Multilayer Amendment to E-NNI 2.0 – Common Part

OIF-ENNI-ML-AM-01.0

April 12, 2013

Implementation Agreement created and approved
by the Optical Internetworking Forum
www.oiforum.com

The OIF is an international non profit organization with over 90 member companies, including the world's leading carriers and vendors. Being an industry group uniting representatives of the data and optical worlds, OIF's purpose is to accelerate the deployment of interoperable, cost-effective and robust optical internetworks and their associated technologies. Optical internetworks are data networks composed of routers and data switches interconnected by optical networking elements.

With the goal of promoting worldwide compatibility of optical internetworking products, the OIF actively supports and extends the work of national and international standards bodies. Working relationships or formal liaisons have been established with IEEE 802.1, IEEE 802.3ba, IETF, IP-MPLS Forum, IPv6 Forum, ITU-T SG13, ITU-T SG15, MEF, ATIS-OPTXS,ATIS-TMOC, TMF and the XFP MSA Group.

For additional information contact:
The Optical Internetworking Forum, 48377 Fremont Blvd.,
Suite 117, Fremont, CA 94538
510-492-4040 Φ info@oiforum.com

www.oiforum.com

Working Group: **Architecture and Signaling**

TITLE: **Multilayer Amendment to E-NNI 2.0 – Common Part**

SOURCE: **TECHNICAL EDITOR**

Evelyne Roch
Huawei Technologies
303 Terry Fox Drive Suite 400
Ottawa, ON K2K 3J1
Canada
Phone: + 1 613 595 1900 x1612
Email: evelyne.roch@huawei.com

WORKING GROUP CHAIR

Remi Theillaud
Marben Products
176 rue Jean Jaures
92800 Puteaux
France
Phone: +33 (1) 79 62 10 22
Email: remi.theillaud@marben-products.com

ABSTRACT: This Implementation Agreement defines Generic Signaling and Routing extensions to OIF E-NNI 2.0 in support of multilayer networks.

1 Table of Contents

2 List of Figures

3 List of Tables

4 Introduction

This document specifies the content and operation of the E-NNI signaling and routing protocol across multilayer networks. It is an amendment to the E-NNI 2.0 and reuses the terms and concepts described in [OIF-E-NNI-Sig-02.0] and [OIF-ENNI-OSPF-02.0].

The ASON architecture [G8080] describes control plane functions on layer networks [G800]. Control plane functions are used to create calls and connections. Most control plane functions are confined to a single layer. Functions that require interactions between layers are described in this document.

It should be emphasized that the ASON architecture supports both single and multiple layer scenarios. Multilayer scenarios include network elements that support more than a single layer, a layer network that supports virtual concatenation and its server layers, transport services that exist where the client layer has no resources in the subnetwork except at its edges, etc. For example, at the edge of a multilayer transport network, a network element may support client layer networks that are not directly supported in the core of a multilayer transport network. The client Characteristic Information (CI) could be adapted, possibly multiple times, onto server layer connections. Such scenarios are supported by the client-layer Network Call Controller (NCC), which has an interlayer interface that enables it to have a relationship with a server-layer NCC. Specifically, the interfaces between NCCs in different layer networks enable an association between calls in a client/server layer relationship, which can recurse with the [G800] layers to mirror a set of “stacked” adaptations. For each switched layer in the stack, an NCC at that layer is created. The decision to use an interlayer NCC interface is driven by policy, as there may be a choice regarding which server layers to use. This is elaborated in the following example taken from [G8080], illustrating the case of two Ethernet clients attached to a common VC-3 network that does not support Ethernet switching. Consider the case of a call for a 40Mbit/s Ethernet service requested over a UNI, shown in

Figure 1. To carry the Ethernet CI, a VC-3 connection is created. The decision by the NCC_{MAC} to make a call to the corresponding $NCC_{VC\text{-}3}$ is driven by operator policy. Both layers are shown with only the VC-3 layer having a network connection. Once the VC-3 connection is established, the Ethernet MAC link connection between the two NCC_{MAC} comes into existence.

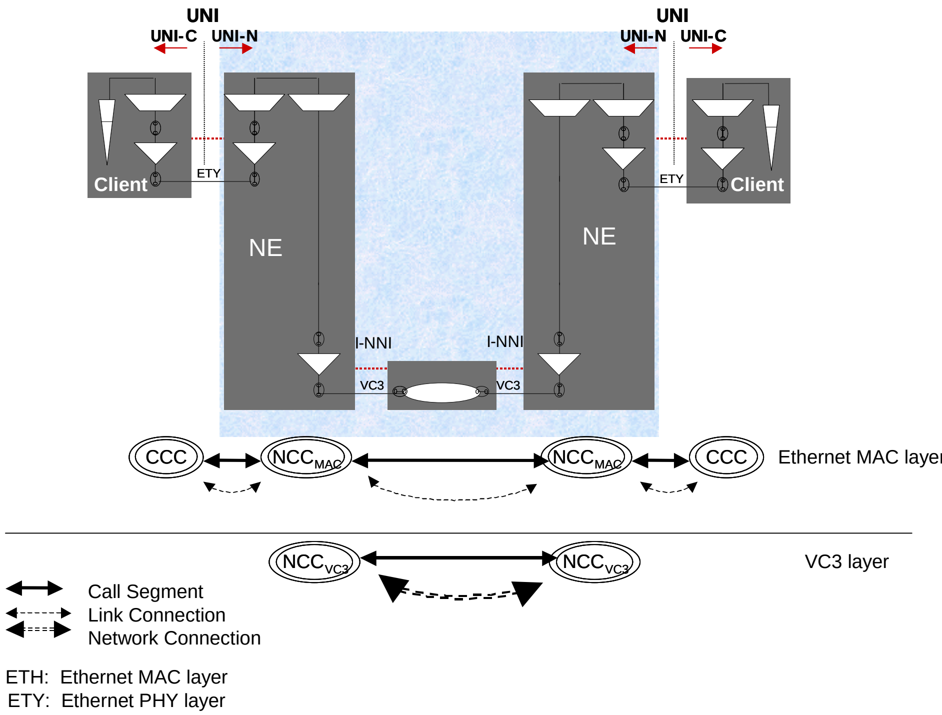

Figure 1: Multilayer Example with Ethernet Client over VC-3 Network.

The E-NNI 2.0 Signaling and Routing specifications [OIF-E-NNI-Sig-02.0] [OIF-ENNI-OSPF-02.0] apply to SONET/SDH, OTN (ODUk) [G709] (2003-03) and Ethernet connection services in support of UNI 2.0 features. This Amendment describes the support of interlayer call invocations to enable a server layer to provide services to a client layer across E-NNI interfaces. It covers the multilayer architecture and reference configurations, signaling, routing (including signaling communications), discovery, reachability and address resolution.

4.1 Problem Statement

The advent of the automatic switched optical network has necessitated the development of interoperable procedures for requesting and establishing dynamic connection services across heterogeneous networks. The development of such procedures requires the definition of:

- Control domains and associated reference points (E-NNI, I-NNI, UNI)
- Services offered by the optical transport network across control domains
- Signaling protocols used to invoke the services across E-NNI interfaces
- Mechanisms used to transport signaling messages
- Routing protocols used to distribute topology across E-NNI interfaces
- Mechanisms for auto-discovery of link adjacency and services across E-NNI interfaces

The User Network Interface and External Network Node Interface (E-NNI) signaling protocols have been completed in [OIF-UNI-02.0-Common] and [OIF-E-

NNI-Sig-02.0]. The OSPF-TE based routing has been completed in [OIF-ENNI-OSPF-02.0]. This amendment adds support for multilayer networks to [OIF-E-NNI-Sig-02.0] and [OIF-ENNI-OSPF-02.0].

4.2 Scope

The scope of this Implementation Agreement is the specification of routing, signaling, and discovery abstract messages, attributes, and flows enabling end-to-end dynamic establishment of transport connections across multiple control domains supporting multiple layer networks. The concrete realization of the abstract messages and attributes based on the RSVP-TE and OSPF-TE protocols can be found in the respective protocol-specific documents [OIF-ENNI-RSVP-02.1] [OIF-ENNI-OSPF-02.1].

4.3 Relationship to Other Standards Bodies

This Implementation Agreement, to the greatest extent possible, uses available global standards documents.

4.4 Merits to OIF

The Multilayer Amendment is a key stride towards the implementation of an open inter-domain signaling protocol that enables dynamic setup, modification and release of various services across multilayer networks. This activity supports the overall mission of the OIF.

4.5 Working Groups

Carrier Working Group
Interoperability Working Group
Networking and Operations Working Group

4.6 Document Organization

This document is organized as follows:

- Section 4: Introduction
- Section 5: Terminology and Abbreviations
- Section 6: Multilayer Architecture Model
- Section 7: Multilayer Feature Set
- Section 8: Security and Logging for Multilayer Signaling
- Section 9: Compatibility with UNI and E-NNI
- Section 10: References

- Appendix I: Use Cases
- Appendix II: Multilayer Requirements
- Appendix III: Routing with Transitional Links
- Appendix IV: Client:Server Ratio

4.7 Keywords

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in [RFC2119].

5 Terminology and Abbreviations

AGC	Access Group Container (see [G.8080])
AP	Access Point
ASON	Automatically Switched Optical Network (see [G.8080])
ATM	Asynchronous Transfer Mode
CC	Connection Controller (see [G.8080] and [G.7713])
CCC	Calling/Called Party Call Controller (see [G.8080])
CI	Characteristic Information (see [G.800])
CP	Connection Point (see [G.800])
CTP	Connection Termination Point (see [G.800])
DDRP	Domain-to-Domain Routing Protocol
DS1	Digital Signal 1
E-NNI	External NNI (see [G.8080])
ETH_FP	Ethernet Flow Point
ETYn	Ethernet PHY layer network of order n
EVPL	Ethernet Virtual Private Line (see [G.8011.2])
FDL	Facilities Data Link
GFP-F	Generic Framing Procedure (GFP) - Framed GFP
I-NNI	Internal NNI (see [G.8080])
LRM	Link Resource Manager (see [G.8080])
MAC	Medium Access Control
MPLS	Multiprotocol Label Switching
NCC	Network Call Controller
NE	Network Element
ODUk	Optical Data Unit of order k
OSPF-TE	Traffic Engineering (TE) Extensions to OSPF Version 2 (see [RFC3630])
OTN	Optical Transport Network (see [G.709])
PHY	Physical
RC	Routing Controller (see [G.8080]
RSVP-TE	RSVP Traffic Engineering (see [RFC3209])
SC	Switched Connection
SC PC ID	Signaling Controller Protocol Controller Identifier (see [OIF-E-NNI-Sig-02.0]
SCN	Signaling Communications Network (see [G.7712])

SDH	Synchronous Digital Hierarchy (see [G.707])
SNP	Subnetwork Point (see [G.8080])
SNPP	Subnetwork Point Pool (see [G.8080])
SONET	Synchronous Optical Network (see [T1.105])
SPC	Soft Permanent Connection
STM-M	Synchronous Transport Module level M
STS-N	Synchronous Transport Signal level N
TAP	Termination and Adaptation Performer (see [G.8080])
TCP	Termination Connection Point (see [G.800])
TNA	Transport Network Assigned (see [G.8080])
UNI	User Network Interface (see [OIF-UNI-020], [G.8080])
UNI-C	UNI Signaling Agent – Client
UNI-N	UNI Signaling Agent – Network
VC	Virtual Component
VCAT	Virtual Concatenation

6 Multilayer Architecture Model

The architectural model of layer networks is described in [G800], which is a unified view of both connectionless and connection-oriented networks. It is a superset of [G809] and [G805]. In all three Recommendations, the relationship between layer networks is described in client/server terms. While [G805] presents only a view of resources, [G809] and especially [G800] describe forwarding as a necessary function to define layers. When two layer networks are in a [G800] client/server relationship, this constitutes a multilayer network. By definition in [G800], client/server relationships are recursive so that more than two layers can comprise a multilayer network.

No control plane entities are described in [G800] but are found in [G8080]. While initially based on [G805], [G8080] incorporates some changes to control plane components for [G800] packet layers. [G8080] describes the architecture of multilayer control planes while [G7712] defines signaling communication networks (SCNs) requirements over which control plane communication occurs.

6.1 Resource Relationships

The entity relationship between NCCs[1] in a client/server call relationship is not restricted in [G8080]. It could be 1:1, 1:n or m:1[2]. Examples of related data planes corresponding to these call relationships are given in this section.

6.1.1 1:1 Call Relationship

An example of a 1:1 client/server call relationship is a 2 Gbps Ethernet call mapped to a single STS-48c/VC-4-16c where GFP-F is the adaptation. The Ethernet service could be an Ethernet Private Line service that has no multiplexing within the Ethernet flow being mapped. Figure 2 is adapted from

[1] Note that each NCC is defined as a component that is responsible for calls in a single layer.

[2] m:n is not precluded

Figure 6 from [G8011.1] to show a specific server subnetwork adaptation to STS-48c/VC-4-16c.

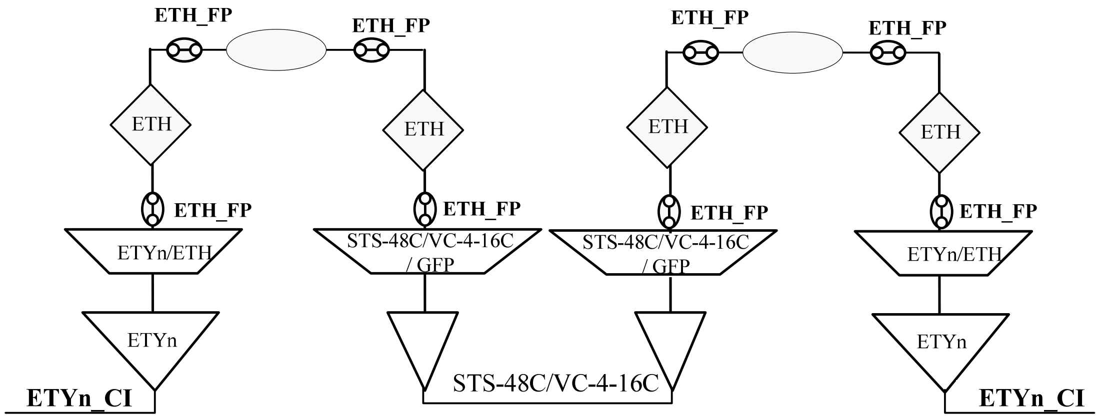

Figure 2: Data Plane example of a 1:1 Call Relationship.

6.1.2 1:n Call Relationship

The VCAT function allows a client-layer call to be supported by multiple server-layer calls. An example of a 1:n call relationship is illustrated using an STS-3c-2v/VC-4-2v VCAT call, assuming each VC-4 connection is in a distinct VC-4 call. Here, the VCAT call illustrates a 1:n client/server call relationship where the VCAT call is related to multiple server layer calls. The Ethernet/GFP layer to VCAT call relationship is 1:1. Note that the Ethernet layer is not actually involved in the 1:n call relationship.

Figure 3 illustrates this example.

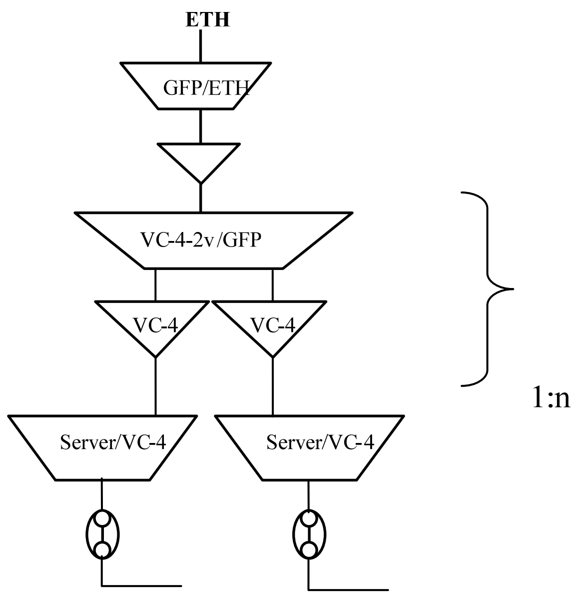

Figure 3: Data Plane example of a 1:n Call Relationship.

6.1.3 m:1 Call Relationship

The client/layer call relationship can be in an m:1 call relationship, inline with [G800] network examples. A packet example from [G8011.2] illustrates this. Here, an Ethernet Virtual Private Line type 2 service is defined as having dedicated access with a shared server layer. Figure 4 is adapted from Figure 6-2 in [G8011.2] and shows multiple VLAN flows multiplexed onto a single server layer connection. Each VLAN flow is the bearer part of the EVPL Type 2 service. Given that a service is supported with a separate call, this constitutes an m:1 client/layer call relationship.

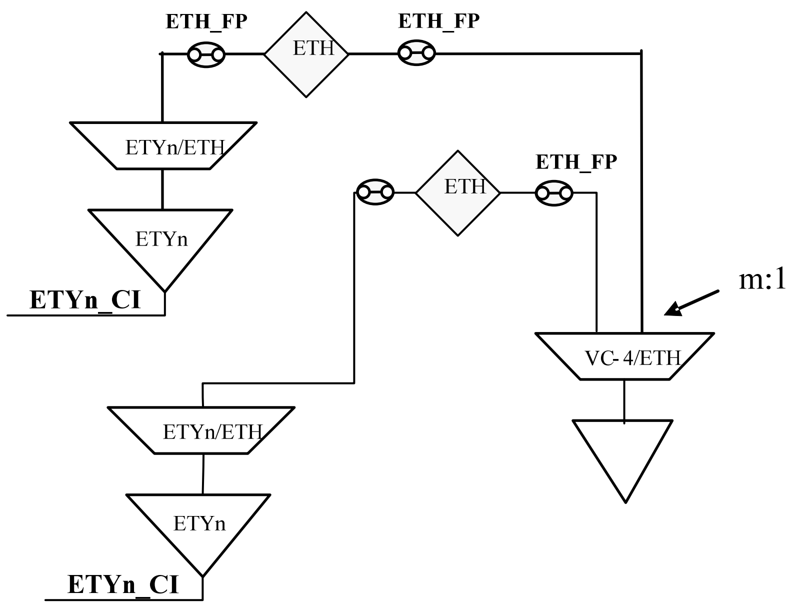

Figure 4: Data Plane example of a m:1 Call Relationship.

6.2 Resource Topology

Several situations can exist at the E-NNI with respect to the transport of a client-layer service. The E-NNI can be in the client layer or in a server layer. Each of the domains may provide flexible connectivity (i.e., switch traffic) at the client layer or at the server layer. This is illustrated in Figure 5:

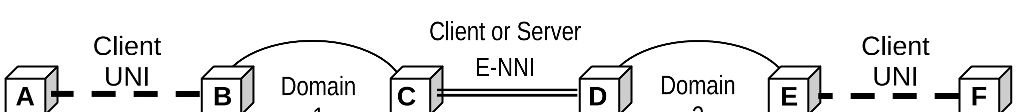

Figure 5: General Network Reference Model.

In the general model, the UNIs represent client layer links, which are the carrier network endpoints for a client layer service connection. The two domains may provide flexibility at the client layer or at a server layer that transports the client layer. The E-NNI represents a link that may be either in the client layer or a server layer. To visualize the possibilities, one can diagram the various combinations of domain flexibility (client or server layer switching) and E-NNI link (client or server layer).

The following set of diagrams shows the possible combinations across the E-NNI between the two domains (bearer plane view). These domains can represent individual network elements or subnetworks composed of many network elements. The diagrams simply serve to present the possible bearer plane situations at the E-NNI. These diagrams use graphic elements defined in [G800] and shown in Figure 6. The following topology descriptions do not drive administrative partitioning.

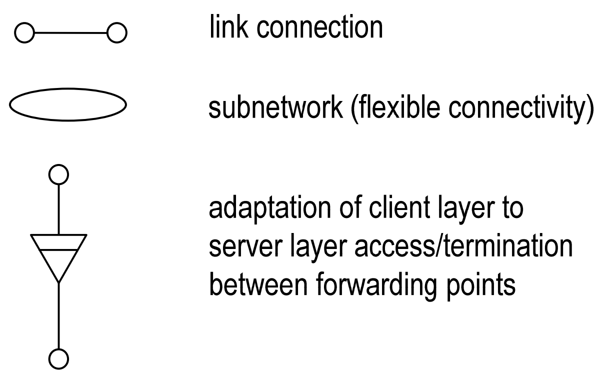

Figure 6: [G800] Symbols.

A single-layer network model can be drawn if both domains switch at the client layer and the E-NNI link is at the client layer.

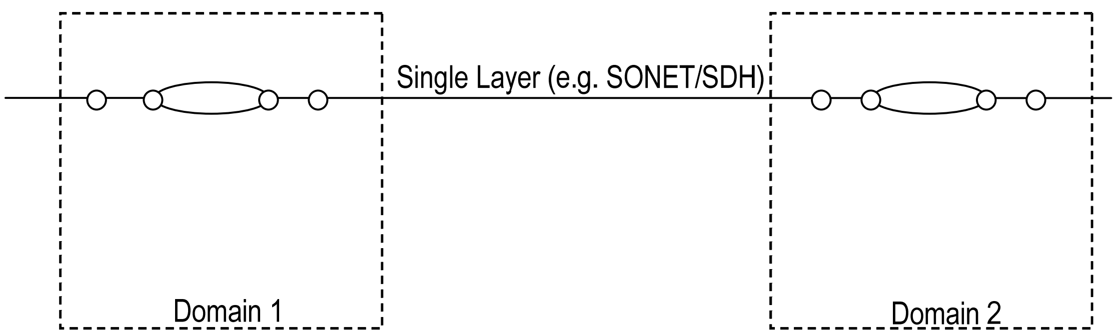

Figure 7: Both Domains Switch at the Client Layer, Client Layer Link.

Figure 8 shows both domains switching at the server layer and adapting the client signal to the server layer at the UNI-N.

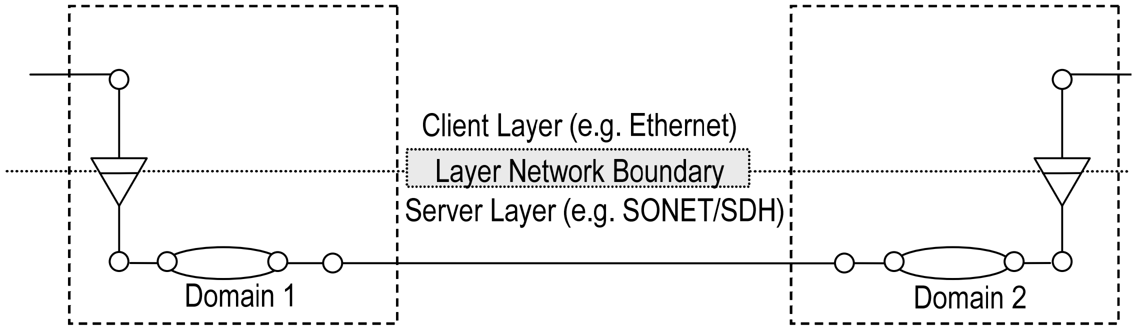

Figure 8: Both Domains Switch at the Server Layer, Server Layer Link.

Several more combinations are possible as shown in Figure 9 through Figure 11, where adaptation between the client and server layers is done at various points depending on the subnetwork (switching) layer in each domain and the layer of the E-NNI link. These combinations are not exhaustive.

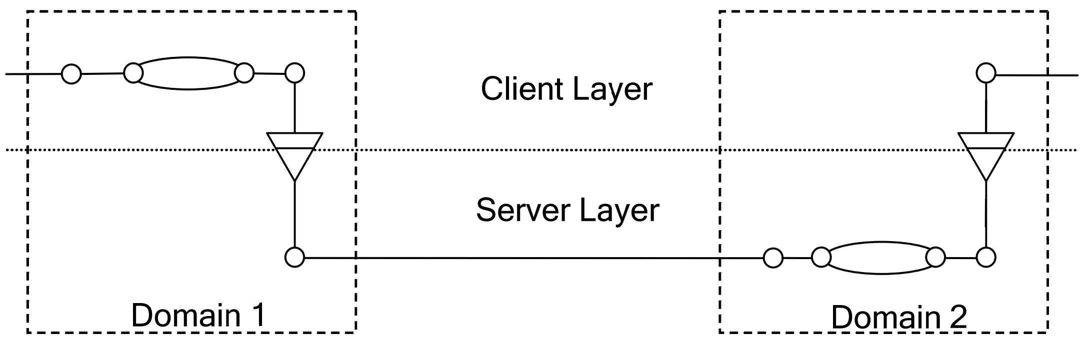

Figure 9: Domain 1 Client Layer Switching, Domain 2 Server Layer Switching, Server Layer Link.

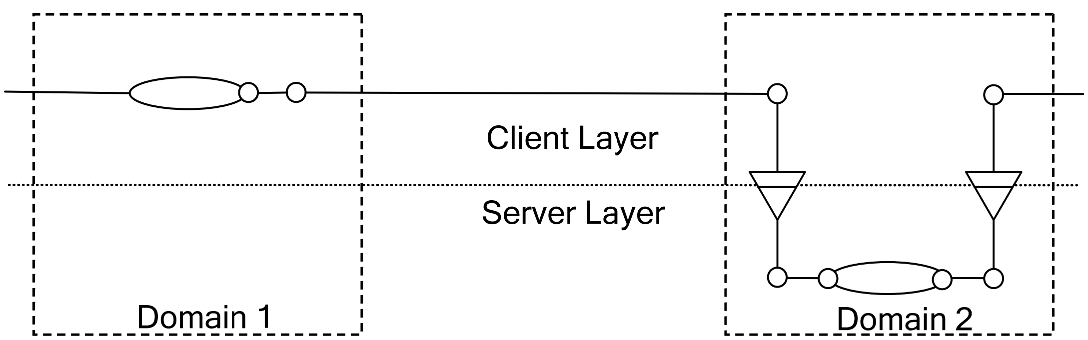

Figure 10: Domain 1 Client Layer Switching, Domain 2 Server Layer Switching, Client Layer Link.

In Figure 11, each server layer is independent from the other and may be realized by different technologies.

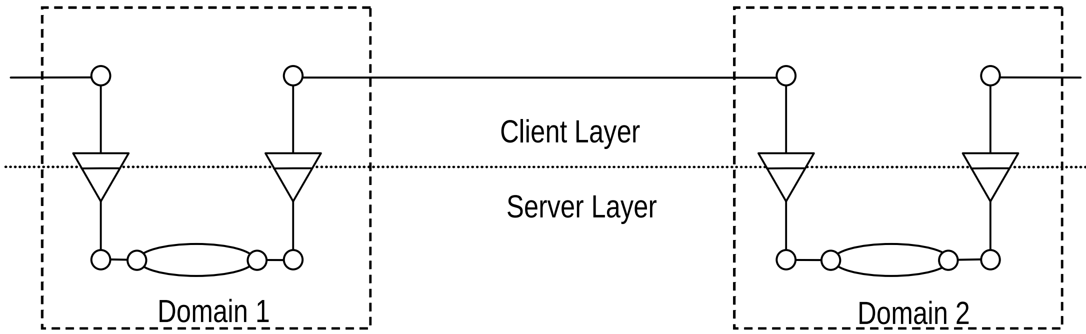

Figure 11: Both Domains Switch at the Server Layer, Client Layer Link.

While a domain may encompass multiple layers, it is also possible to reflect a business relationship where a client layer uses the services of a server layer and both employ a control plane. An example of this configuration is illustrated in the figure below.

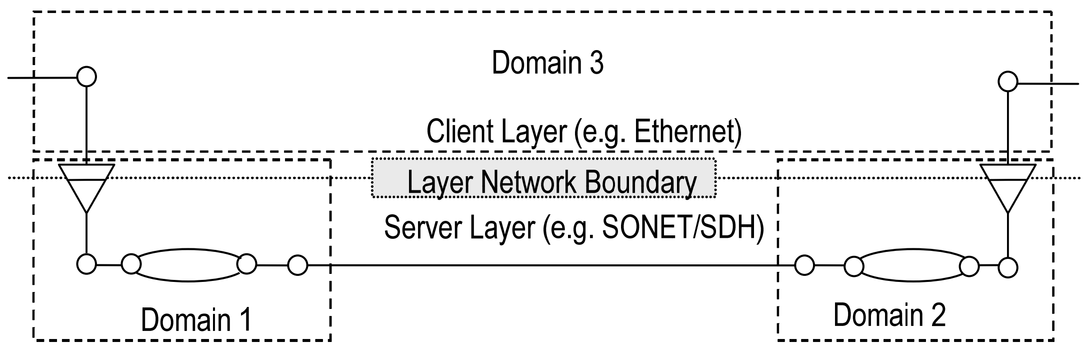

Figure 12: Domains Separated by Layers.

6.3 Interlayer Routing

This section covers multilayer routing constructs and abstraction models. Routing areas are not described but are present and created based on network operator decision.

6.3.1 Tools for Multilayer Topology

A routing topology reflects an organization of the resource topology. In a single-layer network, the resource topology of the network is described in terms of subnetworks and links between them. These translate into the vertices and edges of the routing graph. The layer network comprises the largest subnetwork and the access groups that map the Adapted Information into the Characteristic Information of the layer.

In the multilayer case, connections must transit not only subnetworks but also layer networks, and the layer network is opaque from the point of view of information transfer. The client layer is invisible to the server layer. However, from the routing topology point of view, whether the internal structure of the layer network is exposed or not is a matter of policy in exactly the same way as exposing the structure of a subnetwork is a policy choice. It should be noted that subnetwork or server layer connectivity constraints (e.g. blocking switch fabric) which are not exposed due to server layer abstraction may cause crankback or a connection setup to fail.

Several mechanisms allow integrating layer networks into the overall topology:

- Transitional links [G.800] (discussed in section 6.3.2.1)
- Server layer abstraction that are combined with transitional links
    - Abstract Node (discussed in section 6.3.2.2)
    - Abstract Links (discussed in section 6.3.2.3)
- Client layer abstraction that hide the abstraction from the client layer
    - Abstract Links (discussed in section 6.3.2.4)
    - Pseudo-Nodes (discussed in section 6.3.2.5)

Definitions:

- An abstract node is a logical representation of potential connectivity between points in a given layer; topological details are summarized within an abstract node.
- A transitional link represents potential client-layer resources adaptation into server-layer resources.
- A pseudo-node is a logical representation of potential server layer connectivity between points in a client layer; layer adaptations and other topological details are abstracted within a pseudo-node.

The difference in the scope between a pseudo-node and abstract node is illustrated in Figure 13.

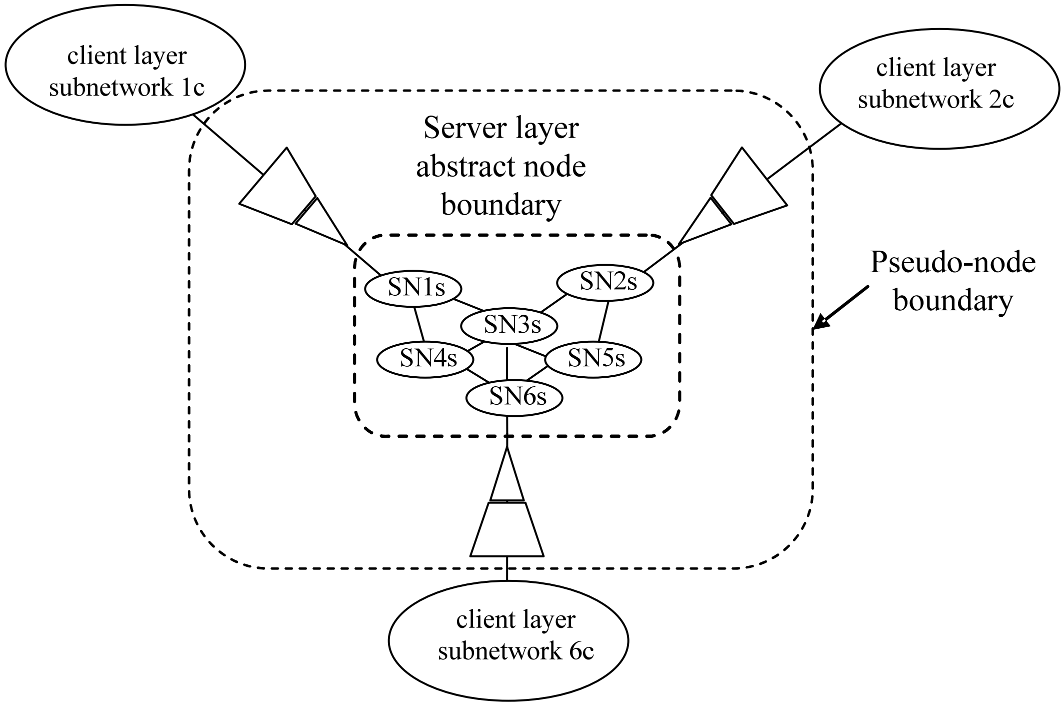

Figure 13: Abstract Node versus Pseudo-node.

A transitional link crosses the layer network boundary, interconnecting two subnetworks operating on different Characteristic Information (CI). Note that a transitional link may exist entirely within a network element. A layer network is shown as a box with rounded corners and access points along its boundary [G800]. A transitional link is annotated with an adaptation/termination symbol, which indicates that there is a layer network boundary somewhere along the link. While the exact location of these functions is important from the point of view of equipment realization, it is not important from the point of view of routing topology. A transitional link may be used between two nodes or routing areas or between a node/routing area and an abstract node as described below.

From a routing topology point of view, the pseudo-node depicts a connected server layer as a single subnetwork, and the transition between the client and server is enclosed within the pseudo-node boundary. The points on the edge of the pseudo-node are those of the client layer CTPs, i.e. the top of the adaptation function. With the pseudo-node, no server layer structure is exposed to the client layer. Any inter-network naming issues are handled within the pseudo-node and are invisible to client layer signaling.

The abstract node hides topology in the same layer. A more general interpretation is to state that handling any name translation occurs within the abstract node edge. An abstract node could thus wrap the subnetworks of a server layer network and be connected to a client layer by transitional links. For example, in Figure 13, a single abstract node could abstract all subnetworks SN1s through SN6s. This implies that when the client layer for subnetwork SN1c receives a signaling request with the transitional link to the abstract node as the next hop, it is responsible for translating that information to a transitional link into subnetwork SN1s.

Using the transitional link representation between routing areas in different layers exposes the nature of the server layer network while using a pseudo-node hides the server layer topology from the client layer. The pseudo-node forces the server to handle name translation, whereas the abstract node promises that the client will handle name translation.

It is possible that different abstraction mechanisms are used between the same pair of nodes depending on the server-layer connection existence. For example, implementations could represent potential server-layer connectivity between two nodes using a pseudo-node representation. Once a connection is established in the server layer, implementations could advertise the resulting link in the client layer. This is a regular link as described in [OIF-ENNI-OSPF-02.0].

6.3.2 Multilayer Abstraction

A control plane routing view of these bearer plane situations must be defined to enable route calculation for client-layer services across server-layer connections. Normally, a routing topology includes nodes and links representing the network elements and the links interconnecting them. The internal details of a network element or node in the routing topology are typically not exposed to the routing subsystem. If hierarchical routing is used, the internal details of a subtending routing domain need not be exposed either.

Five examples of abstraction are covered in this section, in increasing levels of abstraction from fully exposed server layer topology to completely hidden server layer through the use of pseudo-node or client-layer abstract links. If an abstraction mechanism is used, all nodes involved in the abstraction MUST use the same mechanism. If different mechanisms are implemented, path computation may fail.

This document does not define new abstraction concepts. It relies on existing concepts. Abstract nodes and abstract links are described in [OIF-ENNI-OSPF-02.0]. The pseudo-node concept is described in Figure 6.12 of [G8080] but the term is defined in this document.

6.3.2.1 No Abstraction

When no abstraction is used, the server-layer and client-layer topologies combine into a single topology fully visible for routing controllers in both layers. Transitional links are used between the client- and server-layer nodes, as illustrated in Figure 14.

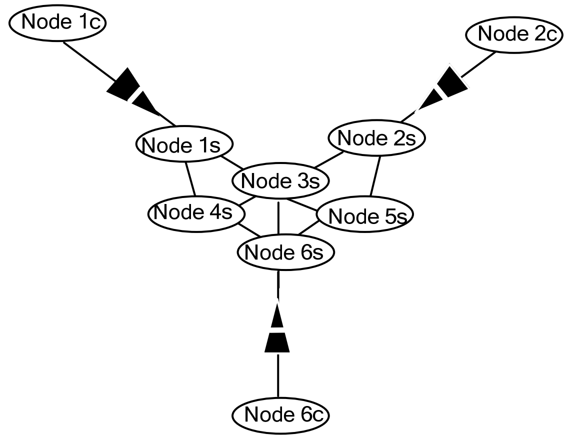

Figure 14: Routing Topology without Abstraction.

Across transitional links, an LRM in the client layer cannot communicate directly with an LRM in a different layer. An RC operating in the client layer and an RC in the server layer is required at respective ends of a transitional link, whether switching is locally available in either layer. The same RC instance MAY provide both of these roles. This differs from multiplexing into one protocol instance, which is also allowed.

Figure 15 illustrates components involved in transitional link representation based on a subset of Figure 14. A server-layer RC on the NE that performs adaptation collects server-layer topology information and ensures that the server-layer link is correctly advertised in the server-layer network, even if no server layer matrix is available underneath the adaptation. The local adaptation information is integrated into a single link in the composite graph by local interactions involving the server-layer RC, TAP and client-layer RC. Part of this interaction must bind the client-layer SNPP (top of adaptation) to the remote server-layer SNPP and this binding is what will be advertised.

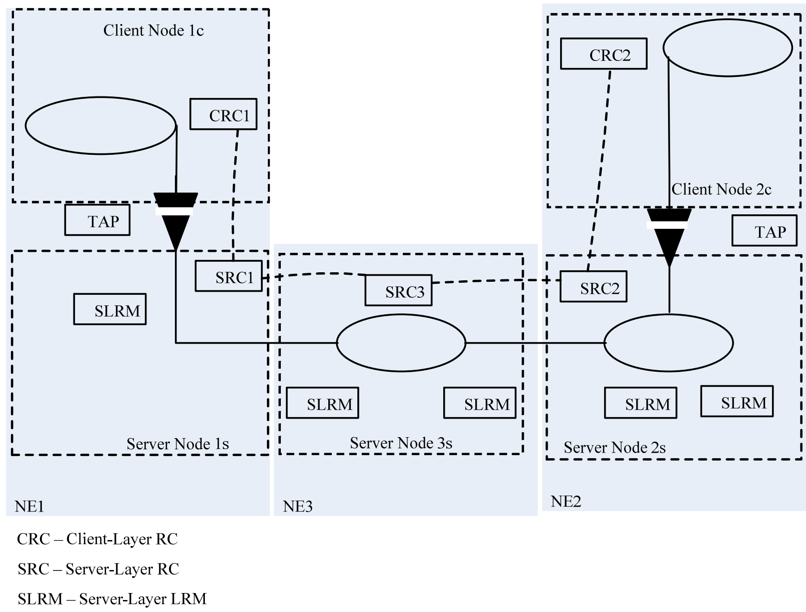

Figure 15: ASON Component Architecture supporting transitional link topology

As a result of the coupling across the transitional links and using Figure 14 as an example, the client layer nodes (1c, 2c and 6c) are able to see all the server layer nodes in the routing graph. Likewise, the server layer nodes (1s through 6s) have a view of all the client layer nodes (1c, 2c and 6c). This allows path computation in either layer to have full visibility of the topology of both layers and perform a path computation fully detailed in both layers.

6.3.2.2 Server Layer Abstraction: Abstract Node

When the server layer is abstracted, transitional links are used as links between the client layer nodes and the abstracted server layer resources. Several configurations are possible using abstract nodes. An example of an abstract node is shown in Figure 16. In this example based on Figure 14, the abstract node hides the internal topology of the server layer; nodes 1s through 6s are not visible to the client layer. In the case of abstract nodes interconnected by abstract links:

- If the node that performs the adaptation is part of an abstract node, this section applies to that node.
- If the node that performs the adaptation is a regular node interconnected with an abstract link to an abstract node, then the description in section 6.3.2.1 applies to that node.

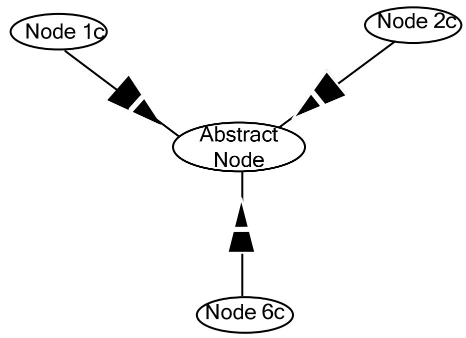

Figure 16: Server Layer Abstract Node.

An abstract node is a structure in a single layer. A link into an abstract node is terminated on a subnetwork. In Figure 16, one end of each transitional link is terminated on a server-layer subnetwork within the abstract node and its SNPP is attached to the abstract node. The other end of each transitional link terminates on a subnetwork in the client layer.

Figure 17 illustrates components involved in abstract node representation.

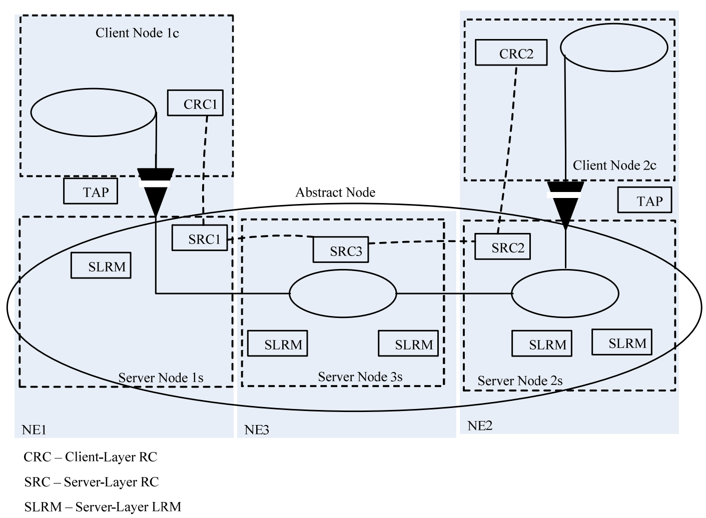

Figure 17: ASON Component Architecture supporting Abstract Node.

In Figure 17, the server-layer nodes 1s and 2s take part in the creation of the abstract node. The nodes at the edge of the abstract node must be provided with a node ID to be used in combination with each link exiting the abstract node. The link identifier assignment is coordinated by nodes that compose the abstract node. The links and abstract node are advertised between CRC1 and SRC1 (and CRC2 and SRC2), however, abstract node internal topology is not exchanged between the abstract node RCs (SRC1 and SRC2) and its adjacent neighbors (CRC1 and CRC2).

The connectivity between subnetworks 1s and 2s is unspecified. The abstract node combines and hides details of nodes and links.

Note that in the example in Figure 17, there are two RCs involved in advertising the abstract node (SRC1 and SRC2). It is possible to use a different model where a subset of RCs is responsible for advertising the abstract node.

6.3.2.3 Server Layer Abstraction: Server Abstract Links

An example of abstract links is shown in Figure 18. In this example based on Figure 14, the abstract links hide the internal topology of the server layer; only the border nodes 1s, 2s and 6s are visible to the client layer. Internal nodes 3s, 4s and 5s are not revealed to the client. Transitional link representation follows the description of section 6.3.2.1. Abstract links are single-layer construct described in [OIF-ENNI-OSPF-02.0].

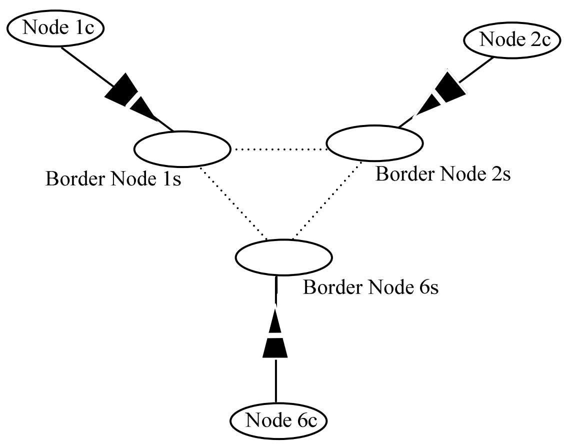

Figure 18: Server Layer Abstract Links between Border Nodes.

6.3.2.4 Client-Layer Abstraction of Server Layer: Client Abstract Links

In the case of abstract links in the client layer, the details of adapting the client to the server layer, if needed, are handled by the control plane view within domains on both sides of the abstract link and are not a concern outside. This is illustrated in Figure 19, which is based on Figure 14. This representation does not reveal anything about how the client-layer nodes 1c, 2c and 6c are connected; it does not reveal the server layer presence. The abstract links are configured on the nodes at both ends of the links.

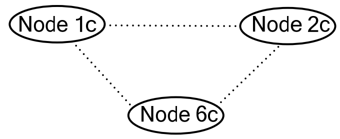

Figure 19: Client-Layer Abstract Links

6.3.2.5 Client-Layer Abstraction of Server Layer: Pseudo-Node

In the pseudo-node case, the details of adapting the client to the server layer, if needed, are handled by the control plane view within the domain represented by that pseudo-node and is not a concern outside. This is illustrated in Figure 20, which is based on Figure 14 but does not reveal how client-layer nodes 1c, 2c and 6c are connected. Only the nodes adjacent to the pseudo-node are aware that node 7c is a pseudo-node. Node 7c looks like a regular node to nodes beyond 1c, 2c and 6c.

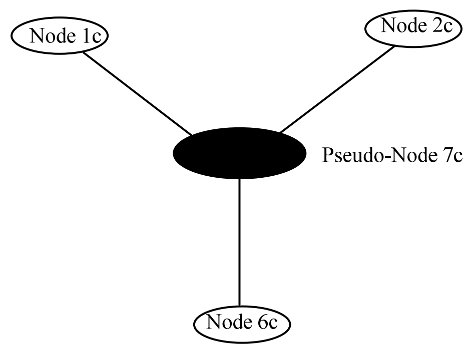

Figure 20: Pseudo-Node Representation.

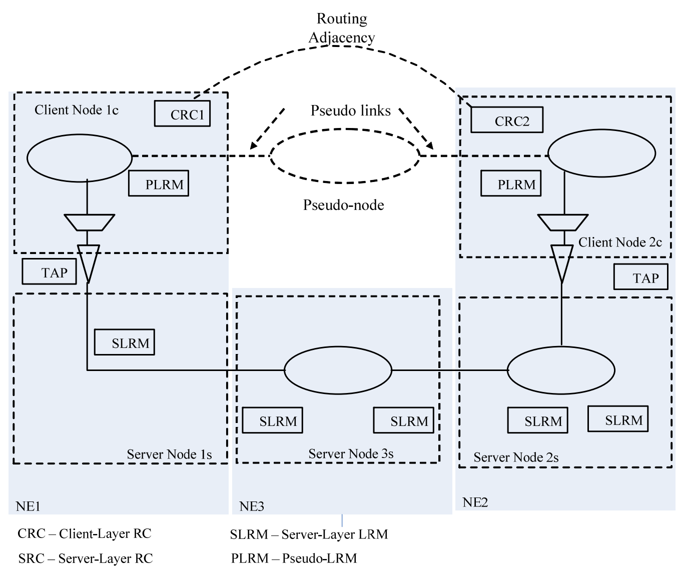

Figure 21: ASON Component Architecture supporting Pseudo-node topology.

Figure 21 illustrates components involved in pseudo-node representation for a subset of the network topology represented in Figure 14. Note that the server-layer RCs are not involved in the pseudo-node representation. NE1 represents an NE without server-layer switching while NE3 represents an NE with server-layer switching. The server-layer RCs would establish adjacencies as is the case for single layer scenarios but this is not illustrated in the figure as it is not related to the pseudo-node. The pseudo-node is modeled as a node and pseudo-links to the Client layer NEs. A pseudo-link is defined as a link with at least one endpoint on a pseudo-node. The pseudo-link is modeled on the client layer NEs as a pair of LRMs (PLRM in the figure). The client nodes surrounding the pseudo-node are provided with a node ID for the pseudo-node. Two interface IDs for the pseudo-link can be generated automatically by the node adjacent to the pseudo-node and responsible for the pseudo-link advertisement. The first interface identifier is the local interface ID, used to identify the link from the advertising node's point of view. The second interface identifier is the remote interface ID, used to identify the link from the pseudo-node's point of view. It is needed to avoid collisions with links advertised by other nodes. By default, the remote interface ID SHOULD be filled with the advertising node's node ID but implementations MUST support the ability for the operator to override this value. The client node CRC is made aware of the pseudo-link via the normal (internal) LRM-RC communication, as for any other link, and advertises both ends of the pseudo-link. The server layer is not involved in pseudo-node advertisements at all. The server layer also has ASON components supporting it and the Pseudo-Node-LRM <–> TAP <–> Server-Layer-LRM transitive relationship allows for bandwidth accounting of the pseudo-link taking into account the server-layer link usage.

6.3.3 Multilayer Multi-Domain Models

When applying abstraction concepts across multilayer E-NNI interfaces, several options are available and abstraction that spans more than one domain requires coordination among domains. This section uses a multi-domain topology model illustrated in Figure 22 that is used as a reference for the examples in this section. The pseudo-nodes and abstract nodes boundaries show a range of possible abstraction constructs that can be used to realize one of the routing models explained in this section. This example shows two E-NNIs between domains A and B, one E-NNI at each layer. Each domain contains client and server layer resources and the domains are interconnected at both layers. This is for illustration purposes only as it is possible to have an E-NNI control-plane only interaction between two client-layer controllers in the absence of a link between client-layer endpoints.

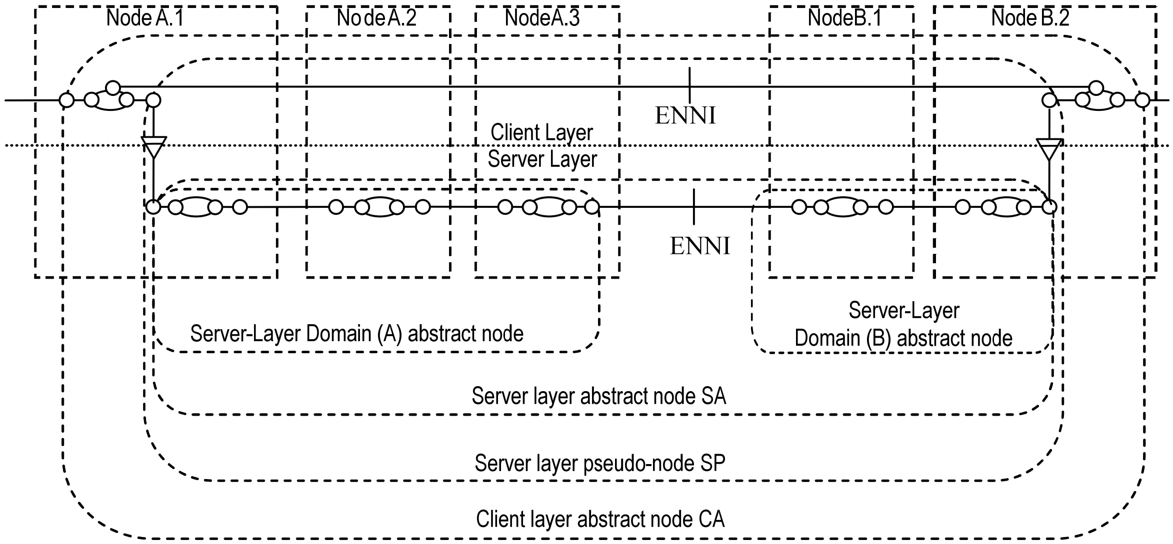

Figure 22: Multilayer Multi-Domain Abstraction.

In a case where no abstraction is desired, only discovery or configuration of the links between the domains is required. The routing topology corresponding to a full representation is shown in Figure 23.

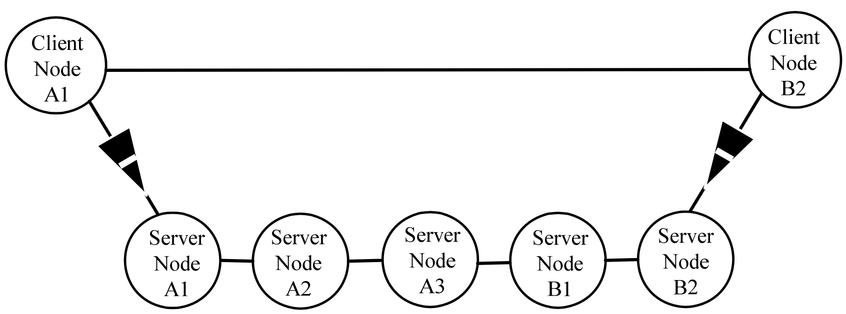

Figure 23: Multilayer Multi-Domain - No Abstraction

In a case where abstraction is desired but kept within a domain, only discovery or configuration of the links between the domains is required. The routing topology corresponding to abstraction contained within a domain is shown in Figure 24. Server domain A is representing connectivity within its domain using an abstract link between SN/A1 and SN/A3. Server domain B is representing connectivity within its domain using an abstract node B.

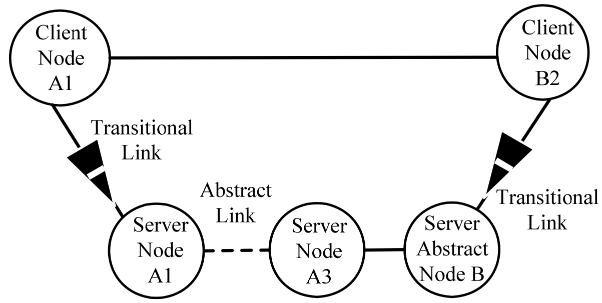

Figure 24: Multilayer Multi-Domain - Single Domain Abstraction

In a case where abstraction is desired to hide the adaptation from the client layer, a pseudo-node can be used. The pseudo-node requires coordination between controllers that participate in the abstraction at the adaptation point, e.g. CN/A1 and CN/B2 in Figure 24, but not from controllers for other nodes or domains, e.g. SN/A1, SN/A3 and SN/B in Figure 24. The participants in the pseudo-node must agree on the pseudo-node name and each participant advertises a set of links into the pseudo-node, representing potential server-layer connectivity. The client routing topology corresponding to pseudo-node abstraction is shown in Figure 25.

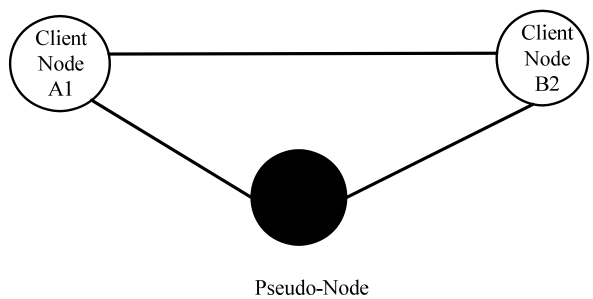

Figure 25: Multilayer Multi-Domain - Pseudo-Node

Other mechanisms of abstraction are available. For example, a client- or server-layer abstract node encompassing multiple domains is possible as shown by the server-layer abstract node SA and the client-layer abstract node CA in Figure 22.

All nodes participating in an abstraction MUST select the same abstraction mechanisms. For example, if pseudo-node representation is used to hide server layer resources by one node, all nodes that perform the same adaptation MUST also use the pseudo-node representation. If one node advertises the server layer resources by advertising a link into a pseudo-node while another node advertises resources into the same server layer by advertising a transitional link, it does not allow paths to be computed across the server layer.

6.3.4 Transitional Links

Transitional links are located where adaptation/termination functions exist connecting a client layer to a matrix in a server layer enabling access to the server layer's flexible connectivity. This construct enables a path computation function to locate the lowest-cost multilayer route, taking into account the attributes of server layer resources in addition to client layer resources. Transitional links apply to 1:1 adaptation as described in [G8080] but also to 1:n (e.g. VCAT/VC-4) and m:1 (e.g. EVPL/SONET).

The transitional link [G.800] shares many attributes of the link defined in G.8080 and described in [G7715.1] as it is represented in a topology graph through use of an arc connecting two vertices. The vertices represent two different matrices. However, unlike a link, the transitional link represents the adaptation of client layer CI into server layer CI. As a result, the set of attributes necessary to describe a transitional link is different than the set of attributes necessary to describe a link.

[G7715.1] defines the attributes of a link as:

- Local SNPP name
- Remote SNPP name
- Layer specific characteristics

Layer specific characteristics include:

- Signal Type
- Link Weight
- Resource Class
- Local Connection Type
- Link Capacity
- Link Availability
- Diversity Support
- Local Client Adaptations Supported

While many of these attributes are generic in nature, not all are appropriate for transitional links. Attributes used to form the topology graph (i.e. Local SNPP ID, Remote SNPP ID) or to provide the network administrator control of how traffic will be routed (i.e. link weight, resource class) are still applicable for transitional links. The Local SNPP ID and Remote SNPP ID are mandatory attributes for transitional links. However, the remaining attributes are not general items, but attributes specific to G.800 links.

The following [G7715.1] layer specific characteristic attributes apply to transitional links:

1. Signal Type is applicable to both client and server layers of the transitional link. (Mandatory)
2. Link Weight (Mandatory)
3. Resource Class (Optional)

4. Link Capacity for Transitional Link (Optional)
   - Describes the relationship between the smallest allocatable unit in the client layer capacity and the corresponding number of client layer units that come from allocating the smallest unit in the server layer, allowing for path computation to be performed by a system unaware of the client/server adaptation technology. For example, it could state the client layer capacity as 16 units for 1 unit of server capacity, or as N Gb/s for 1 unit of server layer capacity. This would facilitate not only TDM client layers, but also packet clients (e.g. MPLS-TP) and also the variability of the ODU TS. A discussion on the usage of this field is included in appendix in section 14.

5. Link Availability (Optional)

6. SRLG (Optional)

7. Link Protection Type (Optional)

8. Local Client Adaptations supported is listed in [G7715.1] as an optional attribute for a G.800 Link. This attribute is mandatory for a transitional link as it identifies the layers being transitioned between (i.e. the client and server layers specifically) in addition to the specific variant of adaptation (e.g. bit sync vs. byte sync) being used.

The following [G7715.1] layer specific characteristic attribute does not apply to transitional links:

1. Local Connection Type. Unnecessary for Transitional Links as adaptations are always configured on TCPs, not on CPs. As a result, there is no variability of configuration, making this attribute redundant.

A discussion on impacts of transitional links on path computation is included in appendix in section 13.

6.3.5 SCN Arrangements

This section covers SCN arrangements for various routing topologies. There are three ways the SCN can be organized:

1. Controllers in all layers are connected by a single SCN.
2. The client controllers are on a client SCN while the server controllers are on a server SCN.
3. Each adjacency has its own SCN.

Figure 26 through Figure 31 illustrate variants of SCN topology for various routing representations. The appendix on SCN and topology choices discusses the variants in more detail.

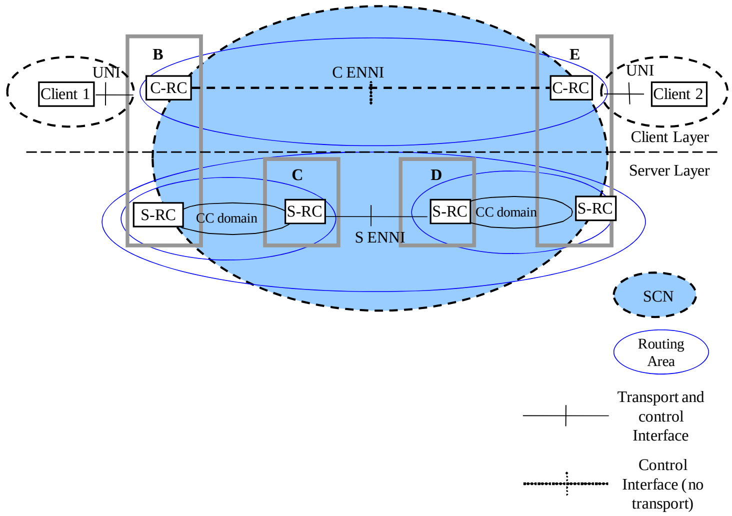

Figure 26: SCN supporting Pseudo-node topology – Single SCN

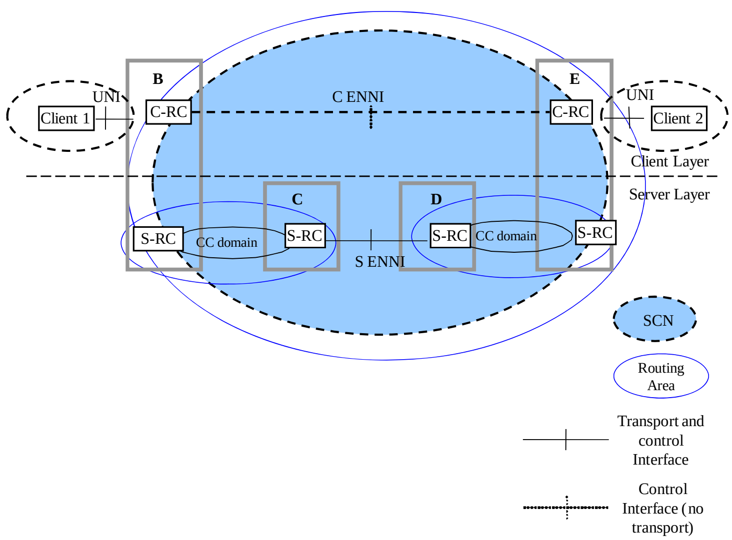

Figure 27: SCN supporting transitional link topology – Single SCN

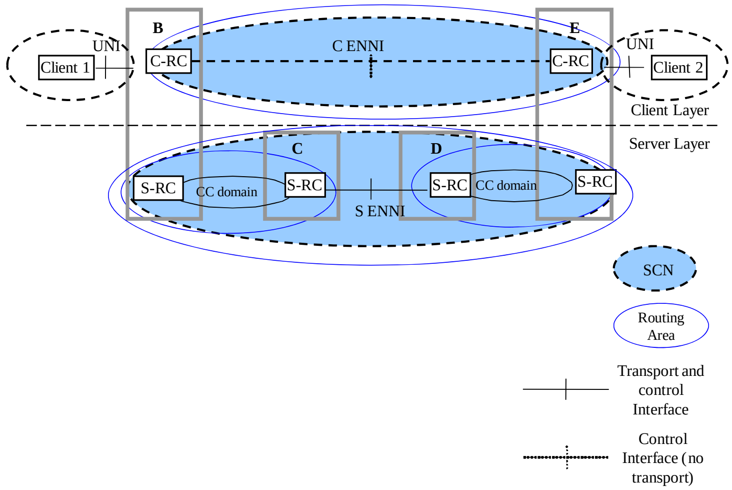

Figure 28: SCN supporting Pseudo-node topology – SCN per layer

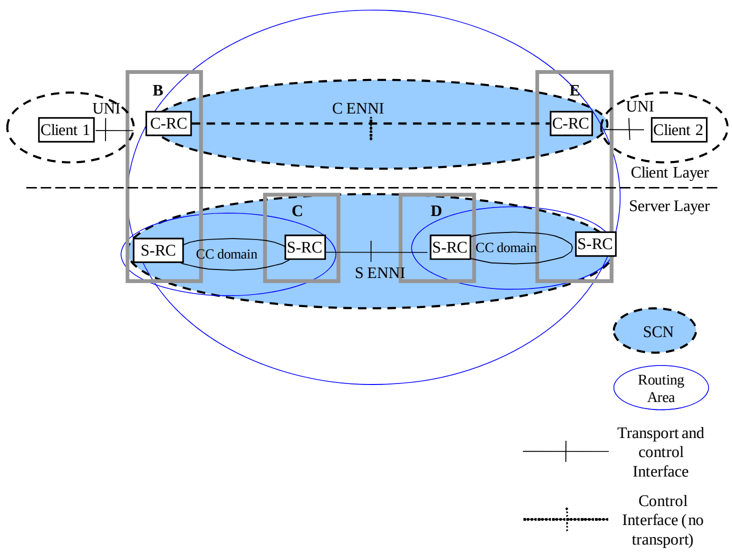

Figure 29: SCN supporting transitional link topology – SCN per layer

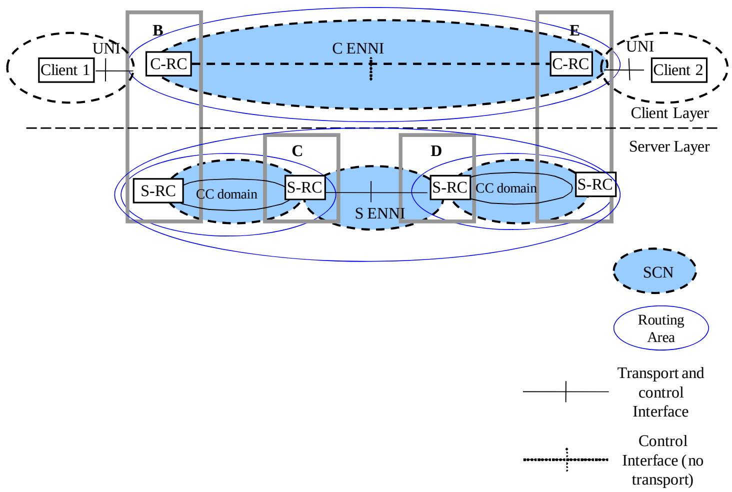

Figure 30: SCN supporting Pseudo-node topology – SCN per adjacency

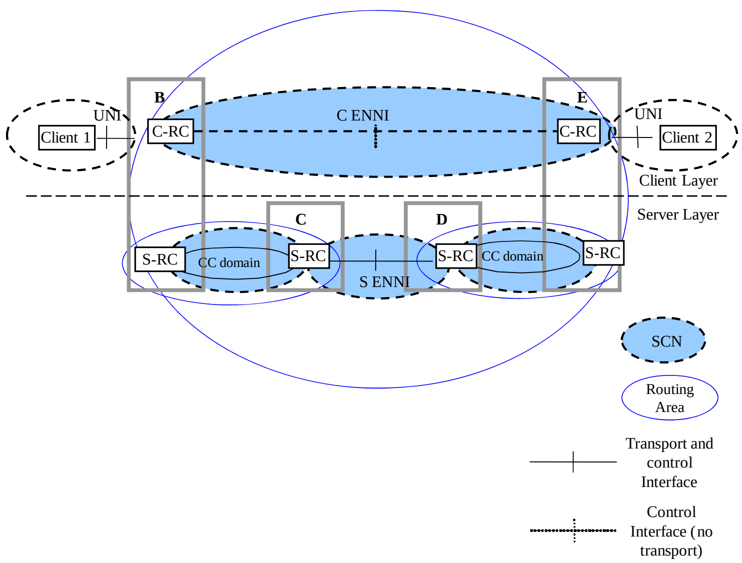

Figure 31: SCN supporting transitional link topology – SCN per adjacency

6.4 Control Plane Component Models

In the ASON architecture, the control plane view of the transport plane is determined by the configuration of routing areas that encompass subnetworks. This section introduces two concepts: a control-plane only E-NNI and SCN gateways. A control-plane only E-NNI is an interface between two NCCs that do not have a link between them in their layer but rely on a server layer to provide connectivity. A control-plane only E-NNI includes routing, signaling and discovery functions. An SCN gateway is a control component that provides forwarding of messages across an SCN boundary when the sender and receiver are on different SCNs. There is no corresponding NCC function at the boundary itself.

The various models described below are examples and do not preclude other arrangements. SCN configuration must be agreed upon by all involved parties as it is not a parameter that can be automatically discovered.

Figure 32 shows a possible arrangement (Control Plane Arrangement 1) that corresponds to the topology shown in Figure 8. To enable client-layer calls between the two client-layer NCCs, a client-layer control-plane only E-NNI is used. This requires a common SCN between the client-layer NCCs.

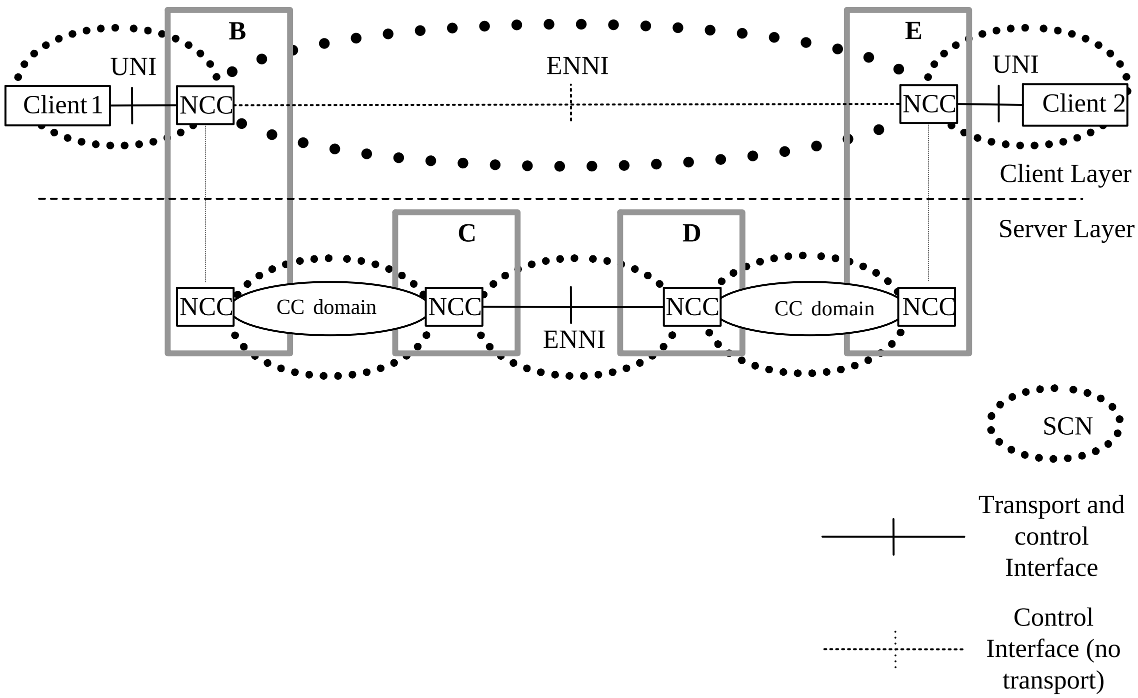

Figure 32: Control Plane Only E-NNI.

Figure 33 shows a second possible arrangement (Control Plane Arrangement 2) that corresponds to the topology shown in Figure 8. To enable client-layer calls between the two client-layer NCCs, a client-layer control plane only E-NNI is used in the client layer. In this model, the client layer SCN is partitioned and gateways are required to cross the SCN boundaries. The gateways are responsible for forwarding messages from one SCN to another. The gateways are single-layer and do not interface with NCCs in other layers.

This arrangement corresponds to a business scenario in which a carrier has client- and server-layers that use a common SCN. Between carriers, another SCN is used for E-NNIs in which the client-layer E-NNI does not have a corresponding transport resource but is used to transfer signaling messages between domains.

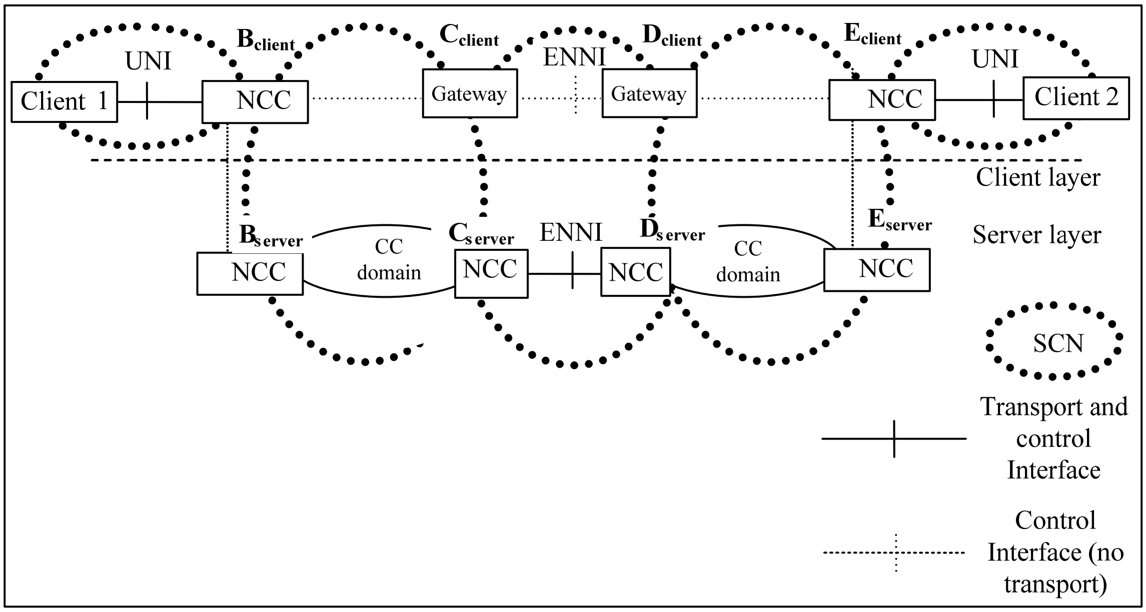

Figure 33: Control Plane E-NNI Gateway – Common SCN across Layers.

Figure 34 shows a third possible arrangement (Control Plane Arrangement 3) that corresponds to Figure 8. To enable client-layer calls between the two client-layer NCCs, a client-layer control plane only E-NNI is used in the client layer. In this model, the server-layer and client-layer SCNs are partitioned and gateways are required to cross the client-layer SCN boundaries. This corresponds to a business scenario in which each carrier uses its own SCN per layer. Between carriers, another SCN is used for each layer.

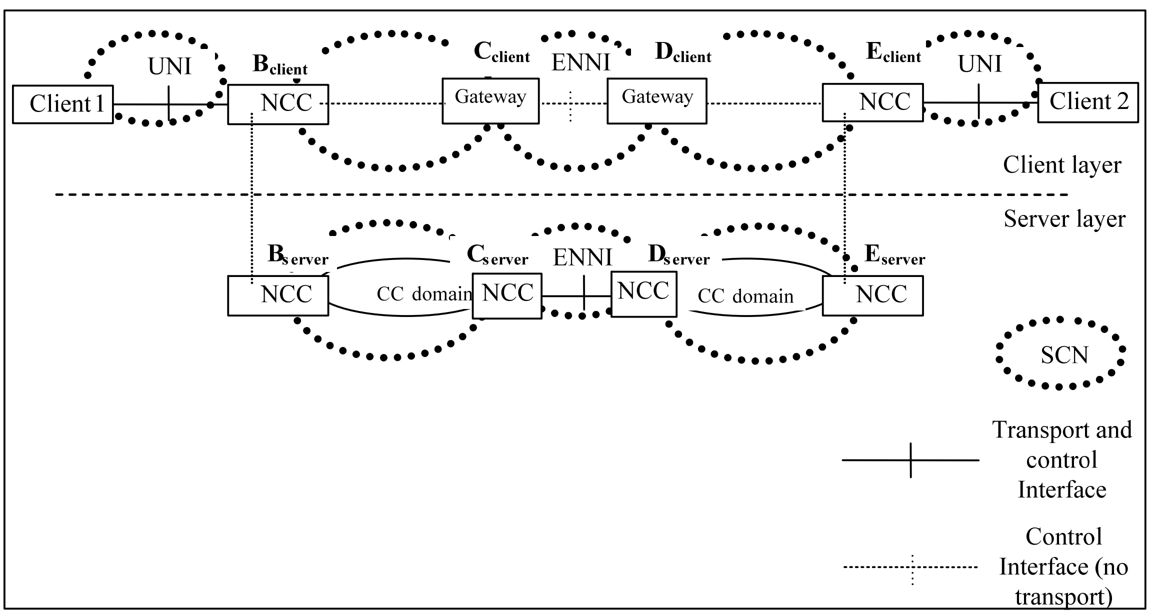

Figure 34: Control Plane E-NNI Gateway – Different SCN across Layers.

6.5 Identifiers and Name Resolution

Within the domains described in the previous section, signaling components are needed to establish calls and connections at both client and server layers. This section defines addressing and signaling interactions that result from the interlayer relationship.

6.5.1 Addressing: TNAs

For the interlayer case where the server layer does not have SNPP links to a client, [G8080] (Section 8.5.1) allows the TNA to represent the set of access points that may be used to support the server-layer call. This is illustrated below in Figure 35. For example, a VCAT layer TNA is assigned for access point z1 and VC-4 layer TNA is assigned to the access point set a1 to a2.

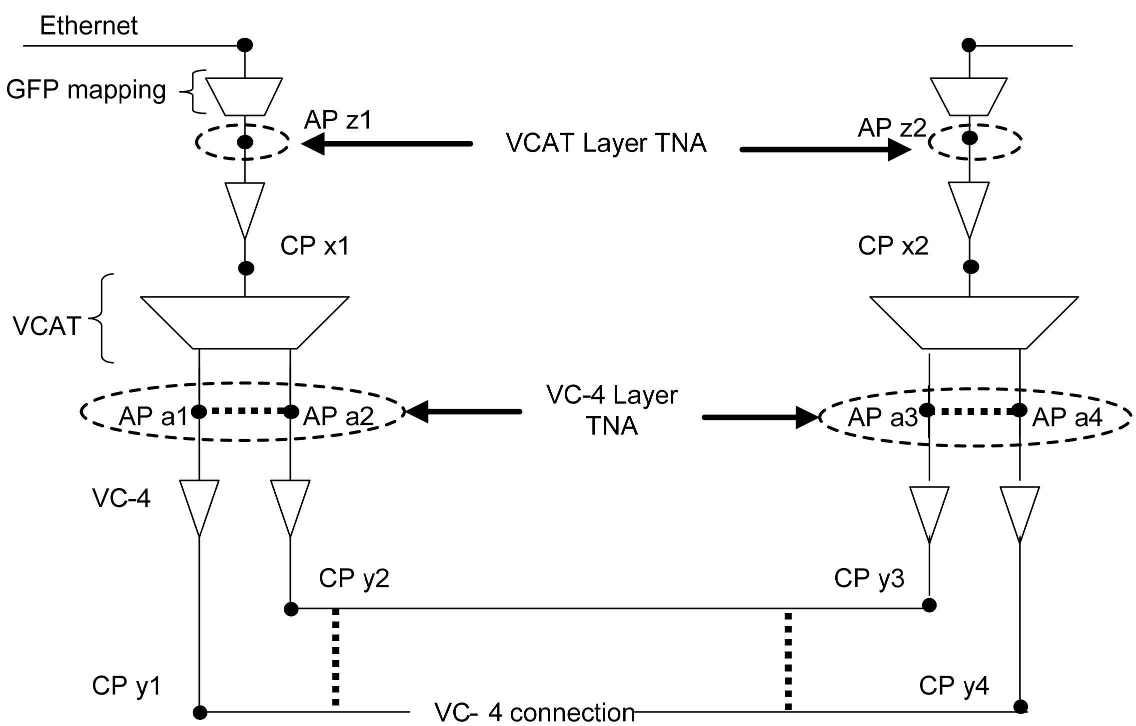

Figure 35: TNAs to Identify Resources in Multilayer Networks.

Client-layer NCCs have visibility of the server-layer TNAs for initiating interlayer call invocations. In Figure 35, the Ethernet layer NCCs have visibility to the VCAT layer TNAs at z1 and z2; and the VCAT layer NCCs have visibility to the VC-4 layer TNAs at a1-a2 and a3-a4.

TNAs for different layers may be in different namespaces depending on the control plane instantiation. There may be OSPF-TE limitations that force the user to put TNAs for different layers in a global namespace.

6.5.2 Addressing: SNPP Names

The SNPP names come from a layer-specific namespaces. Note that within a single layer, routing levels may also have separate SNPP namespaces.

6.5.3 Addressing: Control Components

Identifiers for signaling and routing control components come from separate layer-specific namespaces. At NCCs with interlayer interaction, it is assumed that knowledge of component names at adjacent layers is provided so that a client-layer NCC may invoke a server-layer NCC for a server-layer call.

6.5.4 Address Resolution

In the single-layer case, for processing an incoming call, a UNI-N accesses an address resolution function that maps a destination TNA to an SNPP in the same layer. This SNPP may be at the egress UNI identified by the TNA or could be at an E-NNI on a route to the destination UNI. The address resolution function might be accessed locally or through a directory service.

For interlayer invocation, when a client-layer NCC needs to perform address resolution but is at a layer boundary, the address resolution function needs to be able to return the source and destination TNAs in the server layer so that the client layer can make a call request in the server layer. This resolution function takes as input two SNPPs in the client layer. The first SNPP is at the interlayer boundary where the client-layer CI can be mapped to a server layer. It could be obtained from a path that was created from client-layer path computation. The second SNPP is also at a location where the adaptation occurs and lets the client-layer CI be recovered from the mapping. It is obtained either by address resolution from the destination TNA or from the client-layer path created by the client-layer path computation function. The output is a pair of TNAs with optional logical port identifier. The optional logical port identifier may be specified in cases where the TNA is not sufficient to fully specify the set of required resources in the server layer.

6.6 Interlayer Processing Abstract Model

This section describes modified steps in the NCC, CC, and RC to perform interlayer setup. The interlayer processing refers to the interface between two network call controllers (NCCs) in adjacent layers. It is typically hidden within an implementation and no associated protocol messages are defined. It supports messages for providing call control operations across the interlayer interface. There are no messages exchanged between connection controllers in different layers. These messages are described as “abstract” because the actual realization depends on the internal implementation used. The interlayer processing messages support server-layer call establishment triggered by the client layer and client-layer call establishment over existing server-layer calls.

The interlayer processing abstract model does not apply in the case where the server layer resources are established by the management plane.

In order to illustrate the interlayer processing, an example network and configuration is used. In this example, it is assumed that the address resolution function is configured for interlayer TNA resolution. To provide generality, the model allows client-layer switching resources to connect server-layer subnetworks and client ingress and egress points to not coincide with server-layer adaptations. The examples are based on three layers but the model can be applied to any number of layers in a recursive fashion. An example bearer organization is shown in Figure 36 . Note that layer x+1 resources in SN2 and SN3 are not derived from the link in layer x between SN6 and SN7, but have their own layer x+1 links.

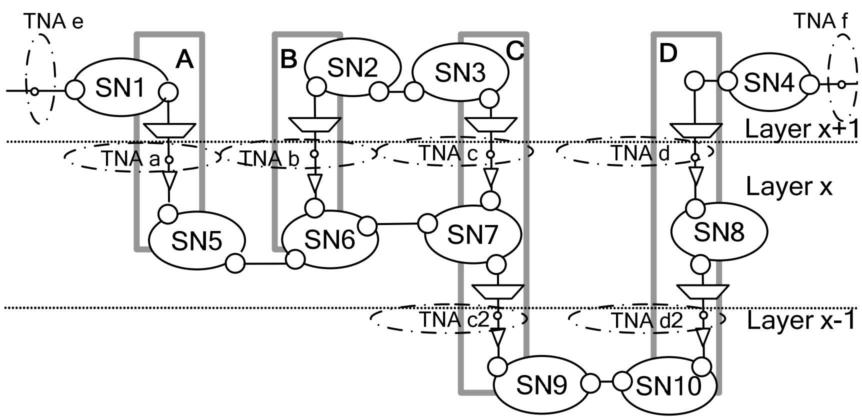

Figure 36: Bearer Example with Client Subnetworks.

6.6.1 Control Plane Organization

Several possible control plane organizations could correspond to Figure 36. A possible signaling control plane organization is shown in Figure 37 and a possible routing control plane organization is shown in Figure 38. In Figure 37, CC domain n corresponds to Subnetwork SNn, and locations of adaptations A, B, C and D are contained in NEs with the same identifiers. For simplicity, several NEs and NCCs were omitted from the figure. For example, there are NEs and NCCs at layer x+1 across the E-NNI between 2 and 3 but they were omitted from the figure.

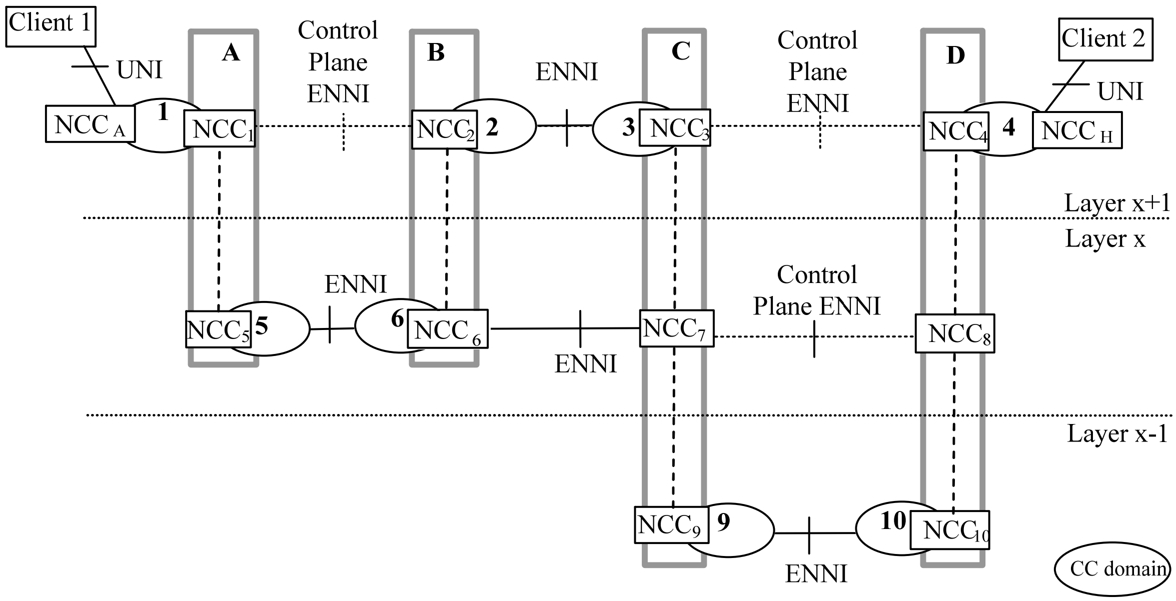

Figure 37: Control Plane Organization of Figure 36 - Signaling.

In this example, the routing topology follows the subnetworks and CC domains except for pseudo-nodes shown in layer x+1. Pseudo-node w represents potential server-layer connectivity through the SN5 and SN6 routing areas, pseudo-node z represents potential connectivity through the SN7, SN8, SN9 and SN10 routing areas and pseudo-node y represents potential connectivity through the server layer routing area encompassing layer x and layer x-1. A transitional link model is used to represent connectivity between layer x and layer x-1.

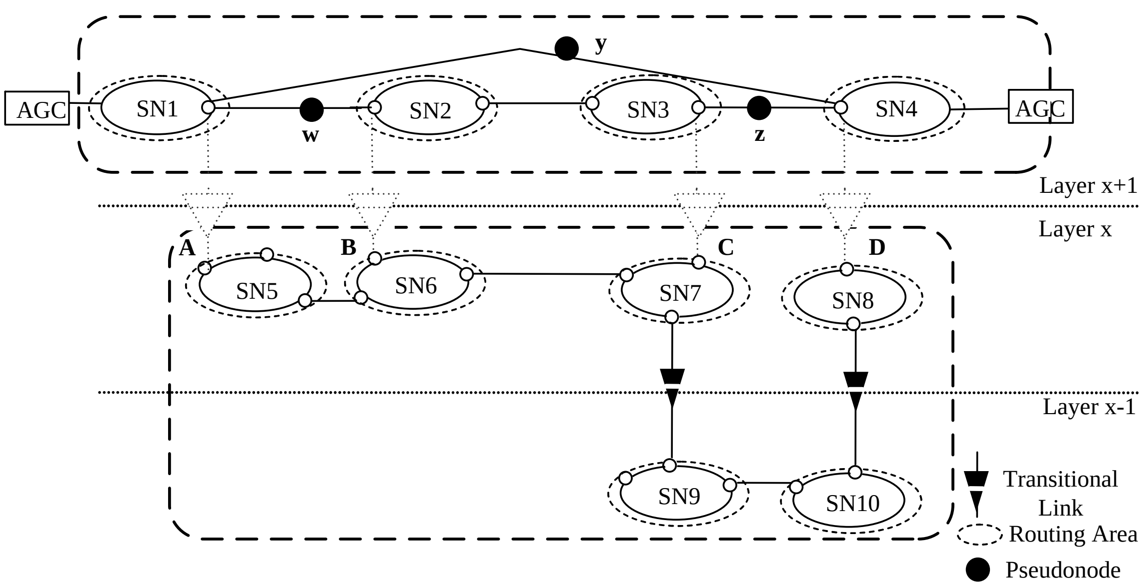

Figure 38: Control Plane Organization of Figure 36 - Routing.

Figure 39 shows a sequence of interactions between call controllers based on the control plane organization of Figure 36. Interactions with other components have been omitted for simplicity but follow the sequence of component interactions of [G8080]. The interlayer call invocation (e.g., NCC1 to NCC5) is analogous to a connection request, and the result should be a connection that the client layer is able to use via an adaptation. The example also assumes path computation returned a path through pseudo-node w, SN2, SN3 and z. A different path computation result would have resulted in involvement of different signaling components and different signaling sequences.

The example assumes call control messages are piggybacked onto connection control messages as described in [OIF-E-NNI-Sig-02.0].

For the interlayer NCC-NCC interaction, Figure 39 illustrates an example where the server-layer call is triggered by the client layer. It is possible that the server-layer call is already in place before the client-layer call (e.g. steps 4 to 6 are performed first, then steps 1 to 3, followed by 7 to 35).

The sequence also illustrates that path computation in the client layer drives the potential interlayer call invocations. Two layer x calls are used in the example. It would also be possible to use a single layer x call between NCC_5 and NCC_8. This is due to subnetworks SN6 and SN7 having a link between them. If path computation returned a route that used pseudo-node y, a single layer x call between NCC_5 and NCC_8 would have been used.

By having two layer x calls, the example shows an interlayer scenario in which client layer resources are used that are not at the ingress or egress points. This is the layer x+1 call created by NCC 2 and NCC3 within subnetworks SN2 and SN3, which is in the "middle" of the client-layer network (i.e. layer x+1).

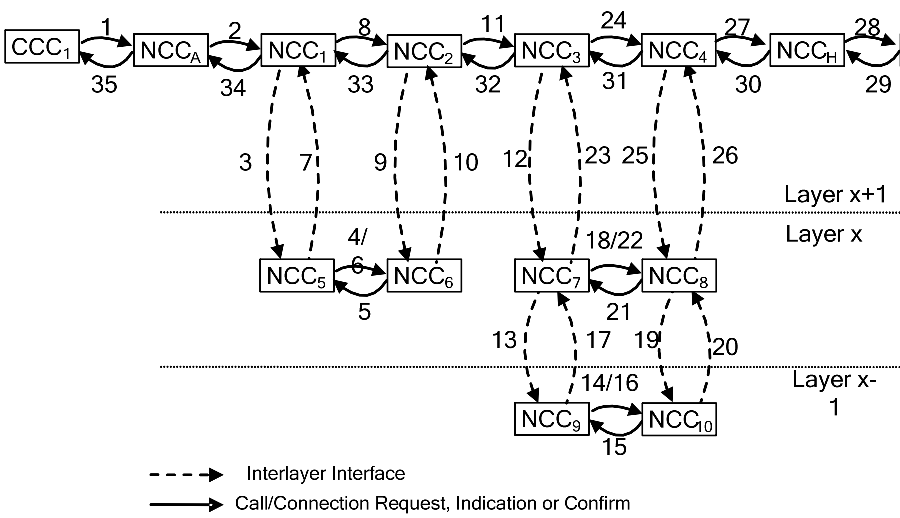

Figure 39: Interlayer NCC Interaction for Figure 36.

Step 1 – NCC_A receives a call setup request from CCC_1 for a call at layer x+1. It invokes an address resolution function with the source and destination TNAs (TNA e and TNA f) to obtain the source and destination SNPPs at layer x+1. The NCC invokes the CC with the SNPPs and other call parameters. The CC invokes path computation with the SNPP pair and obtains a path as follows: SN_1, pseudo-node w, SN_2, SN_3, pseudo-node z, SN_4.[3] The CC then initiates a connection in layer x+1, coordinating with other CCs.

Step 2 – The call/connection setup request progresses to NCC_1. The explicit route now contains pseudo-node w, SN_2, SN_3, pseudo-node z, SN_4. The presence of pseudo-node w as the next hop is an indicator that an interlayer call is necessary. An address resolution function is invoked with the ingress adaptation and egress adaptation client-layer SNPPs to be connected in the server layer as input. A pair of TNAs in the server layer is returned (TNA a and TNA b).

Step 3 – An interlayer call setup request is made between the client NCC (layer x+1) and server NCC (layer x). The request includes the TNAs a and b in the server layer x and any other parameters needed to obtain a server-layer connection that can satisfy the call parameters of the client's call request.

Step 4 – When NCC_5 receives a call setup request from NCC_1 for a call at layer x, it invokes an address resolution function with the source and destination TNAs provided (TNA a and b) to obtain the source and destination SNPPs at layer x. The NCC invokes the CC with the SNPPs and other call parameters. The CC invokes path computation with the SNPP pair and obtains a path as follows: SN_5, SN_6. The CC then initiates a connection in layer x, coordinating with other CCs. The call setup request is forwarded to the peer call controller in layer x, NCC_6.

Step 5 – Call setup indication is sent from NCC_6 to NCC_5. Once the connection is established in layer x at node 5, layer x+1 becomes aware of the new link that results from this server layer connection through the Termination and Adaptation Performer (TAP) and may trigger the discovery process in layer x+1 as determined by local policy. The discovery process is performed between the Discovery Agents (DA) on nodes 1 and 2 and consists of the exchange of names and identifiers for all control plane entities associated with the link. The discovery process is a parallel activity but it has to be completed before Step 8 can proceed below.

Step 6 – A call setup confirm is sent to NCC_6. Once the connection is established in layer x at node 6, layer x+1 becomes aware of the new link that results from this server layer connection through the Termination and Adaptation Performer (TAP) and may trigger the discovery process.

Step 7 – A successful call setup indication is sent to NCC_1.

Step 8 – Using the identifiers and names discovered during the the discovery process stage, the call/connection setup request progresses to NCC_2. The explicit route now contains SN_3, pseudo-node z, SN_4. The connection setup request indicates that the new server layer link is used for the connection between SN1 and SN2.

Step 9 – The new link resulting from the connection in layer x is reserved locally.

---

[3] The CC cannot distinguish pseudo-nodes from subnetworks in the path returned from path computation.

Step 10 – Successful call setup indication indicates the resources have been reserved.

Step 11 – The call/connection setup request progresses to NCC_3. The explicit route now contains pseudo-node z, SN_4. The presence of pseudo-node z as the next hop is an indicator that an interlayer call is necessary. An address resolution function is invoked with the pair of SNPPs in the client layer to be connected in the server layer as input. A pair of TNAs in the server layer is returned: TNAs c and d.

Step 12 – An interlayer call setup request is made between the client NCC (layer x+1) and server NCC (layer x). The request includes the TNAs c and d and any other parameters needed to obtain a server-layer connection that can satisfy the call parameters of the client's call request.

Step 13 – NCC_7 invokes an address resolution function with the TNAs c and d to obtain the local and destination SNPPs. The NCC invokes the CC with the SNPPs and other call parameters. The CC invokes path computation with the SNPP pair and obtains a path as follows: SN_7, SN_9, SN_{10}, SN_8. Because the link between SN_7 and SN_9 is a transitional link, an interlayer request is necessary. An address resolution function is invoked with the pair of SNPPs in the client layer to be connected in the server layer as input. A pair of TNAs in the server layer is returned: TNAs c2 and d2. An interlayer call setup request is made to the server NCC_9 in layer x-1.

Step 14 – When NCC_9 receives a call setup request from NCC_7 for a call at layer x-1, it validates the ingress and egress adaptation SNPPs supplied in the path against the TNAs provided. The NCC invokes the CC with the SNPPs and other call parameters. The CC invokes path computation to validate the supplied path: SN_9, SN_{10}. The CC then initiates a connection in layer x-1, coordinating with other CCs. The call setup request is forwarded to the peer call controller in layer x-1, NCC_{10}.

Step 15 – Similar to step 5

Step 16 – Similar to step 6

Step 17 – Similar to step 7

Step 18 – Similar to step 8

Step 19 – Similar to step 9

Step 20 – Similar to step 10

Step 21 – Similar to step 5

Step 22 – Similar to step 6

Step 23 – Similar to step 7

Step 24 – Similar to step 8

Step 25 – Similar to step 9

Step 26 – Similar to step 10

Step 27-28 – The call/connection setup request progresses to CCC_2.

Steps 28-35 – Call setup indication in layer x+1

6.6.2 Interlayer Processing Abstract Messages

The interlayer processing abstract messages refer to the messages used for interlayer control in the internal interlayer interface.

6.6.2.1 Abstract Messages and Error Codes

Interlayer processing abstract call messages are defined in Table 1. Abstract messages apply at ingress and egress adaptation.

Table 1: Interlayer Processing Abstract Call Messages.

Abstract Message	Message Direction
Call Setup Request	$NCC_{Client} \rightarrow NCC_{Server}$
Call Setup Indication	$NCC_{Server} \rightarrow NCC_{Client}$
Call Release Request	$NCC_{Client} \rightarrow NCC_{Server}$
Call Release Indication	$NCC_{Server} \rightarrow NCC_{Client}$
Call Query Request	$NCC_{Client} \rightarrow NCC_{Server}$
Call Query Indication	$NCC_{Server} \rightarrow NCC_{Client}$
Call Modify Request	$NCC_{Client} \rightarrow NCC_{Server}$
Call Modify Indication	$NCC_{Server} \rightarrow NCC_{Client}$
Call Notify	$NCC_{Server} \rightarrow NCC_{Client}$

Most attributes used in the abstract messages are derived from [OIF-E-NNI-Sig-02.0], as indicated in the abstract message tables. Figure 40 shows the interlayer interface.

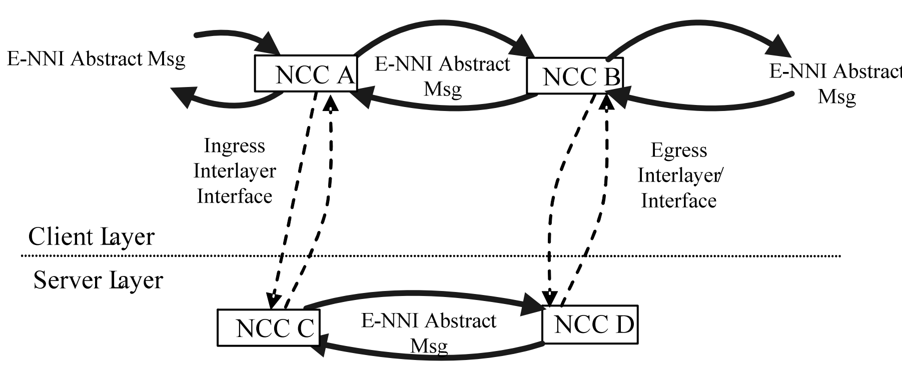

Figure 40: Interlayer Interface

6.6.2.2 Call Setup Messages

6.6.2.2.1 Call Setup Request

The Call Setup Request message is used by the client-layer NCC to request a server-layer call from the server-layer NCC. This does not automatically result in the creation of new server-layer calls, because it may also result in the binding of existing server-layer calls with the requesting client-layer call. Table 2 shows the abstract message and its content.

Table 2: Call Setup Request Abstract Message.

Attribute	Type
Source Server-Layer TNA, Logical Port ID	Mandatory TNA, optional Logical Port ID (Ingress adaptation) Not Applicable (Egress adaptation)[4]
Destination Server-Layer TNA, Logical Port ID	Mandatory TNA, optional Logical Port ID (Ingress adaptation) Not Applicable (Egress adaptation)4
Client-Layer NCC Name	Mandatory
Server-Layer NCC Name	Mandatory
Client-Layer Call Name	Mandatory
Directionality	Mandatory[5]
Ingress Server-Layer SNPP ID/SNP IDs	Optional (Ingress Adaptation) Not Applicable (Egress Adaptation)
Egress Server-Layer SNPP ID/SNP IDs	Not Applicable (Ingress Adaptation) Mandatory (Egress Adaptation)
Explicit Route	Optional (Ingress Adaptation) Not Applicable (Egress Adaptation)
Service Level	Optional[6]
Contract ID	Optional
Encoding Type	Mandatory
Switching Type	Mandatory
Technology specific traffic parameters	Mandatory
Generalized Payload Identifier	Optional
Bandwidth Modification Support	Optional
Server-Layer SRLG Exclusion List	Optional

6.6.2.2.2 *Call Setup Indication*

The Call Setup Indication message is used by a server-layer NCC to respond to a service request from a client-layer NCC. This message provides information needed by the client-layer NCC to complete the client-layer call and connection. In the case of setup rejection, this message includes error codes regarding the rejection. Table 3 shows the abstract message and its content.

---

[4] At the ingress adaptation, the source and destination server layer TNAs are used to initiate the server-layer call request. At the egress adaptation, the server-layer call already exists and only needs to be referenced by the SNP IDs associated with the server layer TCPs.

[5] This differs from UNI and E-NNI where this is an optional connection parameter.

[6] It is mandatory to have a service level associated with each call. If the attribute is not present, a default service level should be applied.

Table 3: Call Setup Indication Abstract Message

Attribute	Type
Client-Layer NCC Name	Mandatory
Server-Layer NCC Name	Mandatory
Client-Layer Call Name	Mandatory
Server-Layer Call Name(s)	Mandatory
Ingress Server-Layer SNPP ID/SNP IDs	Mandatory
Egress Server-Layer SNPP ID/SNP IDs	Mandatory
Server-Layer Call Status	Mandatory
Error Code	Optional
Bandwidth Modification Support	Optional
Server-Layer SRLG	Optional

The following error codes are defined for this message:

- Called Party Busy
- Unauthorized sender (policy error)
- Unauthorized receiver (policy error)
- Invalid source TNA
- Invalid destination TNA
- Unavailable directionality
- Invalid route
- Unavailable service level

6.6.2.3 Call Release Messages

6.6.2.3.1 Call Release Request

The Call Release Request message is used by the client-layer NCC when it no longer requires the server-layer resources. In turn, the server layer may release the server-layer call or only remove the binding between the client- and server-layer calls if the server-layer call is shared between various client-layer calls or the local policy dictates that the server-layer call should remain in place. No error codes are carried in this message. Table 4 shows the abstract message and its content.

Table 4: Call Release Request Abstract Message Attributes.

Attributes	Type
Client-Layer NCC Name	Mandatory
Server-Layer NCC Name	Mandatory
Client-Layer Call Name	Mandatory
Server-Layer Call Name	Mandatory

6.6.2.3.2 Call Release Indication

The Call Release Indication message is used by the server-layer NCC to respond to a release request. This message provides an indication that the binding to the server-layer call has been removed unless the request has been rejected as indicated by an error code. Table 5 shows the abstract message and its content.

Table 5: Call Release Indication Abstract Message Attributes.

Attributes	Type
Client-Layer NCC Name	Mandatory
Server-Layer NCC Name	Mandatory
Client-Layer Call Name	Mandatory
Server-Layer Call Name	Mandatory
Error Code	Optional

The following error codes are defined for this message:

- Unauthorized sender (policy error)
- Unauthorized receiver (policy error)
- Invalid or unknown Client-Layer Call Name
- Invalid or unknown Server-Layer Call Name
- Invalid or unknown SNPP ID or SNP IDs

6.6.2.4 Call Query Messages

6.6.2.4.1 Call Query Request

The Call Query Request message is used by a client-layer NCC to query server-layer call information. This message requests call-specific information for one (or more) established calls. If no Call Names are specified, all calls managed by the server-layer call controller used or available for use by the requesting client layer MUST be returned. If Call Names are specified, then information for all call information matching the server or client layer call names is returned. Table 6 shows the abstract message and its contents.

Table 6: Call Query Request Abstract Message Attributes.

Attributes	Type
Client-Layer NCC Name	Mandatory
Server-Layer NCC Name	Mandatory
(List of) Server-Layer Call Names	Optional
(List of) Client-Layer Call Names	Optional

6.6.2.4.2 *Call Query Indication*

The Call Query Indication message is used by the server-layer NCC to respond to a Call Query Request. If there are multiple calls, the following contents are repeated for each association:

- o Call-specific details

This message is specified as having a local scope. Table 7 shows the abstract message and its content.

Table 7: Call Query Indication Abstract Message Attributes.

Attribute	Type
Client-Layer NCC Name	Mandatory
Server-Layer NCC Name	Mandatory
Error Code	Optional
Call specific details below	
Client-Layer Call Name	Mandatory
Server-Layer Call Name	Mandatory
Call Status	Mandatory
Source Server-Layer TNA Name	Optional
Destination Server-Layer TNA Name	Optional
Local Server-Layer SNPP ID/SNP IDs	Optional
Contract ID	Optional
Encoding Type	Optional
Switching Type	Optional
Technology specific traffic parameters	Optional
Directionality	Optional
Explicit Route	Optional
Generalized Payload Identifier	Optional
Service Level	Optional
Bandwidth Modification Support	Optional

The following error codes are defined for this message:

- Unauthorized sender (policy error)
- Unauthorized receiver (policy error)

- Invalid or unknown Client-Layer Call Name
- Invalid or unknown Server-Layer Call Name

6.6.2.5 Call Notification Messages

6.6.2.5.1 Call Notification

The Call Notification message is used by a server-layer NCC to notify the client-layer NCC of the failure or recovery of an existing server-layer call. Table 8 shows the abstract message and its contents.

Table 8: Call Notification Abstract Message Attributes.

Attribute	Type
Server-Layer NCC Name	Mandatory
Client-Layer NCC Name	Mandatory
Server-Layer Call Name	Mandatory
Client-Layer Call Name	Optional
Call Status	Mandatory
Error Code	Optional

The following error codes are defined for this message:

- Service-affecting defect (resulting in failed connection)
- Non-service-affecting defect (no failed connection)

6.6.2.6 Call Modify Messages

6.6.2.6.1 Call Modify Request

The Call Modify Request message is sent by a client-layer NCC to request a non-disruptive modification to an existing call. This can result in non-disruptive call modification in the server layer through either non-disruptive connection modification or addition or removal of connections to the existing server-layer call. The mechanism used in the server layer is selected by the server layer and described in [E-NNI-2.0-Sig-02.0]. A client-layer call may also modify bandwidth by adding or removing server-layer calls binding to the client-layer connections. This is explained in Section 7.1. Table 9 shows the abstract message and its contents. The only attributes that may differ from the Call Setup Request for an increase in bandwidth is the Traffic Parameters Attribute. For a decrease in bandwidth, the Traffic Parameters Attribute and Server-Layer SNPP ID/SNP IDs may be modified.

Table 9: Call Modify Request Abstract Message Attributes.

Attribute	Type
Client-Layer NCC Name	Mandatory
Server-Layer NCC Name	Mandatory
Client-Layer Call Name	Mandatory
Server-Layer Call Name	Mandatory
Source Server-Layer TNA Name	Mandatory
Destination Server-Layer TNA Name	Mandatory
Server-Layer SNPP ID/SNP IDs	Mandatory
Contract ID	Optional
Encoding Type	Mandatory
Switching Type	Mandatory
Technology specific traffic parameters	Mandatory

6.6.2.6.2 Call Modify Indication

The Call Modify Indication message acknowledges the modification of the call. In the case of a modify rejection, this message includes error codes regarding the rejection. Table 10 shows the abstract message and its contents. In the case of a valid Call Modify Request to decrease the bandwidth of a connection, the Call Modify Indication MUST return success and remove the binding association between the requesting Client-Layer Call and Server-Layer Call. It is the responsibility of the server-layer NCC (1) to inform the management plane of the failure to decrease the bandwidth and (2) to re-attempt the decrease at the management plane’s request or automatically after a timer expires as dictated by local policy. The SNPP ID/SNP IDs may differ from the pair returned for the original call.

Table 10: Call Modify Indication Abstract Message Attributes.

Attribute	Type
Client-Layer NCC Name	Mandatory
Server-Layer NCC Name	Mandatory
Client-Layer Call Name	Mandatory
Server-Layer Call Name	Mandatory
Server-Layer SNPP ID/SNP IDs	Mandatory
Error Codes	Optional

The following error codes are defined for this message:

- Unauthorized sender (policy error)
- Unauthorized receiver (policy error)
- Invalid or unknown Client-Layer Call Name
- Invalid or unknown Server-Layer Call Name

- Invalid or unknown SNPP ID/SNP IDs
- Invalid source TNA
- Invalid destination TNA
- Unavailable Server-Layer Resources

6.6.3 Interlayer Signaling Interface Abstract Attributes

Table 11 lists the abstract attributes to support the interlayer interface call signaling, based on the attributes defined in [G7713], [OIF-UNI-02.0-Common], and [OIF-ENNI-Sig-02.0]. Most attributes are defined in one of these documents but the definitions are replicated here for completeness. In case of discrepancy between the documents, [OIF-ENNI-Sig-02.0] rules.

Table 11: Interlayer Signaling Abstract Attributes.

Abstract Attributes		Reference	Cross Reference
Identification Attributes	Server-Layer Source TNA and optional logical port ID	6.6.3.1.1	Transport Resource Identifier in [G8080]
	Server-Layer Destination TNA and optional logical port ID	6.6.3.1.1	Transport Resource Identifier in [G8080]
	Client-Layer NCC Name	6.6.3.1.2	CallC Name in [G7713]
	Server-Layer NCC Name	6.6.3.1.2	CallC Name in [G7713]
	Client-Layer Call Name	6.6.3.1.3	See [OIF-UNI-02.0-Common] and [OIF-ENNI-Sig-02.0]
	Server-Layer Call Name	6.6.3.1.3	See [OIF-UNI-02.0-Common] and [OIF-ENNI-Sig-02.0]
	Server-Layer SNPP ID/SNP IDs	6.6.3.1.4	[G7713]
Service Attributes	Directionality	6.6.3.2.1	Same in [OIF-UNI-02.0-Common] and [G7713]
	Encoding Type	6.6.3.2.2	[OIF-UNI-02.0-Common], Section 10.13.2.1
	Switching Type	6.6.3.2.3	[OIF-UNI-02.0-Common], Section 10.13.2.2
	Technology specific traffic parameters	6.6.3.2.4	See [OIF-UNI-02.0-Common] Section 10.13.2.3
	Bandwidth Modification Support	6.6.3.2.5	[G7713]
Routing Attributes	Explicit Route	6.6.3.3.1	See [OIF-ENNI-Sig-02.0].
	Server-Layer SRLG	6.6.3.3.2	For diverse paths required for recovery[7]
Policy Attributes	Service Level	6.6.3.4.1	This attribute is the same in [OIF-UNI-02.0-Common] and is a combination of “Class of Service” and “Grade of Service” in [G7713]
	Contract ID	6.6.3.4.2	From [OIF-UNI-02.0-Common]
Miscellaneous Attributes	Server Layer Call Status	6.6.3.5.1	From [OIF-UNI-02.0-Common]
	Error Code	6.6.3.5.2	From [OIF-UNI-02.0-Common]

6.6.3.1 Identification-Related Attributes

6.6.3.1.1 Server-Layer Source and Destination TNA

These attributes are described in Section 6.5.1. The Source and Destination TNAs are translated into server layer SNPP ID/SNP IDs.

6.6.3.1.2 Client- and Server-Layer NCC Name

These attributes are described in Section 6.5.3.

[7] Recovery extensions are outside the scope of this Amendment

6.6.3.1.3 Call Name

This attribute represents a globally- or carrier-unique name assigned to a call. The network call controller at the source of the call assigns the Call Name. When a client layer requests a call setup from a server layer, the server-layer NCC receiving the request assigns the call name. The Call Name is assumed to remain constant across call modification operations, i.e., across the “life” of the call.

6.6.3.1.4 SNPP ID / SNP ID

The combination of the SNPP ID and SNP ID attributes represent the transport plane link connection resource used to support the requested client service over the server-layer resources. It is returned by the server-layer NCC and identifies the TCPs for use by the adaptation from the client layer.

6.6.3.2 Service-Related Attributes

6.6.3.2.1 Directionality

This attribute indicates whether the requested service is for a unidirectional or bidirectional call service.

6.6.3.2.2 Encoding Type

The Encoding Type specifies the encoding format of the signal to be transported. The encoding options are specified in [OIF-ENNI-Sig-02.0].

6.6.3.2.3 Switching Type

This indicates the type of switching that should be performed on a particular link. The switching type options are specified in [OIF-ENNI-Sig-02.0].

6.6.3.2.4 Traffic Parameters

Traffic parameters are specified in [OIF-ENNI-Sig-02.0].

6.6.3.2.5 Bandwidth Modification Support

This indicates whether the call supports non-disruptive bandwidth modification.

6.6.3.3 Routing-Related Attributes

6.6.3.3.1 Explicit Route

This attribute represents a list of links and subnetworks that the requested service traverses. Depending on the amount of routing information exchanged between layers, this attribute may contain detailed route information or an abstracted view.

6.6.3.3.2 Server-Layer SRLG

This attribute represents a set of SRLGs that are collected in the server layer during connection establishment and that the client layer can use to request diversity between connections for recovery purposes.

6.6.3.4 Policy-Related Attributes

6.6.3.4.1 Service Level

This attribute contains information about the service level associated with the requested service. It is necessary to enable a consistent translation of "equivalent" service level information between layers.

6.6.3.4.2 Contract ID

The contract ID is locally determined by the client-layer call controller.

6.6.3.5 Miscellaneous Attributes

6.6.3.5.1 Call Status

This indicates the status of a call.

6.6.3.5.2 Error Code

The error code is used to describe the errors resulting from call actions:

- Called Party Busy
- Unauthorized sender (policy error)
- Unauthorized receiver (policy error)
- Invalid source TNA
- Invalid destination TNA
- Invalid or unknown Client-Layer Call Name
- Invalid or unknown Server-Layer Call Name
- Invalid or unknown SNPP ID/SNP IDs
- Unavailable directionality
- Unavailable service level
- Invalid route
- Service-affecting defect (resulting in failed connection)
- Non-service-affecting defect (no failed connection)
- Unavailable Server Layer Resources

6.7 SCN Models

Section 8 of [OIF-E-NNI-Sig-02.0] describes the SCN architecture that supports communications for single-layer operation. That architecture has two main variants, depicted in [OIF-E-NNI-Sig-02.0] Figure 16, depending on whether the total SCN is covered by a single SCN routing area or not. When distinct SCN RAs are present, manual provisioning of reachable SCN addresses between them is necessary.

This architecture is adequate for single-layer operation, because signaling takes place on a link-by-link basis and end-to-end signaling is not essential. Links are static structures, so manually provisioning the required SCN addresses is feasible.

However, as described in Section 6.6, the multilayer architecture assumes that each layer is supported by its own independent set of signaling associations. The end-to-end path computed across the server-layer network provides a segment of the client's end-to-end connection.

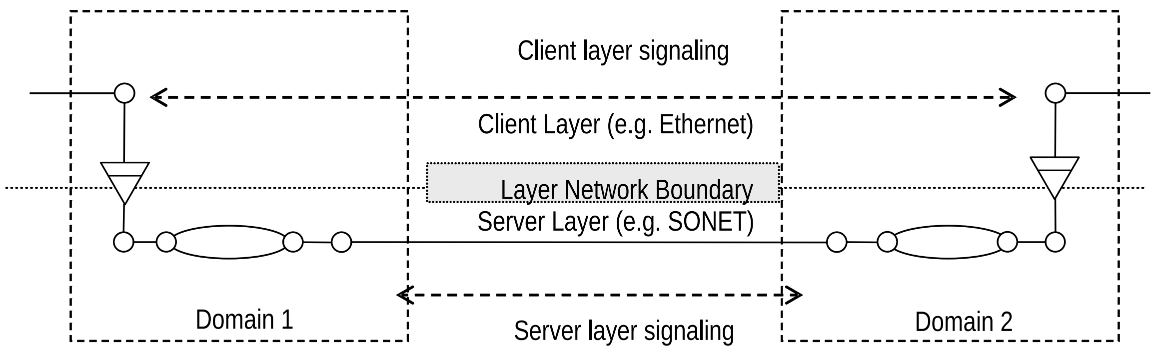

Figure 41: Client and Server Signaling Associations.

The client-layer signaling association is dynamically constructed as a result of the server-layer connection that supports the client's service. A consequence of this design is that the SCN must support message transport for these dynamically created signaling associations, and this leads to several SCN design implications.

To allow for the dynamic construction of a signaling relationship between any two client-layer controllers, the SCN must use the same address space across the entire routing area. All the controllers in a client layer must share a common SCN address space, so the SCN cannot be partitioned into independent SCN domains. This requirement is met by a common SCN address space, as in the client-layer E-NNI shown in Figure 32. The variants with distinct SCN address spaces, as in the client-layer E-NNI in Figure 33 and Figure 34, add additional SCN requirements to multilayer operations.

Within an operator's domain, the SCN may be partitioned for many reasons, and the requirement for a common SCN address space across all layers may not be easy to meet.

To use this implementation agreement in the case of a partitioned SCN, it is necessary to provide a new address space for signaling together with signaling gateways to allow signaling messages to be routed across the set of SCN domains. Such a gateway and protocol extensions are outside the scope of this implementation agreement.

6.7.1 Realization of Client-Layer Control-Plane Adjacencies

There are three different approaches to establish client-layer control plane adjacencies: (1) separate out-of-band adjacency; (2) client-layer adjacency carried by the server layer; and (3) private point-to-point link. Each of these offers multiple choices described below. Note that the second of these requires methods outside the scope of this implementation agreement.

The first and second approach can support Control Plane Arrangements 1, 2 and 3 illustrated in Figure 32, Figure 33 and Figure 34 respectively. The third approach is limited to Control Plane Arrangement 1 illustrated in Figure 32.

6.7.1.1 Approach 1: Separate Out-of-Band SCN Link

Two different scenarios need to be supported when using an out-of-band SCN link to carry client-layer signaling messages:

- A SCN in which a common address space is used by Nodes B and E (Figure 32)
- A SCN in which different address spaces are used by Nodes B and E (Figure 33 and Figure 34)

Because nodes B and E are not necessarily located at the edge of their domains, the ability to create an out-of-band SCN link between B and E is dependent on the structure of the SCN. If a common SCN address space is in use, the controller at the edge of the domain does not need to process the client-layer signaling messages. However, if different SCN address spaces are in use, then the controller at the edge of the domain needs to translate the signaling and routing messages when leaving and entering the domain. This translation can be done through one of two approaches:

- Using of a domain-edge proxy that is agnostic to the signaling information being exchanged[8]
- Establishing a client-layer E-NNI at Nodes C and D[9]

If Nodes B and E share a common SCN address space, or a domain-proxy is available, then Control Plane Arrangement 1 is possible. Control Plane

[8] It should be noted that the location of the domain-edge proxy is independent of the location of the server-layer resources used to support the client-layer link.

[9] It should be noted that the location of the client-layer E-NNI can be the same as or different from the location of the server-layer resources used to support the client-layer link. This is a major difference between Control Plane Arrangements 2 and 3.

Arrangements 2 and 3 are only possible when a client-layer E-NNI is established at Nodes C and D.

6.7.1.2 Approach 2: Client-Layer Adjacency Carried by Server-Layer Signaling Adjacency

Carrying client-layer signaling through the server-layer signaling adjacency can be accomplished in two different ways:

- Adding a sub-object to the call object to carry the client-layer signaling information
- Developing a general purpose mechanism to support communication between client-layer signaling entities

If a general-purpose mechanism is developed for the server-layer signaling function to exchange client-layer signaling messages and this client signaling is passed over the UNI interface, there is a potential that it could be abused by client layer. There is also a potential risk for the server layer to snoop the client layer information.

6.7.1.3 Approach 3: Private Point-to-Point link

Three different options exist for supporting a private point-to-point link dedicated to control plane signaling:

- Out-of-band, potentially established dynamically by the control plane
- In-band (client overhead)
- In-band (client payload)

An out-of-band link carries client-layer signaling messages over a different server-layer trail from the one carrying the client-layer link being controlled. This out-of-band link can only be used when the intervening server layer cross-connects are established.

An in-band link carries client layer signaling messages over the same server-layer trail as the client-layer link connections being controlled. These messages could be contained in the client-layer overhead (i.e., the DS1 FDL) or carried in the client-layer link connections themselves (i.e., in a separate Ethernet VLAN). Because different client-layer links have different capabilities (e.g., overhead availability, existence of multiple link connections, etc.), the definition of how a client-layer carries signaling messages is client-layer-technology specific.

The private point-to-point link approach differs from approach 1 by making the private link unavailable to other SCN traffic. This is done by not advertising the link in the SCN routing topology. Since the private point-to-point link is not advertised, it does not require sharing a common SCN address space with other SCN links.

7 Multilayer Feature Set

This document provides an interlayer description that can help provide automatic establishment of interlayer services across a variety of technologies and following several different user scenarios. Support of various technologies is described in the protocol specific amendments for RSVP-TE [OIF-ENNI-RSVP-02.1] and OSPF-TE [OIF-ENNI-OSPF-02.1]. Protocol definitions may evolve over time to include more technologies.

Regardless of the technologies used, the interlayer interface is designed to support all types of server-to-client relationships described in Section 6.1. The interlayer interface is also designed to support various use cases in which the trigger for server connection establishment may or may not depend upon client-layer requirements. The interlayer interface also allows for a mix of control-plane-enabled layers and manually provisioned layers.

The interlayer interface allows for support of the following features already supported by [OIF-E-NNI-Sig-02.0]:

- Call and Connection Separation
- Non-disruptive modification of service parameters

7.1 Call Modification

Several mechanisms can be used to provide non-disruptive bandwidth modification across multiple layers. Call Modification in the client layer may result in the following in the server layer:

- Server-layer call modification that increases or decreases bandwidth of an existing server-layer connection. For example, an Ethernet 5 Gbps server layer call is realized by an ODUflex[10] 5 Gbps call and connection. The client layer call is increased to 7 Gbps and the ODUflex server layer call and connection is increased to 7Gbps.
- Server-layer call modification that increases or decreases the number of server-layer connections associated with a call. For example, a VC-4-5v client-layer call is realized by one server layer call with five VC-4 server-layer connections. The VCAT client layer call is modified to VC-4-6v by modifying the VC-4 server-layer call to add another VC-4 server layer connection.
- Server-layer call release or server-layer call setup. For example, a VC-4-5v client-layer call is realized by five VC-4 server-layer calls. The VCAT client layer call is modified to VC-4-6v by setting up an additional VC-4 server-layer call. Note that this mechanism may be used to change the client-layer to server-layer relationship by releasing a server-layer call in one server layer and setting up a new server-layer call in another server layer.

[10] ODUflex was not yet supported by the protocol-specific Amendments [OIF-ENNI-RSVP-02.1][OIF-ENNI-OSPF-02.1] at the time this Implementation Agreement was approved.

- No change in the server-layer call or connection itself, but the association between the server- and client-layer call is modified. For example, an Ethernet 300 Mbps client-layer call is realized by an STS-48c/VC-4-16c server-layer call and connection. Modifying the bandwidth of the Ethernet layer to 500 Mbps does not require any modifications to the call and connection in the server layer. In this example, sharing of the server-layer call may be allowed between client-layer calls, and it may also be possible to add four other 500 Mbps Ethernet calls onto the same server-layer call.

The following sub-sections apply to a model where the client and server layers are control-plane enabled. The following message flows assume that when a client-layer call controller receives a Call Modify Request (whether it is an ingress client call controller, an upstream network call controller or a client-layer call controller) and it is at an interlayer boundary, it determines whether the request is for an increase or a decrease and which mechanism for non-disruptive modification is available and best suited to the request. Requests that decrease the bandwidth are sent downstream in the client layer before being sent to the server layer. This ensures downstream client layer call controllers validate the decrease request before the bandwidth is actually decreased in the connections. Requests that increase the bandwidth are sent to the server layer to ensure local bandwidth capacity is available before forwarding the request downstream in the client layer.

The client-layer call modification is illustrated in Figure 42 and Figure 43. It is well suited to many types of relationships between Call and Connection controllers in the server layer as shown in Figure 44 through Figure 47. The examples of this section are based on the message sequences across a multilayer E-NNI based on a server-layer E-NNI and a control-plane only client-layer E-NNI as illustrated in Figure 32.

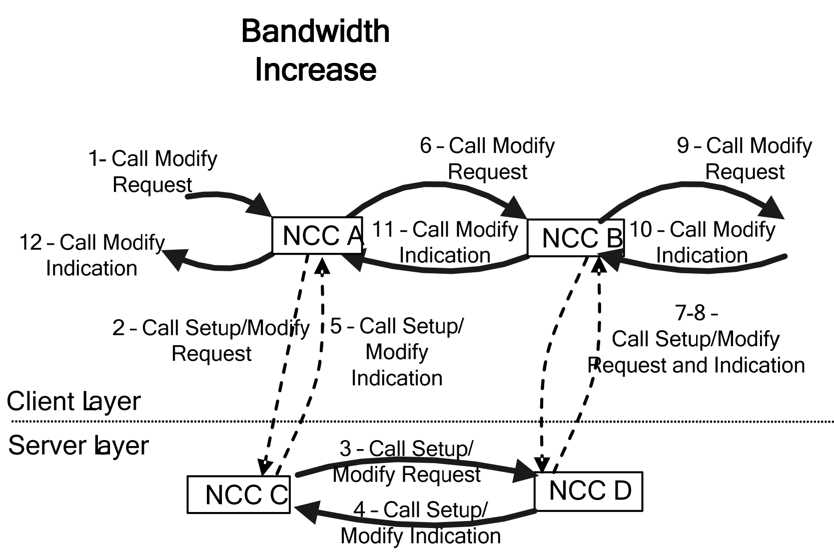

Figure 42: Interlayer Call Modification Message Flow–Bandwidth Increase.

Table 12 – Interlayer Call Modification Steps – Bandwidth Increase

Step	Description
1	When a client-layer NCC receives a Call Modify Request, it validates the request and determines whether it is an increase or a decrease. The following steps are based on an increase.
2	The client-layer NCC sends a Call Modify Request message to the server-layer NCC, including the new traffic parameters that contain the new desired bandwidth.
3-4	Call Modification is performed across the server layer as described in [OIF-E-NNI-Sig-02.0]. This can be achieved by modifying existing server-layer connection(s), by adding server-layer connection(s) or setting up server-layer call(s).
5	A Call Modify Indication is returned to the client-layer NCC and potentially provides a new client-layer SNP ID.
6	Call modification continues in the client layer across the E-NNI with a Call Modify Request as described in [OIF-E-NNI-Sig-02.0].
7-8	The client-layer NCC sends a Call Modify Request message to the server-layer NCC and includes the new traffic parameters and SNP ID.
9-12	Call modification completes in the client layer based on [OIF-E-NNI-Sig-02.0]

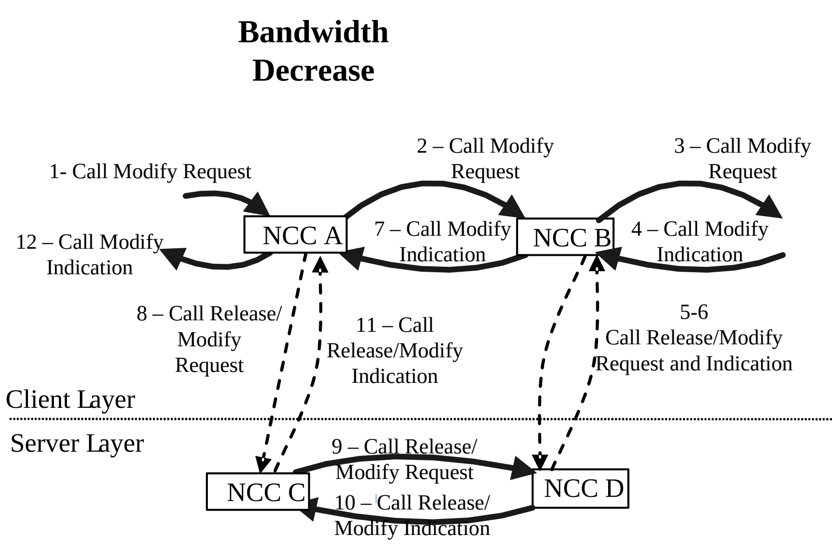

Figure 43: Interlayer Call Modification Message Flow–Bandwidth Decrease.

Table 13 – Interlayer Call Modification Steps – Bandwidth Decrease

Step	Description
1	When a client-layer NCC receives a Call Modify Request, it validates the request and determines whether it is an increase or a decrease. The following steps are based on a decrease.
2-3	Call modification continues in the client layer across the E-NNI with a Call Modify Request as described in [OIF-E-NNI-Sig-02.0].
4	A Call Modify Indication is received at NCC B in client layer as described in [OIF-E-NNI-Sig-02.0].
5-6	The client-layer NCC sends a Call Modify Request message to the server-layer NCC, including the new traffic parameters that contain the new desired bandwidth.
7	Call modification continues in the client layer across the E-NNI with a Call Modify Request as described in [OIF-E-NNI-Sig-02.0].
8	The client-layer NCC sends a Call Modify Request message to the server-layer NCC and includes the new traffic parameters and SNP ID.
9-10	Call Modification is performed across the server layer as described in [OIF-E-NNI-Sig-02.0]. This can be achieved by modifying existing server-layer connection(s), by removing server-layer connection(s) or releasing server-layer call(s).
11	A Call Modify Indication is returned to the client-layer NCC and potentially provides a new client-layer SNP ID.
12	Call modification completes in the client layer based on [OIF-E-NNI-Sig-02.0].

Client-layer call modification by modifying existing server layer connections is well suited to a 1:1 relationship between client- and server-layer calls and a 1:1 relationship between server-layer call and server-layer connections as shown in Figure 44.

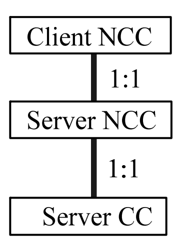

Figure 44: Server Layer 1:1 Relationship between Call and Connections.

Client-layer call modification by adding or removing server-layer connections is well suited to a 1:1 relationship between client- and server-layer calls and a 1:N relationship between server-layer call and server-layer connections as shown in Figure 45. It is also well suited to a 1:N relationship between a client layer call and calls and connections in the server layer as shown in Figure 46.

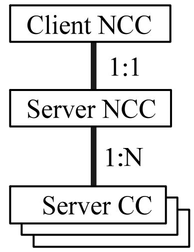

Figure 45: Server Layer 1:N Relationship between Call and Connections

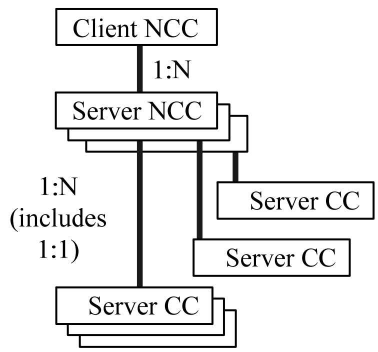

Figure 46: 1:N Relationship between Client- and Server-Layer Call Controllers.

7.1.1 In some scenarios it is possible for the server-layer call to be shared among multiple client-layer calls. This is illustrated in Figure 47. For this case, call modification may only modify the binding of the client layer to server layer call.

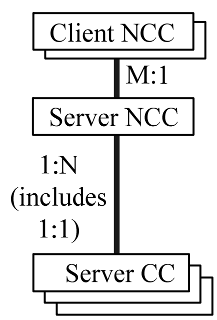

Figure 47: Client-Layer Calls Sharing a Server-Layer Call.

When resources are shared or independently established, or when the server-layer capacity exceeds the client-layer request, the request for a new call or modification to an existing call may not result in any modifications of calls or connections in the server layer. Instead, the server-layer resources are bound to the client-layer call through the same Call Setup Request and Call Setup Indication, unbound using the Call Release Request and Call Release Indication, and the binding is modified using the Call Modify Request and Call Modify Indication. Steps 3-4 of bandwidth increase (Figure 42) and steps 9-10 of bandwidth decrease (Figure 43) are not performed.

7.1.2 Failure Scenarios

A failure to increase the bandwidth in the server layer results in an error code in the Call Modify Indication sent to the client-layer NCC. In this case, steps 6-11 in Figure 42 are skipped and a Call Modify Indication with an error code is sent in step 12.

A failure to decrease the bandwidth in the server layer results in the server-layer NCC informing the management plane of the failure to decrease the bandwidth but does not otherwise impact the message flow. It is the responsibility of the server-layer NCC to re-attempt a bandwidth decrease after a timeout period or when instructed to do so by the management plane, as local policy dictates.

A failure to increase the bandwidth downstream of the client layer NCC B as a result of the destination's CCC or any upstream NCC rejected the request and indicated by an error code in the Call Modify Indication in step 10, results in locally decreasing the bandwidth, i.e., undoing the bandwidth increase, before sending a Call Modification Indication with an error code towards NCC A in step 11. If the decrease is unsuccessful, the behavior is as described in the previous paragraph. Similarly, a failure to increase the bandwidth at NCC B, indicated by an error code in the Call Modify Indication in step 11 results in locally decreasing the bandwidth, i.e., undoing the bandwidth increase, before sending a Call Modification Indication with an error code in the upstream direction in step 12.

A failure to decrease the bandwidth downstream of the client-layer NCC as a result of the destination's CCC rejecting the request and indicated by an error code in the Call Modify Indication in steps 4 and 7 results in no interlayer interactions at the local node. Steps 5-6 and 8-11 of the decrease are skipped, and the Call Modify Indication is sent in the upstream direction with an error code in step 12.

8 Security and Logging for Multilayer Signaling

Methods for securing the OIF's control plane protocols are specified in the OIF's existing signaling and routing implementation agreements. See, for example, [OIF-E-NNI-Sig-02.0]. This amendment introduces no new control plane protocols.

It should be pointed out, however, that control plane protocols active at multiple layers may have different pairs of endpoints each requiring its own protocol security. These different pairs of endpoints may, in fact, have different security policies requiring different security services.

If control plane logging with syslog as specified in [OIF-SLG-01.1] is implemented, then the protocol errors listed in this implementation agreement SHOULD be logged with a Severity higher (i.e., lower-numbered) than Informational. Also, to verify the correct operation of multilayer functionality, the PROT@26041 logging capability can be turned on at one or more network elements implementing the multilayer capability to generate a secure, time-stamped trace of control plane traffic.

9 Compatibility with UNI and E-NNI

The signaling aspects of this implementation agreement are backwards compatible with UNI 2.0 [OIF-UNI-02.0-Common] and E-NNI 2.0 [OIF-ENNI-Sig-02.0] as described in Section 9.1. The routing aspects of this implementation agreement are backwards compatible with E-NNI 2.0 [OIF-ENNI-OSPF-02.0] as described in Section 9.2. Security and logging considerations are covered in Section 8. Other aspects of the multilayer amendment, including discovery, are not included in other OIF IAs and, as such, do not cause any interoperability concerns. This section only covers generic backwards compatibility. Protocol-specific backwards compatibility details are found in the corresponding protocol-specific Implementation Agreements [OIF-ENNI-RSVP-02.1][OIF-ENNI-OSPF-02.1].

9.1 Signaling Backwards Compatibility

This IA does not introduce any new UNI signaling messages or attributes and fully supports all UNI signaling messages and attributes defined in UNI 2.0. This IA describes the relationship between a UNI request at a specific client layer and the mappings of this request across an interlayer interface to establish connectivity at a server layer. This requires compliance with this IA from implementations that provide adaptation between a client and a server layer. Implementations that do not provide adaptation are single layer and only need to comply with the existing UNI or E-NNI IA. In other words, the minimum requirements to provide multilayer signaling across a network that implements UNI and E-NNI is UNI 1.0 and E-NNI 1.0 with the signaling controllers at the adaptation point also compliant with this IA. [OIF-E-NNI-Sig-02.0] describes the compatibility limitations when interworking with E-NNI 1.0.

The scope of this IA is limited to providing services between UNIs at the same layer, whether the UNI is signaled (SC) or not (SPC).

9.2 Routing Backwards Compatibility

This IA introduces two new routing constructs–the pseudo-node and the transitional link. The backwards compatibility issues for these constructs are different and described separately below.

Pseudo-nodes hide all of the details of the layer boundary crossing within the pseudo-node abstraction. For routing controllers outside of the pseudo-node control domain, the routing advertisement appears as an ordinary node. As a result, existing E-NNI Routing 2.0 implementations are still able to compute a path that goes through the pseudo-node. However, they will not be able to participate in the pseudo-node control domain as the specific procedures for client-to-server layer call invocation would not be supported.

Transitional links are a completely new construct requiring new link attribute information (e.g., the client and server layers connected, the adaptation method used, and any adaptation configuration parameters) and a modification to the path computation process (to keep track of a layer stack and adaptation stack). Consequently, transitional links are not supported by existing E-NNI Routing 2.0

implementations. This does not mean that transitional links may not be used within a network consisting of E-NNI Routing 2.0 implementations. There are two approaches in this case:

- It is possible to hide the transitional links within an abstract node (as shown in Figure 48). Note that the mechanism used to perform node abstraction is outside the scope of this implementation agreement. In this example, the client layer subnetworks 2, 3, and 5 participate in the abstract node abstraction as described in this IA, but the client layer subnetworks 1, 4, and 6 are only required to implement the E-NNI Routing 2.0 IAs.
- If the transitional links are not hidden in an abstract node, E-NNI Routing 2.0 implementations will ignore this attribute as well as any LSAs for layers that are not supported. This approach does not allow E-NNI 2.0 implementations to compute path across multiple layers.

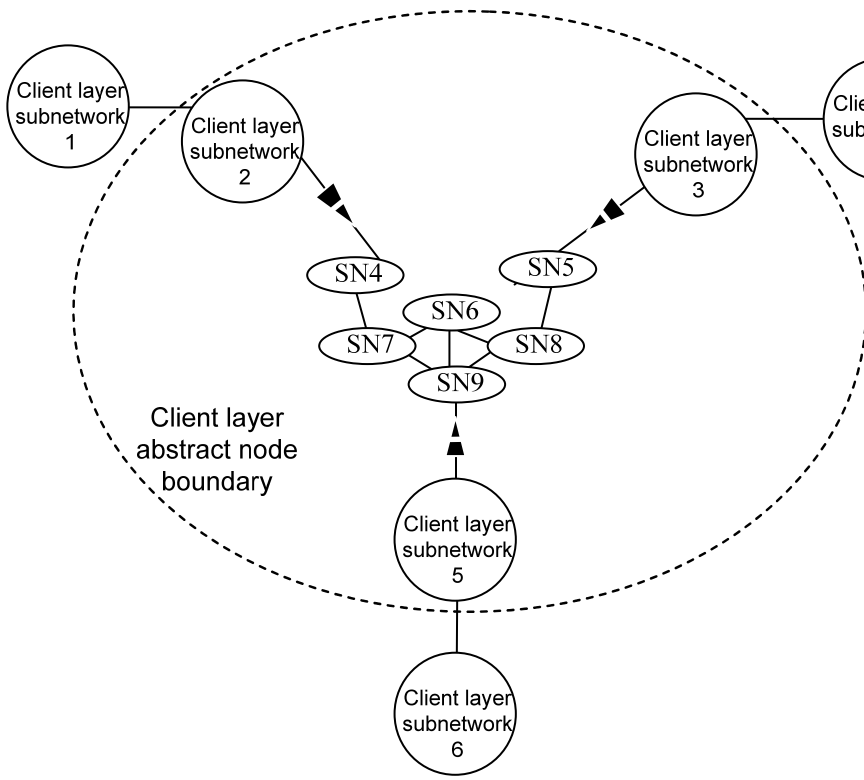

Figure 48: Routing Backwards Compatibility Using an Abstract Node.

10 References

[G709] ITU-T Recommendation G.709/Y.1331 (2003), “Interfaces for the optical transport network (OTN)”, March 2003

[G7712] ITU-T Recommendation G.7712/Y.1703 (2008), “Architecture and specification of data communication network”, June 2008

[G7713] ITU-T Recommendation G.7713/Y.1704 (2009), "Distributed Call and Connection Management (DCM)", November 2009

[G7715] ITU-T Recommendation G.7715/Y.1706 (2002), Amendment 1 (2007), "Architecture and Requirements for routing in the automatically switched optical networks", February 2007

[G7715.1] ITU-T Recommendation G.7715.1(2004), "ASON routing architecture and requirements for link state protocols", February 2004

[G800] ITU-T Recommendation G.800 (2012), "Unified functional architecture of transport networks", February 2012

[G8011.1] ITU-T Rec. G.8011.1/Y.1307.1 (2009), "Ethernet Private Line Service", January 2009

[G8011.2] ITU-T Rec. G.8011.2/Y.1307.2 (2009), "Ethernet Virtual Private Line Service", January 2009

[G805] ITU-T Recommendation G.805 (2000), "Generic functional architecture of transport networks", March 2000

[G8080] ITU-T Recommendation G.8080/Y.1304 (2012), "Architecture for the Automatically Switched Optical Network (ASON)", February 2012

[G809] ITU-T Recommendation G.809 (2003), "Functional architecture of connectionless layer networks", March 2003

[OIF-ENNI-OSPF-01.0] OIF Implementation Agreement, "External Network-Network Interface (E-NNI) OSPF-based Routing - 1.0 (Intra-Carrier)", January 2007

[OIF-ENNI-OSPF-02.0] OIF Implementation Agreement, "External Network-Network Interface (E-NNI) OSPF-based Routing 2.0 (Intra-Carrier) Implementation Agreement", July 2011[OIF-ENNI-SIG-01.0] OIF Implementation Agreement OIF-E-NNI-Sig-01.0, "Intra-Carrier E-NNI Signaling Specification", February 2004

[OIF-E-NNI-Sig-02.0] OIF Implementation Agreement, "OIF E-NNI Signaling Specification (OIF E-NNI 2.0)", April 2009

[OIF-ENNI-OSPF-02.1] OIF Implementation Agreement, "Multilayer Amendment to E-NNI 2.0 OSPFv2-based Routing", April 2013

[OIF-ENNI-RSVP-02.1] OIF Implementation Agreement, "Multilayer Amendment to E-NNI 2.0 – RSVP-TE Signaling", April 2013

[OIF-SLG-01.1] OIF Implementation Agreement OIF SLG-01.1, “OIF Control Plane Logging and Auditing with Syslog”, July 12, 2010

[OIF-UNI-01.0-R2] OIF Implementation Agreement OIF-UNI-01.0-R2, “User Network Interface (UNI) 1.0 Signaling Specification, Release 2: Common Part”, February 2004

[OIF-UNI-02.0-Common] OIF Implementation Agreement, “User Network Interface (UNI) 2.0 Signaling Specification: Common Part,” February 2008.

[RFC2702] IETF RFC 2702, Requirements for Traffic Engineering over MPLS

[RFC2119] IETF RFC2119, Key words for use in RFCs to Indicate Requirement Levels

[RFC3630] IETF RFC3630, Traffic Engineering (TE) Extensions to OSPF Version 2

11 Appendix I: Use Cases

11.1 SCN and Topology Choices

The choice of which topology representation to choose is outside of the scope of this document, however this appendix gives some considerations for both network and network element designers.

Transitional links, pseudo-nodes, and abstract nodes provide views of both topology and policy. The transitional link description allows for the creation of a full topology graph so that all details of the server topology are available for path computation. It is useful to compare this with a single routing level in which all details are available.

Just as a layer network may have several routing levels to control scaling or topology visibility, the multilayer network also needs to control scaling and topology visibility. The transitional link representation allows for multiple levels within a layer, so in principle, visibility and scaling can be controlled to any desired degree. However the special case of wrapping the whole server layer network is very common as that best fits the completely independent layer model.

The pseudo-node wraps the complete server layer so that a client layer has no visibility into the server layer topology and only has a signaling relationship with the server layer. One consequence of this wrapping is that some manual provisioning is necessary; however that provisioning is limited to the client layer only and can be limited to provisioning a node ID for the pseudo node for each link that enters that node.

An abstract node wraps a set of nodes in the same layer so that internal topology is now hidden. Like the pseudo-node, it requires provisioning of the abstract node ID on the NEs at the edge of the abstract node. Unlike the pseudo-node, link advertisements into an abstract node create a routing association between the controllers at the end of the links.

In addition to the three possible topology forms, variants are also created by the choice of function assignment to equipment. This creates two more variations depending on whether the server layer access link is between NEs or is within a single NE. Yet more variation is created by SCN partitioning choices.

Coupling between the layers is one aspect of design, and coupling between network elements is another aspect. The associations among the components needed to support the three views are distribution independent and create coupling among the network elements involved, which depends on how the layers are placed on the network element. SCN partitioning choices force some associations to be supported between different SCNs as described in section 6.3.4.

The pseudo node clearly creates the least coupling because it only requires a signaling association with the server. Transitional links and abstract nodes need both routing and signaling associations between the NEs and are perhaps most suitable when both layers are on the same NE. Abstract nodes hide internal topology without creating new routing levels.

12 Appendix II: Multilayer Requirements

This section captures requirements that were used as the basis for the development of this Implementation Agreement. Not all requirements are supported in this version of the Implementation Agreement. For example, not all SCN configurations are supported.

12.1 Addressing of Signaling Entities

- It must be permitted to associate TNAs with access points at layers for which there are no UNI resources, but are involved in the endpoint(s) of call.
- It must be possible to assign TNAs to access points at different layers in a multilayer control plane environment.
- TNAs for different layers may be in different namespaces depending on the control plane instantiation.
- SNPPs for different layers MUST be distinguishable, as they are in different namespaces. An SNPP has multiple components that may include routing area identifier, subnetwork identifier, and resource context identifier. Note that they may be assigned the same value, but this would be by operational intent.
- Node IDs (smallest subnetwork) at different layers are in different address spaces and MUST be distinguishable. This is a corollary of the fact that SNPPs at different layers are in different namespaces. Note that Node IDs at different layers may be assigned the same value, but this would be by operational intent.
- A VCAT adaptation with a specific server CI MAY have a single TNA associated with it for any possible numbers of constituents. That is, the same TNA may be used when a VCAT group changes size. For example, the same TNA may be used for an instance from the set VC4-1v through VC4-256v. This accommodates the “floating” nature of a VCAT layer.
- Should a VCAT function be capable of using different fundamental signals, e.g., VC-3 or VC-4, each different signal MAY have a TNA associated with it for all possible numbers of constituents.
- SC PC IDs at different layers are in different identifier spaces. Note that SC PC IDs may be assigned the same value, but this would be by operational intent.

12.2 Addressing of Routing Entities

- Given different network elements participate in different sets of network layers, and network layers may be assigned to different business entities, it MUST be possible to utilize separate confederations of routing controllers for one or more layer supported by a network element.

- When separate confederations of routing controllers are used in different layers, the routing controllers MUST limit the routing information exchanged with peer routing controllers to the information specific to that confederation.
- RC IDs are used to identify the originator of routing information being exchanged in a confederation of routing controllers. Since the confederations may be separately administered, the RC ID namespace must be specific to the confederation. Note that the same value MAY be used by routing controllers in different confederations supported on the same network element, but this would be by operational intent.

12.3 Address Resolution

- The binding between SC PC IDs at different layers in an interlayer instance MAY be configured or the binding may be set by the implementation.
- The address resolution function that maps a TNA to an SNPP identifier in the same layer MUST be available to the NCC. Note that the SNPP identifier may identify an E-NNI link transitioning into a neighboring domain instead of identifying the far-end UNI point. Subsequent domains may, as a result, perform the resolution function again.
- The interlayer address resolution function that maps an SNPP in the client layer to a TNA in the server layer MUST be available to the client layer NCC. This includes local and remote SNPPs resolution into local and remote TNAs.

12.4 Services Between Layers

- It MUST be possible to recognize when the processing of the call is at the edge of an interlayer boundary. This is to trigger an interlayer call invocation. An example mechanism for this is an indication in a path returned from a route server.
- When a choice between existing client- or server-layer resources exists, the decision to use client- or server-layer resources MUST be based on local policy.
- When a choice of different server layers exists to satisfy an interlayer request, local policy MUST be used. Note that local policy may include interactions with other control plane components, i.e., routing controllers, to identify which server layer to use.

12.5 SCNs

- When multiple SCNs exist in a single layer, if connectivity is possible and allowed by policy, it MUST be possible to have control messages traverse them.

- The multilayer control plane solution MUST be able to support configurations where the server layer SCN is distinct from the client layer SCN.
- The multilayer control plane solution MUST be able to support the following SCN topologies:
    - Single SCN across all interfaces at all layers.
    - Multiple SCNs across the server layer, common SCN across the client layer.
    - Common SCN across the server layer, multiple SCNs across the client layer,
    - Multiple SCNs across the server layer, multiple SCNs across the client layer, symmetrical E-NNI. See Figure 34.
    - Multiple SCNs across the server layer, multiple SCNs across the client layer, no correspondence between the SCNs at the client and server layer.
    - Common SCN shared between adjacent layers. Figure 33 shows this scenario.

12.6 Routing

- When separate confederations of routing controllers are used to perform routing in different layers, the path computation performed by each layer is independent.
- In order to facilitate independent computation when separate confederations of routing controllers are used in different layers, the client layer routing topology MUST contain a representation of any potential connectivity made available by a server layer.
- The Client layer representation of server layer potential connectivity MUST be provided using attributes and identifiers appropriate for the client layer.

12.7 Discovery

- The flexibility provided by a server layer allows any two client layer network elements to potentially form adjacencies as a result of the establishment of server-layer connections. Since the control plane requires the establishment of signaling and routing adjacencies to allow the use of the link resulting from the establishment of a server-layer connection, the control plane needs to be able to identify the protocols and operational parameters to configure these adjacencies. The protocol and operational parameters MUST be specified by the network operator or MAY be exchanged in the server layer signaling.

- The specification of protocol and operational parameters used on a link SHOULD be done in the form of policy that is driven by the identity of the adjacent network element.

12.8 Client/Server Entity Relationships

- The multilayer control plane solution MUST be able to support a 1:1 relationship between a client layer link connection and server layer trail. For example, a 2Gbps Ethernet call can be mapped to a single STS-48c/VC-4-16C.
- The multilayer control plane solution MUST be able to support a 1:n relationship between a client layer link connection and a server layer trail. For example, a 250 Mbps Ethernet call can be mapped to 2 VC-4 calls to form a VC4-2v call.
- The multilayer control plane solution MUST be able to support a m:1 relationship between a client layer link connection and a server layer trail. For example, two Ethernet EVPL calls at 10 Mbps could be multiplexed onto the same VC-3.
- Each layer has its associated NCCs and therefore different call names MUST be allocated at different layers. The call names come from the same name space and cannot be the same for two calls, even if the calls are in different layers.
- The multilayer control plane solution MUST be able to support client-layer control-plane only ENNIs. This allows the client connection to be supported as combination of mapped server and intermediate client layer connections.
- The multilayer control plane solution MUST be able to support client layer subnetworks. This allows the client connection to be supported as combination of mapped server and intermediate client layer connections.
- Client layer link attributes MUST be able to inherit server layer link attributes, e.g. risks.

12.9 Service Initiation

- A server layer call MAY exist before a client layer call uses it. These resources may be used to satisfy a client layer call request.
- The ability to create server layer calls through a management system MUST be supported. The server layer calls terminate at the server layer access points and can later be used by one of its client layers. Those "SPC" server layer calls are owned by the management system and are therefore not deleted when the client layer calls are "detached" from them.
- A server layer MAY create a call (at its layer) if a client layer request cannot be satisfied with the existing calls at the server layer.

- When client layer call is released in an interlayer arrangement, local policy SHOULD be used to determine if the associated server layer calls may be released. Note that the server layer MUST take into account the type of relationship (1:1, 1:n, m:1, m:n) with the client layer and whether the call was triggered by the management system or the client layer.

12.10 Call Modification

- Call and connection setup MUST complete in the server layer before continuing in the client layer.
- Call and connection release MUST complete in the client layer before continuing in the server layer.
- Bandwidth increase across multiple layers MUST be done in the server layer first.
- Bandwidth decrease across multiple layers MUST be done in the client layer first.
- Bandwidth increase/decrease SHOULD be triggered at the source server layer NCC.

13 Appendix III: Routing with Transitional Links

13.1 Background

The introduction of Transitional Links into E-NNI 2.0 Routing changes the method used for per-layer routing. The E-NNI 1.0 and 2.0 routing specifications follow the method defined for MPLS-TE [RFC2702], which treats the layer of a service request as a constraint to be applied to a link, causing the link to be removed from the graph if that layer was unavailable on a link. This creates a sub-graph which is used for path computation. An example for a three layer network is shown in Figure 49.

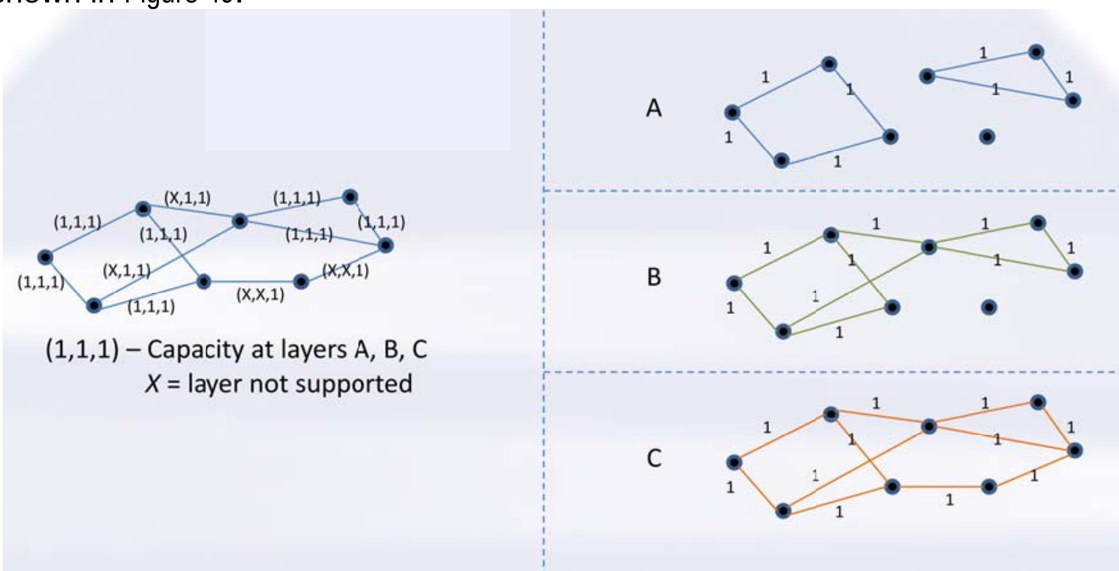

Figure 49: Layer-specific Graph Filtering

Transitional links require a change from this method as they represent adapting the signal at one layer to be carried over another. This means transitional links connect the per-layer sub-graphs, allowing paths to be computed that previously were not possible. Figure 50 shows the same network as Figure 49, but adds transitional links.

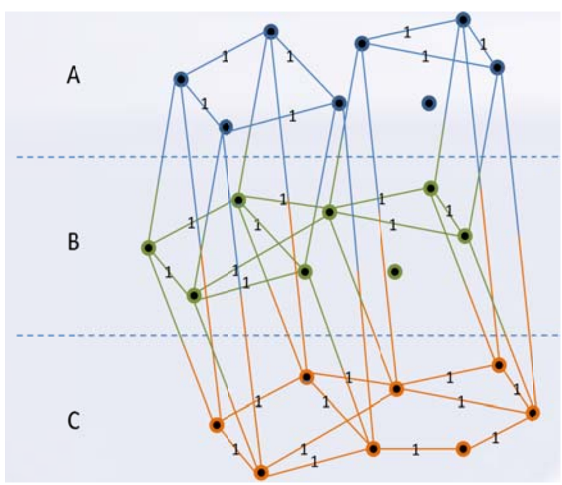

Figure 50: Multi-Layer Graph Including Transitional Links.

This change has a number of ramifications to path computation:

1) The vertex naming used in Figure 49 does not sufficiently distinguish the per-layer vertex shown in Figure 50. Additional layer identification needs to be recorded along with the vertex name to remove ambiguity.
2) The layer constraint previously used applied in path computation needs to change to a layer stack. This layer stack is used to identify both the in-layer links that can be used, as well as the server to client transitional links that may be used. Maintaining the layer stack keeps track of a transition from a client to server layer so it can be reflectively performed when returning to the client layer. As an example, a service starting in layer A using a path that transitions from A to C cannot use a C to B to A transition to return to layer A.

13.2 Transitional link representation and graph storage

While storing a graph containing transitional links is an implementation specific detail, it is not necessary to change existing link storage mechanisms. This is made possible by advertising transitional links as an attribute of a link end. This recognizes the condition that traversing a transitional link is performed as a part of traversing a physical link.

As a result, it is possible to handle transitional links in-line with a few changes to path computation. For example, path computation can be performed with the following changes:

1) Addition of a layer id to the candidate key used to identify candidate paths. This recognizes the need to develop paths to each layer a node appears in.
2) Addition of a layer stack that provides context as a vertex is being evaluated. This is used as follows:
    a. At initialization, the path computation function places the layer of the service request on the stack.
    b. The layer stack is stored with each candidate path. This saves the state of the candidate path computed. This stack is used when a candidate path is selected for further extension.
3) Creation of additional candidate path entries when a link from the vertex being processed contains an applicable transitional link, specifically:
    a. Transitional links with the current layer shown as the client layer. These entries recognize the ability to transition from a client layer to server layer as a part of a path to the destination. Attribute information for the candidate path entry (e.g. cost) reflects both the physical attribute as well as any transitional link specific attributes.
    b. Transitional links with the current layer as the server layer and the layer list matches the entries in the layer stack. These entries recognize the ability for the service being routed to return to a client layer.
4) Manipulation of the layer stack when transitional links create candidate path entries as follows:
    a. When the candidate path entry is being created for a transitional link associated with a physical link where the client layer of the transitional link matches the layer at the top of the stack, the layer list associated with a transitional link will be pushed onto the stack.

b. When the candidate path entry is being created for a transitional link associated with a physical link where the server layer and ordered layer list of the transitional link matches the top and subsequent entries of the layer stack, the entries of the layer list are popped from the layer stack.

14 Appendix IV: Client:Server Ratio

14.1 Background

The introduction of transitional links to facilitate multi-layer network graphs creates two issues previously unseen in single-layer networks. These are:

1. The introduction of layers into the network graph that may be unknown to the system performing path computation. This limits the ability for path computation to determine if a server layer is useful in meeting a client layer demand.
2. The need for scaling link cost parameters to deal with opportunity forgone cost.

These issues require introduction of a technology independent method to describe the capacity relationship between the client and server layers. This need is address through the introduction of an optional attribute for transitional links – the Client:Server ratio. This is described as the *Link Capacity for Transitional Link* in section 6.3.4.

14.2 Server Layer Capacity

When a service request is received at a domain ingress, it includes traffic parameters detailing the layer and capacity required. These parameters are required to be understood by the domain ingress node as it is necessary to perform policy validation for service requested. However, the domain does not need all internal equipment to support the technology of the layer requested. In these cases, a multi-layer network graph describes how to transition the service requested into a different layer. This is called adaptation of the client layer to a server layer.

To facilitate selection of applicable server layer networks and links within the server layer with appropriate capacity, path computation needs to understand how the capacity of the client layer is converted into the capacity of the server layer. When the path computation process does not have any awareness of how this conversion is performed, external information is needed. The Client:Server Ratio is a value optionally advertised by the node where the adaptation will be performed, and provides a path computation process with this needed information. In general, this ratio can be used recursively over more than two layers.

Implementations need to be able to handle the case where path computation does not have awareness of a technology and the client:server ratio is not provided. In this case, the implementation may choose to ignore the transitional link.

This information may be used with the following method:

When path computation encounters a transitional link, it will push the server layer onto the layer stack (see section 13). This stack entry may be extended with the layer capacity required as shown in Figure 51. This value is generated by applying the Client:Server Ratio to the client layer capacity. As links are considered within the graph, their available capacity is checked against the top entry in the layer stack.

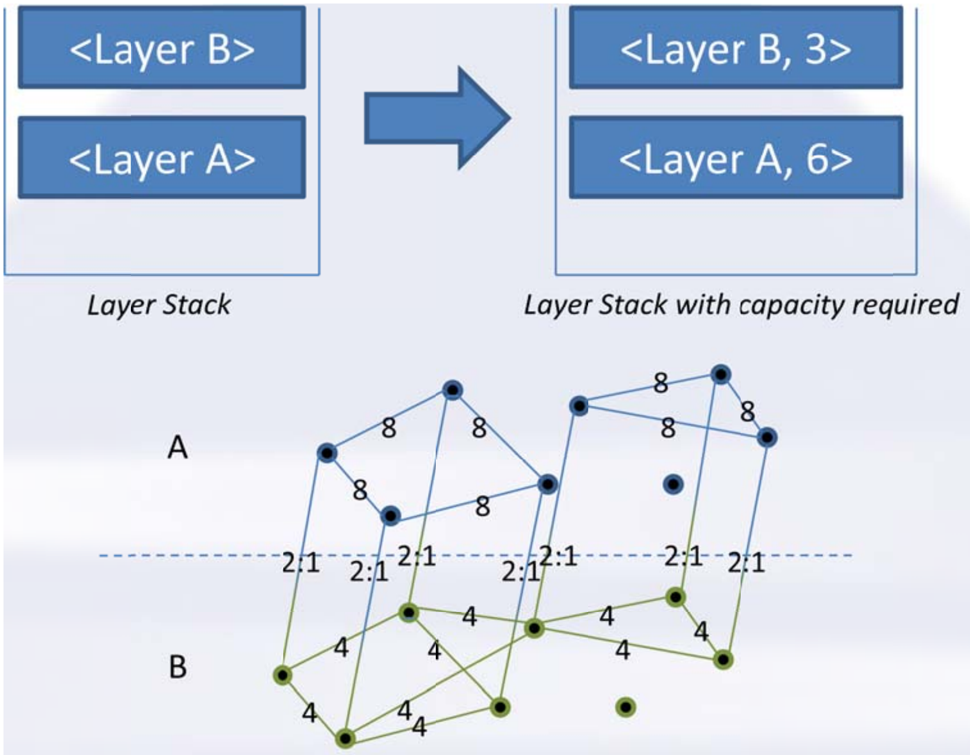

Figure 51: Use of Client:Server Ratio with Service Request Capacity conversion (request = 6xA)

When a transitional link returning to the client layer is encountered, the layer stack entry is popped. This pop operation would also include popping of the layer capacity value.

14.3 Server Layer Cost Scaling

Each layer found in a multi-layer network has different costs associated with providing a connection. While the cost-per-bit for different layers may be similar, the granularity of connection sizing is many times different. This leads to a different amount of stranded capacity in a server layer. Since stranded capacity negatively affects cost for service delivery, it needs to be accounted for when selecting a connection path.

To reflect the higher cost resulting from mismatch between client capacity required and server selected, the server layer link cost needs to reflect the inefficiency of the client signal being carried by that link. This requires understanding the relationship of client and server capacity, which is reflected in the Client:Server Ratio.

In general, this ratio can be used recursively over more than two layers.

This information may be used with the following method:

When path computation encounters a transitional link, it will push the server layer onto the layer stack (see section 13). This stack entry may be extended with a cost factor derived from the Client:Server Ratio associated with the transitional link as well as the lower entries in the layer stack as shown in Figure 52. The cost factor in the top layer stack entry is applied to the cost of all server layer links encountered by path computation.

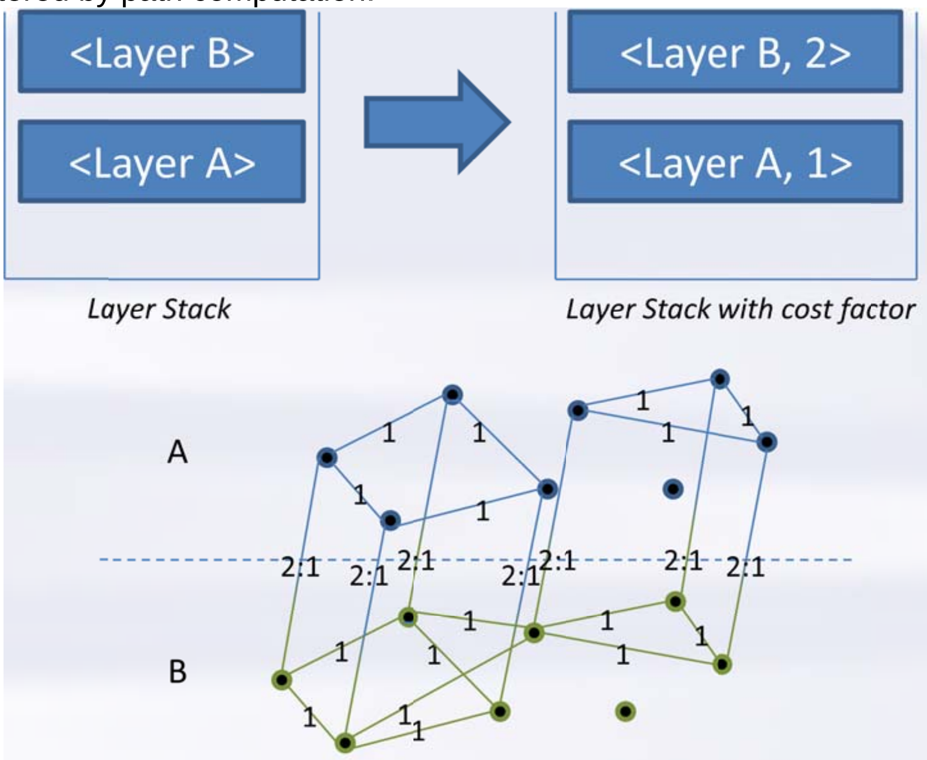

Figure 52: Use of Client:Server Ratio with cost scaling

When a transitional link returning to the client layer is encountered, the layer stack entry is popped. This pop operation would also include popping of the layer cost factor.

15 Appendix V: List of companies belonging to OIF when document is approved

Acacia Communications
ADVA Optical Networking
Agilent Technologies
Alcatel-Lucent
Altera
AMCC
Amphenol Corp.
Anritsu
Applied Communication Sciences
AT&T
Avago Technologies Inc.
Broadcom
Brocade
Centellax, Inc.
China Telecom
Ciena Corporation
Cisco Systems
ClariPhy Communications
Cogo Optronics
Comcast
Cortina Systems
CPqD
CyOptics
Dell, Inc.
Department of Defense
Deutsche Telekom
EigenLight.com
Emcore
Ericsson
ETRI
EXFO
FCI USA LLC
Fiberhome Technologies Group
Finisar Corporation
France Telecom Group/Orange
Fujitsu
Furukawa Electric Japan
GigOptix Inc.
Hewlett Packard

Hitachi
Hittite Microwave Corp
Huawei Technologies
IBM Corporation
Infinera
Inphi
Intel
IPtronics
JDSU
Juniper Networks
Kandou
KDDI R&D Laboratories
Kotura, Inc.
LeCroy
LSI Corporation
Luxtera
M/A-COM Technology Solutions, Inc.
Marben Products
Metaswitch
Mindspeed
Mitsubishi Electric Corporation
Molex
MoSys, Inc.
MultiPhy Ltd
NEC
NeoPhotonics
Nokia Siemens Networks
NTT Corporation
Oclaro
Optoplex
PETRA
Picometrix
PMC Sierra
QLogic Corporation
Reflex Photonics
Semtech
SHF Communication Technologies
Skorpios Technologies
Sumitomo Electric Industries
Sumitomo Osaka Cement

TE Connectivity
Tektronix
Tellabs
TELUS
TeraXion
Texas Instruments
Time Warner Cable
TriQuint Semiconductor
u2t Photonics AG
Verizon
Vitesse Semiconductor
Xilinx
Xtera Communications
Yamaichi Electronics Ltd.