarchy –only the source functions are shown. The MPLS TFP's can be shown at the same horizontal level in a diagram by means of a simple translation as shown in 6(b). This can be extended to represent a label stack of arbitrary depth. The result of this translation is that the MPLS TFP's, and therefore FP's, link flows, flow domain flows, flow domains and links can all be shown in a single layer network. An example is shown in Figure 7.

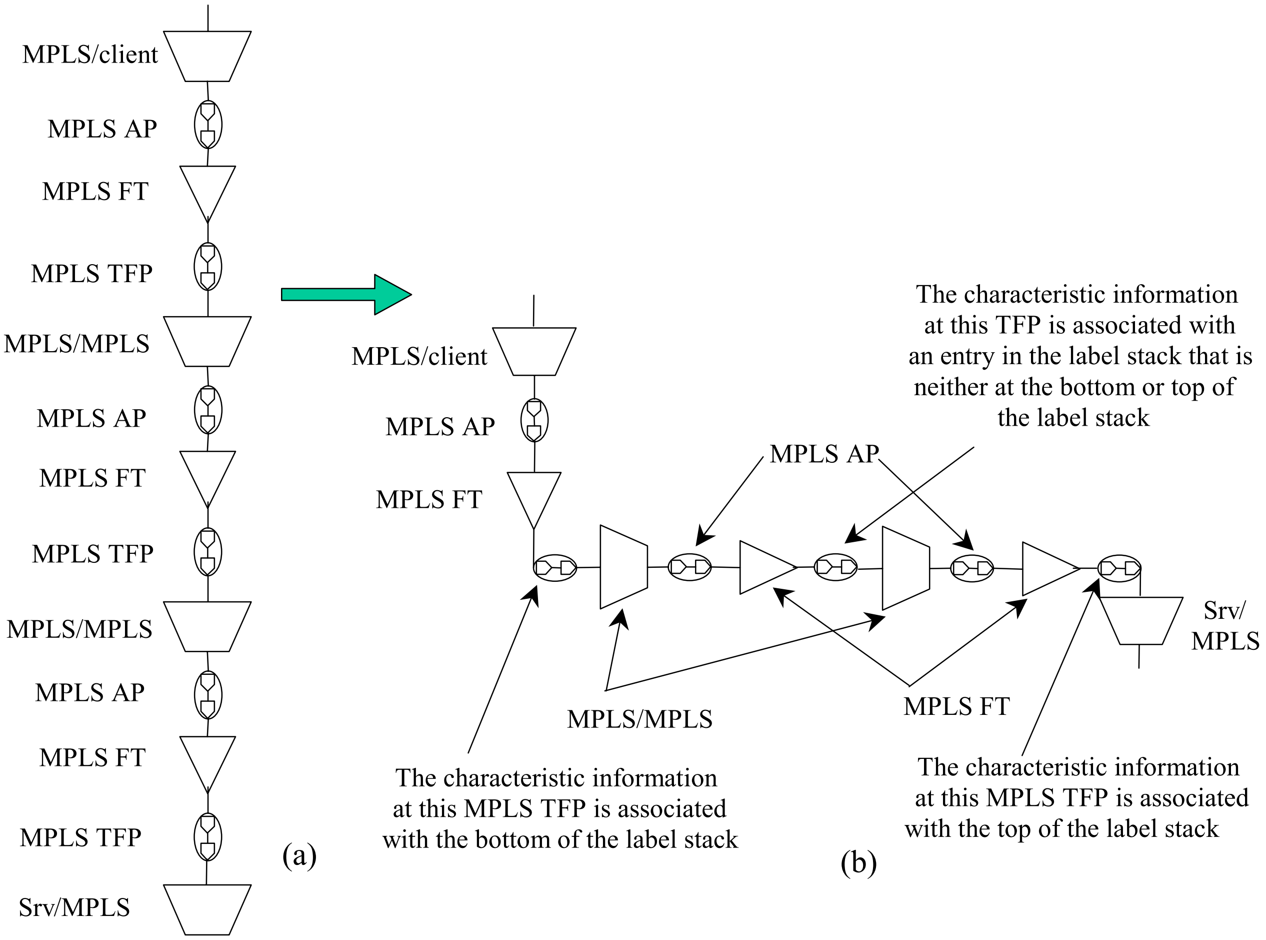

MPLS/client
MPLS AP
MPLS FT
MPLS TFP
MPLS/MPLS
MPLS AP
MPLS FT
MPLS TFP
MPLS/MPLS
MPLS AP
MPLS FT
MPLS TFP
Srv/MPLS
(a)
MPLS/client
MPLS AP
MPLS FT
The characteristic information at this TFP is associated with an entry in the label stack that is neither at the bottom or top of the label stack
MPLS AP
Srv/ MPLS
MPLS/MPLS
MPLS FT
The characteristic information at this MPLS TFP is associated with the bottom of the label stack
The characteristic information at this MPLS TFP is associated with the top of the label stack
(b)

Figure 6/G.8110/Y.1370 - Translation between sublayer and layer network viewpoints (only source direction is shown)

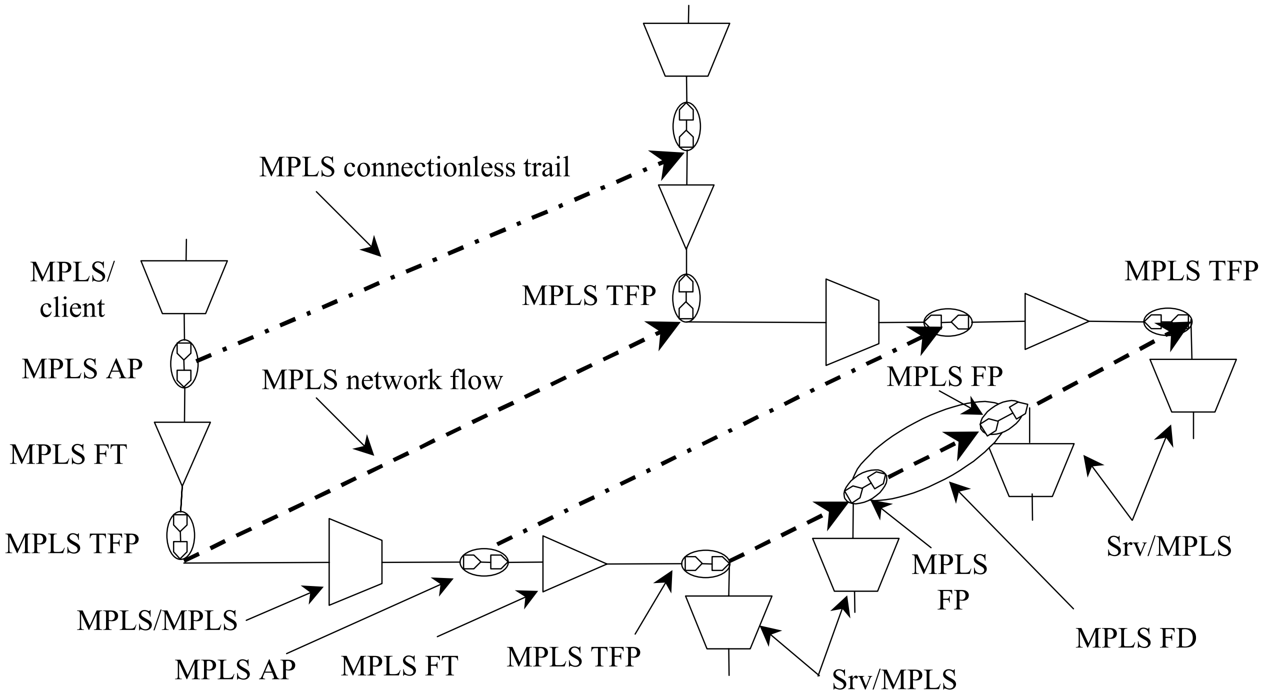


Figure 7/G.8110/Y.1370 - Representing MPLS network flows and connectionless trails in a single MPLS layer network

NOTE - The convention in this Recommendation is that sublayering is used to illustrate the relationships between transport entities (hierarchy) whilst the single layer network viewpoint is used to illustrate topology and partitioning. The transformation whereby all of the MPLS sublayers can be shown in a single layer network is only possible because all of the MPLS sublayers have the same characteristic information and belong to the same address space. Flow points (or connection points) associated with different types of characteristic information are always shown in different layer networks.

The relationship between an MPLS layer network and MPLS sublayers is as follows:

- An MPLS layer network may support hierarchy such that an MPLS layer network contains one or more sublayers. The sublayers associated with a layer network must all belong to the same address space. The context of the address space is that of the addresses of the MPLS access points.
- Where two MPLS sublayers have different address spaces then each sublayer is associated with a different MPLS layer network.

Within any MPLS flow point pool link there may be flow points from different levels of the label stack depending on the structure of the label stacks supporting the link ends. An example is shown in Figure 8 along with the resultant layer network topology.

The network topology of an MPLS layer network can be partitioned according to the partitioning rules described in G.809.

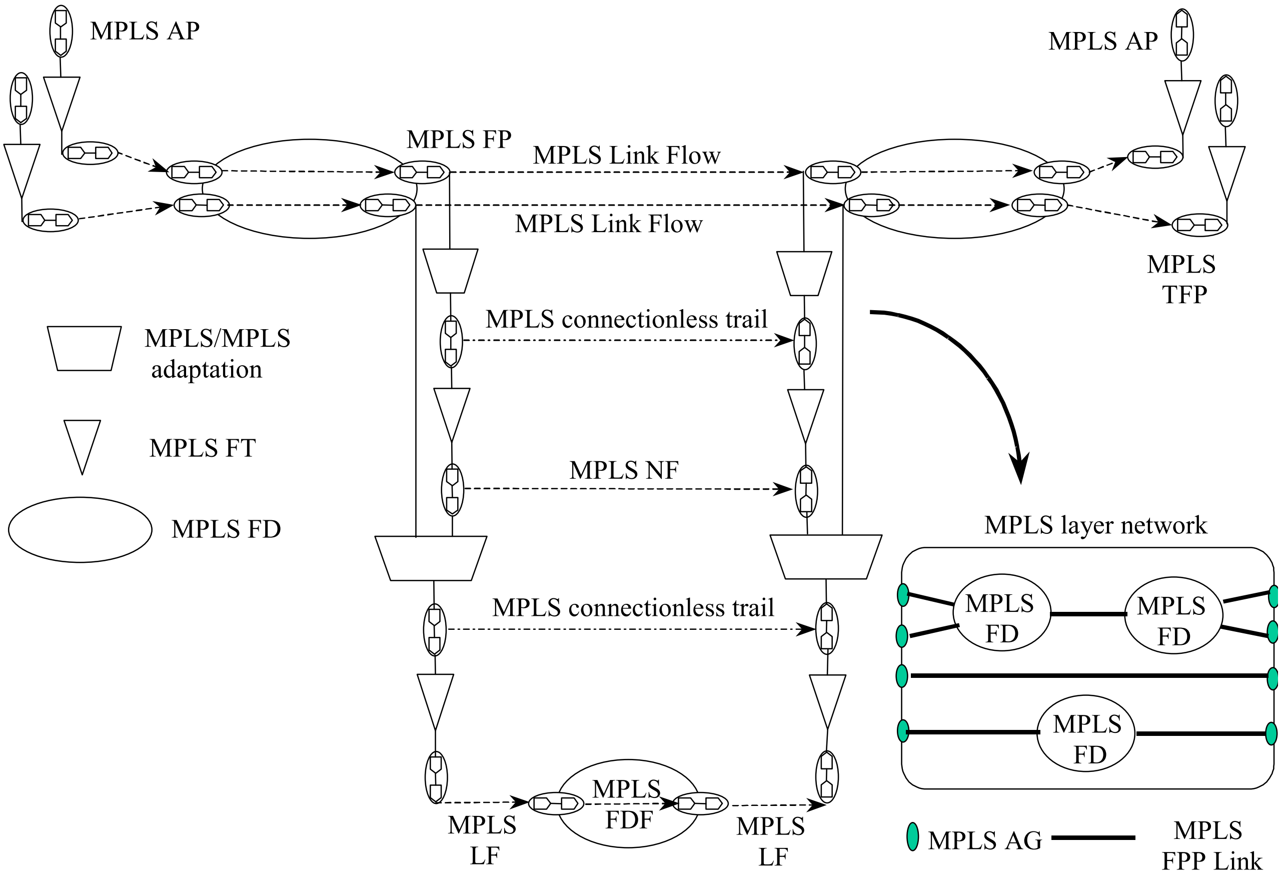


Figure 8/G.8110/Y.1370 - Flow points in an MPLS link and their relationship within the MPLS hierarchy

7.3 MPLS Label Behaviour

7.3.1 Reserved Labels

Label values 0 to 15 are reserved. Four of the reserved label values are defined in RFC 3032 and are described in Table 1. Note that label value 3 is only sent in the control plane and never in the bearer plane. MPLS labelled packets with label values of 0,1 and 2 are directed by an adaptation sink function toward an FTP.

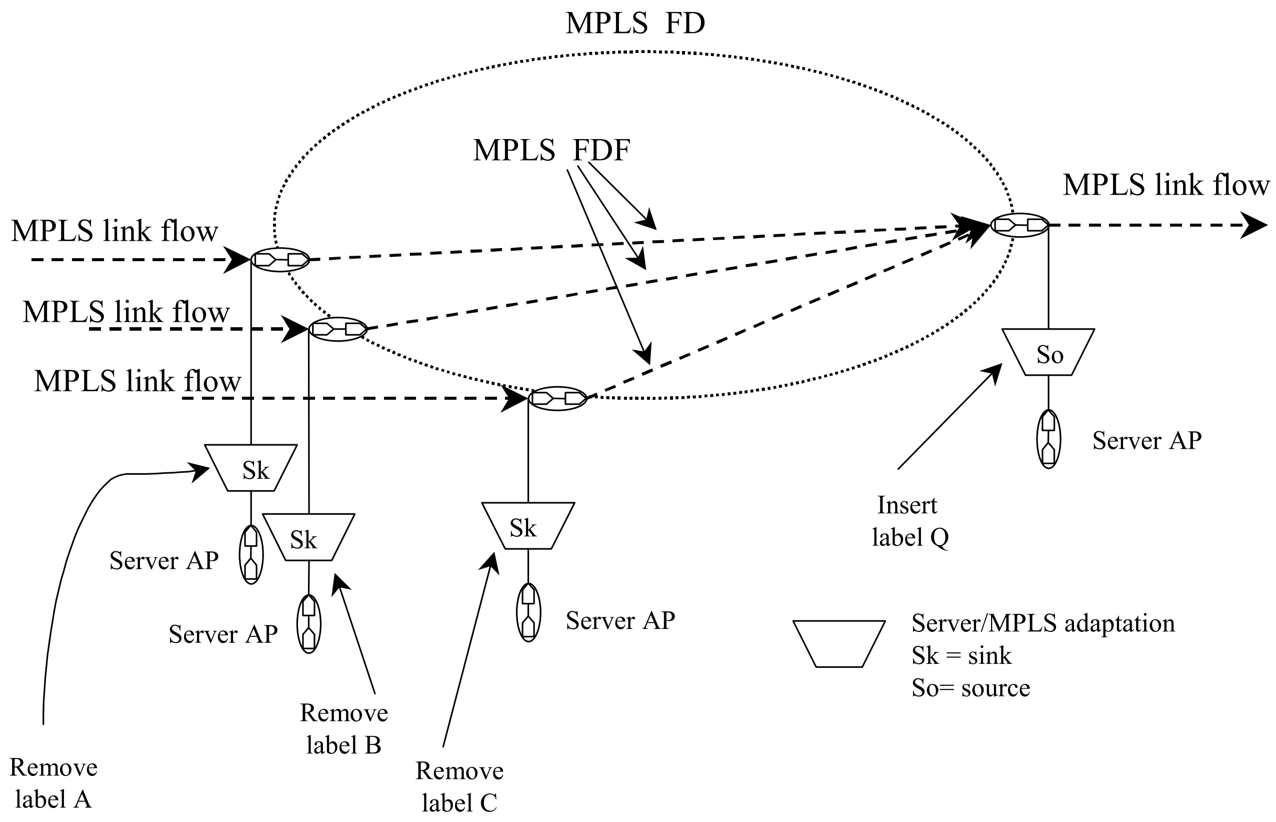
The functional models for each of the reserved labels is described in Annex B.

Table 1/G.8110/Y.1370 - Reserved Label Values

Label value	Label Name	Description
0	IPv4 Explicit Null Label	This indicates that the label stack must be popped and that forwarding of the payload, an IPv4 packet, must be based on the IPv4 header. This label value is only valid at the bottom of the stack.
1	Router Alert Label	When received the packet is processed locally. Forwarding is determined by the client header, but a Router Alert Label should be pushed on at egress. This value is legal anywhere in the label stack except at the bottom.
2	Ipv6 Explicit Null Label	This indicates that the label stack must be popped and that forwarding of the payload, an IPv6 packet, must be based on the IPv6 header. This label value is only valid at the bottom of the stack.
3	Implicit Null Label	In the control plane the last hop of the LSP advertises a label value of 3 to indicate that the MPLS header is to be removed and the forwarding is based on the MPLS payload. The Implicit Null value never appears in an MPLS header
4-13		Reserved
14	OAM Alert Label	Label for MPLS OAM packets as described in Y.1711. It is not used in the G.809 model.
15		Reserved

7.3.2 Label merge

As already described in the functional model the label field is associated with the MPLS link and not with the MPLS characteristic information. Consequently different label field values may be used on different links. This is also referred to as label swapping. Merging occurs when MPLS_CI traffic units arriving at an MPLS flow domain on different MPLS links are directed toward a single MPLS flow point on an outgoing MPLS link. All traffic units traversing this flow point are assigned the same outgoing label by the associated server/MPLS adaptation source. This is illustrated in Figure 9. The multipoint-to-point flow that merging creates is also referred to as a multipoint-to-point LSP tree.



(Note that in this example the server layers are indicated as not being MPLS. It is also possible to have MPLS server layers in which case the adaptation functions would be MPLS/MPLS and the access points would be MPLS AP's.)

Figure 9/G.8110/Y.1370 - Merging of MPLS link flows

Merging in the MPLS layer network removes the ability to distinguish between traffic sources in that layer network. Deconstruction of the merged flow is only achieved by demultiplexing to a client (sub) layer network (that is a client of the (sub) layer that created the merge. This requires that the client layer either:

- Is connectionless in the sense that each unit of characteristic information contains both a source and destination address. In this case resolution of the source and sink is simple.
- Provides point-to-point connectivity between each source and sink. This is normally effected in an MPLS client layer network by means of point-to-point MPLS link flows above the MPLS sublayer that has created the merging. This is illustrated in Figure 10.

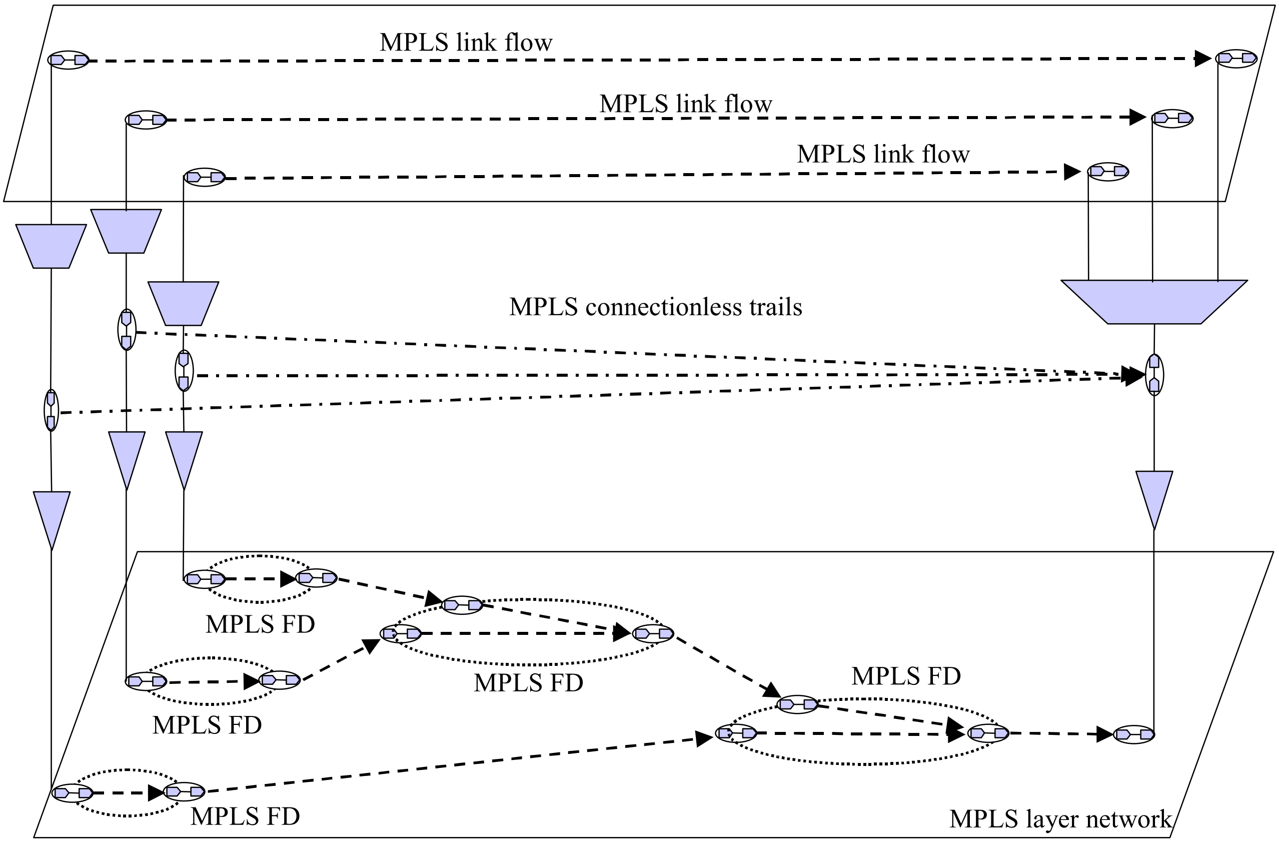
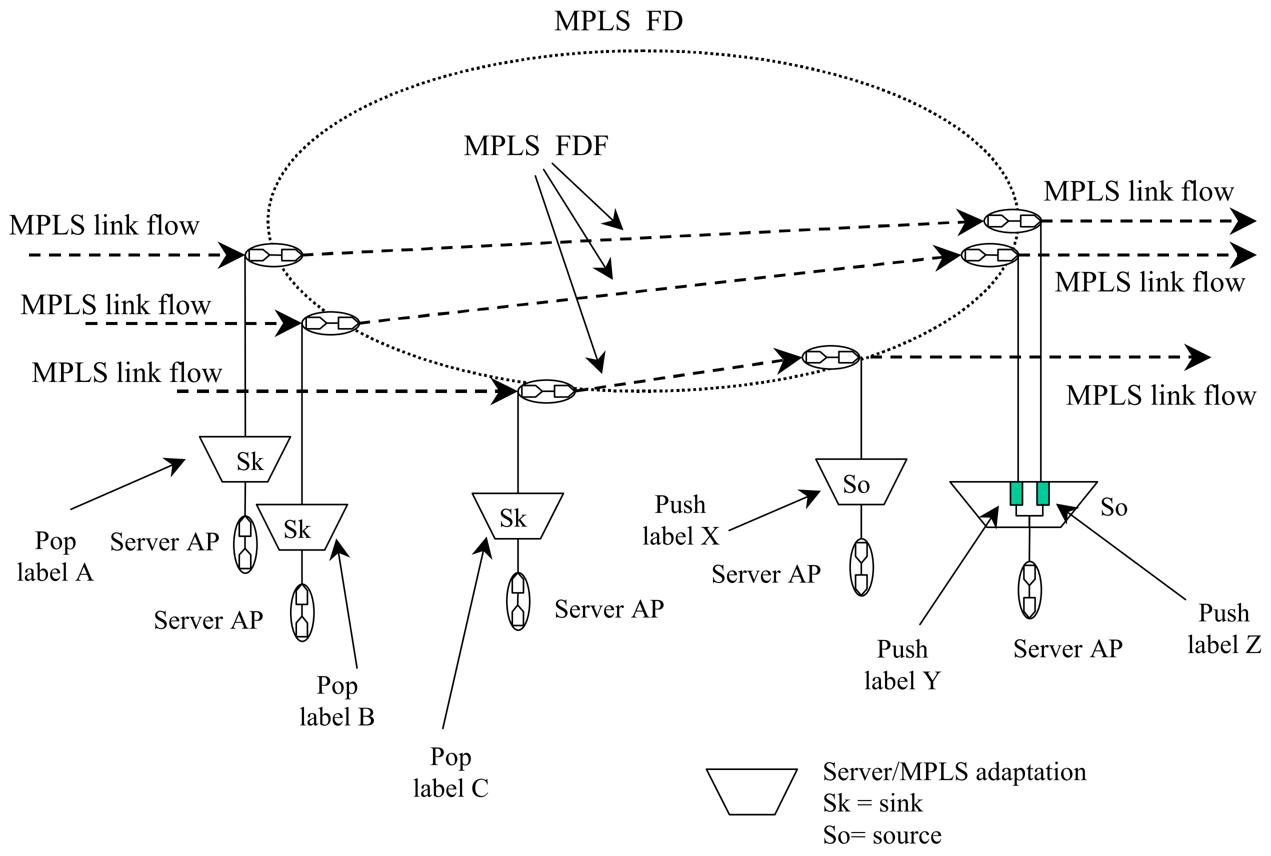


Figure 10/G.8110/Y.1370 - Server layer network merging in support of multiple client layer network point-to-point flows

Where an MPLS flow domain does not support merging then any traffic units that arrive at an MPLS flow domain at different ingress flow points must also egress the flow domain via different flow points. The egress flow points can be in different MPLS links or in the same link. This is shown in Figure 11.



(Note that in this example the server layers are indicated as not being MPLS. It is also possible to have MPLS server layers in which case the adaptation functions would be MPLS/MPLS and the access points would be MPLS AP's.)

Figure 11/G.8110/Y.1370 - MPLS flow domain that does not support merging

7.3.3 Global Label Space

When any incoming traffic units with the same label, regardless of the link on which they arrive at an MPLS matrix, are forwarded in the same manner, with respect to an outgoing flow point, (or flow points where ECMP is present), the label is said to be from the global label space.

In Figure 9 for, example, any MPLS traffic units arriving with the same label, regardless of which link they arrive on, are forwarded in the same manner, in this case to a single output flow point. The label Q inserted at the egress adaptation function may or may not have the same value as that of the incoming label.

The global label space is also known as per-platform label space. The word “scope” can be substituted for the word “space” such that the terms global label space and global label scope are interchangeable.

7.3.4 Interface Label Space

A per-interface label space is a label space where an MPLS label value is only unique for a flow point within a link. In Figure 11 for example labels A, B and C values are set independently and can have the same or different values. Labels X, Y and Z have the property that they can be set to any valid value with the only restriction being that Y does not equal Z.

7.3.5 Support for Multiple Label Spaces

The labels present on a link may be taken from either global or per-interface label spaces. An individual label can only belong to one label space with respect to the link. There can be multiple instances of global or per interface label spaces on a link.

7.4 Penultimate Hop Popping (PHP)

Penultimate hop popping (PHP) is a label stack processing feature, which when enabled, "pops" (or discards) the MPLS header and forwards the payload over the next link. Where penultimate hop popping is not used the MPLS label switched path (LSP) is equivalent to an MPLS network flow composed of contiguous MPLS link and flow domain flows as shown in Figure 12.

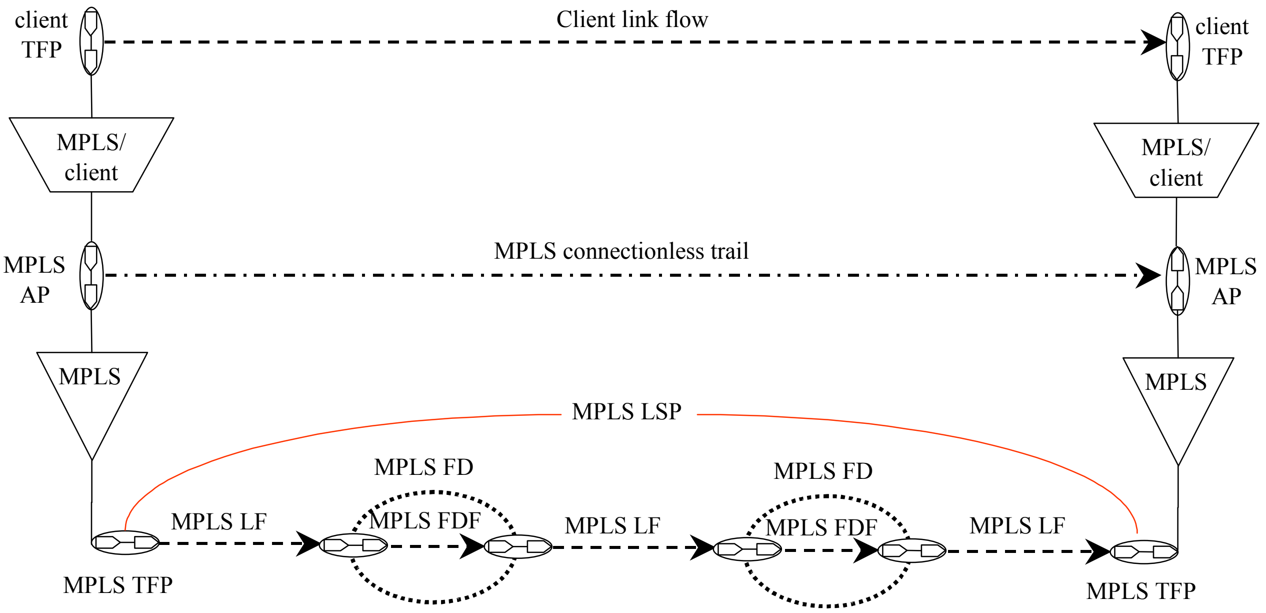


Figure 12/G.8110/Y.1370 - MPLS without Penultimate Hop Popping

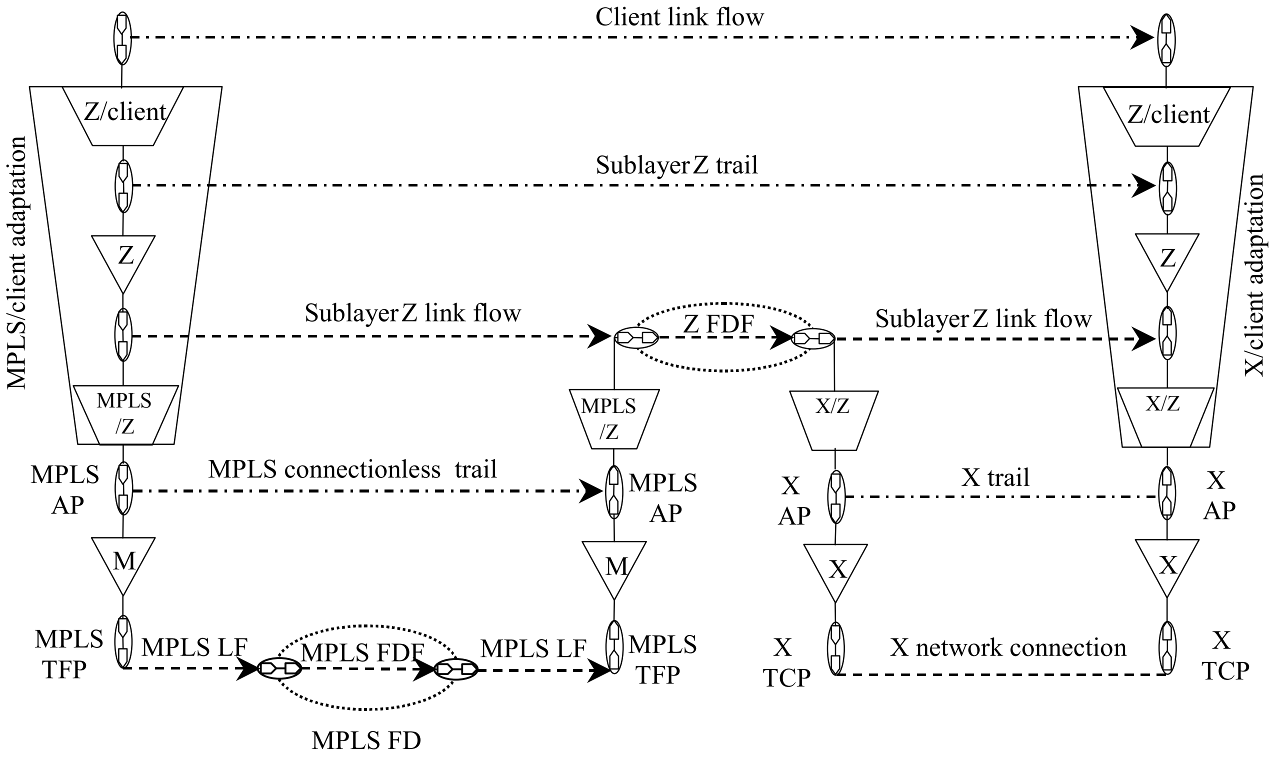
When penultimate hop popping is used, each of the hops in the LSP, with the exception of the final one, is equivalent to the transit of an MPLS link. However in PHP the last hop is associated with a link in another layer network. This layer network is made visible by the expansion of the MPLS/client adaptation function as shown in Figure 13. This resultant layer network is denoted as Z. The characteristic information of layer network Z is equivalent to the payload of the unexpanded MPLS/client adaptation function. It is composed of client characteristic information plus any client specific information added as part of the unexpanded MPLS/client adaptation function. The characteristic information of the Z sublayer therefore corresponds to a label stack entry or an IP packet. The final Z link is supported by a non-MPLS based technology denoted as X. In the example of Figure 13 this is a connection oriented technology and as such the Z flow is supported by a trail in the X layer network.

The LSP for the case of PHP is shown in Figure 14.

The relationship between label stack entries in RFC3032, MPLS traffic unit characteristic information and the characteristic information transferred on the final link of an LSP when PHP is present is shown in Figure 15.

Note that the MPLS/client adaptation at the source of the LSP is unaware that penultimate hop popping has occurred and as such all three functions, Z/client adaptation source, Z flow termination source and MPLS/Z adaptation source are encapsulated within an MPLS/client adaptation function. They are shown here for modelling purposes but their combined behaviour is the same as that of the MPLS/client adaptation itself.

The Z trail offers no trail overhead. The client link flow therefore derives its integrity from Z's server trails, which are themselves disjoint and as such cannot provide validated end-to-end information transfer as a service to the client.



In this example the sublayer Z link flow is supported by a connection-oriented trail in technology X.

Figure 13/G.8110/Y.1370 - Penultimate Hop Popping in MPLS

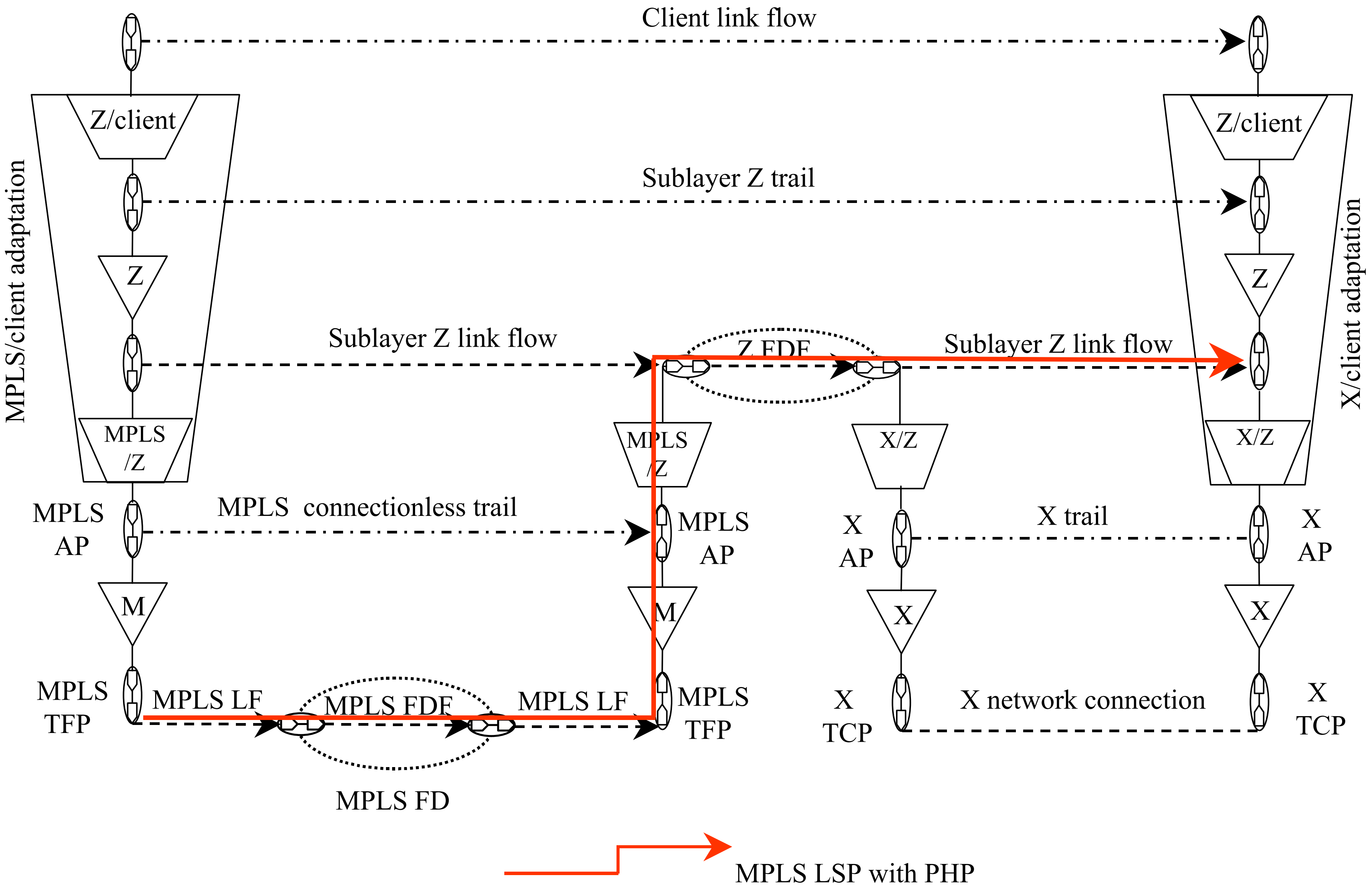
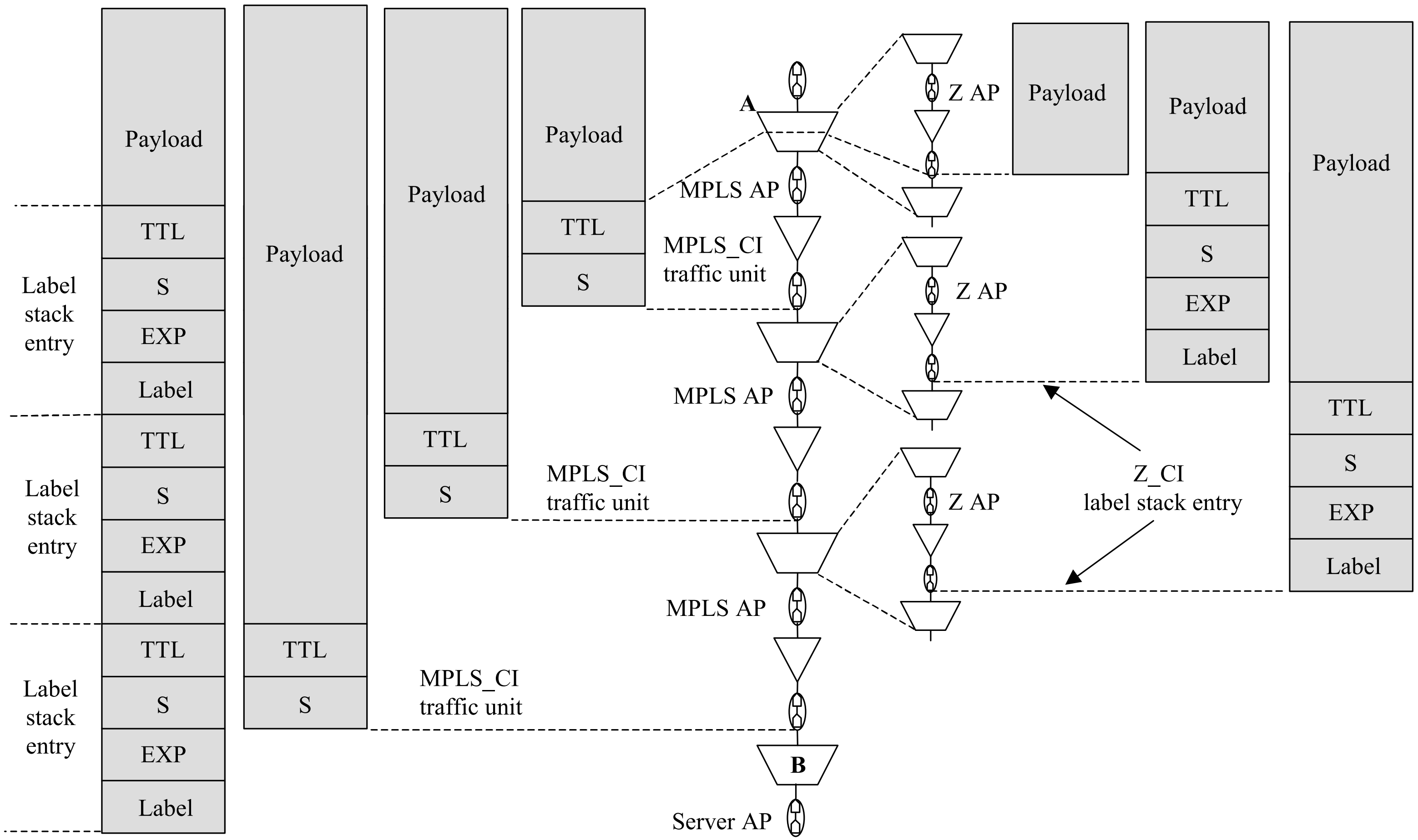


Figure 14/ G.8110/Y.1370 - MPLS LSP with Penultimate Hop Popping (note the LSP is shown slightly to one side for convenience)



RFC3032 label stack	MPLS traffic unit characteristic information	Characteristic information on last link of an LSP when PHP is present (expansion of adaptation)

A - MPLS client or IP client. Adaptation expanded according to client specific/MPLS server specific processes
B -MPLS server or non-MPLS server. If PHP at this level were present the adaptation would be expanded in same manner as others

Direction of information processing is from bottom to top. Note that the characteristic information present on the last link of an LSP with PHP corresponds to a label stack boundary.

Figure 15/G.8110/Y.1370 - Relationship between label stack entries and characteristic information

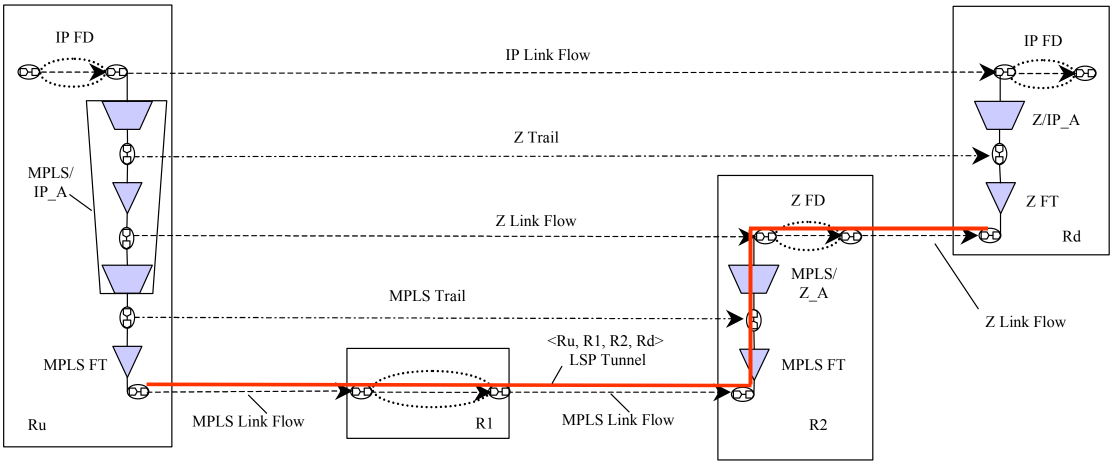
From a network perspective only the MPLS/Z and X/Z adaptations are visible at the penultimate LSR as all other functions associated with the Z layer network are encapsulated in the MPLS/client_A_So or X/client_A_Sk. The processes associated with the MPLS/Z_A_Sk and X/Z_A_Sk are described in Table 2.

Table 2/G.8110/Y.1370 - Assignment of processes to MPLS/Z_A_Sk and X/Z_A_So

Transport Processing Function	Processes
MPLS/Z_A_Sk Client CI is MPLS label stack entry	Extract and process the S bit from the MPLS shim header associated with the MPLS server Process TTL and EXP fields according to sections 13.2 and 13.3 for the MPLS shim header associated with Z
MPLS/Z_A_Sk Client CI is IP packet	Extract and process the S bit from the MPLS shim header associated with the MPLS server Process TTL and EXP fields according to sections 13.2 and 13.3 for the IP header associated with Z
X/Z_A_So	Map the client Z characteristic information to create X_AI. Processes are X specific

7.5 LSP Tunnels

An LSP can be used to form a tunnel between routers that are not directly connected. An example is shown in Figure 16 where there is an IP link flow between routers Ru and Rd, where Rd is a transit router and the routers are connected via intermediate label switched routers R1 and R2. The IP flow between router Ru and Rd travels through an LSP that forms an LSP tunnel <Ru, R1, R2, Rd> , where Ru is the transmit end of the tunnel and Rd is the receive end of the tunnel. In this example the LSP tunnel includes a penultimate hop pop at R2.



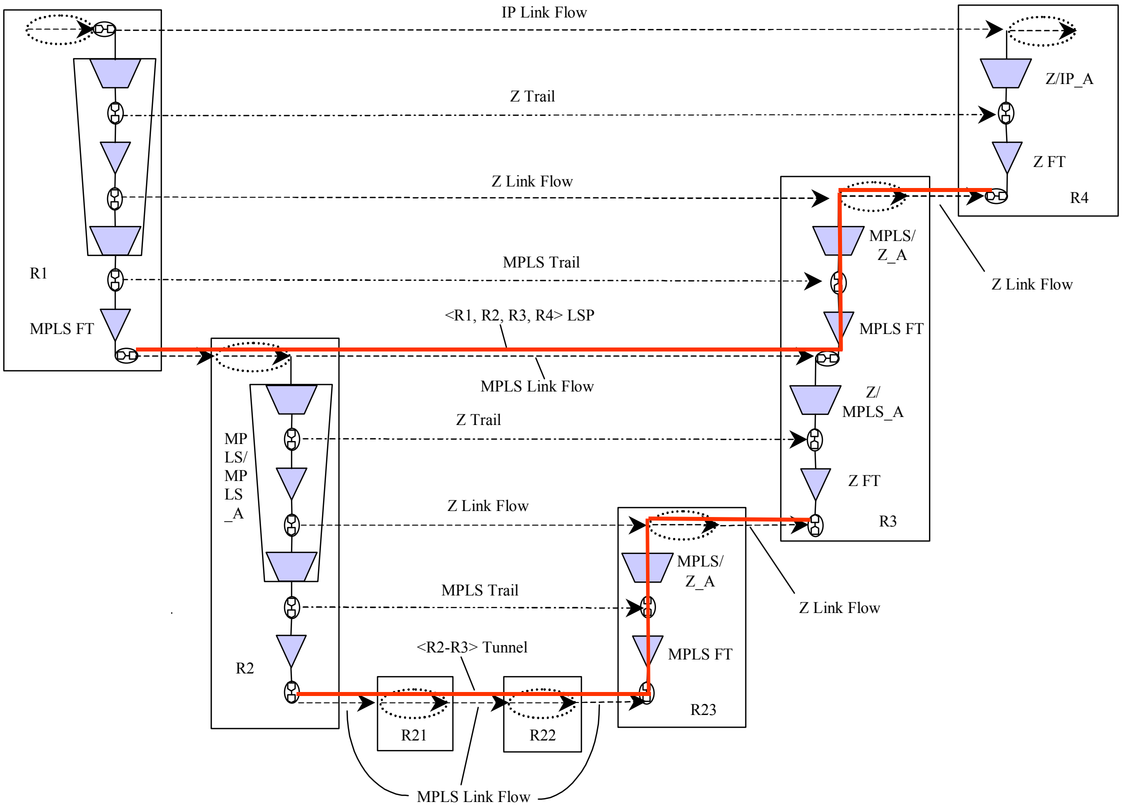
(Note – the tunnel is shown slightly aside from the link flows for diagrammatic convenience)

Figure 16/G.8110/Y.1370 - Example of an LSP tunnel

Figure 17 shows an LSP with the path <R1, R2, R3, R4>. Penultimate hop popping occurs at R3. This LSP represents a tunnel between the end points of the IP link flow between R1 and R4. This link flow is supported by a Z trail, where the characteristic information of the Z layer represents an IP packet.

The end points of the MPLS link flow between R2 and R3 represent the end points of an LSP tunnel, R2-R3, formed by the LSP with the path <R2, R21, R22, R23, R3> . There is a penultimate hop pop present in this LSP at R23. The MPLS link flow between R2 and R3 is supported by a Z trail, where the characteristic information of the Z layer represents a label stack entry.

Note that whilst Figure 17 shows the end points of a single MPLS link flow as the end points of a tunnel, a tunnel in general can support multiple link flows which are multiplexed and demultiplexed into/from the tunnel via adaptation functions. A tunnel can also be constructed from any valid LSP construct, e.g. a point-to-point LSP or a multipoint-to-point LSP tree.

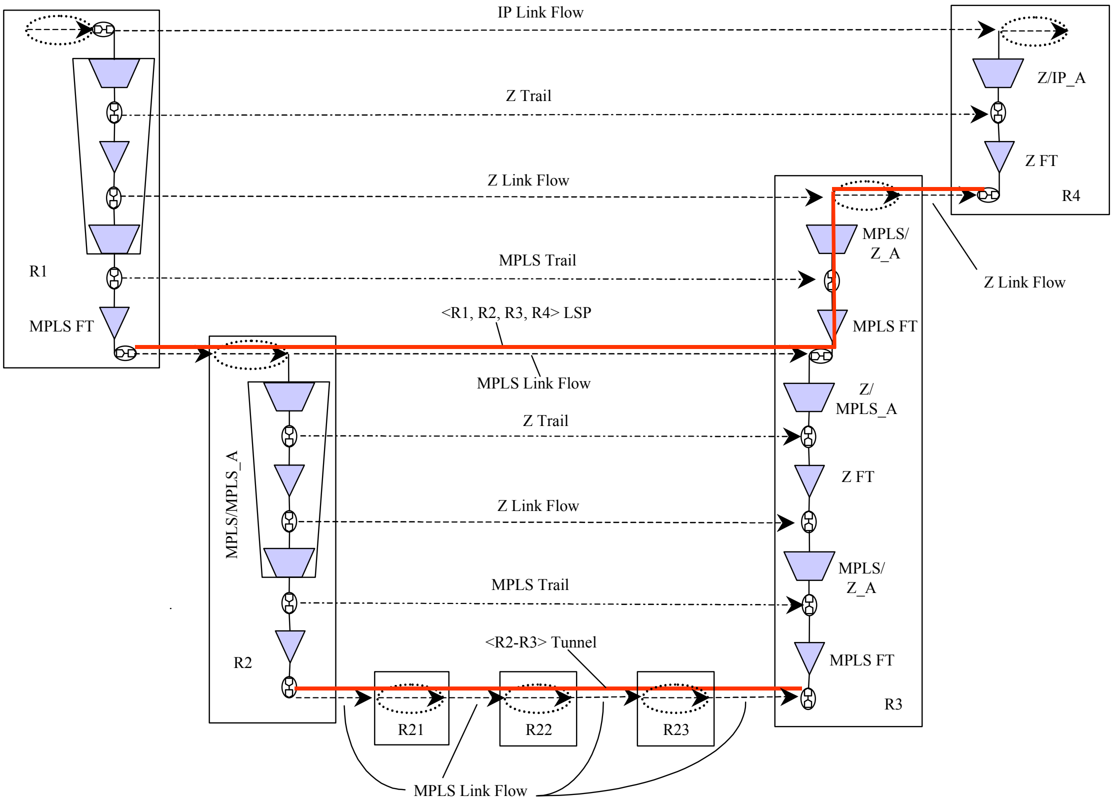


(Note – the tunnel is shown slightly aside from the link flows for diagrammatic convenience)

Figure 17/G.8110/Y.1370 - An LSP tunnel within an LSP where the server layer LSP has a PHP

Figure 18 shows an LSP with the path <R1, R2, R3, R4>. Penultimate hop popping occurs at R3. This LSP represents a tunnel between the end points of the IP link flow between R1 and R4. This link flow is supported by a Z trail, where the characteristic information of the Z layer represents an IP packet.

The end points of the MPLS link flow between R2 and R3 represent the end points of an LSP tunnel, R2-R3, formed by the LSP with the path <R2, R21, R22, R23, R3> . There is no penultimate hop popping present in this LSP, which is equivalent to a network flow. The MPLS link flow between R2 and R3 is supported by an MPLS trail.



(Note – the tunnel is shown slightly aside from the link flows for diagrammatic convenience)

Figure 18/G.8110/Y.1370 - An LSP tunnel within an LSPs where the server layer LSP has no PHP

The LSP tunnel concept can be applied recursively, where an MPLS link flow, which is part of an LSP, in a client layer is tunnelled through a server layer LSP.

8 MPLS shim header based functional architecture based on G.805

8.1 MPLS Layer Network

The MPLS layer network provides the transport of adapted information through an MPLS trail between MPLS access points.

The MPLS layer network characteristic information) is transported over an MPLS network connection. The MPLS layer network contains the following transport processing functions, transport entities and topological components (see Figure 19):

- MPLS trail
- MPLS trail termination source (MPLS_TT_So)

- MPLS trail termination sink (MPLS_TT_Sk)
- MPLS network connection (NC)
- MPLS link connection (LC)
- MPLS subnetwork connection (SNC)
- MPLS subnetwork (SN)
- MPLS link

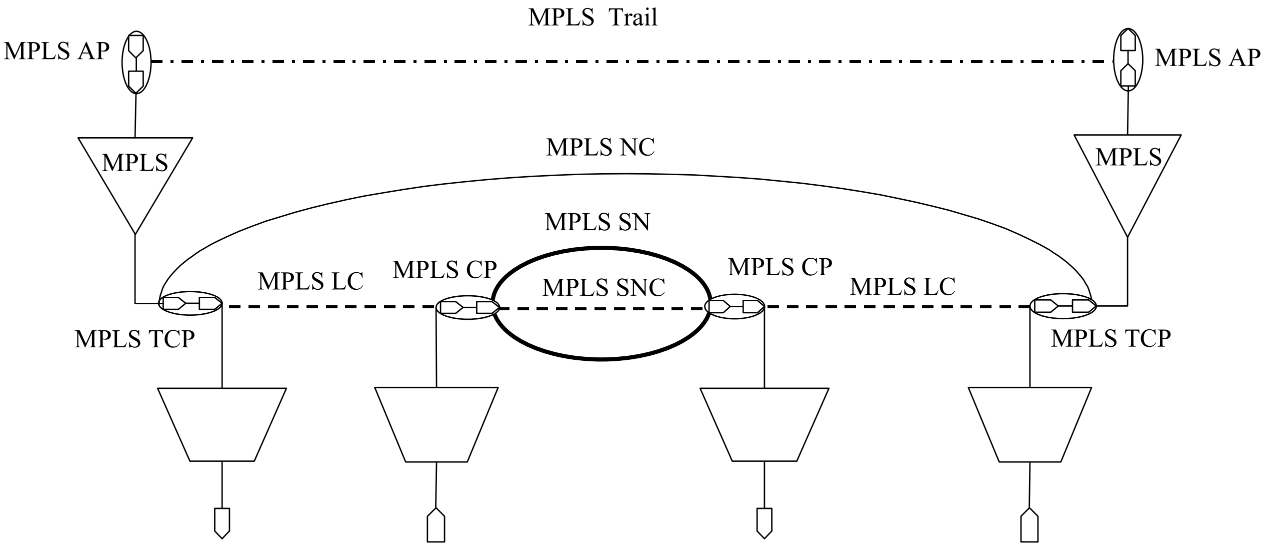
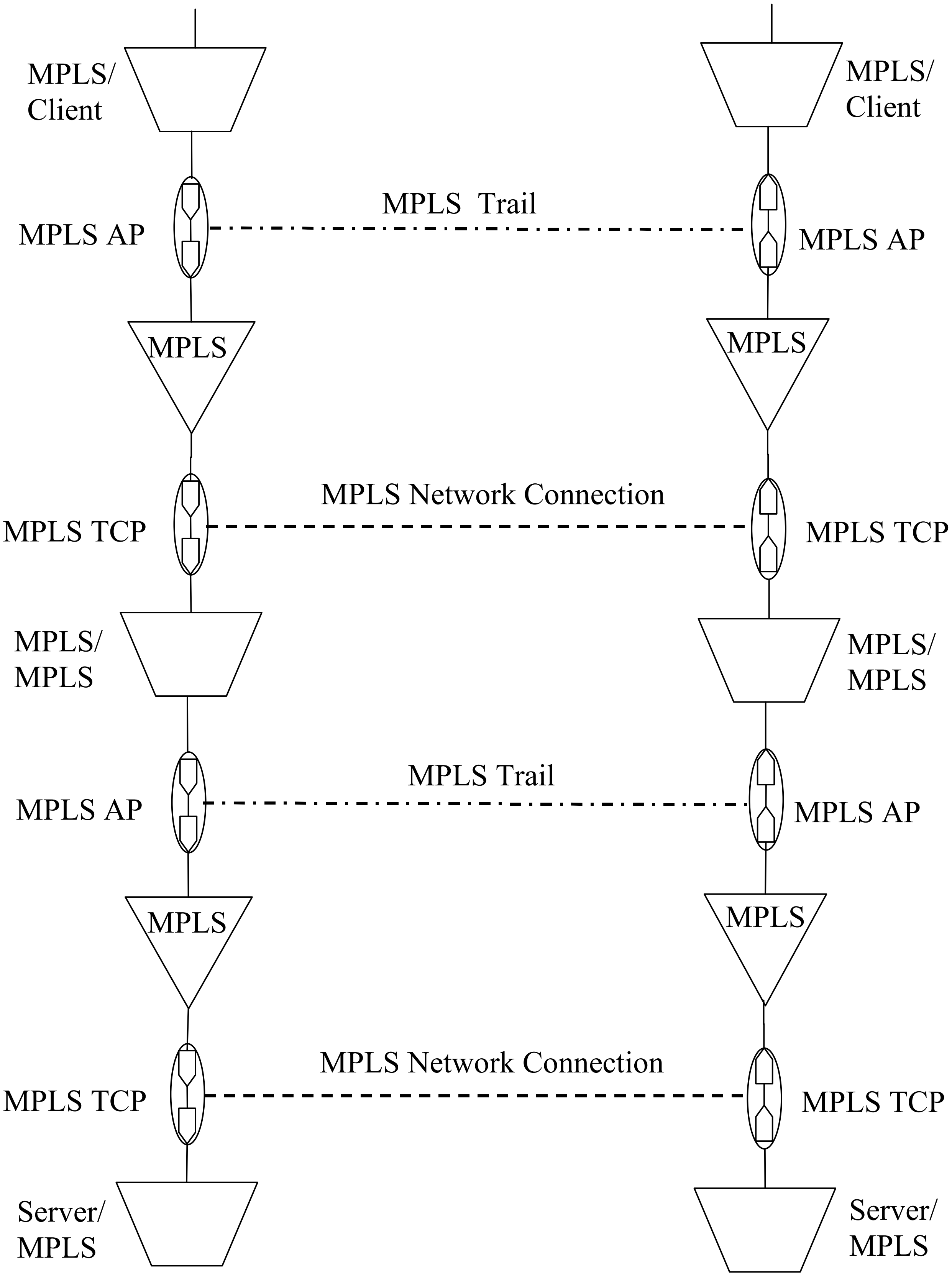


Figure 19/G.8110/Y.1370 – MPLS layer network example

The MPLS layer network may be employed recursively to describe an MPLS hierarchy implemented as a label stack. This is described by the use of sub-layering. A transport network based on MPLS can be decomposed into a number of independent transport sublayer networks with a client/server association between adjacent sublayer networks. An example of MPLS sublayers and their structure and the adaptation functions is shown in Figure 20.

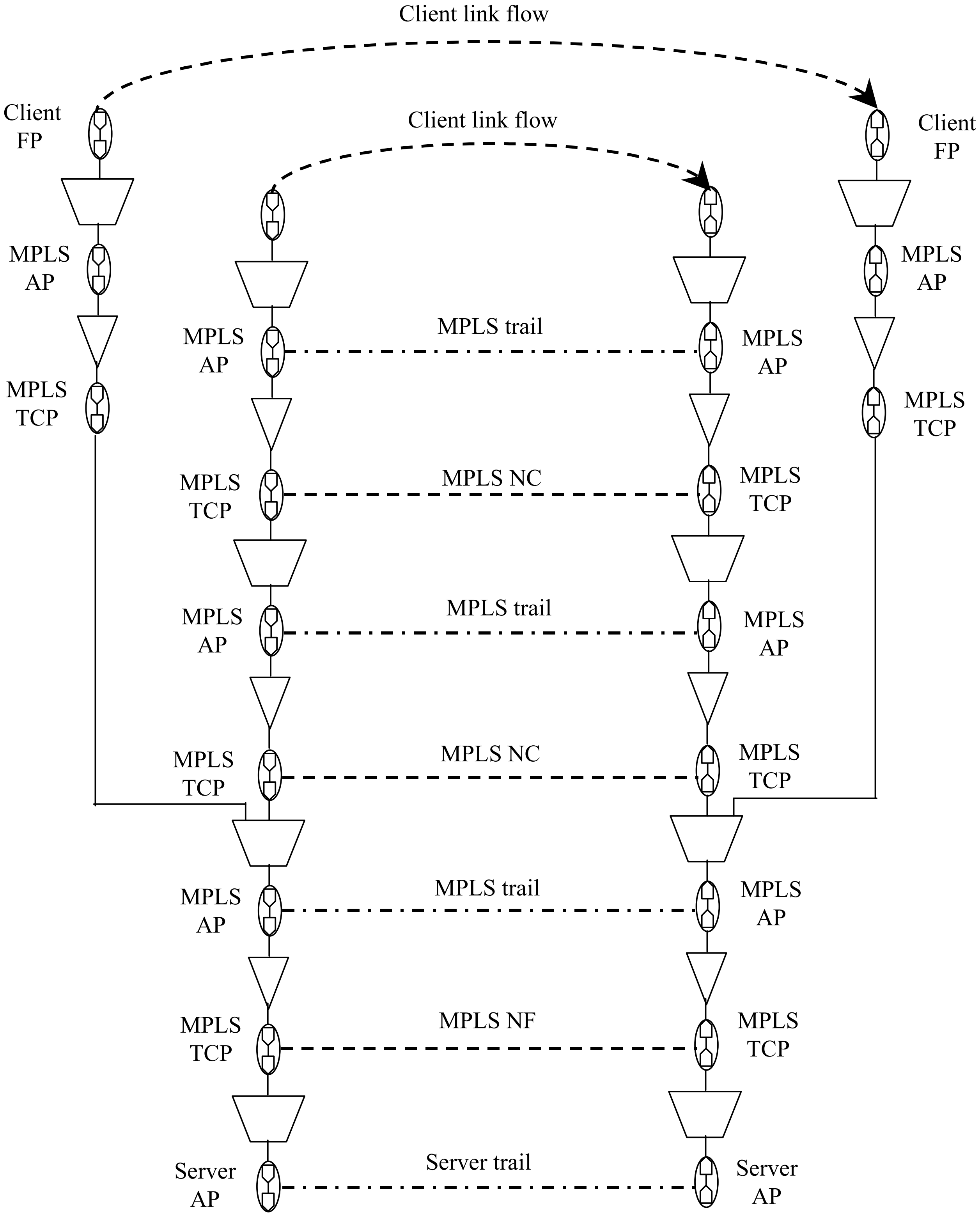
The label stack is related to the MPLS sublayers such that diagrammatically the bottom of the stack is associated with the MPLS sublayer at the top of the diagram (where the client is not MPLS), whilst the top of the label stack is associated with the MPLS sublayer at the bottom.

MPLS allows for the creation of an arbitrary depth of sublayers, formed by the label stack. An example is shown in Figure 21.



Two levels are shown in this example. Additional sublayers can be added as required. The bottom of the stack is at the top.

Figure 20/G.8110/Y.1370 - Example of MPLS hierarchy illustrated using sub-layering



The outermost client connection (or flow) is supported by an MPLS hierarchy with a stack depth of two, whilst the inner client connection (or flow) is supported by a MPLS stack of depth three. As such, other than the bottom of the stack an MPLS sublayer has no designated depth

Figure 21/G.8110/Y.1370 - Example of MPLS stack depths

8.1.1 MPLS Topological Components

The MPLS topological components are:

- MPLS layer network
- MPLS subnetwork
- MPLS link
- MPLS Access Group

The MPLS layer network can be partitioned into one or more MPLS subnetworks interconnected by MPLS links.

8.1.1.1 MPLS Layer Network

The MPLS layer network is defined by the complete set of MPLS access groups that may be associated for the purpose of transferring information. The information transferred is characteristic of the MPLS layer network and is termed MPLS characteristic information. The topology of the MPLS layer network is described by MPLS access groups, MPLS subnetworks and the MPLS links between them. The structures within the MPLS layer network and its server and client layer networks are described by the components below.

8.1.1.2 MPLS Subnetwork

An MPLS subnetwork is defined by the set of MPLS connection points that are available for the purpose of transferring information. In general, MPLS subnetworks may be partitioned into smaller subnetworks interconnected by MPLS links. The matrix is a special case of an MPLS subnetwork that cannot be further partitioned.

8.1.1.3 MPLS Link

An MPLS link consists of a subset of the MPLS connection points at the edge of one MPLS subnetwork or MPLS access group that are associated with a corresponding subset of MPLS connection points at the edge of another MPLS subnetwork or MPLS access group for the purpose of transferring MPLS characteristic information. The MPLS link represents the topological relationship and available capacity between a pair of MPLS subnetworks, or an MPLS subnetwork and an MPLS access group, or a pair of MPLS access groups.

Multiple MPLS links may exist between any given MPLS subnetwork and MPLS access group or pair of MPLS subnetworks or MPLS access groups.

8.1.1.4 MPLS Access Group

An MPLS access group is a group of co-located MPLS trail termination functions that are connected to the same MPLS subnetwork or MPLS link.

8.1.2 MPLS Transport Entities

The MPLS transport entities are:

- MPLS Link Connection
- MPLS Network Connection
- MPLS Subnetwork Connection
- MPLS trail

8.1.3 MPLS Transport Processing Functions

The MPLS transport processing functions are:

- MPLS trail termination function
- MPLS to client layer network adaptation functions

8.1.3.1 MPLS Trail Termination

The MPLS trail termination source (MPLS_TT_So) performs the following processes between its input and output:

- inserts the 8-bit TTL field
- Output the resulting MPLS_CI

The MPLS trail termination sink (MPLS_TT_Sk) performs the following functions between its input and output:

- Extract and terminate the 8-bit TTL field
- Output the resulting MPLS_AI

8.1.3.2 MPLS to client layer network adaptation functions

The MPLS/client adaptation functions are described in section 10.

8.1.4 MPLS Reference Points

The MPLS reference points (Figure 19) are:

- MPLS Access Point (AP)
- MPLS Connection Point (CP)
- MPLS Termination Connection Point (TCP)

8.1.4.1 MPLS Access Point

An MPLS Access Point (MPLS AP) represents the binding between an MPLS trail termination function and one or more MPLS/client, or MPLS/MPLS, adaptation functions.

8.1.4.2 MPLS Connection Point

An MPLS Link connects to an MPLS Subnetwork or another MPLS Link via an MPLS Connection Point. This connection point is provided through the Server/MPLS, or MPLS/MPLS adaptation function.

8.1.4.3 MPLS Termination Connection

An MPLS Termination Connection Point, (MPLS TCP), connects an MPLS Trail Termination (MPLS_TT) function with an MPLS Link.

8.2 MPLS Layer Network Partitioning

The description of MPLS layer network partitioning is the same as section 7.2 with the following exceptions

- G.809 entities are translated to G.805 entities according to table C.1 of annex C.

8.3 MPLS Subnetwork Behaviour

8.3.1 Reserved Labels

The reserved label space is as described in section 7.3.1 except that:

- where Y.1711 is supported, the OAM alert label, 14, is used.

8.3.2 Label merge

Merging is not supported in the MPLS shim header based architecture based on G.805.

8.3.3 Global Label Space

Labels that belong to a global label space (also known as per-platform label space) do not have their context defined by the link on which they are received. As a result they are unique for the matrix. In a connection-oriented context only one LSP is associated with a particular label value taken from the global label space.

8.3.4 Interface Label Space

The alternative to the use of a global label space is a per-interface label space where an MPLS label value is only unique for a connection point within a link.

8.3.5 Support for multiple label spaces

Multiple label spaces may be supported as described in section 7.3.5.

8.4 Penultimate Hop Popping (PHP)

Penultimate hop popping is as described in section 7.4 with the following exceptions:

- G.809 entities are translated to G.805 entities according to table C.1 of annex C.

8.5 LSP Tunnels

The description of LSP tunnels is the same as section 7.5 with the following exceptions:

- G.809 entities are translated to G.805 entities according to table C.1 of annex C
- LSP tunnels are point-to-point

9 MPLS Hierarchies

9.1 G.809 MPLS Hierarchies

MPLS hierarchies implemented as label stacks according to the G.809 model are described in section 7. The assumption in section 7 is that all of the MPLS hierarchy, and therefore all of the MPLS sublayer networks, are described using the G.809 model.

An example of the relationship of an MPLS flow domain to the flow points in such a label stack is shown in Figure 22. The recursive nature of the sublayering is such that the flow domain is associated with flow points in multiple sublayers.

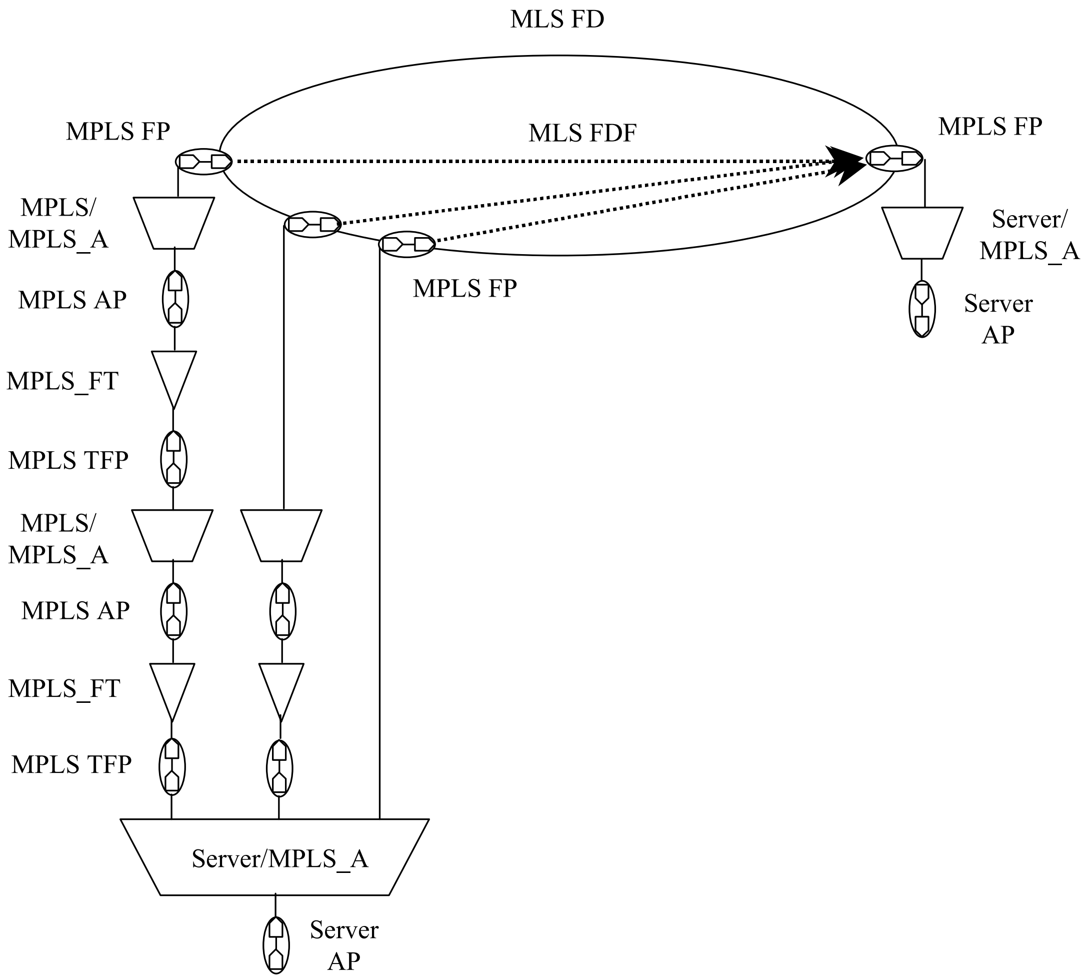


Figure 22/G.8110/Y.1370 - Relationship between an MPLS flow domain and the sublayers of a label stack

9.2 G.805 MPLS Hierarchies

MPLS hierarchies implemented as label stacks according to the G.805 model are described in section 8. The assumption in section 8 is that all of the MPLS hierarchy, and therefore all of the MPLS sublayer networks, are described using the G.805 model.

An example of the relationship of an MPLS subnetwork to the connection points in such a label stack is shown in Figure 23. The recursive nature of the sublayering is such that the subnetwork is associated with connection points in multiple sublayers.

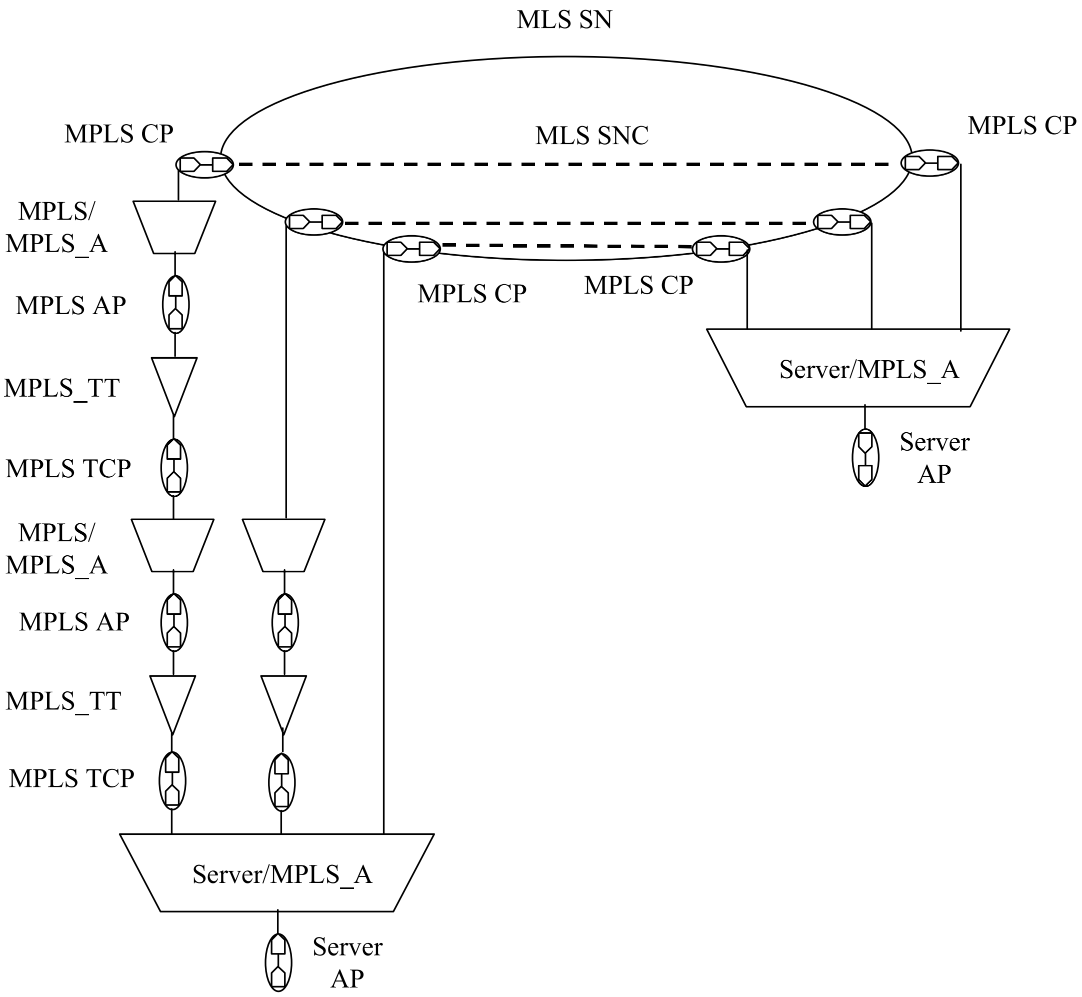


Figure 23/G.8110/Y.1370 - Relationship between an MPLS subnetwork and the sublayers of a label stack

9.3 Heterogeneous MPLS Hierarchies

An MPLS hierarchy may also be implemented such that sublayer networks based on G.805 and sublayers based on G.809 both exist within the same MPLS hierarchy.

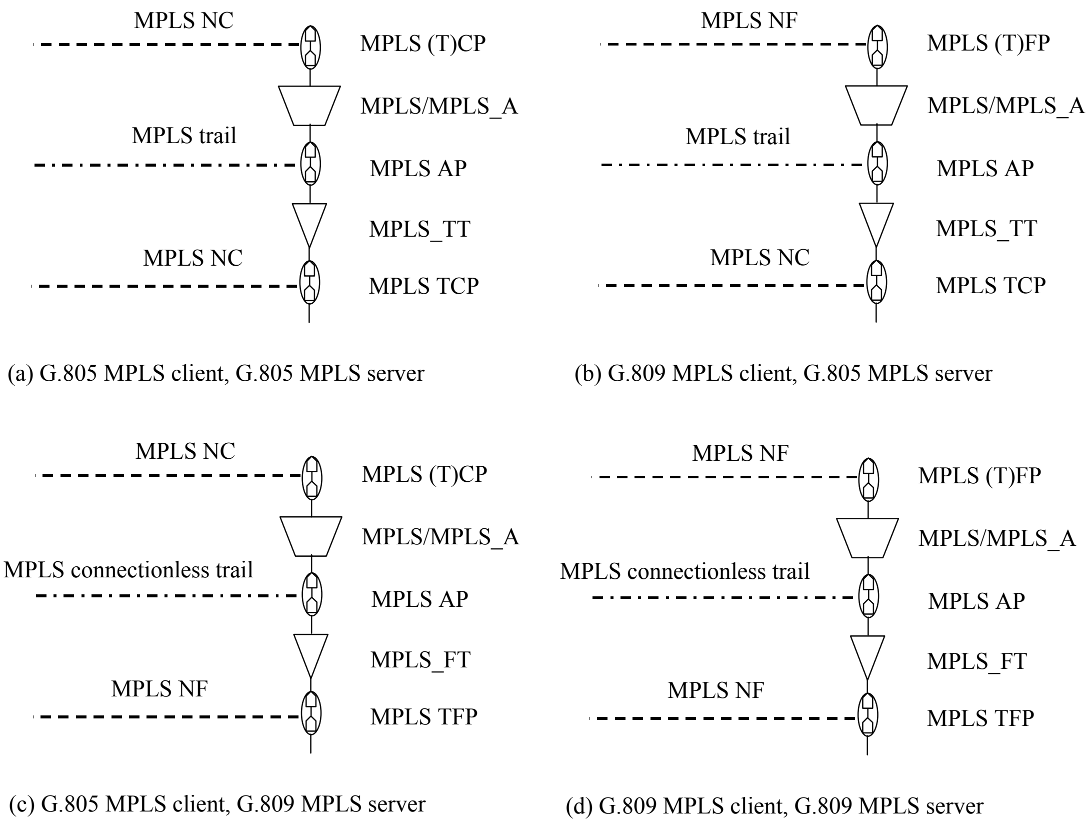
For sublayers between the bottom and top of the label stack a G.805 sublayer may therefore have either:

- a G.805 client, or
- a G.809 client

Similarly, for sublayers between the bottom and top of the label stack a G.809 sublayer may therefore have either:

- a G.805 client, or
- a G.809 client

These relationships are illustrated in Figure 24.



(a) G.805 MPLS client, G.805 MPLS server

(b) G.809 MPLS client, G.805 MPLS server

(c) G.805 MPLS client, G.809 MPLS server

(d) G.809 MPLS client, G.809 MPLS server

Figure 24/G.8110/Y.1370 - MPLS/MPLS Client Server Relationships

These client/server relationships affect the MPLS/MPLS_A functions in the following ways:

- For an MPLS sublayer described using G.809 the MPLS/MPLS_A to/from this sublayer may be bound to:
  - Termination flow points or flow points as described in section 7 for a G.809 MPLS client.
  - Termination connection points or connection points for a G.805 client
- For an MPLS sublayer described using G.805 the MPLS/MPLS_A to/from this sublayer may be bound to:
  - Termination connection points or connection points as described in section 8 for a G.805 client
  - Termination flow points or flow points for a G.809 client

In general an MPLS sublayer, whether it is based on G.805 or G.809, can support:

– G.809 MPLS client sublayers

– G.805 MPLS client sublayers

– G.805 and G.809 MPLS client sublayers

At the top of the label stack a non-MPLS server layer can support

– G.809 MPLS client sublayers

– G.805 MPLS client sublayers

– G.805 and G.809 MPLS client sublayers

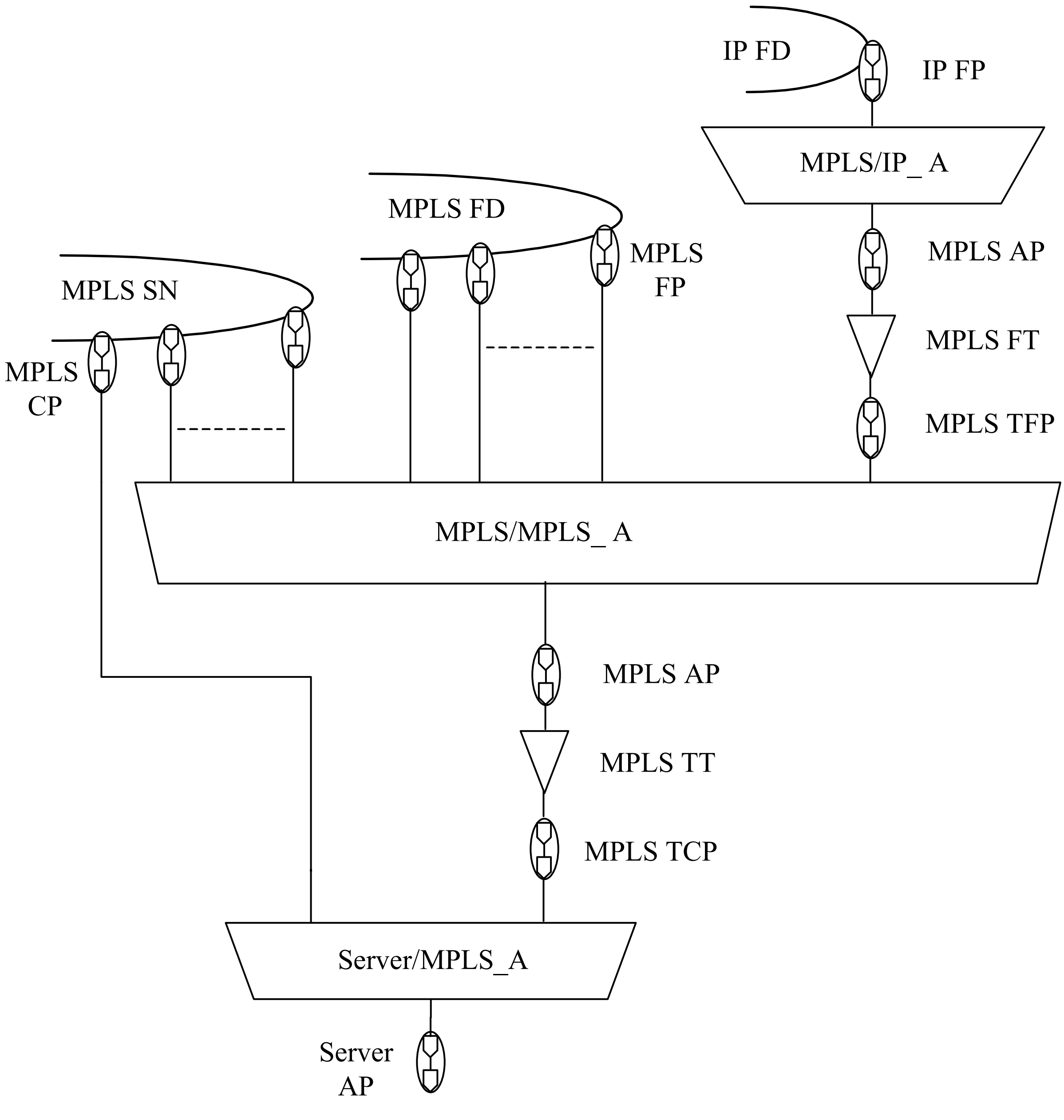
At the bottom of the label stack a non-MPLS client layer may be supported by

– G.809 MPLS server sublayers

– G.805 MPLS server sublayers

– G.805 and G.809 MPLS server sublayers

An example of an MPLS hierarchy containing both G.805 and G.809 sublayers is illustrated in Figure 25.



(Note that the adaptation function in this example is associated with both (T)CPs and (T)FPs)

Figure 25/G.8110/Y.1370 - MPLS Hierarchy containing both G.805 and G.809 based sublayers

The following rules apply within an MPLS hierarchy containing both G.805 and G.809 based layer networks:

- MPLS FP’s can only be bound to flow domains and never to subnetworks
- An MPLS flow domain is shared across all G.809 sublayers within a layer of the hierarchy
- MPLS TFP’s can only be bound to flow terminations and never to trail terminations
- MPLS CP’s can only be bound to subnetworks and never to flow domains
- An MPLS subnetwork is shared across all G.805 sublayers within the hierarchy
- MPLS TCP’s can only be bound to trail terminations and never to flow terminations
- MPLS (T)FP’s and (T)CP’s never exist together in the same layer network.

The set of access points associated with G.805 based MPLS sublayers are completely separate from the set of access points associated with G.809 based MPLS sublayers.

10 Server/client associations

Three forms of adaptation function are considered in this recommendation:

- MPLS/client adaptation, where the client is not MPLS. In this case the adaptation function is associated with the bottom of the label stack.
- MPLS/MPLS adaptation, where the client is MPLS.
- Server/MPLS adaptation, where the server is not MPLS. In this case the adaptation function is associated with the top of the label stack

The adaptation functions and their main processes are shown in Figure 26.

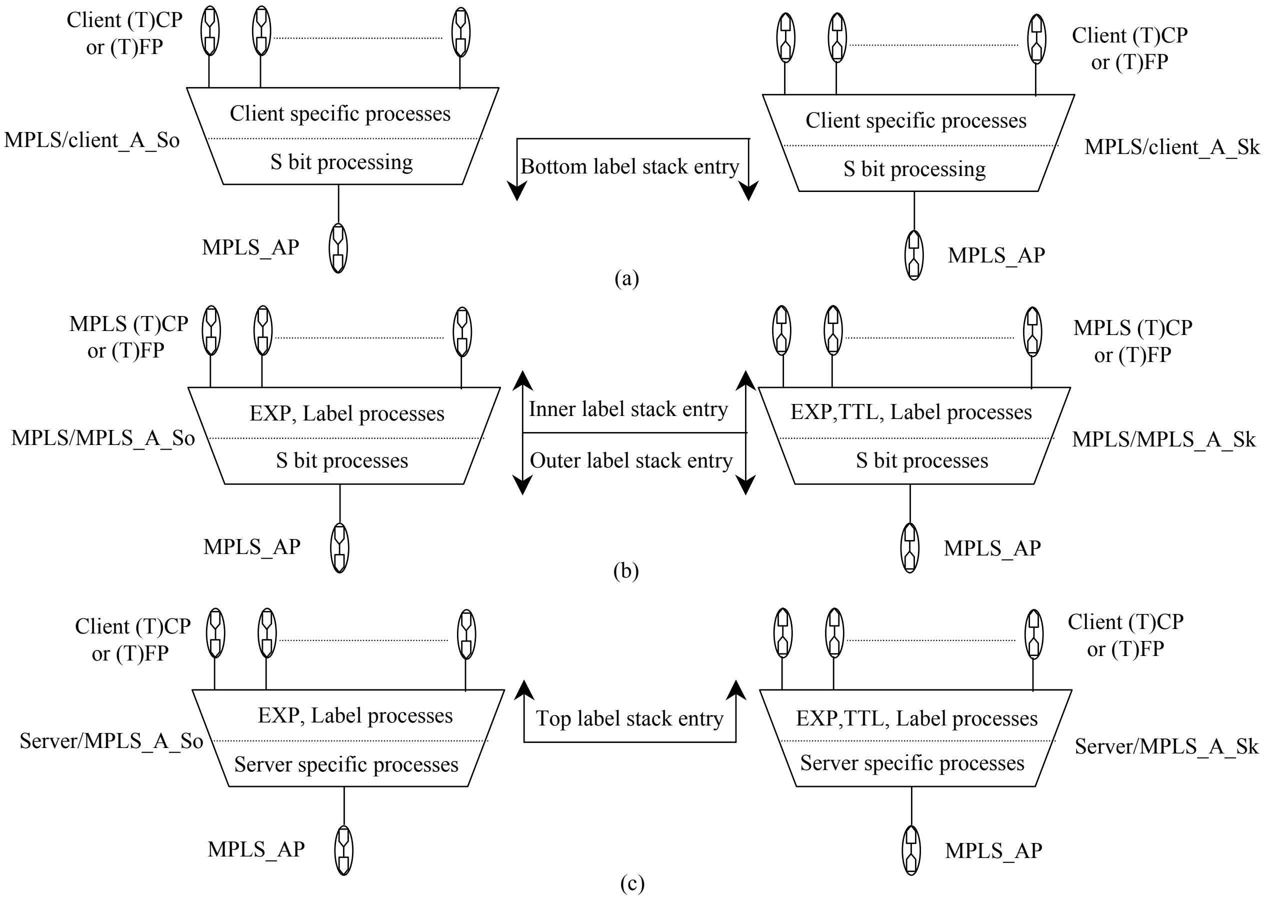


Figure 26/G.8110/Y.1370 - Server/client associations and main processes.
(a) MPLS/client source and sink,
(b) MPLS/MPLS source and sink, and
(c) server/MPLS source and sink.

10.1 MPLS/client adaptation

The MPLS/client adaptation (MPLS/Client_A) is considered to consist of two types of processes: client-specific processes and server-specific processes. The description of client-specific processes, except where:

- they are related to TTL and Diff-Serv processing behaviour for an IP client, as described in section 13,

are outside the scope of this Recommendation.

10.1.1 MPLS/IP adaptation

The MPLS/IP adaptation source (MPLS/IP_A_So) performs the following server specific processes between its input and output:

- Map the IP packet to the payload of the MPLS packet.
- Insert a 1-bit S field set to 1. This indicates that the client is not MPLS.

- Output the resulting MPLS_AI

The MPLS/IP adaptation sink (MPLS/IP_A_Sk) performs the following server specific processes between its input and output:

- Extract and process the 1-bit S field.
- Extract the IP packet from the payload of the MPLS_AI.

Note that the IP layer network may be either IP version 4 or version 6. Where it does not matter for the purpose of description the MPLS/IP prepend is used. Where it is necessary to be more precise, MPLS/IPv4 or MPLS/IPv6 are used as appropriate.

10.1.2 MPLS/MPLS adaptation

The MPLS/MPLS adaptation function provides the MPLS link end functionality.

The MPLS/MPLS adaptation source (MPLS/MPLS_So) performs the following processes between its input and its output:

- Client specific processes
  - Insert the same value 20-bit MPLS Label into each MPLS_CI traffic unit associated with a particular (termination) flow point or (termination) connection point.
  - Insert 3-bit EXP field according to the processes defined in section 13.3. The MPLS_CI plus the 20-bit label plus the EXP field is equivalent to a label stack entry.
  - Multiplex the MPLS labelled packets
- Server specific processes
  - Insert a 1-bit S field set to 0. This indicates that the client is MPLS and therefore the bottom of the stack has not been reached.
  - Map MPLS labelled packet to the payload of the MPLS_AI traffic unit of the server MPLS sub-layer

The MPLS/MPLS adaptation sink (MPLS/MPLS_Sk) performs the following processes between its input and its output:

- Server Specific processes
  - Extract and process the 1-bit S field
  - Extract the MPLS labelled packet of the client MPLS sub-layer from the payload of the MPLS_AI
- Client Specific processes
  - Demultiplex the MPLS_AI by means of the 20-bit label value
  - Remove the 20-bit label
  - Process the 3-bit EXP field as described in 13.3.
  - Process TTL according to the processes described in section 13.2. When the TTL is decremented and has expired, the traffic unit is discarded.
  - Output the MPLS_CI traffic unit.

10.2 Non-MPLS Server/MPLS adaptation

The Server/MPLS adaptation function provides the MPLS link end functionality.

The Server/MPLS adaptation function is considered to consist of two types of processes: client-specific processes and server-specific processes. The client specific processes are associated with the MPLS_CI traffic units, which ingress/egress via the MPLS (T)FP/FPP. Server specific processes are outside the scope of this Recommendation.

The Srv/MPLS adaptation source (Srv/MPLS_A_So) performs the following processes between its input and output:

- Insert the same value 20-bit MPLS Label into each MPLS_CI traffic unit associated with a particular flow or connection point
- Insert EXP field according to processes described in section 13.3
- Multiplex the MPLS Labelled packets
- Server layer related specific processes

The Srv/MPLS adaptation sink (Srv/MPLS_A_Sk) performs one of the following processes between its input and output:

- Server layer related specific processes
- Demultiplex the MPLS labelled Packets using the 20-bit label value
- Remove the 20-bit Label
- Process EXP according to section 13.3.
- Process TTL according to section 13.2. When the TTL is decremented and has expired, the traffic unit is discarded.

11 MPLS Network Control

For further study.

12 MPLS survivability techniques

12.1 Protection techniques

For further study.

12.2 Network restoration

For further study.

13 MPLS and support of the Diff-Serv Architecture

The use of MPLS for support of Differentiated Services (Diff-Serv), is described in RFC 3270. Diffserv terminology discusses both the traffic and the treatment of traffic, and deconstructs it in a hierarchical fashion.

The relevant traffic definitions are:

- Behaviour Aggregate (BA): This is a collection of packets with a common Diff-Serv code point (DSCP) transiting a link in a particular direction.

- Ordered Aggregate (OA): This is a set of BAs that share an ordering constraint.

And the associated treatment definitions are:

- Per-Hop-Behavior (PHB): Which is how an LSR will treat a BA.
- PHB-Group: This is a set of PHBs with a common constraint and therefore must be implemented relative to each other.
- PHB Scheduling Class (PSC): A PHB group with the minimum common constraint being a requirement for microflow ordering. This is how an LSR will treat an OA.

Three Diff-Serv tunnelling models (if and how PHB information is propagated between sublayers) are described:

- The uniform model, with or without penultimate hop popping,
- The pipe model, with no penultimate hop popping,
- The short pipe model, with or without penultimate hop popping.

These models are discussed in section 13.3.

Diff-Serv information encoded in either the IP header or the MPLS shim header is used to select the Per Hop Behaviour (PHB), as described in RFC 3270, that determines the scheduling treatment and, where appropriate, the drop precedence of the packet.

Two forms of LSP are defined within RFC 3270:

- E-LSP: an EXP inferred PHB scheduling class (PSC) LSP. The PSC and drop precedence is inferred directly from the EXP field in the MPLS shim header.
- L-LSP: a label only inferred PHB scheduling class (PSC) LSP. The scheduling treatment is inferred from the 20-bit label in the MPLS shim header. The drop precedence to be applied is carried in the EXP field contained in the MPLS header.

The LSP Diff-Serv information contained within an MPLS header is as shown in Table 3.

Table 3/G.8110/Y.1370 - Relationship between LSP type, Per Hop Behaviour and MPLS header fields

Type of LSP	Per Hop Behaviour	
	PHB Scheduling Class	Drop Precedence
E-LSP	EXP field	
L-LSP	Label	EXP field

The relationship between the PHB and the LSP Diff-Serv information is determined by means of mappings that are established either by means of a preconfigured mapping or by means of a mapping that is explicitly signalled at label setup. This is illustrated in Table 4.

Table 4/G.8110/Y.1370 - Mapping of Per Hop Scheduling Class and Drop Precedence to PHB as a function

LSP Type	LSP Diff-Serv Information Component	Mapping	Mapping Mechanism
E-LSP	PSC plus Drop Precedence	EXP ↔ PHB	Explicitly signalled at label setup OR Pre-configured mapping
L-LSP	PSC	Label – PSC	PSC is explicitly signalled at time of label establishment
	Drop Precedence	EXP ↔ PHB mapping This is a function of the PSC supported on the LSP	Mandatory specified EXP/PSC ↔ PHB mapping

Each of the Diff-Serv tunnelling models utilises the LSP Diff-Serv information (EXP for E-LSP, label and EXP for L-LSP) in the MPLS shim header in different ways. The LSP Diff-Serv processing can be described by a combination of:

- Reference diagrams illustrating the transport entities and transport processing functions of interest
- Descriptions of the LSP Diff-Serv information processing that occurs in each type of transport processing function in the reference models

The reference diagrams are described in 13.1, TTL processing for each tunnelling model is described in 13.2, whilst the LSP Diff-Serv information processing for each tunnelling model is described in 13.3.

13.1 Reference Diagrams for Uniform, Pipe and Short Pipe models

Two reference diagrams are used, one for all three tunnel models without penultimate hop popping and one for uniform and short pipe models with penultimate hop popping.

13.1.1 Reference Diagram for Uniform, Pipe and Short Pipe models with no PHP

The reference model for describing the uniform, pipe and short pipe models in the absence of penultimate hop popping is shown in Figure 27. The tunnel of interest is represented by the processing associated with the tunnel’s label stack entry, whilst the information that is tunnelled (the client) is represented by the tunnelled label stack entry or IP header. Where the client layer is MPLS the MPLS/client adaptation corresponds to an MPLS/MPLS adaptation and processing is wholly MPLS based. Where the client is IP the MPLS/client adaptation corresponds to an MPLS/IP adaptation and processing includes IP Diff-Serv processing.

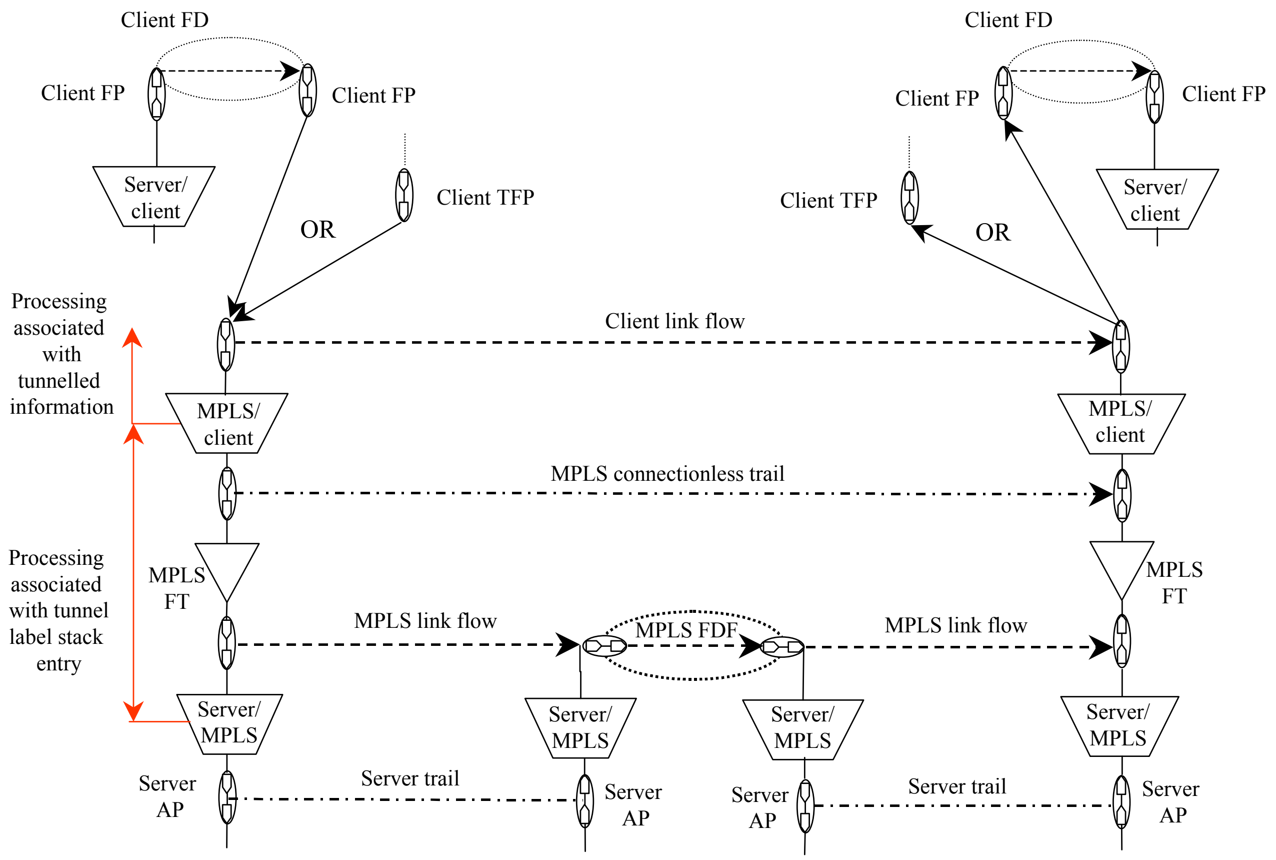


Figure 27/G.8110/Y.1370 - Reference diagram for Uniform, Pipe and Short Pipe models without PHP

For the server/MPLS adaptation the following cases apply:

- the server layer network is MPLS and the adaptation function is of the form MPLS/MPLS adaptation. In this case the processes of the MPLS/MPLS adaptation sink function are dependent on the nature of the MPLS server tunnel mode. The label stack mechanism allows tunnelling to nest to any depth. There is also no requirement for consistent tunnelling models across levels so each tunnel may operate in a different tunnelling mode to its client or its server. Note that in the reference diagram the processes associated with MPLS/client adaptation source function, where the client is MPLS are dependent on the servers tunnelling mode.
- The server layer network is IP. This is not considered further in this Recommendation.

The server is neither MPLS nor IP. This case is described for each tunnel model using the server/MPLS (server is not MPLS) notation. The server layer network may be either connection oriented or connectionless. In the reference diagram it is shown as connection oriented. Server layer processes associated with the adaptation function are not described.

For the server/client adaptation the following cases apply:

- The client is MPLS and the server is MPLS, so the adaptation is MPLS/MPLS_A
- The client is IP. The server may then be:
  - MPLS so the adaptation is MPLS/IP_A

- Any other technology that supports IP so the adaptation is server/IP_A

13.1.2 Reference Diagram for Uniform and Short Pipe models with PHP

The reference model for describing the uniform and short pipe models in the presence of penultimate hop popping is shown in Figure 28. The tunnel of interest is represented by the processing associated with the tunnels label stack entry, whilst the information that is tunnelled (the client) is represented by the tunnelled label stack entry or IP header. Where the client layer is MPLS the MPLS/client adaptation corresponds to an MPLS/MPLS adaptation and processing is wholly MPLS based. Where the client is IP the MPLS/client adaptation corresponds to an MPLS/IP adaptation and processing includes IP Diff-Serv processing.

The use of penultimate hop popping creates the Z sublayer. The characteristic information of this sublayer is dependent on the client and the resultant expansion of the MPLS/client adaptation. If the client is MPLS the characteristic information corresponds to a label stack entry boundary, if it is IP it corresponds to an IP packet.

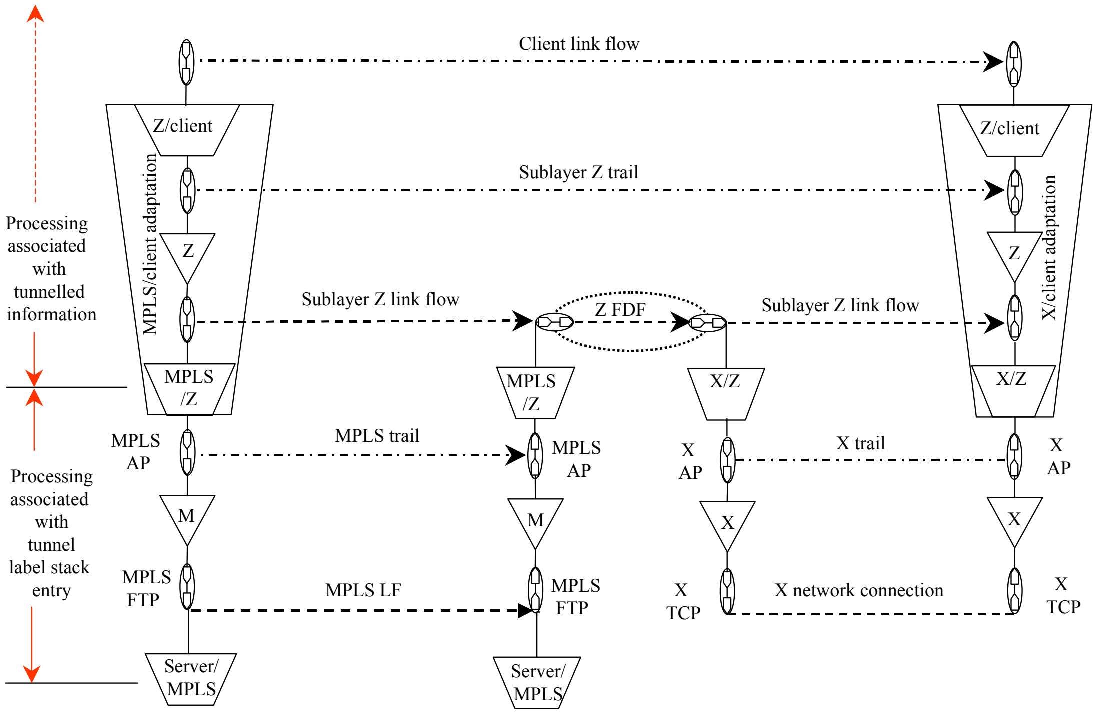


Figure 28/G.8110/Y.1370 - Reference diagram for Uniform and Short Pipe models with PHP

For the server/MPLS adaptation the following cases apply:

- the server layer network is MPLS and the adaptation function is of the form MPLS/MPLS adaptation. In this case the processes of the MPLS/MPLS sink function are dependent on the nature of the MPLS server tunnel mode. The label stack mechanism allows tunnelling to nest to any depth. There is also no requirement for consistent tunnelling models across levels so each tunnel may operate in a different tunnelling mode to its client or its server. Note that in the reference diagram the processes associated with the MPLS/client adaptation source function, where the client is MPLS are dependent on the servers tunnelling mode.

- The server layer network is IP. This is not considered further in this Recommendation.
- The server is neither MPLS nor IP. This case is described for each tunnel model using the server/MPLS (server is not MPLS) notation. The server layer network may be either connection oriented or connectionless. In the reference diagram it is shown as connection oriented. Server layer processes associated with the adaptation function are not described.

The X server layer is any valid server layer technology into which the Z CI can be mapped and as such can be either connection oriented or connectionless. In the reference diagram it is shown as connection oriented.

The processes associated with the transport processing functions are the same as for the uniform and short pipe models without PHP, except for MPLS/Z adaptation and the transport processing functions downstream of it. Note that the functions Z/client adaptation, Z flow termination and MPLS/Z adaptation in the MPLS/client adaptation are not described as these are all encapsulated within the containing MPLS/client adaptation.

13.2 MPLS TTL Behaviour

The Time-To-Live (TTL) field can be processed in a number of ways depending on LSP type as described in RFC 3443.

The TTL behaviour for each of the Diff-Serv tunnelling models, uniform, pipe and short pipe is provided in this section by means of tables that describe the TTL processing that occurs in each of the transport processing functions in the appropriate reference diagram.

The following prepended signals, _AI_TTLVALUE and _CI_TTLVALUE, are signals that contain information related to TTL values and are used to describe information flows between transport processing functions as described in the following sections. These signals are present between transport processing functions in equipment but are not transported between equipment over trails or flows. As such they are not described separately from the characteristic information and adapted information that are transported on transport entities. The AI_TTLVALUE and CI_TTLVALUE are not part of MPLS traffic units. They provide transport processing functions in one layer network with information related to the TTL value obtained from another layer network. Depending upon the Diff-Serv model these may or may not be used by a layer network as part of its own TTL processing.

The server/client adaptation functions of Figure 27 are not described in the following tables as they are not necessary to explain behaviour.

13.2.1 Uniform Model without Penultimate Hop Popping

The transport processing functions and processes for the uniform model without penultimate hop popping are described in Table 5.

Table 5/G.8110/Y.1370 - Transport processing functions and Diff-Serv TTL processing in the uniform model without PHP.

Transport Processing Function	TTL Processing
MPLS/client_A_So (client is IP)	Generate the MPLS_AI_TTLVALUE from the received IP_CI_TTLVALUE
MPLS/client_A_So (client is MPLS)	Generate MPLS_AI_TTLVALUE from the received MPLS_CI_TTLVALUE
MPLS_FT_So	The received MPLS_AI_TTLVALUE is copied into the TTL field of the MPLS_CI traffic unit Generate the MPLS_CI_TTLVALUE from the TTL field of the MPLS traffic unit
Server/MPLS_A_So (server is not MPLS)	Terminate the MPLS_CI_TTLVALUE and do not process it any further.
Server/MPLS_A_So (server is MPLS)	Generate MPLS_AI_TTLVALUE from the received MPLS_CI_TTLVALUE
Server/MPLS_A_Sk (server is not MPLS)	Decrement the TTL field in the MPLS shim header by 1. If the TTL ≤ 0 then the packet is not forwarded. If the packet is forwarded then use the decremented TTL value to generate MPLS_CI_TTLVALUE
Server/MPLS_A_Sk (server is MPLS)	If the server layer is running in either pipe or short pipe mode then: Terminate the received MPLS_AI_TTLVALUE and do not process it any further Decrement TTL field in the MPLS_AI traffic unit by 1. If TTL ≤ 0 then packet is not forwarded. Generate the MPLS_CI_TTLVALUE from the decremented TTL field in the MPLS_AI traffic unit If the server layer is running in uniform mode then: Overwrite the TTL field in the MPLS_AI traffic unit with the received MPLS_AI_TTLVALUE Generate MPLS_CI_TTLVALUE from MPLS_AI_TTLVALUE
MPLS_FT_Sk	Terminate the received MPLS_CI_TTLVALUE and do not process any further. Remove the TTL field from MPLS_CI traffic unit and generate a copy of it as MPLS_AI_TTLVALUE
MPLS/client_A_Sk (client is MPLS)	Overwrite the TTL field in the MPLS_AI traffic unit with the received MPLS_AI_TTLVALUE Generate MPLS_CI_TTLVALUE from MPLS_AI_TTLVALUE
MPLS/client_A_Sk (client is IP)	Overwrite the TTL field in the IP header with the received MPLS_AI_TTLVALUE Generate IP_CI_TTLVALUE from MPLS_AI_TTLVALUE

13.2.2 Pipe and Short Pipe Models without Penultimate Hop Popping

The transport processing functions and processes for the pipe and short pipe models, without penultimate hop popping, are described in Table 6.

Table 6/G.8110/Y.1370 - Transport processing functions and Diff-Serv TTL processing for the pipe and short pipe models without PHP.

Transport Processing Function	**TTL Processing**
MPLS/client_A_So (client is IP)	Generate the MPLS_AI_TTLVALUE from the received IP_CI_TTLVALUE
MPLS/client_A_So (client is MPLS)	Generate the MPLS_AI_TTLVALUE from the received MPLS_CI_TTLVALUE
MPLS_FT_So	Terminate the received MPLS_AI_TTLVALUE and do not process any further. The TTL value in the MPLS_CI traffic unit is set administratively to a value less than or equal to 255. Generate MPLS_CI_TTLVALUE from the administratively set TTL value.
Server/MPLS_A_So (server is not MPLS)	Terminate the received MPLS_CI_TTLVALUE and do not process any further.
Server/MPLS_A_So (server is MPLS)	Generate the MPLS_AI_TTLVALUE from the received MPLS_CI_TTLVALUE
Server/MPLS_A_Sk (server is not MPLS)	Decrement the TTL field in the MPLS_AI traffic unit by 1. If the TTL ≤ 0 then the packet is not forwarded. If the packet is forwarded then use the decremented TTL value to generate MPLS_CI_TTLVALUE
Server/MPLS_A_Sk (server is MPLS)	If the server layer is running in either pipe or short pipe mode then: Terminate the received MPLS_AI_TTLVALUE and do not process it any further Decrement TTL field in the MPLS_AI traffic unit by 1. If TTL ≤ 0 then packet is not forwarded. Generate the MPLS_CI_TTLVALUE from the decremented TTL field in the MPLS_AI traffic unit If the server layer is running in uniform mode then: Overwrite the TTL field in the MPLS_AI traffic unit with the received MPLS_AI_TTLVALUE Generate MPLS_CI_TTLVALUE from MPLS_AI_TTLVALUE
MPLS_FT_Sk	Terminate the received MPLS_CI_TTLVALUE and do not process any further. Remove the TTL field from MPLS_CI traffic unit and generate a copy of it as MPLS_AI_TTLVALUE
MPLS/client_A_Sk (client is MPLS)	Terminate the received MPLS_AI_TTLVALUE and do not process any further Decrement TTL field in the MPLS_AI traffic unit by 1. If TTL ≤ 0 then packet is not forwarded. Generate the MPLS_CI_TTLVALUE from the decremented TTL field in the MPLS traffic unit

Transport Processing Function	TTL Processing
MPLS/client_A_Sk (client is IP)	Terminate the received MPLS_AI_TTLVALUE and do not process any further Decrement TTL field in the IP header by 1. If TTL ≤ 0 then packet is not forwarded. Generate the IP_CI_TTLVALUE from the decremented TTL field in the IP header

13.2.3 Uniform Model with Penultimate Hop Popping

The transport processing functions and processes for the uniform model with penultimate hop popping, are described in Table 7.

From a network level perspective it is sufficient to describe the processes associated with the MPLS/client and X/client adaptation functions rather than by the functions that they encapsulate.

Table 7/G.8110/Y.1370 - Transport processing functions and Diff-Serv TTL processing for the uniform model with PHP.

Transport Processing Function	TTL Processing
MPLS/client_A_So (client is IP) MPLS/client_A_So (client is MPLS) MPLS_FT_So Server/MPLS_A_So (server is not MPLS) Server/MPLS_A_So (server is MPLS) Server/MPLS_A_Sk (server is not MPLS) Server/MPLS_A_Sk (server is MPLS) MPLS_FT_Sk	TTL processing in these functions is exactly the same as it is the Uniform model without PHP
MPLS/Z_A_Sk Z is equivalent to an IP packet	Overwrite received MPLS_AI_TTLVALUE into the TTL field of the IP header, recalculate the CRC
MPLS/Z_A_Sk Z is equivalent to MPLS label stack entry	Overwrite received MPLS_AI_TTLVALUE into the TTL field of the MPLS shim header
X/Z_A_So	No TTL processing
X_TT_So	
X_TT_Sk	
X/client_A_So (client is MPLS)	Decrement TTL field in the MPLS shim header by 1. If TTL ≤ 0 then packet is not forwarded. Generate the MPLS_CI_TTLVALUE from the decremented TTL field in the MPLS header
X/client_A_So (client is IP)	Decrement TTL field in the IP header by 1. If TTL ≤ 0 then packet is not forwarded. Generate the IP_CI_TTLVALUE from the decremented TTL field in the IP header

13.2.4 Short Pipe Model with Penultimate Hop Popping

The transport processing functions and processes for the uniform model with penultimate hop popping, are described in Table 8.

From a network level perspective it is sufficient to describe the processes associated with the MPLS/client and X/client adaptation functions rather than by the functions that they encapsulate.

Table 8/G.8110/Y.1370 - Transport processing functions and Diff-Serv TTL processing for the short pipe model with PHP.

Transport Processing Function	TTL Processing
MPLS/client_A_So (client is IP) MPLS/client_A_So (client is MPLS) MPLS_FT_So Server/MPLS_A_So (server is not MPLS) Server/MPLS_A_So (server is MPLS) Server/MPLS_A_Sk (server is not MPLS) Server/MPLS_A_Sk (server is MPLS) MPLS_FT_Sk	TTL processing in these functions is exactly the same as it is the short pipe model without PHP
MPLS/Z_A_Sk Z is equivalent to an IP packet	MPLS_AI_TTLVALUE is terminated with no further processing. The TTL of the Z traffic unit is unchanged
MPLS/Z_A_Sk Z is equivalent to MPLS label stack entry	
X/Z_A_So	No TTL processing
X_TT_So	
X_TT_Sk	
X/client_A_So (client is MPLS)	Decrement TTL field in the MPLS shim header by 1. If TTL ≤ 0 then packet is not forwarded. Generate the MPLS_CI_TTLVALUE from the decremented TTL field in the MPLS header
X/client_A_So (client is IP)	Decrement TTL field in the IP header by 1. If TTL ≤ 0 then packet is not forwarded. Generate the IP_CI_TTLVALUE from the decremented TTL field in the IP header

13.3 MPLS EXP Behaviour

RFC 3032 describes the EXP field as being reserved for experimental use. RFC 3270 describes the application of the EXP field for MPLS support of Diff-Serv. This Recommendation considers the use of EXP as described in RFC 3270 and other applications are for further study.

The EXP behaviour for each of the Diff-Serv tunnelling models; uniform, pipe and short pipe, is provided in this section by means of diagrams that describe the EXP processing that occurs in each of the transport processing functions in the appropriate reference diagram.

The PHB remarking due to traffic conditioning functions is for further study.

The diagrams use the following conventions:

Incoming PHB is denoted as iPHB and Outgoing PHB is denoted as oPHB.

M represents Diff-Serv information conveyed in the encapsulated header – the “tunnelled Diff-Serv Information”, whilst m represents the Diff-Serv information conveyed in the encapsulating header – the “LSP Diff-Serv information”, as described in section 2.6 of RFC 3270.

M_i or (m_i) represents the synatx coding of the Diff-Serv information in the appropriate MPLS or IP header. In an LSR where changing the EXP value is allowed (as described in section 3.2.1 of RFC 3270) the incoming Diff-Serv information is swapped to outgoing Diff-Serv information M_j (M_j may or may not equal M_i). Where changing the EXP bits is not supported then the incoming Diff-Serv information M_i is copied in the outgoing Diff-Serv information (and is equal to M_i)

NOTE 1 – The figures below assume a non-MPLS server layer for the Server/MPLS source adaptation function. When the server layer is an MPLS tunnel, the behaviour depends on the tunnel model as specified in sections 13.3.1 (uniform model), 13.3.2 (pipe model) and 13.3.3 (short pipe model) for the MPLS/Client source adaptation function. The only exception is applicable when the server layer for the incoming Server/MPLS sink adaptation function is an MPLS tunnel using the pipe model: in this case the Server/MPLS source adaptation function must set the EXP field equal to the incoming value (as specified in section 13.3.2 for the Server/Client source adaptation function).

NOTE 2 – The figures below assume a non-MPLS server layer for the Server/MPLS sink adaptation function.

When the server layer is an MPLS tunnel, the behaviour depends on the tunnel model as specified in sections 13.3.1 (uniform model), 13.3.2 (pipe model) and 13.3.3 (short pipe model) for the MPLS/Client source adaptation function.

13.3.1 Uniform model without penultimate hop popping

The transport processing functions and processes for the uniform model without penultimate hop popping are described in Figure 29.

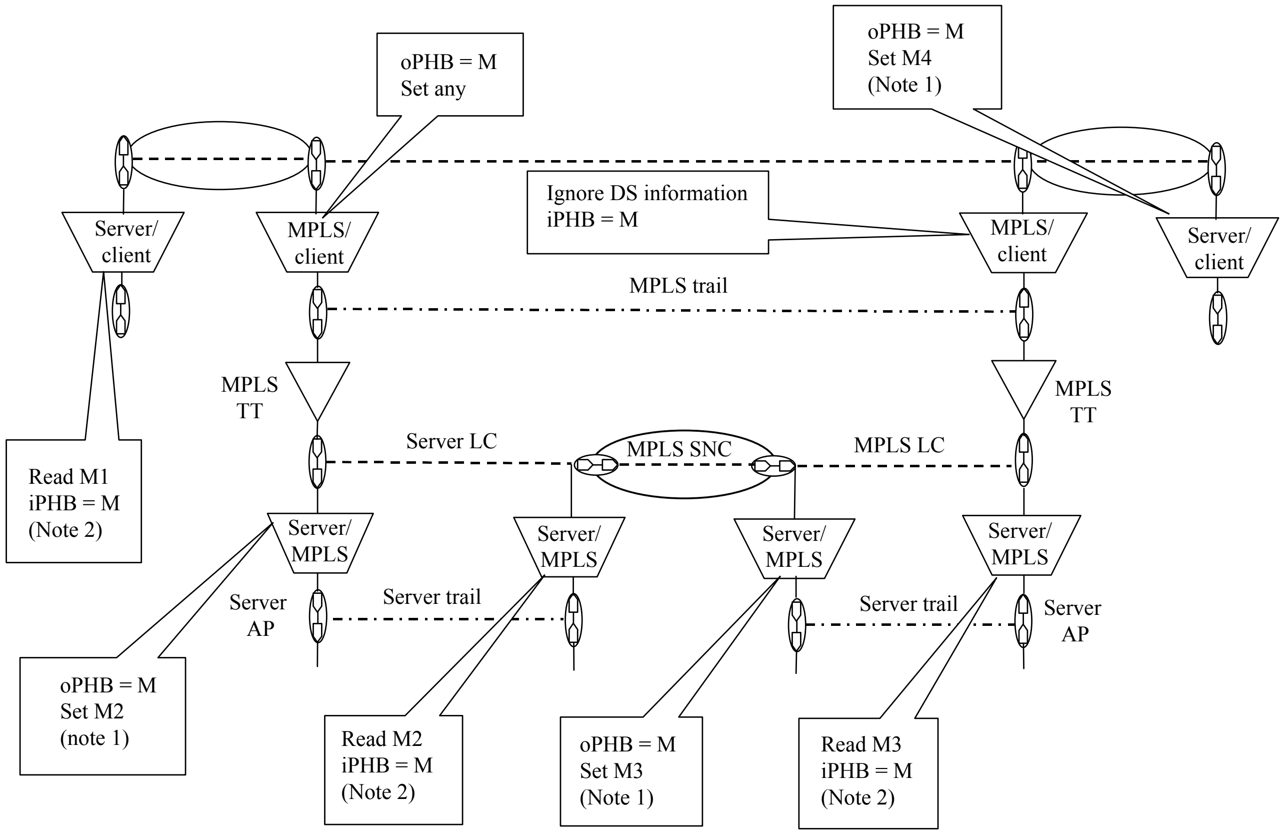


Figure 29/G.8110/Y.1370 - Reference diagram for Uniform model without PHP

13.3.2 Pipe model without penultimate hop popping

The transport processing functions and processes for the pipe model without penultimate hop popping are described in Figure 30.

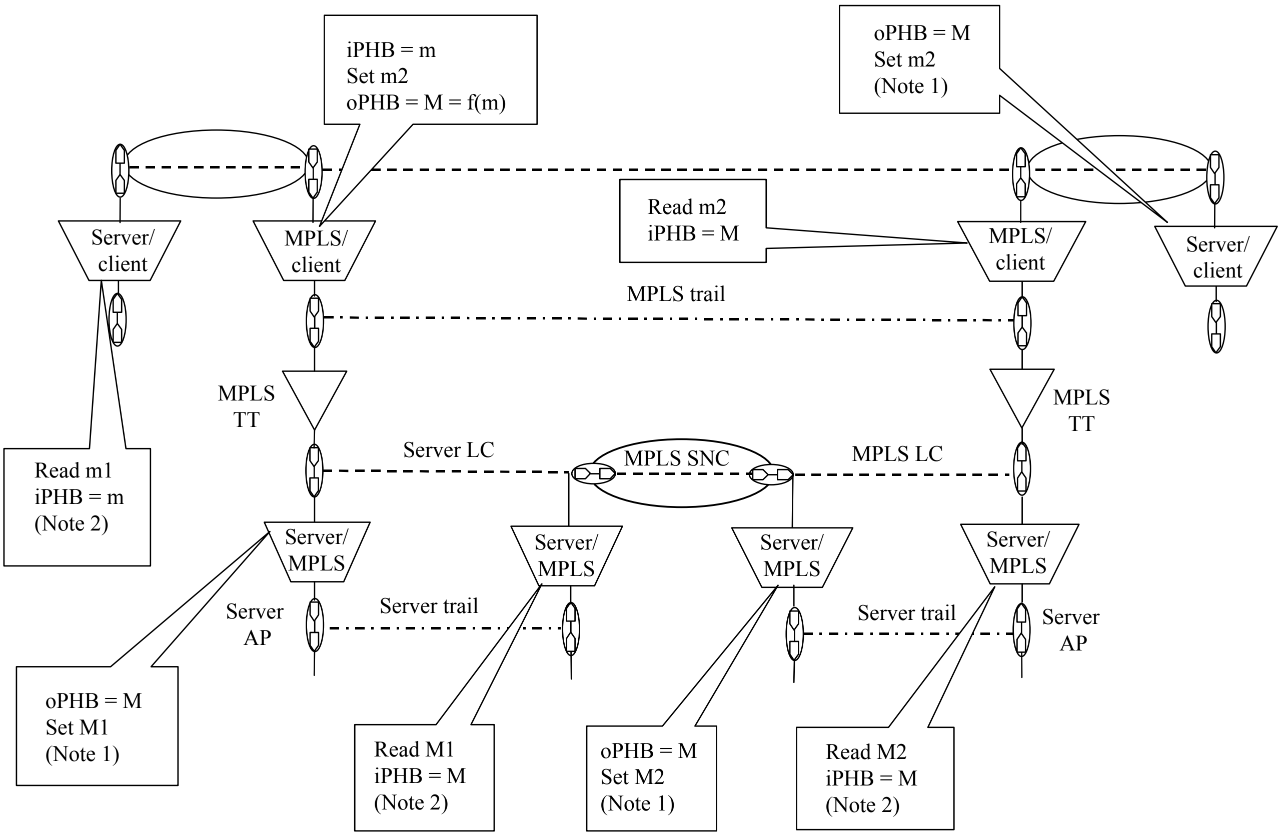


Figure 30/G.8110/Y.1370 - Reference diagram for Pipe model without PHP

13.3.3 Short pipe model without penultimate hop popping

The transport processing functions and processes for the short pipe model without penultimate hop popping are described in Figure 31.

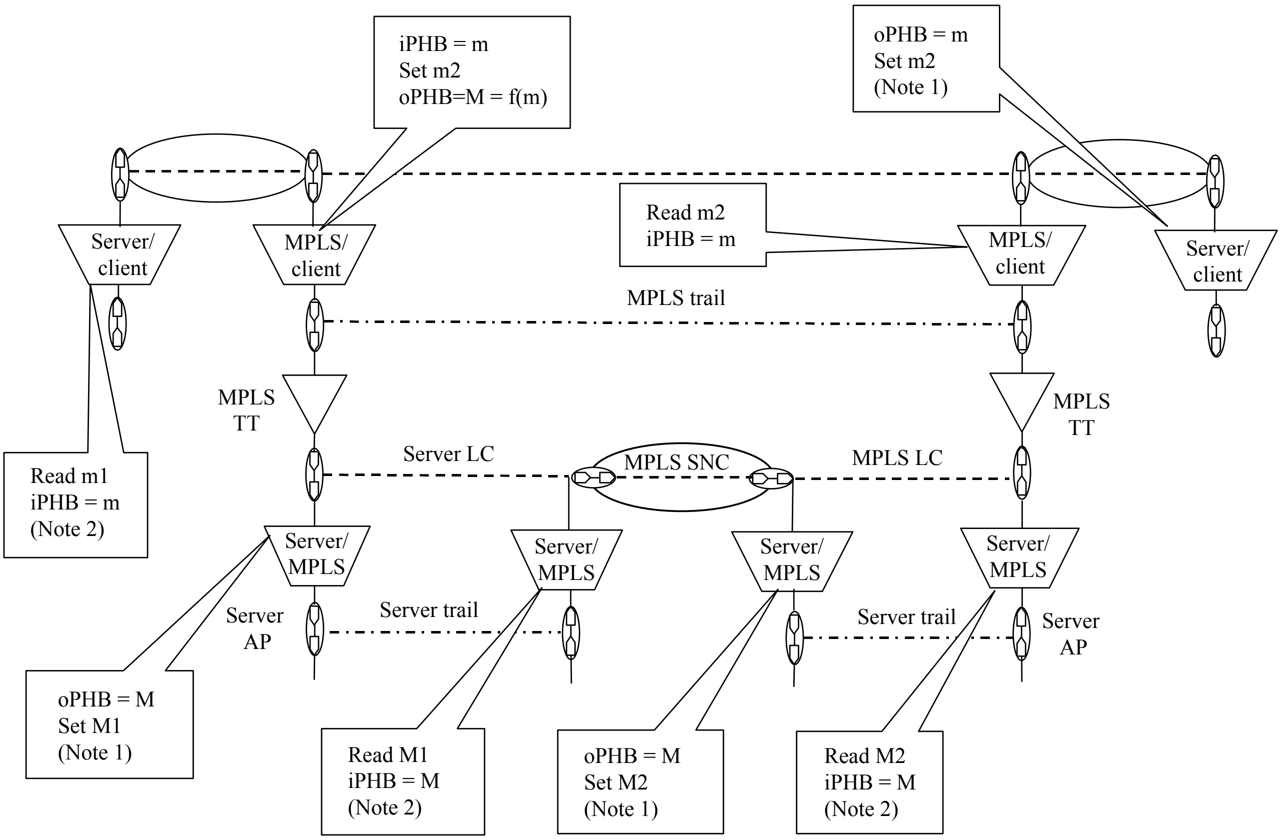


Figure 31/G.8110/Y.1370 - Reference diagram for Short pipe model without PHP

13.3.4 Uniform model with penultimate hop popping

The transport processing functions and processes for the uniform model with penultimate hop popping are described in Figure 32.

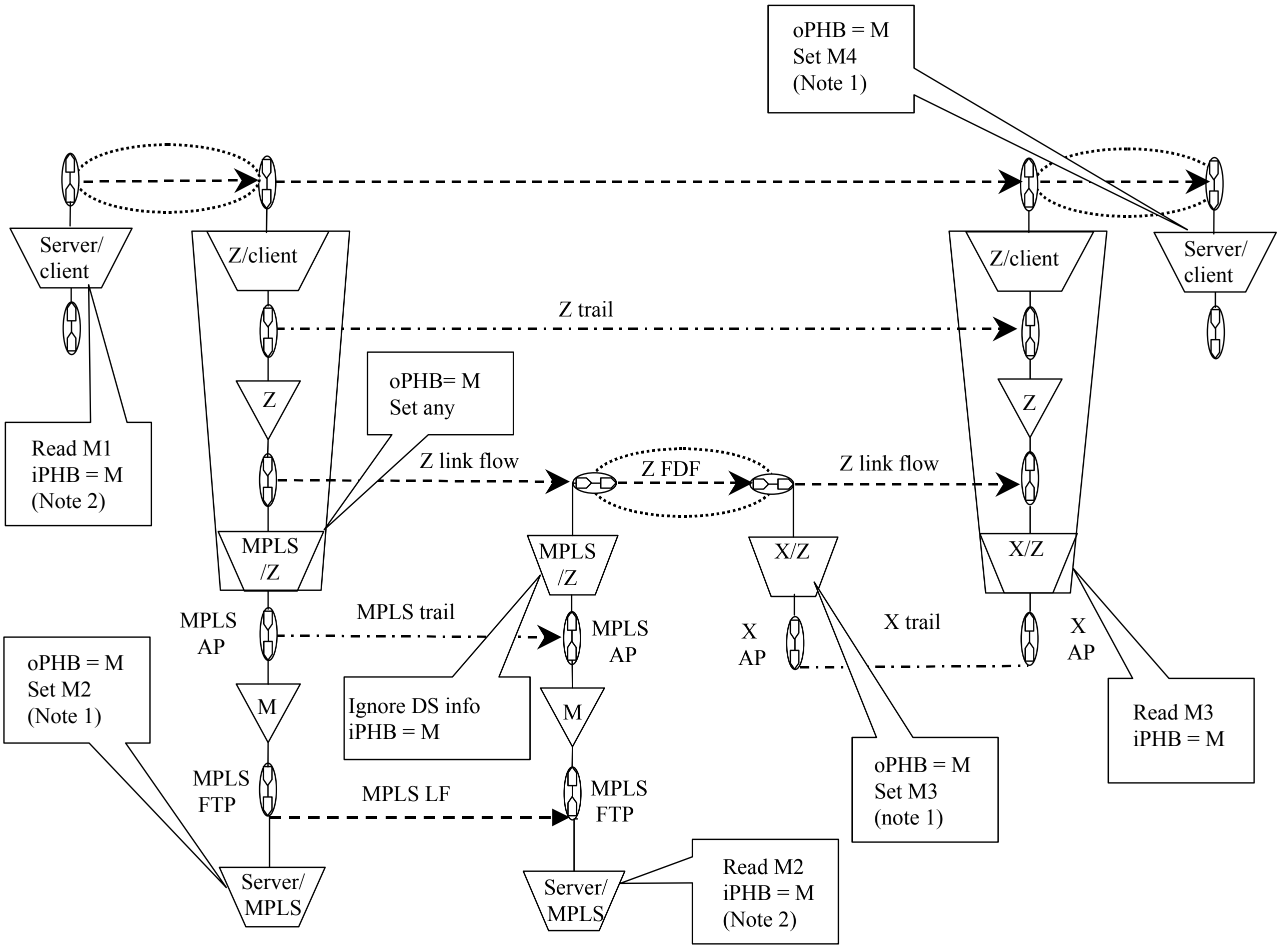


Figure 32/G.8110/Y.1370 - Reference diagram for Uniform model with PHP

13.3.5 Short pipe model with penultimate hop popping

The transport processing functions and processes for the short model with penultimate hop popping are described in Figure 33.

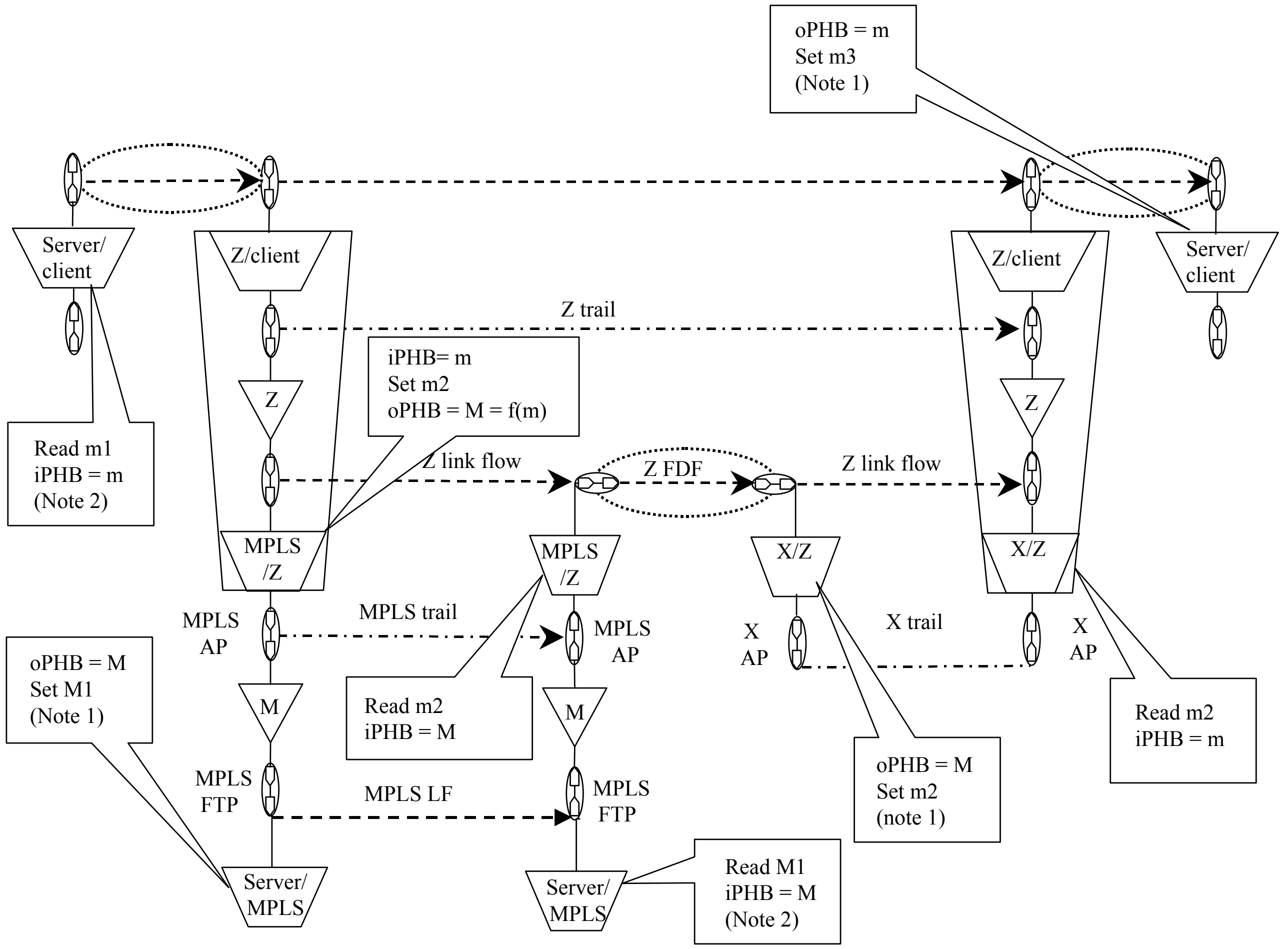


Figure 33/G.8110/Y.1370 - Reference diagram for Short pipe model with PHP

13.4 LSP merging and Diff-Serv support

In the G.809 model E-LSP and L-LSP merging are supported with the following restrictions:

- E-LSPs can only be merged into a single E-LSP if they support exactly the same set of Behaviour Aggregates (BAs).
- L-LSPs can only be merged into a single L-LSP if they support exactly the same Per Hop Behaviour Scheduling Class (PSC).

In the G.805 model neither E-LSP or L-LSP merging are supported.

Annex A

Functional Model for Fragmentation of Packets in an MPLS Network

In an IP network it is possible to receive IP packets that are too large to be transmitted on an outgoing link. To allow the packets to be transferred on the link the IP packets may be fragmented. A similar situation may occur in MPLS where labelled packets are too large for the outgoing link. MPLS however does not provide a process within MPLS itself for doing this, rather it relies upon the fragmentation mechanism of IP to deal with this situation. RFC 3032 "MPLS Label Stack Encoding" describes the processes for dealing with fragmentation of MPLS packets. If DF (Don't Fragment) is not set then the labelled packet may be silently discarded or fragmentation may be attempted. If the DF bit is set the packet must be discarded and an error message sent, according to the processes described in RFC 3032.

The description of the previous paragraph indicates that the adaptation function associated with the MPLS link is not transparent to the content of the information of its client layer networks. By processing information from client layer networks, without terminating the client layer trail, the integrity of the trail of the client is compromised.

To ensure semantic and syntactic consistency of information transfer the adaptation function must perform processes that are the equivalent to going up through the layer networks, (reading information), until the MPLS/IP adaptation function is reached, fragmenting the packets according to the processes in RFC 3032 and then reconstructing the label stack (by prepending the fragments with the same label headers that would have been present if there had been no fragmentation) in the opposite direction.

The functional model for this is provided in Figure A-1. The traversing of the label stack in both directions is encapsulated within the adaptation function.

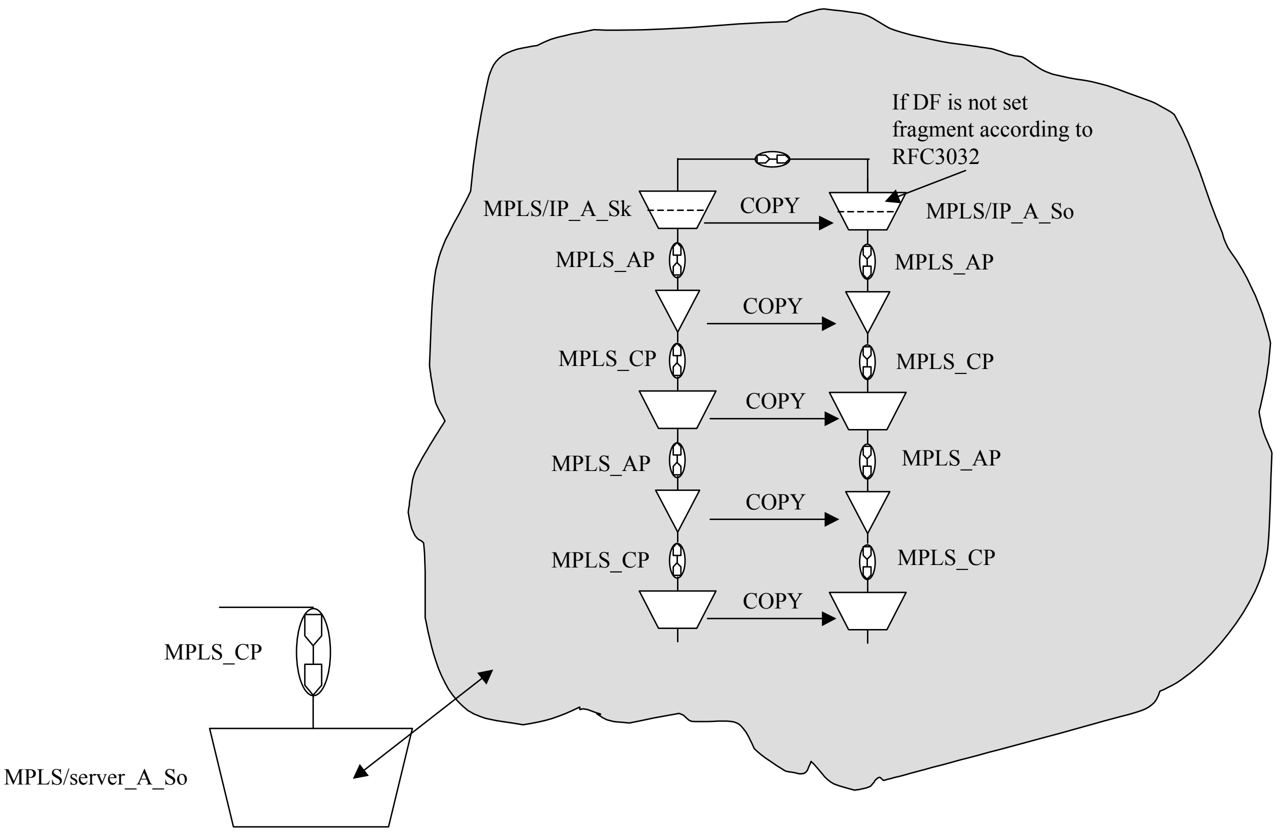


Figure A-1/G.8110/Y.1370 - Fragmentation in MPLS

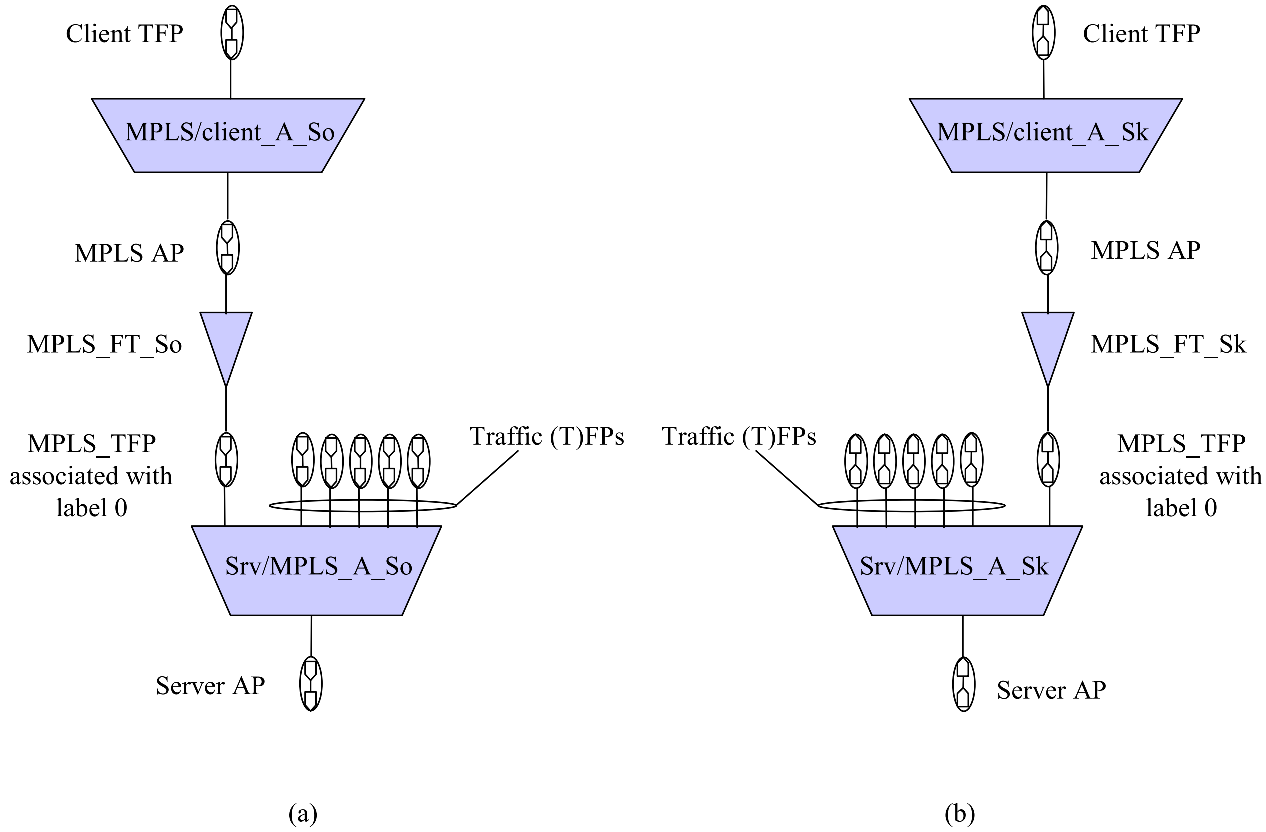
Annex B

Reserved Label Processing

This annex describes the functional models associated with MPLS reserved labels.

Reserved Label – 0: IPv4 Explicit Null

The processing of MPLS traffic units with a label value of 0 by transport processing functions is shown in Figure B.1. The model for the G.805 description is the same, except that the (T)FPs are replaced with (T)CPs, flow termination functions are replaced with trail termination functions, and the flows are replaced with connections.



(NOTE - the server layer may be either MPLS or a non-MPLS server layer. A non-MPLS server is shown in the figure)

Figure B.1/G.8110/Y.1370 – IPv4 Explicit Null Processing
(a) Source transport processing functions
(b) Sink transport processing functions

MPLS traffic units with label 0 are multiplexed by the server/MPLS adaptation source via

- A TFP associated with label 0 for an MPLS layer network described using G.809. Or
- A TCP associated with label 0 for an MPLS layer network described using G.805.

MPLS traffic units with label 0 are demultiplexed by the server/MPLS adaptation sink and directed toward:

- A TFP associated with label 0 for an MPLS layer network described using G.809. Or
- A TCP associated with label 0 for an MPLS layer network described using G.805.

If the client of the MPLS/Client_A_Sk function is IPv4, the S bit is equal to 1, and the bottom of stack has been reached. The packet is then forwarded according to the IPv4 processing contained within the adaptation function. This is a legal operation according to RFC3032.

If the client of the MPLS/Client_A_Sk function is MPLS, the S bit is equal to 0, and the bottom of stack has not been reached. Such a packet is illegal according to RFC3032.

Reserved Label –1: Router Alert Label

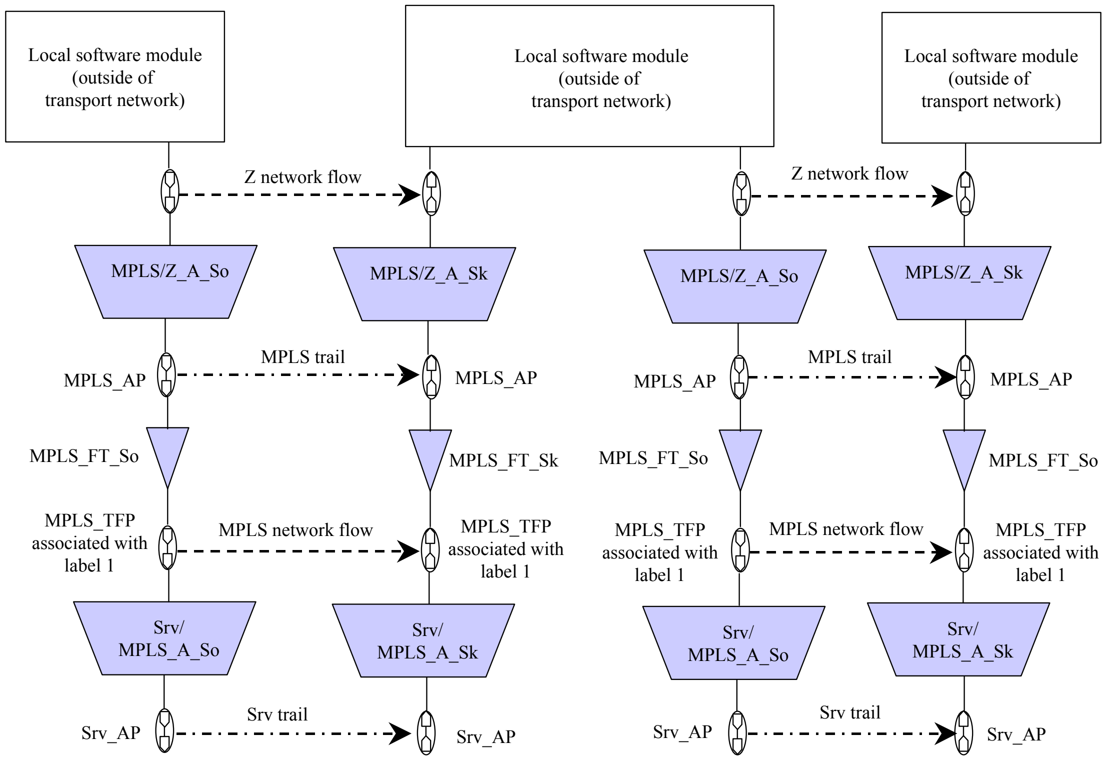
The processing of MPLS traffic units with a label value of 1 by transport processing functions is shown in Figure B.2. The model for the G.805 description is the same, except that the (T)FPs are replaced with (T)CPs, flow termination functions are replaced with trail termination functions, and the flows are replaced with connections.

The Router Alert Label allows a software module in one network element to communicate with a software module in another. The local software module generates an MPLS packet which is presented to the transport network as a label stack entry (corresponding to the characteristic information of the Z network flow) and an additional MPLS header with label value 1.

MPLS traffic units with label 1 are demultiplexed by the server/MPLS adaptation sink and directed toward:

- A TFP associated with label 1 for an MPLS layer network described using G.809. Or
- A TCP associated with label 1 for an MPLS layer network described using G.805.

The label stack entry output from the MPLS/Z_A_Sk is passed to a local software module for processing. This processing is assumed to be outside of the transport network. If after processing the packet is to be forwarded, the forwarding is determined from the label at the top of the label stack presented to the software module. The local software module then presents the transport network with a label stack entry and an additional MPLS header with label value 1 is pushed on.

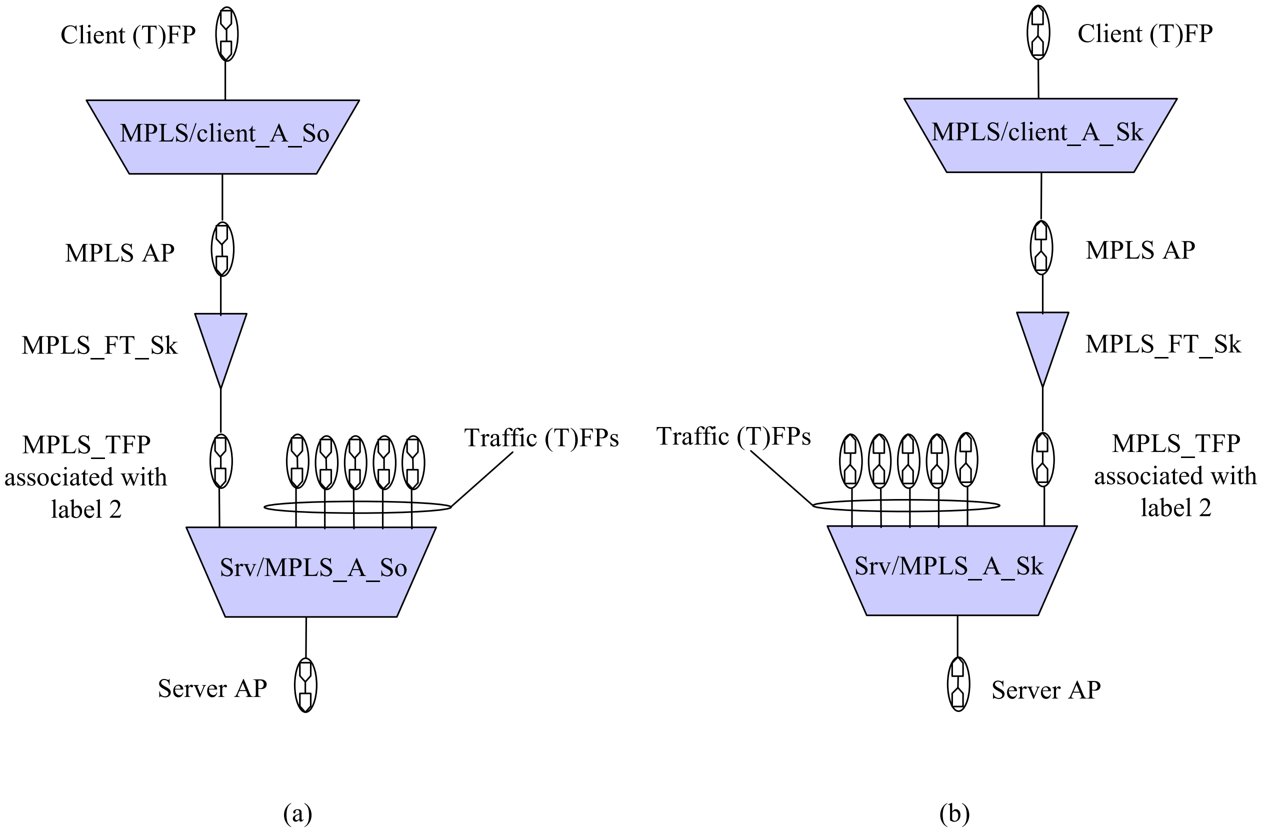


(The CI on the first Z flow need not be the same as that on the second in the figure depending on the processing.)

Figure B.2/G.8110/Y.1370 - Router Alert Processing

Reserved Label- 2: IPv6 Explicit Null

The processing of MPLS traffic units with a label value of 2 by transport processing functions is shown in Figure B.3. The model for the G.805 description is the same except that the (T)FPs are replaced with (T)CPs, flow termination functions are replaced with trail termination functions, and the flows are replaced with connections



(Note that the server layer may be either MPLS or a non-MPLS server layer. A non-MPLS server is shown in the figure)

Figure B.3/G.8110/Y.1370 – IPv6 Explicit Null Processing

(a) Source transport processing functions
(b) Sink transport processing functions

MPLS traffic units with label 2 are multiplexed by the server/MPLS adaptation source via

– A TFP associated with label 2 for an MPLS layer network described using G.809. Or

– A TCP associated with label 2 for an MPLS layer network described using G.805.

MPLS traffic units with label 2 are demultiplexed by the server/MPLS adaptation sink and directed toward:

– A TFP associated with label 2 for an MPLS layer network described using G.809. Or

– A TCP associated with label 2 for an MPLS layer network described using G.805.

If the client of the MPLS/Client_A_Sk function is IPv6, the S bit is equal to 1, and the bottom of stack has been reached. The packet is then forwarded according to the IPv6 processing contained within the adaptation function. This is a legal operation according to RFC3032.

If the client of the MPLS/Client_A_Sk function is MPLS, the S bit is equal to 0, and the bottom of stack has not been reached. Such a packet is illegal according to RFC3032.

Reserved Label –3: Implicit Null

This label value only appears in the control plane and never in the transport plane.

Other reserved label values

For further study.

Annex C

G.809 to G.805 Translation

The description of penultimate hop popping, LSP tunnels and support of the Diff-Serv Architecture in the G.809 model can be applied in the G.805 model with the translations shown in table C.1

Table C.1/G.8110/Y.1370 - Translation between G.809 and G.805

G.809 construct	G.805 construct
MPLS_FT, MPLS_FT_So, MPLS_FT_Sk	MPLS_TT, MPLS_TT_So, MPLS_TT_Sk
MPLS connectionless trail	MPLS trail
MPLS link flow	MPLS link connection
MPLS network flow	MPLS network connection
MPLS flow domain flow	MPLS subnetwork connection
MPLS TFP	MPLS TCP
MPLS FP	MPLS TFP
MPLS FPP link	MPLS link
MPLS flow domain	MPLS subnetwork

Note that a G.809 flow point pool has no defined counterpart in G.805. However, a flow point pool is analogous to the set of connection points associated with a link in G.805.

Annex D

MPLS and IP Multiplexing

Where MPLS is used to support IP traffic the server layer technology that supports an MPLS hierarchy may also be used to transport IP traffic. The server layer must therefore provide an adaptation function that supports multiple clients. An example of such an adaptation function, in the form of a compound adaptation source function, is shown in Figure D.1. The characteristic information that is presented to the adaptation function can be one, or more, of the following:

(A) Penultimate hop popped traffic where the characteristic information corresponds to an IP packet.

(B) Penultimate hop popped traffic where the characteristic information corresponds to a label stack entry.

(C) MPLS characteristic information

(D) IP characteristic information

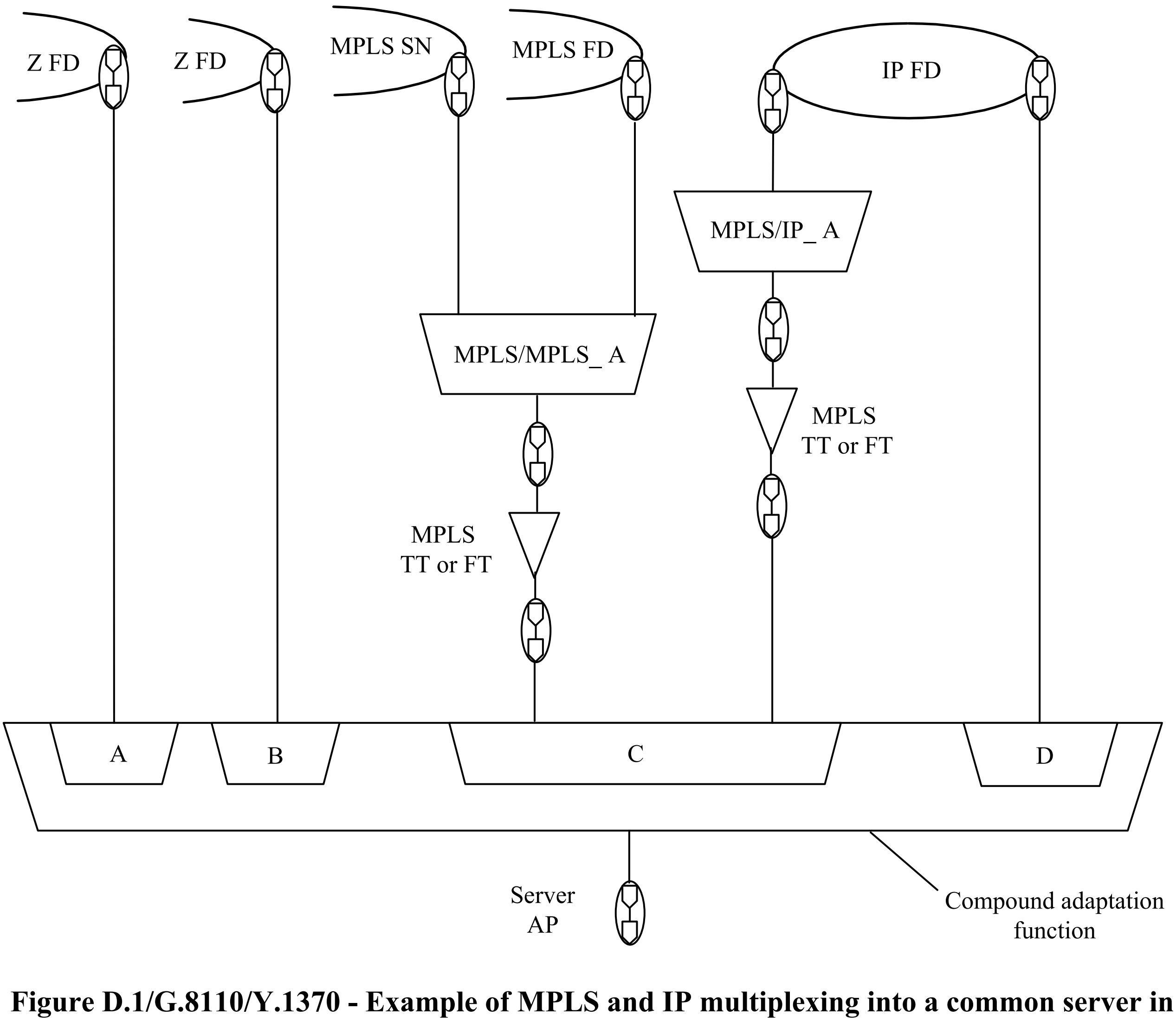


Figure D.1/G.8110/Y.1370 - Example of MPLS and IP multiplexing into a common server in the source direction

An example of such an adaptation function, in the form of a compound adaptation sink function, is shown in Figure D.2. The characteristic information that is presented to the adaptation function can be one, or more, of the following:

(C) MPLS characteristic information.

(D) IP characteristic information.

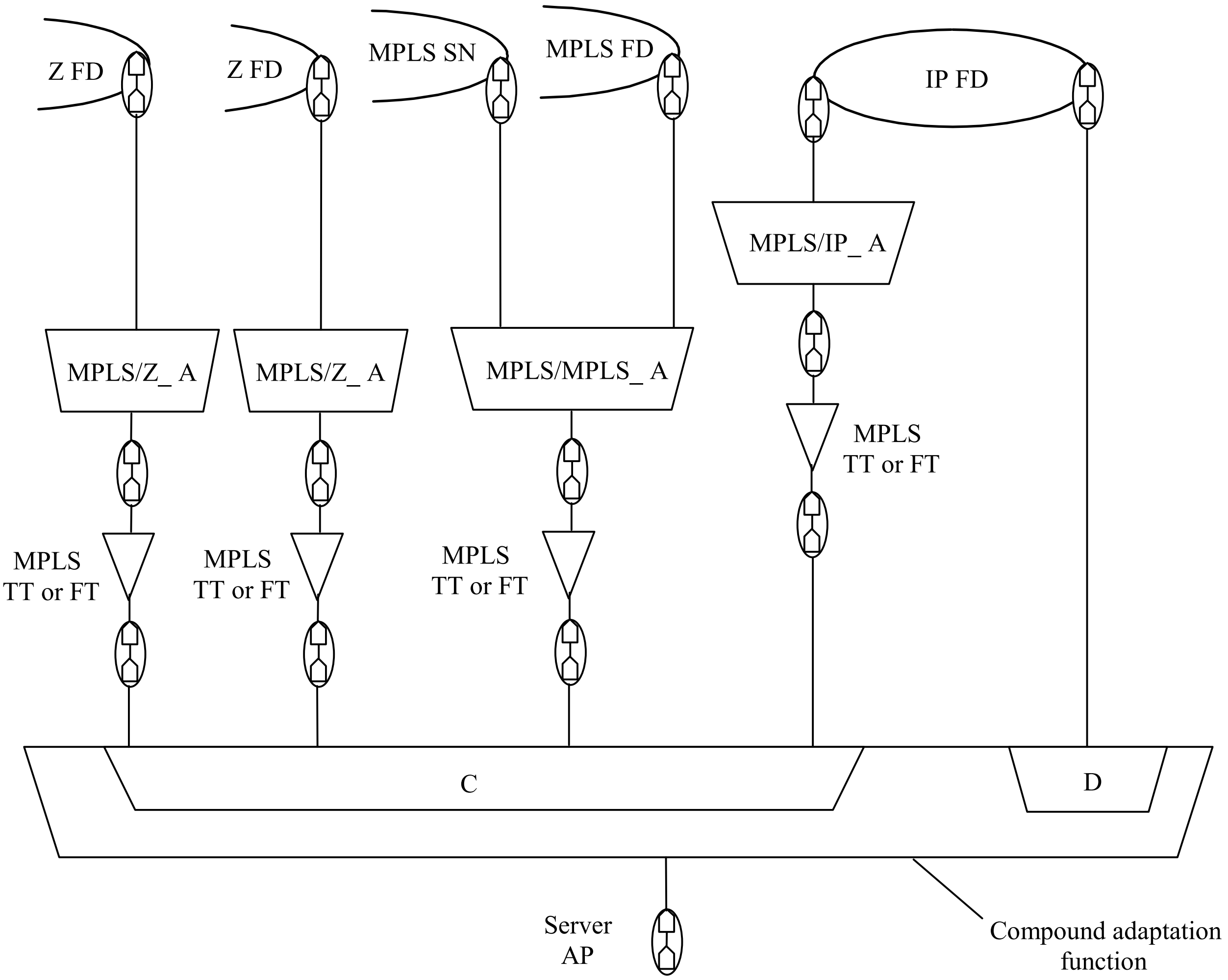


Figure D.2/G.8110/Y.1370 - Example of MPLS and IP demultiplexing in the sink direction

Note that when drawing a network diagram both (C) and (D) can, if necessary, be expanded to show substructure that allows any Z flows and trails associated with PHP to be shown. Such an expansion does not change the functionality of either (C) or (D).

Appendix I

Functional Model for describing the use of ECMP in MPLS networks

Equal Cost Multi Path (ECMP) is an unspecified mechanism that allows all of the members of a set of equal cost paths between a source node and destination node to be used.

Although there are no standardised mechanisms common means of implementing it include:

- Random selection of an outgoing link on a per packet basis. This can cause out-of-order packets
- Round robin selection of an outgoing link on a per packet basis. This can cause out-of-order packets
- Flow-based selection using hashing on fields in the underlying packet transported in MPLS. This preserves packet order for the flow concerned.
- Flow based selection based on hashing of underlying labels at a lower level in the label stack

ECMP implementations often limit the number of equal cost multi paths that may be supported and this can if necessary be set independently to the number of next hop nodes.

An example is provided in Figure I.1. The ECMP mechanism at A identifies two equal cost paths via B and C toward G. Similarly the ECMP mechanism at B identifies two equal cost paths to G via D and E. The traffic is then routed as shown. It should be noted that ECMP implementations often limit the number of equal cost multi paths that may be supported.

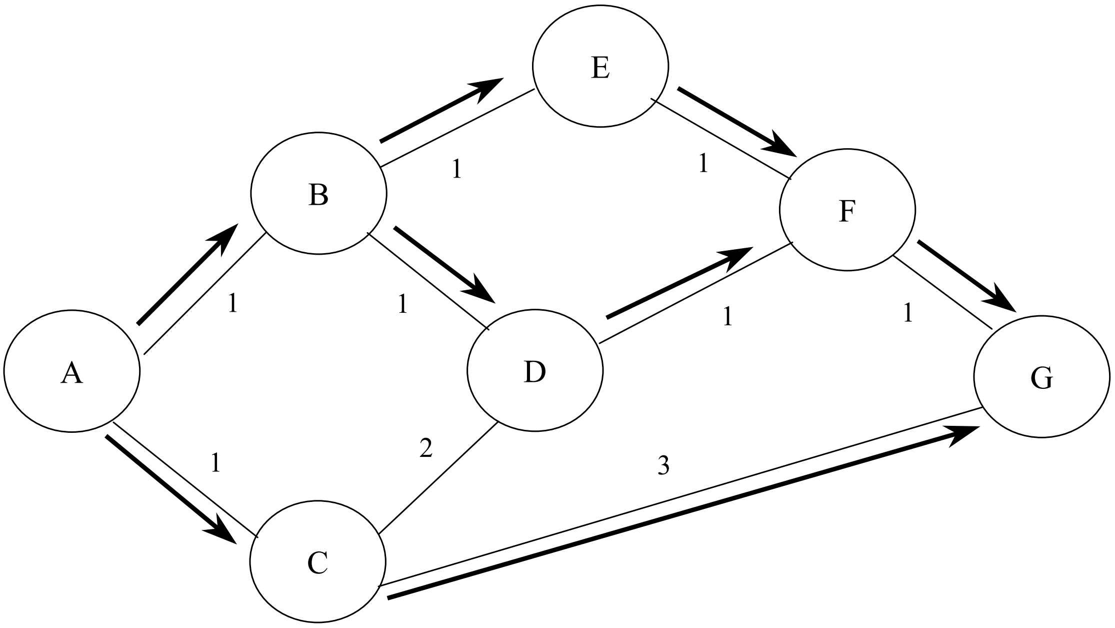


Figure I.1/G.8110/Y.1370 - Load Balancing with ECMP

As ECMP only uses alternate routes with equal costs it can used for coping with parallel links between two nodes in a network.

ECMP as a Process associated with an Adaptation Function

Round robin forwarding needs no detailed explanation at the network level. The appropriate output link is chosen according to the algorithm employed and the packet of concern forwarded as appropriate.

An ECMP hash process takes place within the MPLS/MPLS_A_Sk or Server/MPLS_A_Sk functions. For this to occur the adaptation function cannot be transparent to the content of the information of its client layer networks.

To ensure semantic and syntactic consistency of information transfer the adaptation function must perform processes that are the equivalent to going up through the layer networks, (reading information), until the appropriate adaptation function is reached so that the appropriate field(s) are hashed. This is achieved by copying the MPLS labelled packet that is to be forwarded and then reading through the fields of the copied packet until the appropriate point is reached, as illustrated in Figure I.2. The original packet is then forwarded as required.

The Server/MPLS adaptation source or MPLS/MPLS adaptation source functions are not involved in the ECMP process. They simply assign the appropriate label to packets according to the flow point used to enter the adaptation function.

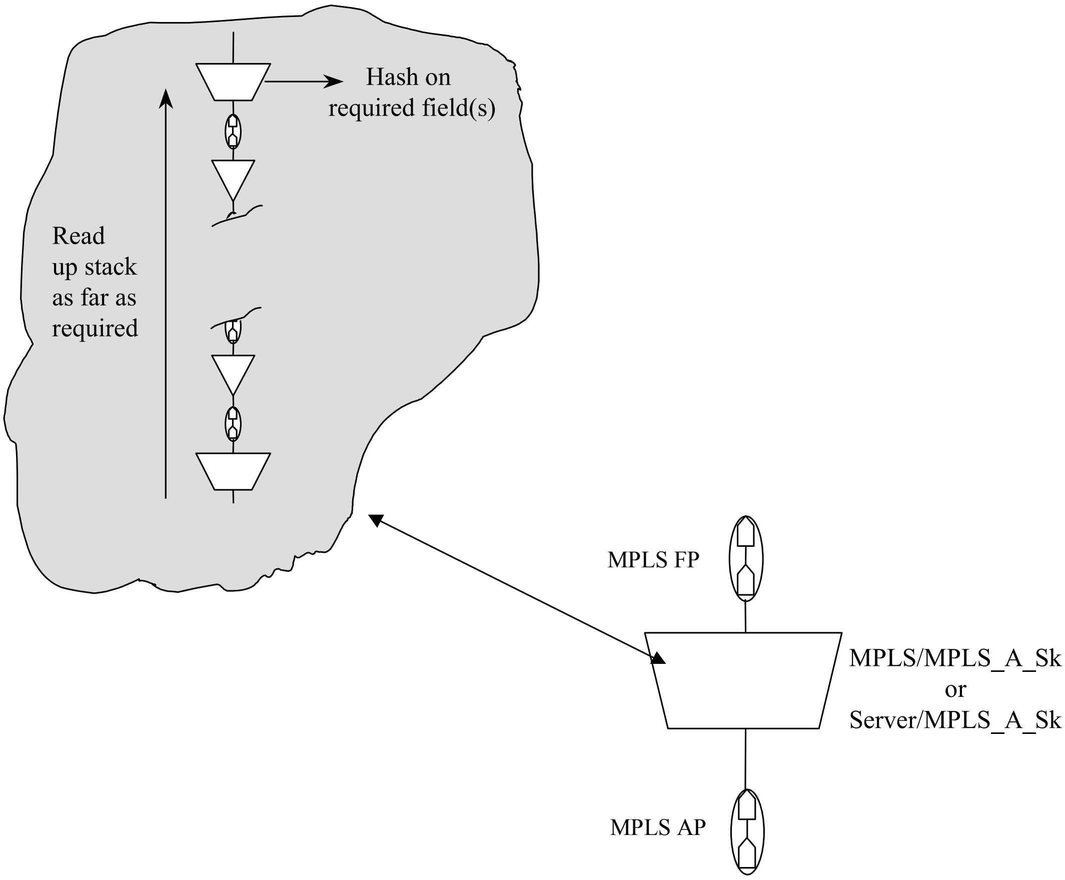


Figure I.2/G.8110/Y.1370 - ECMP Processing

Diagrammatic Convention for Illustrating ECMP

In the absence of ECMP the flow shown in Figure I.3(a) represents a point-to-multipoint flow, where the information at the ingress flow point is copied to both the output flow points. The information flowing through I1, E1 and E2 is therefore the same. There is no load balancing. For a multipoint-to-point flow as illustrated in Figure I.3(b) the flows at I1 and I2 are aggregated (multiplexed) at E1.

These flows are represented using the diagrammatic conventions of G.809.

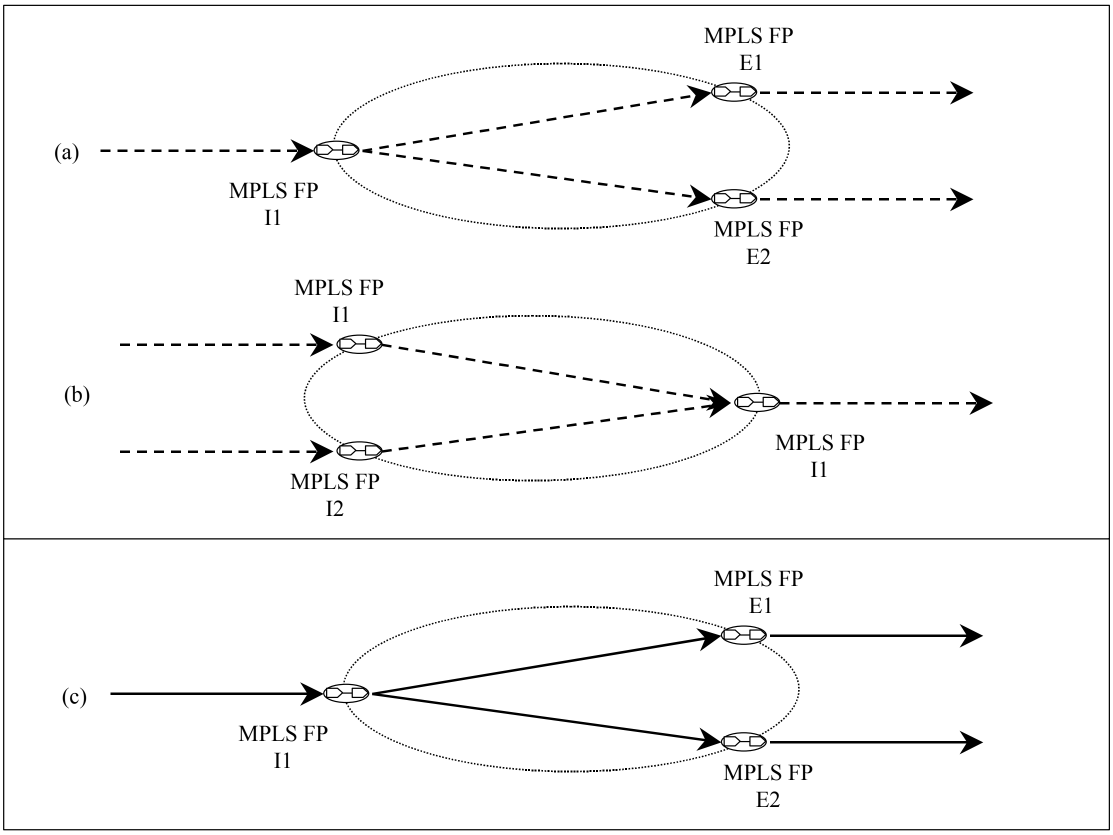


Figure I.3/G.8810/Y.1370 - Modelling flows without ECMP (a) point-to-multi-pont, (b) multipoint-to-point and with ECMP (c)

In a connectionless network where each packet has both a source address and a destination address, e.g. an IP network, the flow entering I1 can be described, or designated, as an aggregation of tuples that include the source and destination address whilst the flows I1-E1 and I1-E2 can be described by means of sub-sets of the tuples present at the ingress of I1. When the packets in a flow are label based the forwarding is such that every packet entering a flow domain via a particular flow point is forwarded across the flow domain in exactly the same way

When ECMP is present the MPLS traffic units associated with a flow point are forwarded based on information other than the label associated with the adaptation function. The information flowing between I1 and E1 in Figure I.3(a) is now no longer the same as the information flowing between I1 and E2. The flow arriving at I1 is demultiplexed into smaller flows based on the ECMP mechanism employed.

To distinguish between flows that are subject to ECMP from those that are not, ECMP effected flows are illustrated by means of a solid arrow as shown in Figure I.3(c).

ECMP in an MPLS network described using G.809

Layer networks that contain LSPs that have been setup using Label Distribution Protocol (LDP) can be modelled using G.809 as described in section 7 of this Recommendation. When LDP is used without ECMP the traffic will not be split. When ECMP is used with LDP then traffic splitting will occur as shown by example in Figure I.4.

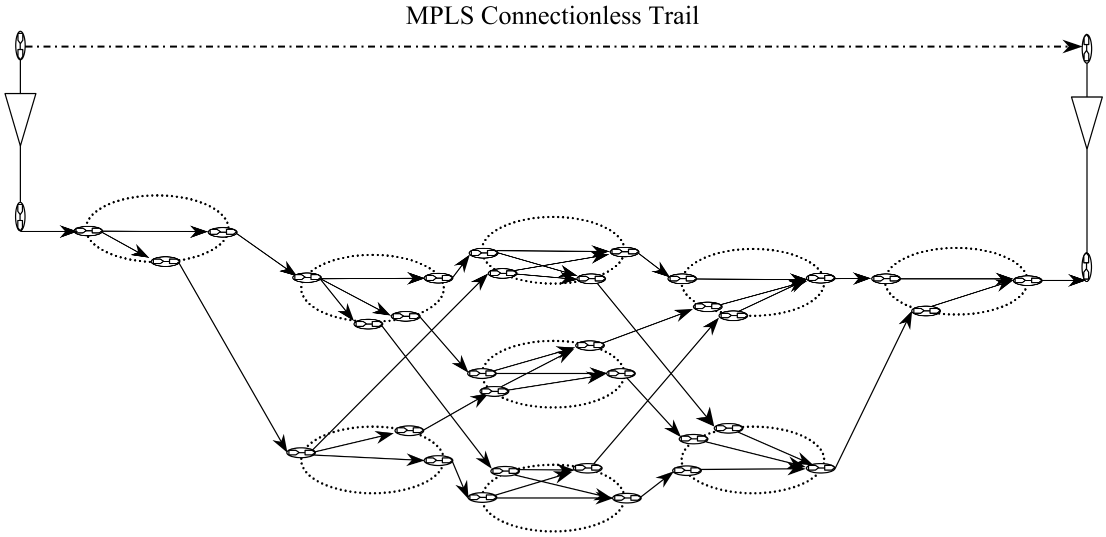


Figure I.4/ G.8110/Y.1370 - Example of an ECMP based LSP

Note that in this example each flow domain has ECMP activated. The effect of ECMP can be seen as inverse multiplexing of the client link.

This process can be repeated by means of the client/server relationship where a link flow in the client is supported by a connectionless trail in the server layer network. However, there is no requirement for a server layer using ECMP to deliver traffic to a single flow point on a single destination flow domain – two separate flow points on the same flow domain can also be supported. This is illustrated in Figure I.5. The result is the creation of dynamic links that respond to the service offered by the server.

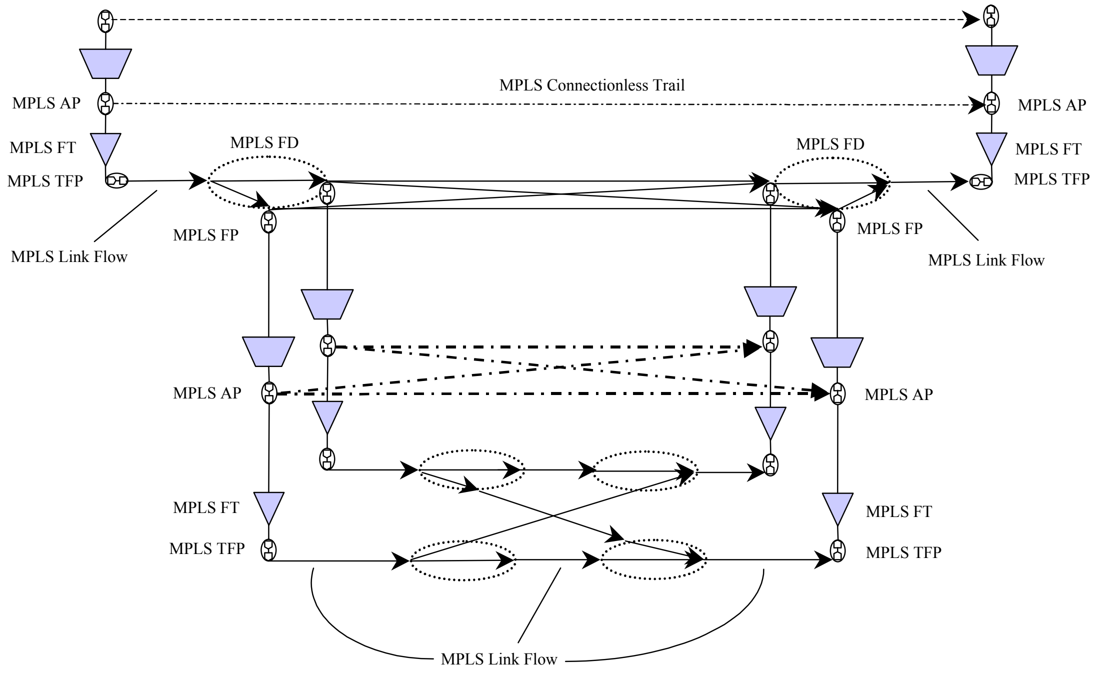


Figure I.5/G.8110/Y.1370 - Example of the effect of ECMP in an MPLS G.809 Hierarchy

This behaviour can be understood by utilising the fact that the MPLS connectionless trail acts at a per packet basis. The effect of ECMP at the lowest layer is to change the relationship between the source and sink of the connectionless trail. The connectionless trail is now alternating between two sink access points and this is driven by the ECMP process. For any particular packet there is only one source access point and one sink access point. For any particular response to the ECMP process all packets with the same response have a trail association with the same sink. The dynamic association between the source and sink of the trail drives a dynamic response in the client layer. This results in a dynamic link that is created between flow points in the client layer network. This link is created in response to a server layer process – the service offered by the trail.

ECMP in an MPLS network described using G.805

In an MPLS layer network where connections are setup using RSVP-TE, the use of ECMP can be considered in two ways:

– Where there is no LSP hierarchy. In this case if more than one LSP is configured to the same destination with equal cost the ECMP is enabled prior to the LSPs by the client layer network, which then distributes traffic between the LSPs as appropriate. As such there is no splitting within such an LSP.

– Where an LSP hierarchy is present. This is for further study.