

SNC AND PROTECTION

Version 3.0

1 INTRODUCTION

The basic create SNC service provided in Phase I of the SSIM (now MTNM) model allows the NMS to create basic SNCs within a limited set of subnetworks applying limited routing constraints.

The MTNM model will advance in these capabilities in Phase II. The following changes are covered in this appendix:

- Increase in the range of subnetworks supported over the interface (adding a mesh subnetwork). The subnetwork topologies are explained in the document
- Enhancements to the SNCs to include a specific SNC type, Static Protection Level and Protection Effort all of which are described in the document
- Improvements in the routing constraint capability to cover inclusions and exclusions:
 - Routing constraints apply to all network topologies other than singleton (e.g. Ring, mesh) where the network operator has effectively chosen to delegate some level of control over the routing to the underlying devices (EMS and NEs) ¹.
 - These constraints may be as extreme as the NMS laying out the entire route or as simple as a single NE inclusion
- Introduction of the cross-connect object

The Version 3 of MTNM model features also:

- SNC modification
- SNC multiple route management

Note:

In this Appendix, unless otherwise stated, wherever the term create SNC is used, it is understood that createAndActivateSNC and createSNC both apply and also wherever delete SNC is used, it is understood that deleteAndDeactivateSNC and deleteSNC both apply.

2 SNC PROPERTIES AND PARAMETERS

An SNC is defined by a number of parameters. The key parameters to be added/enhanced in this release are justified and described below.

2.1 Service Quality Related Parameters – Protection

Aspects of resilience of an SNC are described using StaticProtectionLevel.² During the creation of an SNC the NMS will request a particular StaticProtectionLevel but may also offer the EMS some freedom to

¹ If an EMS supports subnetworks other than Singleton, it is expected to support some degree of network routing. The degree depends upon the type of subnetwork.

² In addition **ResilienceScheme** and **ResilienceLayer** were discussed, however, for Phase two of the MTNM

chose a different StaticProtectionLevel to that requested in the create data; this is achieved via the ProtectionEffort. These parameters are described below.

2.1.1 StaticProtectionLevel [mandatory parameter]

StaticProtectionLevel is a statement of the internal resiliency of the SNC (internal to the subnetwork). The more resilient an SNC is the less the chance of the transmission being interrupted. In general, the more resilient an SNC is the more bandwidth it will consume inside the subnetwork.

The StaticProtectionLevel relates to protection that is provided by using specific dedicated or specific shared resources in the protection path (as opposed to the dynamic protection provided by systems that reroute on failure). The parameter RerouteOnFailure (not covered in this contribution) addresses these dynamic protection capabilities. These augment the other capabilities described in this section. An SNC can be set to any of the values shown below and also set to reroute on failure.

The StaticProtectionLevel will take one of the following values: {PREEMPTIBLE, UNPROTECTED, PARTIALLY_PROTECTED, FULLY_PROTECTED, HIGHLY_PROTECTED}.

The StaticProtectionLevel values are defined as follows:

- **PREEMPTIBLE:** May have resources taken to recover another SNC.
- **UNPROTECTED:** An SNC that will fail if any resource in its route fails.
- **PARTIALLY_PROTECTED:** Protection exists but has at least one shared node, shared link or shared link and node.
- **FULLY_PROTECTED:** An SNC that will not fail if any single managed resource along its route fails (excluding the originating and terminating nodes for the SNC); for example, an SNC that is diversely routed at any layer.
- **HIGHLY_PROTECTED:** A higher level of protection than is possible by simple diverse path routing. A highly protected subnetwork should each be able to experience a single failure without affecting traffic. No shared facilities and NEs excluding originating and terminating NEs. Typically this is achieved in a SONET/SDH environment using dual ring interworking, where the proper use of links enhances survivability over that offered by simple diverse path routing. Highly Protected is used when the NMS wishes to request the highest available protection level from the EMS. If a level greater than simple diverse routing is available, it must be provided. If this can be done in multiple ways and is not further specified by the NMS, the choice of highly protected SNC is made by the EMS.

The StaticProtectionLevel does not have any bearing on the externally visible shape and traffic flows of the SNC.

For example an ST_SIMPLE SNC can have a StaticProtectionLevel of FULLY_PROTECTED and an ST_ADD_DROP_A can have a StaticProtectionLevel of PREEMPTIBLE.

The StaticProtectionLevel, an SNC totally contained within a SONET BLSR, that is unprotected at its layer will have its StaticProtectionLevel set to FULLY_PROTECTED.

An SNC contained within a BLSR and is pre-emptible will have StaticProtectionLevel set to PREEMPTIBLE. Although all SNCTypes can take any StaticProtectionLevel, the StaticProtectionLevel that may be selected for an SNC does depend upon the subnetwork that the SNC is to be created in. The following list identifies possible StaticProtectionLevels for each subnetwork type:

interface only **ProtectionLevel** and **ResilienceQuality** will be fully supported. **ResilienceScheme** relates to the protection architecture including switch/restoration time being employed (i.e. restoration, UPSR) while **ResilienceLayer** relates to the layer the protection is applied (i.e. Line, Optical MS).

Subnetwork	StaticProtectionLevel supported				
	Preemptible	Unprotected	Partially Protected	Fully Protected	Multiply Protected
Singleton	No	Yes ³	No	No	No
Chain	Yes ⁴	Yes	Yes ⁵	Yes ⁶	Yes ⁷
PSR	No ⁸	Yes	No ⁹	Yes	No ¹⁰
SPRing	Yes	Yes ¹¹	No ¹²	Yes	Yes ¹³
OpenPSR	No ¹⁴	Yes	No ¹⁵	No ¹⁶	No ¹⁷
OpenSPRing	Yes	Yes ¹⁸	No ¹⁹	Yes	Yes ²⁰
Mesh	Yes	Yes	Yes	Yes	Yes

In all cases the EMS will comply with requests that fall within the range of protection that it perceives within the subnetwork. So it is possible that although the interconnectivity between two adjacent NEs when viewed in a subnetwork looks protected, e.g. 1+1 MSP, in the server layers outside the view of the EMS, this protection may be invalidated, e.g. the fiber of the 1+1 MSP may run down the same duct. Similarly it is possible that fibers on either side of a UPSR use the same duct or optical layer bearer. In both of these cases the EMS would provide an optimistic presentation of the level of protection achieved. It is assumed that network planning has been carried out to avoid or at least minimize such occurrences.

³ The singleton does not support any form of network protection and equipment protection is not considered.

⁴ Assumes 1:1 point to point system covered supporting extra traffic.

⁵ Assumes a chain with some fibers protected and others not protected where the SNC uses some of both types of fibre

⁶ Only when it involves two adjacent NEs at the layer of connection and all components (e.g. fibers) are diverse.

⁷ Only when it involves two adjacent NEs at the layer of connection and all components (e.g. fibers) are diverse.

⁸ Assumes PSR configuration is standard form and therefore always has no extra traffic capability

⁹ Assumes PSR configuration is standard form and therefore never supports MSP

¹⁰ Assumes PSR has no MSP and therefore can not support anymore than one layer of protection

¹¹ Assumes Standard NUT (Non-pre-emptible Unprotected Traffic) capability available in MSSPring

¹² Assumes SPRing configuration is standard and does not support SNC over mix of NUT and protection bandwidth

¹³ For example MSSPring ring with four fibers

¹⁴ Assumes PSR configuration is standard form and therefore always has no extra traffic capability

¹⁵ Assumes PSR configuration is standard form and therefore never supports MSP

¹⁶ Assumes PSR configuration is standard form and therefore only offers one layer of protection.

¹⁷ Assumes PSR has no MSP and in this case ring is broken so no complete protection in any layer

¹⁸ Assumes Standard NUT (Non-pre-emptible Unprotected Traffic) capability available in MSSPring

¹⁹ Assumes SPRing configuration is standard and does not support SNC over mix of NUT and protection bandwidth

²⁰ For 4 fiber BLSR

2.1.2 ProtectionEffort [mandatory parameter]

The ProtectionEffort is a statement of the requirement of the StaticProtectionLevel to be applied at creation of the SNC. ProtectionEffort will take one of the following values: {WHATEVER, SAME, SAMEORBETTER, SAMEORWORSE}.

The StaticProtectionLevel is ordered from worst to best in the following list: PREEMPTIBLE, UNPROTECTED, PARTIALLY_PROTECTED, FULLY_PROTECTED, HIGHLY_PROTECTED.

The EMS attempts to provide the StaticProtectionLevel requested; if this is not possible it attempts to provide better or worse than the requested protection level according to the ProtectionEffort. However, if the NMS requests StaticProtectionLevel PartiallyProtected with ProtectionEffort SAMEORBETTER, the EMS may attempt to provide FULLY_PROTECTED first.

For example, if ProtectionEffort of SAMEORBETTER is specified for a 3-ended ST_ADD_DROP_A to be created in a SPRing Subnetwork and a StaticProtectionLevel of PARTIALLY_PROTECTED is requested, it is acceptable for the EMS to return an ST_ADD_DROP_A SNC that has a StaticProtectionLevel of FULLY_PROTECTED.

If ProtectionEffort WHATEVER is specified, the EMS will attempt to protect at the requested StaticProtectionLevel, and if it cannot do so or chooses not to do so, it is free to choose any other protection level.

Clearly if an SNC is requested with a StaticProtectionLevel of HIGHLY_PROTECTED and a ProtectionEffort of SameOrBetter this is equivalent to a ProtectionEffort of SAME as there is no better protection than HIGHLY_PROTECTED, however, the combination should be allowed and interpreted in this way. Likewise a combination of StaticProtectionLevel PREEMPTIBLE and a ProtectionEffort of SAMEORWORSE should be allowed and interpreted as StaticProtectionLevel of SAME.

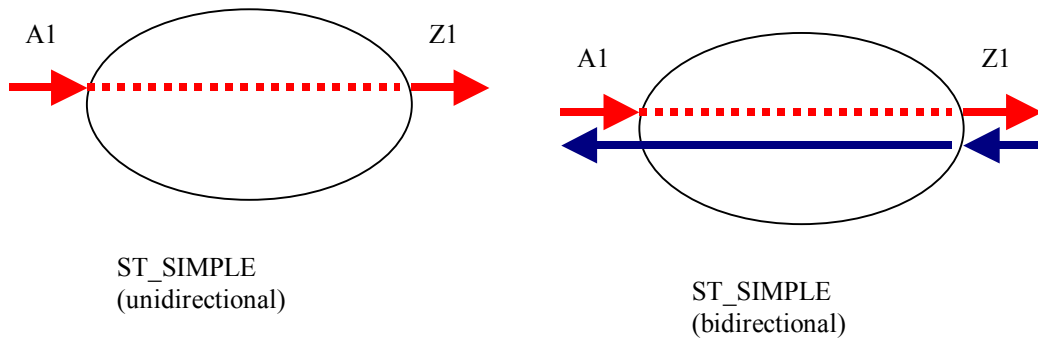
2.2 Traffic flow into and out of an SNC

2.2.1 SNCType overview [mandatory parameter]

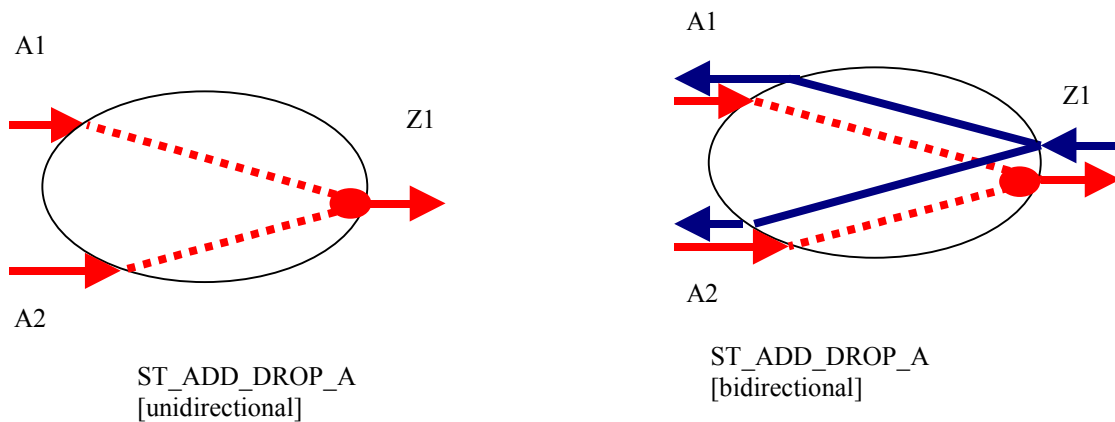
The SNCType reflects the dataflow from outside the subnetwork. The internal configuration of the SNC is not intended to be expressed via this attribute. Both bidirectional and unidirectional SNCs can be created for each type identified below unless otherwise indicated.

There are 2 types of SNC Types: implicit and explicit. The diagrams that follow show the different implicit SNCTypes: {ST_SIMPLE, ST_ADD_DROP_A, ST_ADD_DROP_Z, ST_DOUBLE_ADD_DROP, ST_INTERCONNECT, ST_DOUBLE_INTERCONNECT, ST_OPEN_ADD_DROP}.

ST_SIMPLE	For simple two-ended connection types. This is used in cases where we have one A and one Z.
------------------	---------------------------------------------------------------------------------------------

**Figure 1 ST_SIMPLE****ST_ADD_DROP_A**

For connections involving an AddDrop type that have three ends where two incoming signals are switched by a third end-point. In the bidirectional case, one input in the other direction is broadcast to the two outgoing endpoints. In the unidirectional case the Aend always represents ingress of traffic to the SNC and the Zend egress.

**Figure 2 ST_ADD_DROP_A****ST_ADD_DROP_Z**

Same as ST_ADD_DROP_A, but with Aend as single source, and/or reliable TP.

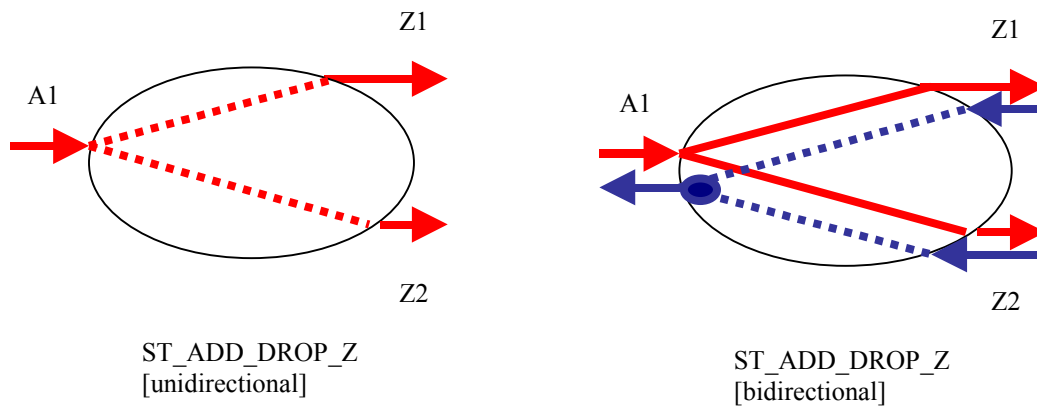


Figure 3 ST_ADD_DROP_Z

ST_DOUBLE_ADD_DROP For connections involving an AddDrop type where the configuration is symmetrical four ended.

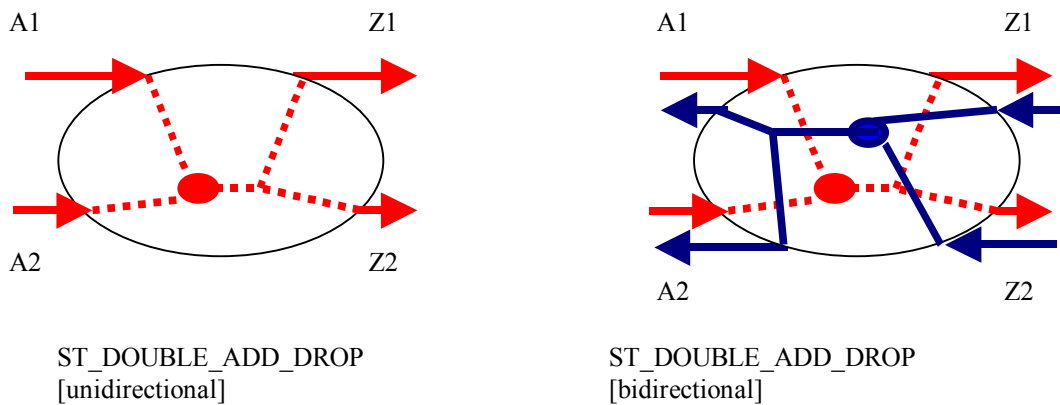


Figure 4 ST_DOUBLE_ADD_DROP

ST_INTERCONNECT For subnetworks performing Drop and Continue in DRI SNCs (See the network examples in 2.2.2 below). In this case the incoming traffic from one Aend is available in the transmit and receive of the second Aend, the third end selects the input from the two Aends. The reverse direction traffic in a unidirectional connection, passes from the Zend to the First Aend.

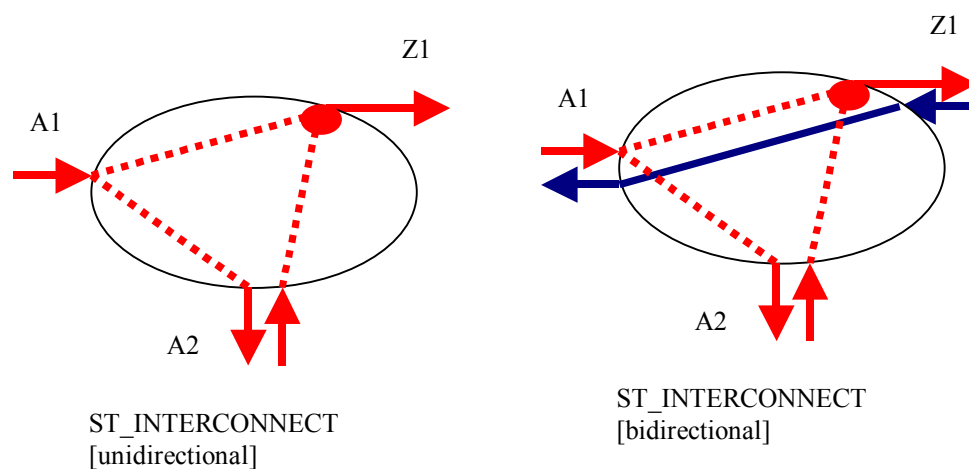


Figure 5 ST_INTERCONNECT

ST_DOUBLE_INTERCONNECT This is a symmetrical form of the ST_INTERCONNECT where a subnetwork provides two interconnect ports, one for the A side traffic and one for the Z side.

Note:

A unidirectional ST_DOUBLE_INTERCONNECT is not valid; instead a unidirectional ST_INTERCONNECT SNC would be used.

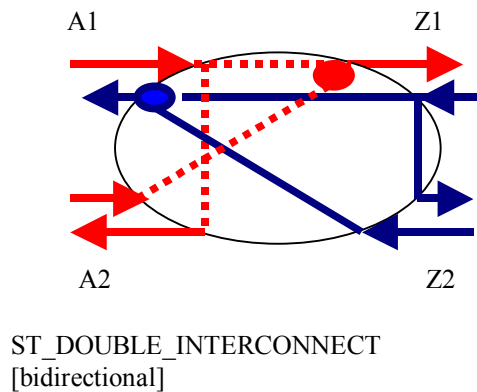


Figure 6 ST_DOUBLE_INTERCONNECT

ST_OPEN_ADD_DROP Used in a double open ring scenario.

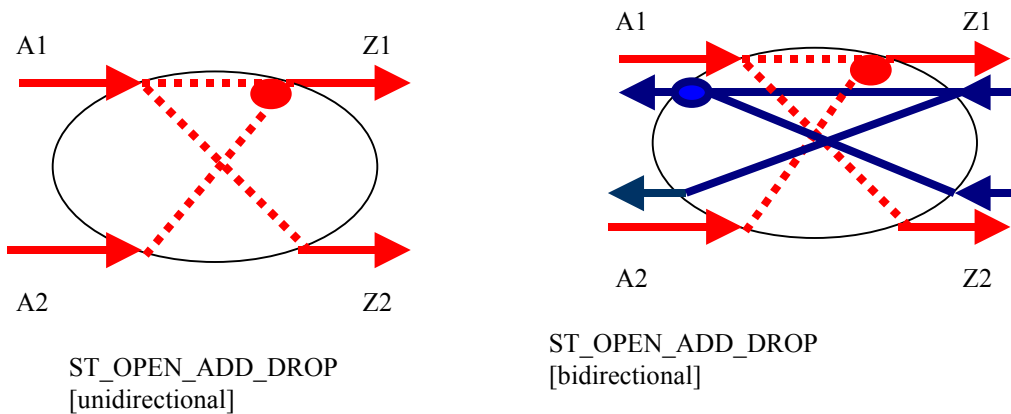


Figure 7 ST_OPEN_ADD_DROP

ST EXPLICIT

1. An SNC that may be described using an implicit SNC type may not use the ST_EXPLICIT type. The ST_EXPLICIT is used to describe a complex SNC that cannot be accurately described using the other defined SNC types.
2. aEnd and zEnd lists must contain the same number of TPs, e.g. n.
3. For every i between 1 and n the ith aEnd entry and the ith zEnd entry form a pair defining an unidirectional traffic flow from the aEnd TP (source) to the zEnd TP (sink).
4. A single TP may be used in the aEnd list (i.e. as a source) an arbitrary number of times.
5. A single TP must not be used more than twice in the zEnd list (i.e. as a sink).
6. If a TP is used twice in the zEnd list, it is assumed that there is a service selector at this TP.

Figure 8 shows 2 examples of SNCs that must use ST_EXPLICIT.

Example 1 shows a bidirectional SNC where the Z-endpoint of is source connected and sink connected to different bidirectional CTPs (a single ST_SIMPLE SNC cannot convey the endpoints of this bidirectional SNC). The Aendpoint-Zendpoint pairs for this SNC are $A_1Z_1=\{J, L1snk\}$, $A_2Z_2=\{L2src, J\}$. Ordering of the pairs is not significant.

Example 2 shows a bidirectional SNC with a protection switch which are connected to source and sink TPs of different bidirectional CTPs at the Z-endpoints (a single ST_ADD_DROP SNC cannot convey the endpoints in this case) The Aendpoint-Ziendpoint pairs for this SNC are $\{L, J1snk\}$, $\{L, K1snk\}$, $\{J2src, L\}$, $\{K2src, L\}$. Ordering of the pairs is not significant.

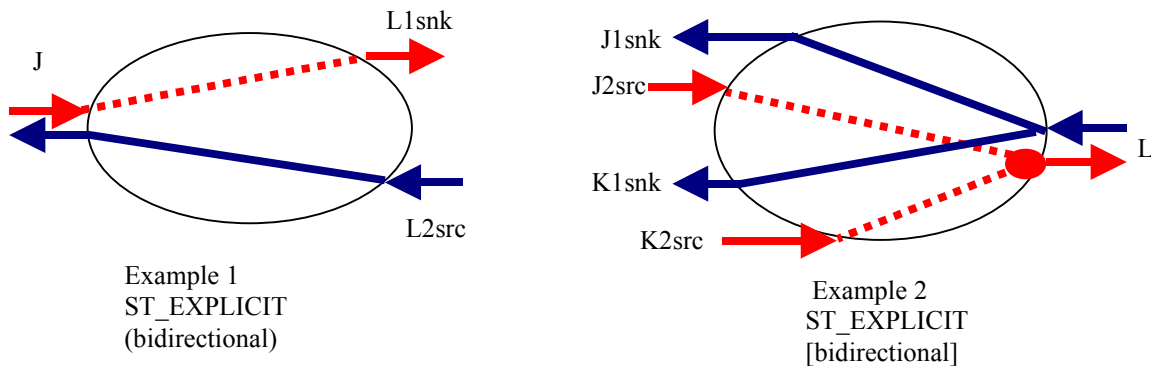


Figure 8 ST_EXPLICIT

2.2.2 Network Examples of SNC types

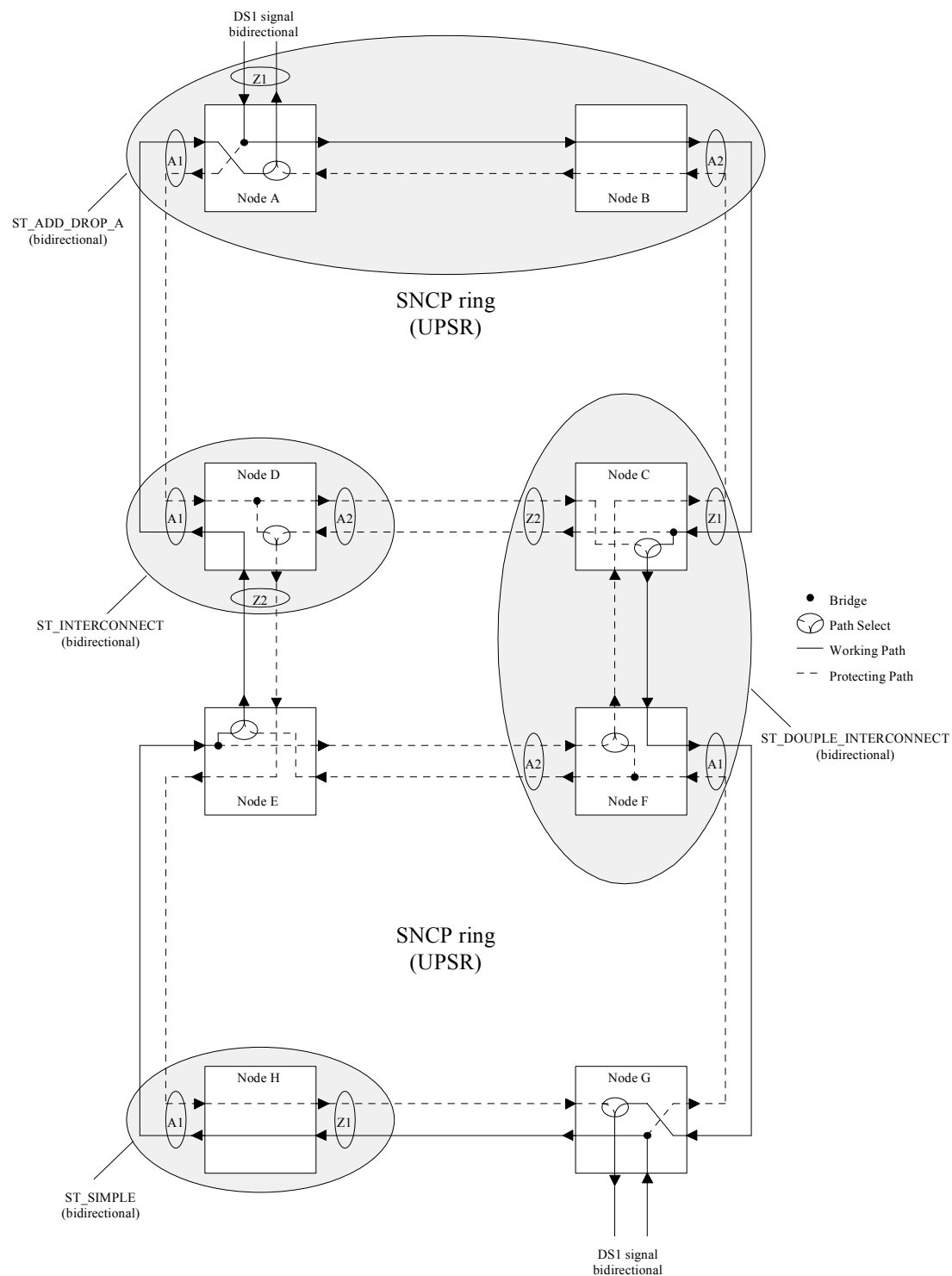


Figure 9 Example Subnetworks Explaining SNC Types ST_ADD_DROP_A, ST_DOUBLE_INTERCONNECT and ST_SIMPLE

To achieve the traffic flow identified with the six subnetworks shown above, the following bidirectional SNCs are required:

- ST_ADD_DROP_A with StaticProtectLevel of Unprotected (and three ports) would be created in the MLSN of nodes A/B, and in node G
- ST_DOUBLE_INTERCONNECT with StaticProtectLevel of Unprotected would be created in the MLSN of nodes C/F
- ST_INTERCONNECT with StaticProtectLevel of Unprotected would be created in node D and node E
- ST_SIMPLE with StaticProtectLevel of Unprotected would be created in node H

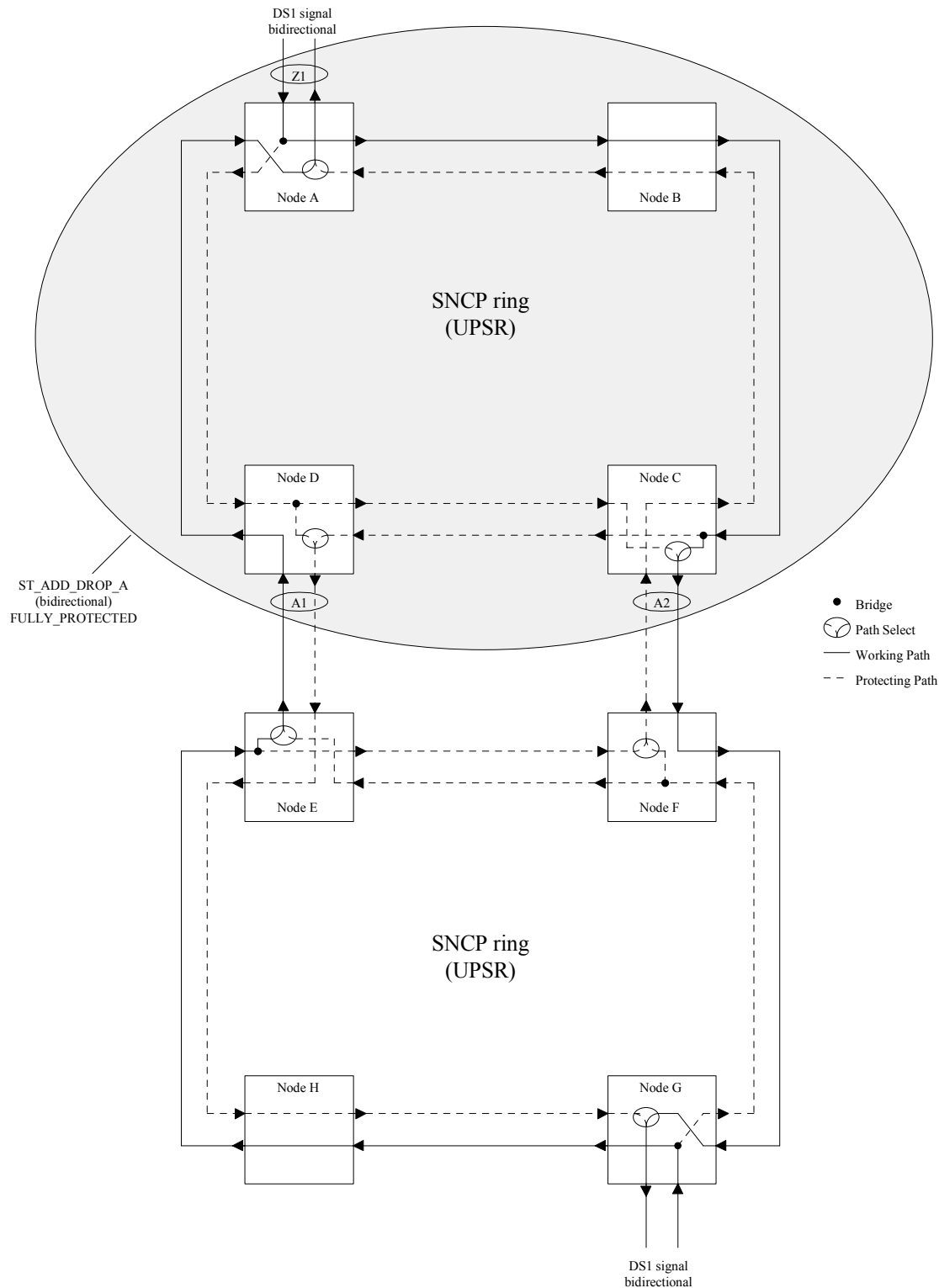


Figure 10 Example Subnetworks Explaining SNC Types ST_ADD_DROP_A

To achieve the traffic flow identified with the five subnetworks shown above, the following bidirectional SNCs are required:

- ST_ADD_DROP_A with StaticProtectLevel of FullyProtected would be created in the MLSN of nodes A/B/C/D.
- ST_INTERCONNECT with StaticProtectLevel of Unprotected would be created in node E and node F
- ST_ADD_DROP_A with StaticProtectLevel of Unprotected (and three ports) would be created in node G
- ST_SIMPLE with StaticProtectLevel of Unprotected would be created in node H

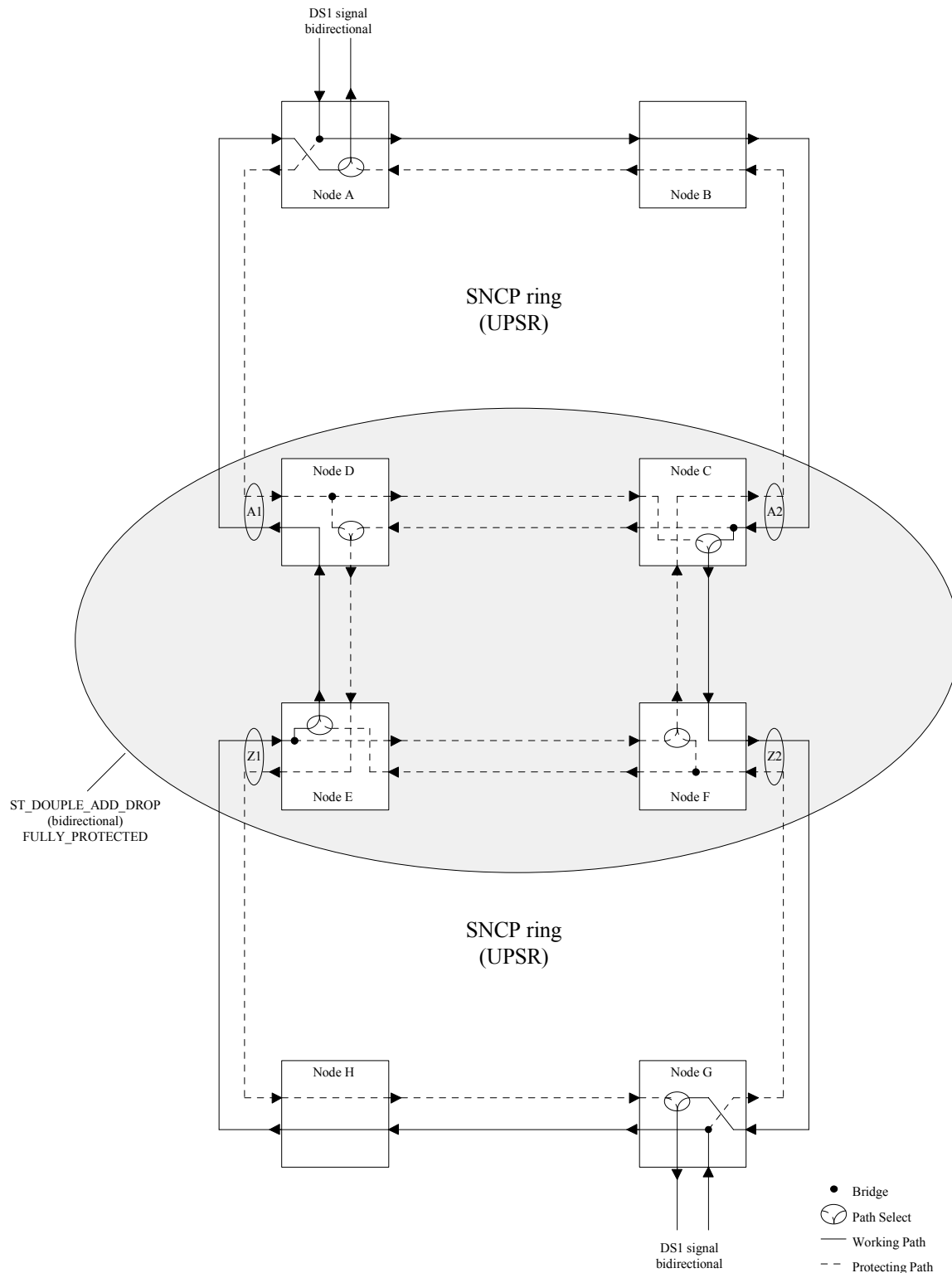


Figure 11 Example Subnetworks Explaining SNC Types ST_DOUBLE_ADD_DROP

To achieve the traffic flow identified with the five subnetworks shown above, the following bidirectional SNCs are required:

- ST_ADD_DROP_A with StaticProtectLevel of Unprotected (and three ports) would be created in node A and node G
- ST_SIMPLE with StaticProtectLevel of Unprotected would be created in node B and node H
- ST_DOUBLE_ADD_DROP with StaticProtectLevel of FullyProtected (and four ports) would be created in the MLSN of C/D/E/F

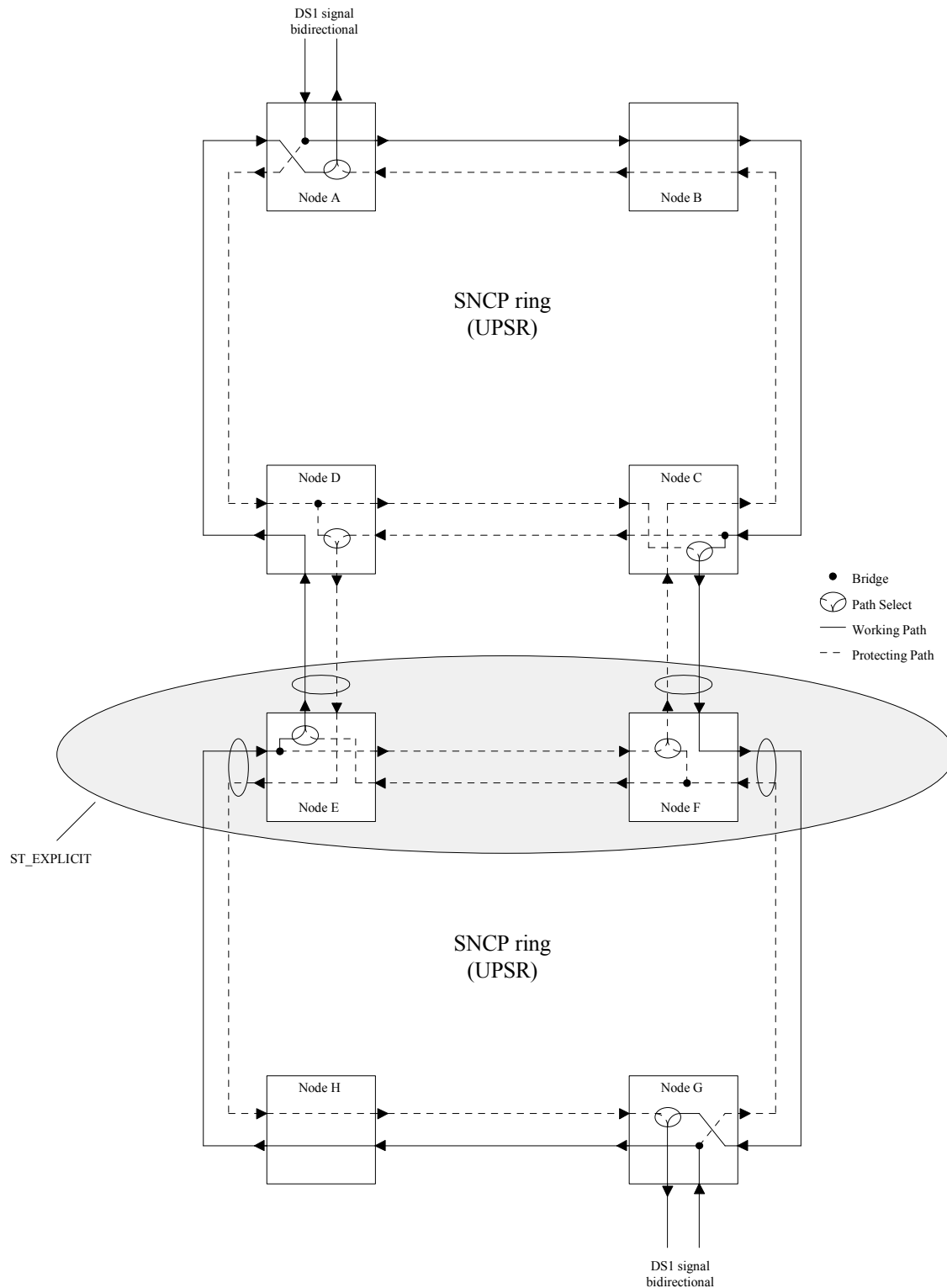


Figure 12 Example Subnetworks Explaining SNC Types ST_EXPLICIT

To achieve the traffic flow identified with the seven subnetworks shown above, the following bidirectional SNCs are required:

- ST_ADD_DROP_A with StaticProtectLevel of Unprotected (and three ports) would be created in node A and node G
- ST_SIMPLE with StaticProtectLevel of Unprotected would be created in node B and node H
- ST_INTERCONNECT with StaticProtectLevel of Unprotected would be created in node C and node D
- ST_EXPLICIT would be created in the MLSN of nodes E/F

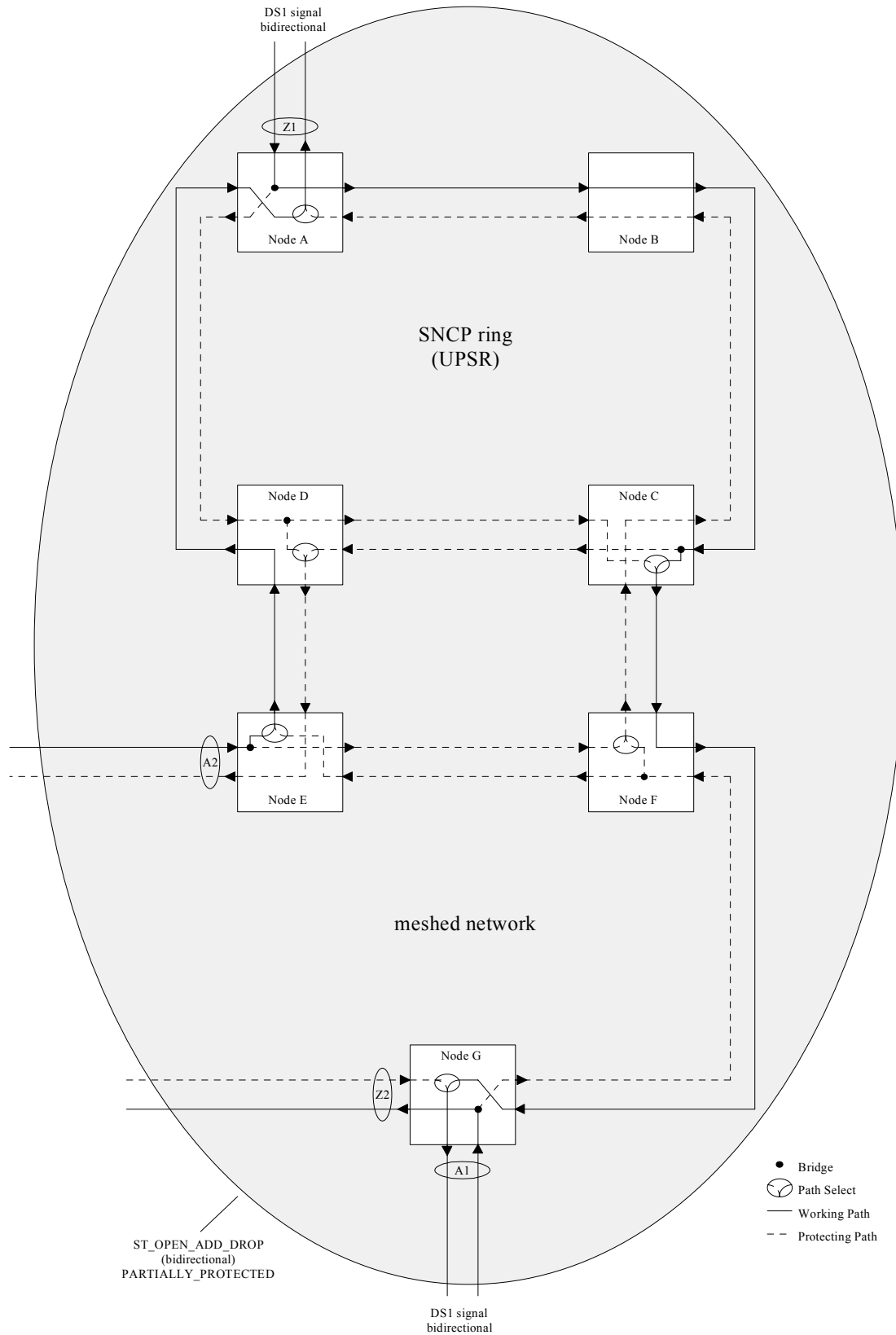


Figure 13 Example Subnetworks Explaining SNC Types ST_OPEN_ADD_DROP

To achieve the traffic flow identified with the subnetwork shown above, the following bidirectional SNC is required:

- ST_OPEN_ADD_DROP with StaticProtectLevel of partially protected would be created in the MLSN of nodes A/B/C/D/E/F/G

2.2.3 Semantic Rules for Aend and Zend

An SNC identifies the TPs that it connects via a pair of lists of points. A single SNC can be described by many A and Zends. There are two distinct styles of SNC, implicit and explicit, that both use the same SNC structure but they use the Aend/Zend lists of the SNC structure in distinctly different ways. The style of SNC is coded into the SNCType value with all but ST_EXPLICIT being implicit in style and structure usage.

For both styles of usage a unidirectional SNCs uses the Aend as the source (transmit) and Zend(s) as the sinks (receive). In bidirectional SNCs the choice of Aend and Zend is determined from the SNC type and in some cases is arbitrary.

2.2.3.1 Implicit SNC Structure Usage

A number of commonly used SNC external shapes have been identified and each modeled as a specific type that includes a set number of Aend/Zends and that has implicit internal traffic flow semantics (that are documented below).

For the implicit SNC types the Aend and Zend list simply represent the points of entry and exit of traffic on the SNC, the internal flow is conveyed by the type. Some of the SNCs defined as implicit are asymmetric in number of Aends and Zends.

It is expected that these implicit SNCs will cover a majority of the cases of connectivity encountered in the network today. It may be decided in later releases to increase the range of types covered in this way.

The implicit SNCTypes described are:

- ST_SIMPLE
- ST_ADD_DROP_A
- ST_ADD_DROP_Z
- ST_INTERCONNECT
- ST_DOUBLE_INTERCONNECT
- ST_DOUBLE_ADD_DROP
- ST_OPEN_ADD_DROP

2.2.3.2 Explicit SNC Structure

Unlike the implicit SNC usage, the explicit SNC type has a pairwise matching of the Aends and Zends. The SNC will have an n-entry Aend List and an n-entry Zend List. The tuples are pairwise matched: (A1,Z1), (A2, Z2), ..., (An,Zn). In this manner any arbitrary cross-connect may be described. In some cases, the TP in the Aend List could be a part of the Zend List also.

The SNCType for an explicit SNC is ST_EXPLICIT.

2.2.3.3 Usage of ST_EXPLICIT

An SNC that may be described using an implicit SNC type may not use the ST_EXPLICIT type. ST_EXPLICIT is to be used to describe all cases of complex SNC that cannot be covered by an implicit type. It is expected that a majority of the cases of use will be complex bi-directional protected SNCs.

2.3 Cross-Connect

A cross-connect represents a physical connection within a managed element. A cross-connect is atomic and is identified, similarly to an SNC in a singleton subnetwork, based on its external shape. That is, a cross-connect is identified in this interface by its A end(s) and Z end(s), its "SNC" type, and its directionality.

Other characteristics of SNCs do not apply to CCs. In particular, the static protection level does not apply to CCs, as equipment protection is not modeled by this interface. As well, the SNC state does not apply to CCs as a CC represents a physical connection and not a logical, potential, or reserved set of connections. As such, a CC may only be "active" or "inactive". A "partial CC" actually corresponds to a different CC in this interface as it has a different external shape (e.g., a "half" bidirectional simple CC is not modeled as a partial CC, but rather as a "complete" unidirectional simple CC).

An "active" CC means that no further intervention on the CC itself is required from the NMS to activate it. The fact that a CC is "active" does not necessarily imply that there is traffic flow on the network.

2.4 Protection Relationships in an Implicit SNC – ProtectionRole [optional parameter]

The **ProtectionRole** parameter is conveyed via the TPParamList (for SNC and cross-connect) and is used to indicate the protection relationship of the Aends and Zends Source TPs.

The ProtectionRole is mandatory for the Source TPs in cases of revertive switch.

The ProtectionRole is optional for the Source TPs in cases of a non-revertive switch. In the non-revertive case the protection role is assigned by the EMS to indicate the Primary from the Secondary TP, where the Primary is the default position of the switch selector.

When applicable, the Source TPs take on one of the following ProtectionRole values:

- **PRIMARY** The Primary Source TP is where the traffic is switched from.
- **BACKUP** The Backup Source TP is where the traffic is switched to on failure of the Primary.

In the case when there is no relevant protection role, there will be no ProtectionRole parameter in TPParamList. There is no ProtectionRole for the reliable end (i.e. in case of revertive switch).

The expected usage of ProtectionRole varies with SNCType:

- ST_SIMPLE: No ProtectionRole parameter.
- ST_ADD_DROP_A: Used in Aend.
- ST_ADD_DROP_Z: Used in Zend.
- ST_INTERCONNECT: Used in Aend.
- ST_DOUBLE_INTERCONNECT: Used in Aend and Zend.
- ST_DOUBLE_ADD_DROP: For bi-directional case used in Aend and Zend. For the unidirectional case used in Aend.

- **ST_OPEN_ADD_DROP:** For this non-symmetrical SNC, the ProtectionRole is used in a mix of Aend and Zend. For bi-directional and unidirectional cases, it is used in the A1 and Z2 ends (that act as the protection for Z1). For bi-directional case, it is also used in the A2 and Z1 ends (that act as the protection for A1).

2.5 Independence of SNCType and StaticProtectionLevel

The SNCType describes the externally visible traffic flow from/to the endpoints of the SNC, including how the input signals are combined/selected/broadcasted to generate the output signals.

The StaticProtectionLevel describes how the traffic flow inside the SNC is protected against failures.

These two attributes are independent. For example, an ST_SIMPLE SNC has no externally visible protection (one input goes to one output), but may have a StaticProtectionLevel of FULLY_PROTECTED if the internal path from the input to the output is fully protected. Any single failure inside this subnetwork connection will not result in the failure of the SNC, because, externally, the traffic flow remains the same: one input still goes to one output.

Conversely, an ST_ADD_DROP_A SNC has some built-in “protection functionality” (two inputs go to one selector and the best one is selected as output), but can have a StaticProtectionLevel of UNPROTECTED if the two paths from the two inputs to the output are unprotected. A single failure inside the subnetwork connection will result in the failure of the SNC, because, externally, the traffic flow is affected: one of the input signals is ignored and can not make it to the output signal even if the other input signal has failed.

Note that a failure of an SNC does not necessarily imply failure of the carried traffic, since the SNC may only be a part of a “larger protected connection” that may survive such failures.

2.6 Network Routed SNCs

For all subnetwork types other than TOPO_SINGLETON the EMS presents an abstraction of the network that treats a set of NEs as a single manageable entity. When a subnetwork contains more than one NE the EMS (or underlying network components) can perform a routing function to route a requested SNC across the network.

3 UNIDIRECTIONAL SNCS AND POINT-TO-MULTIPOINT

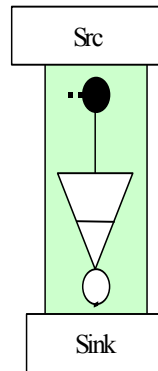
3.1 Models

3.1.1 Unidirectional Model

Both the TP and the SNC support aspects of directionality.

A TP may be unidirectional in function (representing a sink or source) and in addition a bidirectional TP may be solely connected in one of its two directions. Both of these concepts will be represented in the TP.

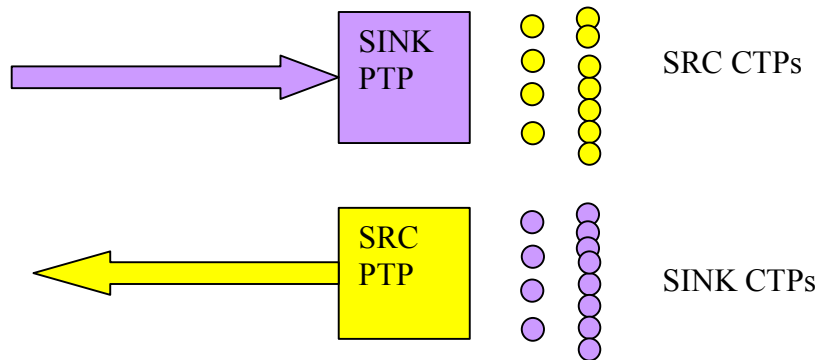
An SNC may be bidirectional or unidirectional. By convention, the A1 end of the SNC is connected to the source so that the signal flows from A to Z.



The relationship between the directionality of the PTP and the contained CTPs is shown in the following diagrams.

In the G805 TP model, the Sink and the Src nature of the TP is applicable across the adaptation layers and alternates as we traverse the hierarchy.

In the collapsed model in TMF MTNM Version 2.0, all the layers are not visible. Therefore the Src and the Sink concepts need to be specified as follows:

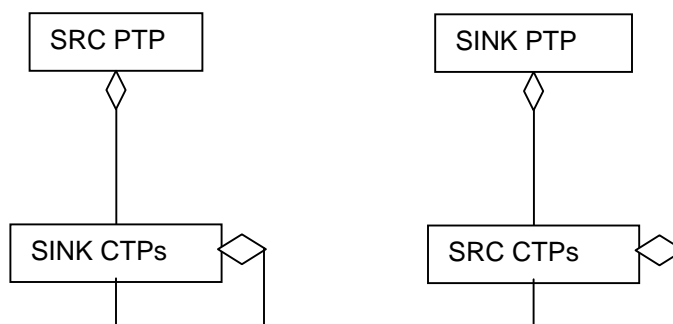


As shown in the figure,

A SINK PTP has all of its potential CTPs as SRC CTPs, which can then be used in creating SNCs or to terminate and map.

A SRC PTP has all of its potential CTPs as SINK CTPs, which can then be used in creating SNCs or to terminate and map.

A Bidirectional PTP can have unidirectional CTPs as contained CTPs, but the entire hierarchy below the first unidirectional CTP will have the same directionality. For example, a bidirectional STM4 can have a SRC and SINK VC4 CTP, but all VC12s under the SRC VC4 CTP will be SRC VC12 CTP, and similar relationship between SINK VC4 CTPs and SINK VC12 CTPs.



3.1.2 Point-to-multipoint Model

The point-to-multipoint concept is supported via an **overlay model**. This model is founded on the recognition that the “legs” of a point-to-multipoint connection are in fact completely independent, the only common aspect being that they share at least the same source. By using individual SNCs to represent the branches of the Point-to-Multipoint connectivity, the issues of sequence of creation and deletion and of migration from a point-point to a point-to-multipoint of other model styles are eliminated. Each SNC is individually managed²¹.

3.2 Enhanced Capabilities for point-to-multipoint over the interface

The unidirectional SNC and point-to-multipoint capabilities are supported using the same creation and deletion operations as for bi-directional SNCs.

However, to allow for network environments where point-to-multipoint resources are at a premium, or where the resources benefit pre-connection configuration, a capability to indicate to an EMS that a single point-to-point SNC is the first of many SNCs in a point-to-multipoint configuration will be provided. It should be noted that an EMS should not prevent point-to-multipoint connectivity even if the indication was not provided with the first SNC. This potential future setup indicator capability can be provided in any of the SNC create and delete operations.

3.2.1 Creation and modification of Point-to-multipoint configuration

In order to create point-to-multipoint configurations the existing create SNC operations are used repeatedly to add many uni-directional SNC²² to a single TP as a source. Uni-directional SNCs may be any of the types listed in this document. In addition, an SNC may be added to any capable TP that is already sourcing one or more SNCs connected at any stage and any of the SNCs involved in a point-to-multipoint may be deleted at any stage.

A single bidirectional TP may be involved as a source in many unidirectional SNCs and as a sink in only one SNC. As a consequence a point-to-multipoint configuration from a bidirectional TP may include one bidirectional SNC.

Setting the transmission parameters at a shared TP of any of the SNCs of a point-to-multipoint configuration will operate on the TP and as a consequence be reflected through all of the SNCs that share that TP.

3.2.1.1 Preparing the resource for multipoint operation – PotentialFutureSetupIndicator [transmission parameter]

In some cases the network resources may benefit from preparation for point-to-multipoint configuration. To enable the NMS to inform the EMS that an SNC being created is likely to be part of a point-to-multipoint configuration a new transmission parameter has been added (PotentialFutureSetupIndicator). The NMS need not indicate that an SNC is to be part of a point-to-multipoint prior to adding a further SNC, however, not indication the intention to build a point-to-multipoint may cause the EMS/network to have to rearrange resources and potentially hit traffic to satisfy the subsequent requests. The PotentialFutureSetupIndicator transmission parameter is associated with the TPs.

²¹ I.e. they are created individually and deleted individually, there is no point-multi-point SNC. On this basis each SNC of a point-to-multipoint configuration possesses all of the protection attributes etc.

²² For ATM, a bi-directional SNC with zero flow of traffic in one direction is considered as unidirectional

3.2.1.2 Indicating the degree of tolerable impact – GradeOfImpact [mandatory parameter]

To allow the NMS to indicate to the EMS what level of traffic impact is tolerable when using create&activate to create an SNC or delete&deactivate to delete an SNC and consequently when building/adjusting a point-to-multipoint configuration a new enum has been added *GradesOfImpact_T* {GOI_HITLESS, GOI_MINOR_IMPACT, GOI_MAJOR_IMPACT}²³. If the EMS can't meet the *GradesOfImpact* requested by the NMS, the operation will fail and an appropriate exception will be raised.

Note: There is no interrelationship between GradeOfImpact and ProtectionLevel.

3.2.2 Retrieval of a Point-to-multipoint configuration

The operation getSubnetworkConnectionsWithTPList populated with the shared point of the point-to-multipoint configuration is used to retrieve all SNCs of a point-to-multipoint configuration that use the TP referenced.

3.2.3 Migration from point-to-point to point-to-multipoint

Assume a Point-to-Point SNC is created, with PotentialFutureSetupIndicator set to "RSU_POINT_TO_POINT" for Source TP. If a second Point-to-Point SNC create&activate operation is sent using the same Source TP, the EMS creates the requested broadcast and changes the value of the Source TP PotentialFutureSetupIndicator to "RSU_BROADCAST". For NEs that do not need any configuration change, the source TP value is set to "RSU_ANY_CONFIG". No AVC is sent for this parameter.

3.2.4 Comments on deletion

If a point-to-multipoint configuration is applied to a complex subnetwork (such as mesh) it is likely that resources will be shared within the subnetwork between several SNCs of the point-to-multipoint configuration. As a consequence it is the responsibility of the EMS to ensure that deletion of a single SNC does not "damage" other SNCs of the point-to-multipoint configuration. The NMS shall specify the desired GradeOfImpact for this operation.

3.3 Subnetworks

A subnetwork is an aggregation of NEs and topological links defined by the EMS. Via the EMS the NMS is able to establish SNCs within the subnetwork.

3.3.1 Creating an SNC in a Subnetwork

An SNC may connect any CTPs in the subnetwork including CTPs that are contained in PTPs that are edgepoints of the subnetwork and also CTPs that are contained in PTPs that are not at the edge of the Subnetwork.

²³ This is applicable to any SNC create operation

3.3.2 The Mesh Subnetwork

The Mesh subnetwork is used to cover cases where the EMS manages, as a single entity, a number of NEs that are arranged in a complex topology. An EMS may expose multiple MESH subnetworks.

3.4 Routing Constraints

A routing constraint can be applied by the NMS and supplied to the EMS during the creation of an SNC.

The routing constraints considered are broken down into three distinct groups:

- a) Inclusions – to be supported by this release as indicated in the following section
- b) Exclusions – to be supported by this release as indicated in the following section
- c) Route Attributes – NOT supported by this release

The constraints identified can be considered for routing of SNCs that are:

- a) Fully within the span of control of the EMS – to be supported by this release
- b) Specified with one end outside the span of control of the EMS – NOT supported by this release

In an SNC create operation the NMS may specify inclusion constraints or exclusion constraints but not both inclusion and exclusion.

Routing constraint support is optional in this release. Where an item is indicated as not mandatory, the EMS may throw the exception EXCPT_UNSUPPORTED_ROUTING_CONSTRAINT.

3.4.1 Exclusion

This capability allows the NMS to require the EMS to avoid a particular set of resources when making its routing choices. For example this could allow the NMS to cause an SNC to avoid a PTP that is about to be taken out of service.

As noted above, to be compliant to this interface it is not mandatory for an EMS to support exclusions and an exception has been provided to allow an EMS to reject an SNC create operation that includes exclusion data.

The following exclusions are all optional:

- a) a list of NEs within the subnetwork
- b) a list of TPs (PTPs and/or CTPs)
- c) the resources that are used by a list of previously built SNCs within the subnetwork²⁴
 - i. this is clearly qualified to allow the reuse of the resources at the two ends of the SNC within the subnetwork (e.g. same NE and same specified edge PTP).

²⁴ Other similar but more sophisticated capabilities to support route separacy have not been covered here, but should be considered in a later release.

In this release it is considered sufficient that the resources identified by the NMS to be excluded must not be used by the EMS in the SNC returned, i.e. for an EMS that supports exclusion it is mandatory that all of the items identified in the exclusion list by the NMS must be excluded from the route²⁵.

3.4.2 Inclusion

This capability allows the NMS to request that the EMS uses a particular set of resources when making its routing choices. For example this would allow the NMS to prescribe the exact route of an SNC including channels etc.²⁶

The following inclusions are all optional:

- a) A number of lists:
 - i. a list of NEs
 - ii. a list of TPs (PTPs and/or CTPs including CTPs of the server layer to SNC being routed)
 - iii. a list of specified cross-connects
 - iv. additional info
- b) Giving a degree of description:
 - i. a full route description
 - ii. a partial route description
- c) The NMS will inform the EMS by means of an attribute in the request as to whether the inclusion contains a full route description or a partial route description

The available capacity provided by the direct servers used by an existing SNC (built at the same layer rate as the newly specified SNC) – is NOT SUPPORTED.

In this release it is considered sufficient that the resources identified by the NMS to be included must be used by the EMS in the SNC returned, i.e. the inclusion is mandatory.

It is clearly possible for an NMS to request inclusion of resources that are not necessarily appropriate for use within the requested SNC. The following sections detail cases considered.

3.4.2.1 Request in an inclusion for inappropriate resources

The following cases should be rejected:

- Resources in the wrong layer
- Resources that do not exist

3.4.2.2 Request in an inclusion for resources already in use

An NMS may request resources for creation&activation of a bi-directional SNC that are:

²⁵ It was recognized that longer term a best effort and degree of success measure would be beneficial.

²⁶ This capability removes the need for the SONET specific BLSR direction and timeslot implementation supported in release 1 of the interface and as a consequence these SONET specific items will be deleted from the release 2 interface definition. It should be noted that the East/West aspect of BLSR direction can be recovered from the protection group information.

- a) already in use in an existing Active or Partial bi-directional SNC recognized by the EMS
 - i. the request should be rejected if the resource can not be appropriately share by the existing SNC and the new request, for example, a CTP can not be involved in two SNCs at the layer of the CTP, so a request to use the CTP again should be rejected
- b) involved in a connection in the network but that are not part of an SNC recognized by the EMS
 - i. If the request is for inclusion of a specific cross-connect and the resources in the network are already used in a cross-connect that matches the request exactly (i.e. same points and same SNC type) and the cross-connect in the network is not represented in an SNC known to the EMS, then the EMS shall make use of the cross-connect (i.e. idempotent behavior)
 - ii. If the request is for inclusion of a specific cross-connect and the resources in the network are already used in a cross-connect that does not match the request (e.g. wrong SNCType) reject the requested SNC and then produce an SNC including the previously “unknown” cross-connect and provide it to the NMS by normal means (for the NMS to resolve as necessary).

3.4.3 Route Attributes

In addition to the specific individual resources that are used in the routing of an SNC, the overall properties of the SNC also have to be considered. These include the various costs of the route relating to both the quality of the route provided and also to the lost potential for routing other SNCs

Route attributes are:

- Constrain the route cost – NOT SUPPORTED
- equipment cost
- delay
- etc.

Considering that the MTNM model does not include the details of a cost model, it is clearly inappropriate at this stage to include this capability. It has been decided to defer this capability to a later release.

3.4.4 Constraints beyond the span of control of the EMS

Specification of constraints beyond the span of control of the EMS is potentially necessary for Network Routed SNCs. In this document it is considered the multi-route management, useful to provision pre-calculated backup routes by NMS to EMS..

3.4.5 Interaction between constraints

If the inclusion and exclusion list identify the same resources in such a way that the request is effectively invalid, then the request should be rejected.

3.4.6 Network Routing

For SNCs where rerouting is allowed there are a number of cases that need to be considered. These cases are controlled via attributes at the interface.

The following table shows the usage of networkRouted, rerouteAllowed and networkReroute.

Network Routed	Yes	No	N/A
Reroute Allowed - Network Reroute			
No (Not allowed to be rerouted from EMS neither the network)	Was last routed in the network	Was last routed by the EMS (or the NMS if full route was specified)	Only valid at creation time (of modification time initiated by modifySNC). Once the SNC is created, will tell where it was routed
Yes – No (Allowed to be rerouted by the EMS not the network)	SNC was initially routed by the network and no EMS rerouting took place yet	SNC was routed or rerouted by the EMS	
Yes – Yes (Allowed to be rerouted by the network only)	SNC was routed or rerouted by the network	SNC was initially routed by the EMS and no network rerouting took place yet	
Yes – Not Set (Allowed to be rerouted by either the EMS or the network)	Network Routed indicates who last rerouted the SNC		

Table 1 : routing attributes usage

3.5 Actual Route Discovery

Regardless of the routing policy and constraints applied to the network the NMS may choose from time to time to retrieve the route of a particular SNC from the EMS. The EMS shall provide a service to allow the NMS to retrieve the route information for any specified SNC (i.e. it is mandatory that the EMS provides the service and also responds with valid route information. However, it is clearly not mandatory for the NMS to make use of this service).

The route information, in the form of cross-connects, should identify the resources allocated to the SNC at the time of the request²⁷ and the way in which these resources are used²⁸. For example for a resilient SNC that is implemented in the Subnetwork using subnetwork connection protection, all details of the route including normal and alternative paths should be provided. If the SNC is using some form of dynamic rerouting then it is clearly acceptable that only the route includes only the single thread of cross-connects that is supporting the Traffic (as there may be many alternative potential paths that could be selected). Only resources in the layer of the SNC should be identified, so for a VC12 SNC, for example, only cross-connects that deal with passing signals of VC12 characteristic information will be supplied.

A fully detailed explicit route discovery service will be supported in this release.

²⁷ In situations where the EMS does not support a capability to reroute (e.g. on failure) after the original route set-up has occurred, the response to the request will be the same each time the request is made. However for an EMS that does support a rerouting capability, the route request will yield a snap shot of the current state and information provided may clearly change from one request to the next.

²⁸ In this release it is assumed that the EMS will not pass back exclusion/inclusion constraint used during the routing.

4 SNC MODIFICATION

For this release, the following modification types are foreseen:

- **Add / Remove Protection Leg:**

the EMS should only try to modify the SNC by applying or removing the legs provided in SNC data. When adding or removing a protection leg, the EMS should compute the differencing of cross connect on the common NE.

- **SNC Rerouting:**

the EMS should use the routing constraints, if any, to reroute the SNC from end to end.

- **Add / Remove Backup Routes:**

more routes can be assigned to a given SNC. *See related chapter.*

4.1 Scenarios of Add / Remove Protection Leg

1. Change the static protection level from unprotected to fully protected, and vice versa. See Figure 1. This configuration is not applicable for singleton subnetworks. The NMS may specify either the full route, a partial route, any routing constraints or nothing.

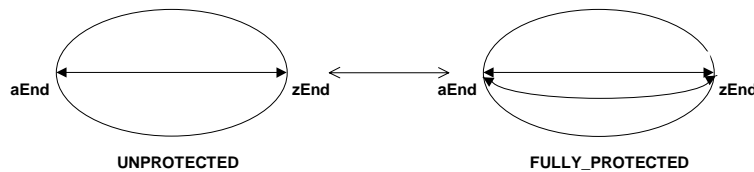


Figure 1: ModifySNC: UNPROTECTED <-> FULLY_PROTECTED

2. Change the shape (sncType) from ST_SIMPLE to ST_ADD_DROP and vice-versa with aEnd (or zEnd) as the reliable TP. See Figure 2. The NMS may specify either the full route, or only the switch port.

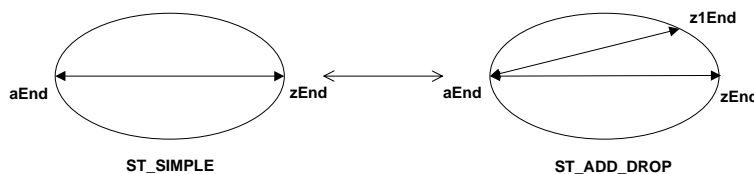


Figure 2: ModifySNC: ST_SIMPLE <-> ST_ADD_DROP; aEnd is reliable TP

3. Change the shape (sncType) from ST_SIMPLE to ST_ADD_DROP and vice-versa, where neither aEnd nor zEnd is the reliable TP (the ME with x-conn of type ST_ADD_DROP is 'in the middle'). See Figure 3. This configuration is not applicable for singleton subnetworks. The NMS may specify either the full route, or only the switch port.

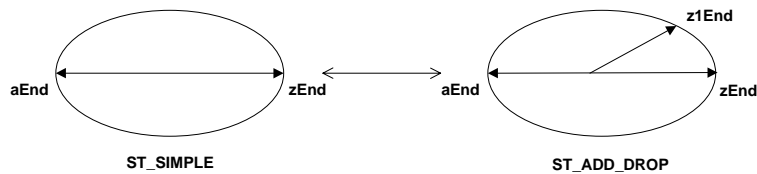


Figure 3: ModifySNC: ST_SIMPLE <-> ST_ADD_DROP; aEnd is not the reliable TP

4. Change the static protection level from unprotected to partially protected, and vice versa. See Figure 4. This configuration is not applicable for singleton subnetworks. The NMS may specify the full route or just the switch port (in case of add a protection leg) or just the switch cross connect (in case of add and remove a protection leg)

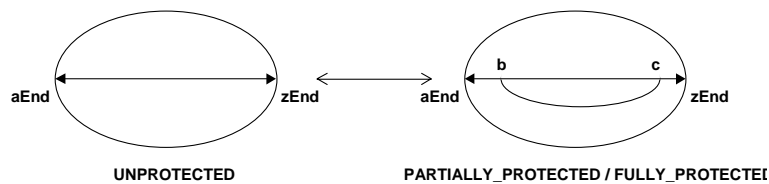


Figure 4: ModifySNC: UNPROTECTED <-> PARTIALLY_PROTECTED

4.2 Scenario of SNC Rerouting

The NMS may either specify the full new route or some routing constraints or just invoke reroute without parameters.

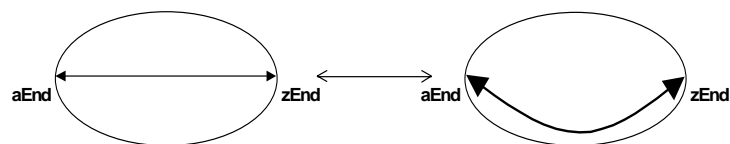


Figure 5: ModifySNC: simple reroute

4.3 Scenario of Add / Remove Backup Route

Add/Remove route: one SNC may have more routes, one intended and the others backup. It is possible to add a new route, activate it and deactivate the former route. See also related chapter.

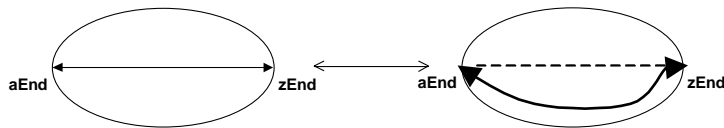


Figure 6: add new route to an SNC

4.4 Impacts on Static Protection Level and SNC type

The modification of the SNC's **Static Protection Level** does not impact the **SNC type**. The EMS may support the modification of an unprotected SNC to being a partially, fully or highly protected SNC while maintaining the same SNC type.

Concerning **SNC type** modifications, the MTNM interface only defines the transitions between ST_SIMPLE and ST_ADD_DROP_A or ST_ADD_DROP_Z to be mandatory.

Note that ST_ADD_DROP_Z bi-directional is new feature of Version 3.

The relation between these four basic modification steps and the corresponding values for SNCType and staticProtectionLevel are shown:

Modification Step				SNCType	staticProtection Level	↔	SNCType	staticProtection Level
unprotected	↔	fully protected		ST_SIMPLE	UNPROTECTED	↔	ST_SIMPLE	FULLY PROTECTED
unprotected	↔	Y-protected	↔	ST_SIMPLE	UNPROTECTED	↔	ST_ADD_DROP	UNPROTECTED
unprotected	↔	Y-protected ('middle' branch)		ST_SIMPLE	UNPROTECTED	↔	ST_ADD_DROP	UNPROTECTED
unprotected	↔	partially protected		ST_SIMPLE	UNPROTECTED	↔	ST_SIMPLE	PARTIALLY PROTECTED

Table 1: Relationship between SNCType/staticProtectionLevel values

4.5 Modification Behaviors

There are two main EMS behaviors when modifying an SNC: **preserving or not the SNC name**.

If the EMS does not preserve the SNC name, then the EMS will create a new pending SNC from an existing pending or active SNC. It is similar to createSNC but the created SNC is made out an existing one. It is then possible to swap the SNCs, i.e. deactivate the former SNC and activate the new one in one shot.

If EMS supports multiple routes, then it has to preserve SNC name through operations on routes (add/remove and switch).

The nativeEMSName of the new SNC must be the same as the nativeEMSName of the original SNC (the SNC to be modified).

4.5.1 SNC state in case the EMS does not preserve the SNC name

If createModifiedSNC is used:

A new modified SNC will be created in state PENDING. And an Object Creation notification of the newly created (modified) SNC in state PENDING.

The original SNC is left in its original state.

The original SNC to be modified may be in PENDING state or in ACTIVE state.

When swapSNC or modifySNC is used:

SUCCESS:

The original SNC is deleted or moved back to pending depending on the retainOldSNC parameter and the new (modified) SNC is put in state ACTIVE. The following is then notified:

An Object Deletion notification of the original SNC or a SNC state change to partial.

A state change notification of the new (modified) SNC from PENDING to ACTIVE.

The original SNC to be modified must be in ACTIVE state.

FAILURE:

The original SNC could not be removed and/or the new (modified) SNC could not be put in ACTIVE state.

The SNCState for both SNCs will be SNCS_PARTIAL

If the resulting SNC state is PARTIAL, such an SNC can only be set to state ACTIVE by applying operation swapSNC() or modifySNC() again.

During "activation" of an SNC, the TPs of the SNC are configured, and the necessary cross-connects are established for the SNC.

Operation swapSNC() and modifySNC() can be called repeatedly and eventually should succeed (except in conflict cases where the SNC remains in SNCS_PENDING or SNCS_PARTIAL). Should the missing or excess cross-connects be changed in the MEs via a craft interface, for example, prior to communications to those MEs being re-established, the swapSNC() / modifySNC() command ultimately should succeed when communication to the MEs is re-established (even though all cross-connects already exist).

SUPPORTING DOCUMENT: SNC AND PROTECTION

In the following Tables, the state changes related to the operations are shown.

Initial State of SNC to be modified	Operation	Intermediate State	Operation	Final State
PENDING	createModifiedSNC()	PENDING for both	N/A	
ACTIVE	createModifiedSNC()	ACTIVE for the old SNC PENDING for the new SNC	swapSNC()	ACTIVE for the new SNC. If the retainOldSNC parameter is false, the old SNC has been deleted If true, the old SNC is left in PENDING
ACTIVE	ModifySNC()	N/A	N/A	ACTIVE

Table 2: state changes related to the createModifiedSNC() and swapSNC() operations in case of success

Initial State of SNC to be modified	Operation	Intermediate State of modified SNC	Operation	State of modified SNC
PENDING	createModifiedSNC()	PENDING for the original SNC, N/A for the new one	N/A	
ACTIVE	createModifiedSNC()	ACTIVE for the old SNC. N/A for the new one	swapSNC()	PARTIAL for both
ACTIVE	ModifySNC()	N/A	N/A	PARTIAL for both

Table 3: state changes related to the createModifiedSNC() and swapSNC() operations in case of failure

Retaining the old SNC

This parameter in SNCModifyData can be used if the original SNC must be preserved in order to swap back to that SNC at some point in time. It is applicable only in case EMS does not preserve the SNC name.

4.5.2 SNC state in case the EMS preserves the SNC name

If createModifiedSNC is used:

The addressed SNC route is modified. If the SNC was in PENDING or PARTIAL state, then the state is unchanged. If the SNC was in ACTIVE state, then the output state is PARTIAL. In case the SNC has several routes, then the administrative state of the addressed route will always transit to LOCKED state.

If modifySNC is used:

SUCCESS:

SNC state is set to ACTIVE. In case the SNC has several routes, then the administrative state of addressed route is set to UNLOCKED.

This operation cannot be applied to a PENDING SNC .

FAILURE:

SNC state is set to PARTIAL. In case the SNC has several routes, then the addressed route administrative state is anyway set to UNLOCKED.

During “activation” of an SNC, the TPs of the SNC are configured, and the necessary cross-connects are established for the SNC.

Operation activateSNC(), modifySNC(), switchSNC() can be called repeatedly and eventually should succeed (except in conflict cases where the SNC remains in SNCS_PENDING or SNCS_PARTIAL). Should the missing or excess cross-connects be changed in the MEs via a craft interface, for example, prior to communications to those MEs being re-established, the activateSNC()/modifySNC()/switchSNC() command ultimately should succeed when communication to the MEs is re-established (even though all cross-connects already exist).

In the following Table 4, 5 the state changes related to the operations are shown, in case SNC has only one route.

Initial State of SNC to be modified	Operation	Intermediate State of modified SNC	Operation	Final State of modified SNC
PENDING	createModifiedSNC()	PENDING	activateSNC()	ACTIVE
ACTIVE	createModifiedSNC()	PARTIAL	activateSNC()	ACTIVE
ACTIVE	ModifySNC()	N/A	N/A	ACTIVE

Table 4: state changes related to the createModifiedSNC() and activateSNC() operations in case of success

Initial State of SNC to be modified	Operation	Intermediate State of modified SNC	Operation	Final State of modified SNC
PENDING	createModifiedSNC()	PENDING	activateSNC()	PARTIAL
ACTIVE	createModifiedSNC()	PARTIAL	activateSNC()	PARTIAL
ACTIVE	ModifySNC()	N/A	N/A	PARTIAL If no XCs are activated, the SNC state is not-existing

Table 5: state changes related to the createModifiedSNC() and activateSNC() operations in case of failure

An already active SNC can be modifyActivated again; the EMS is allowed to not send the commands to the ME a second time for the cross connect establishment however the commands may be sent for the transmission parameters. While in SNCS_PARTIAL state, it is possible to activate a modified SNC again, this corresponds to a retry.

4.5.3 TolerableImpact

In addition tolerableImpact and tolerableImpactEffort (of type ProtectionEffort_T) for the purpose of qualifying the conditions under which an SNC modification may be performed, are provided as parameters of the operation.

Parameter tolerableImpactEffort is a qualification of the requirement that the tolerableImpact as specified, is met.

The NMS may ask for different grades of impact. The EMS can only verify the intent of whether the grade of impact could be met. In actuality when commands are sent to the network, unforeseen circumstances could make a hitless request to fail, causing some service hit. Other times, NMS could accept a major impact, where the EMS can disconnect and connect if suitable tolerableImpact is requested.

4.5.4 Route Change

A route change is notified using the existing NT_ROUTE_CHANGE notification in case operation **createModifiedSNC()** does not result in exception.

Field "route" of the route change event may then be set to emptyList. Persistency of the NT_ROUTE_CHANGE notifications are expected to be maintained at the same level as other configuration change notifications.

4.6 Multiple Route Management

MTNM V2 definition allows the (implicit) association between a SNC and only one route.

MTNM V3 allows to define, for a given SNC, its NOMINAL or INTENDED route, plus zero, one or more BACKUP routes.

The INTENDED route could be defined as the preferred, or default route for a given service. Practically, the intended route could be simply the first time provisioned route, or the preferred route for a number of factors, from network engineering to intrinsic media reliability.

The BACKUP or ALTERNATIVE route (from now on “bkp route”), which is partly or totally different from intended (but with same end points), is useful mainly for restoration and maintenance purposes.

Concerning the “SNC management mode of operation”, it is considered the case of “pending state supported” and “sharing of cross-connects not supported”. Anyway it is possible to include the case of “pending state not supported”, but EMS shall be able to e.g. create&activate a bkp route in a hitless way – which could also imply the capability of cross connects sharing.

The following figures show some examples of restoration scenarios involving SNCs with 2 routes, the INTENDED and BACKUP ones:

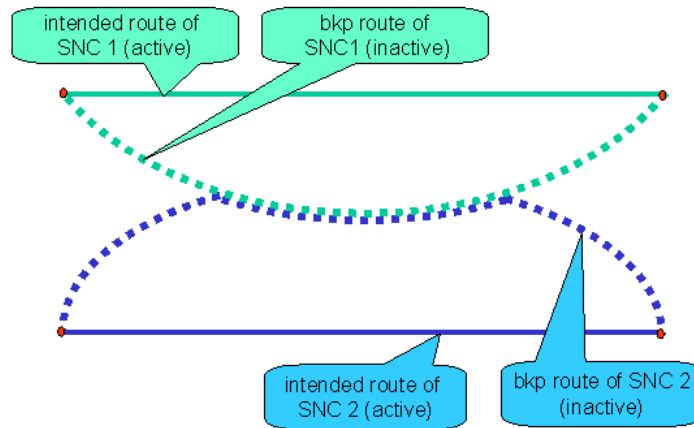


Figure 1: sharing of inactive backup routes, no failures

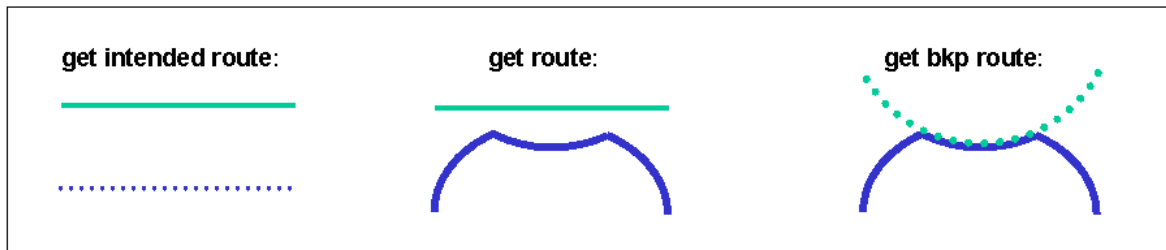
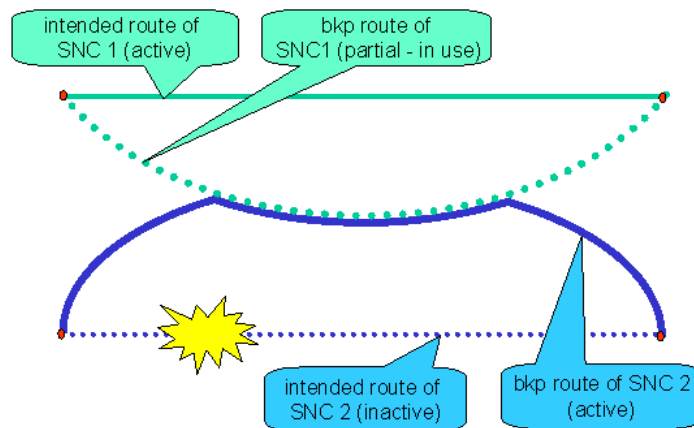


Figure 2: sharing of inactive backup routes, failure on high prio SNC

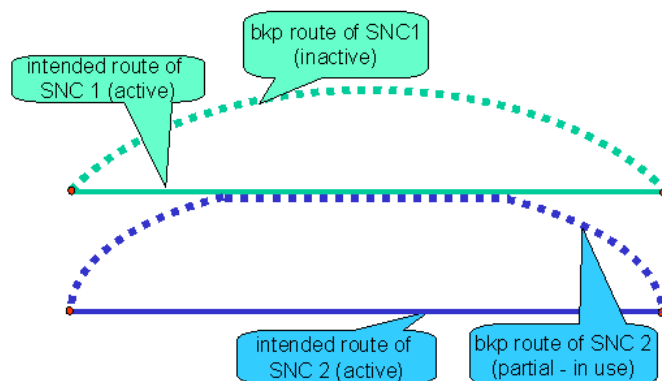


Figure 3: sharing of intended route, no failures

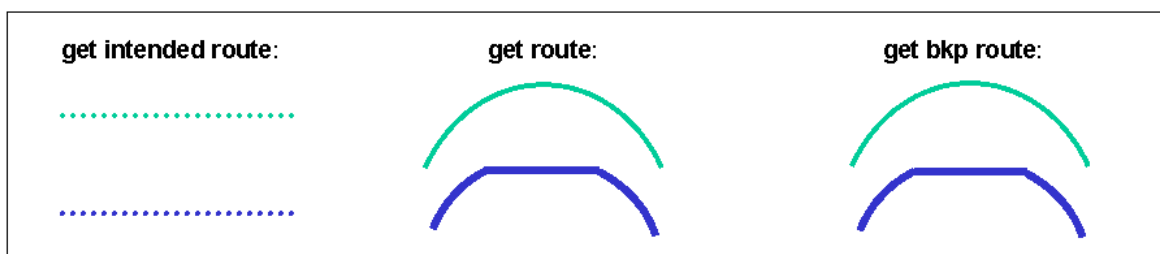
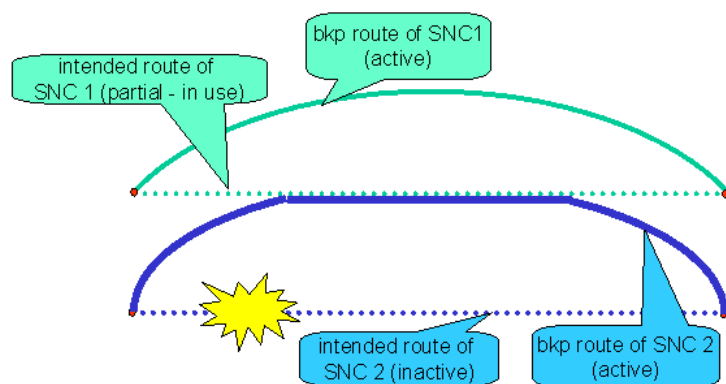


Figure 4: sharing of intended route, failure on high prio SNC

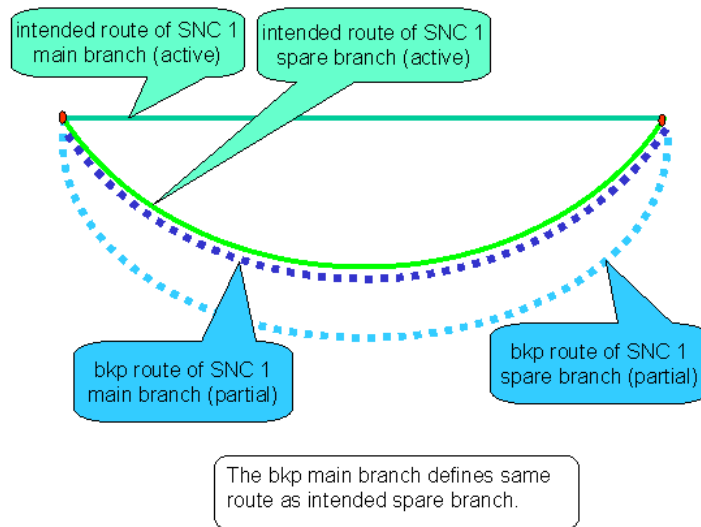


Figure 5: SNCP protected SNC with backup route, no failures

Note that in figure 5 the SNC 1 is an end to end protected SNC via SNCP. According to current definitions, the route retrieved by `getRoute()` operation includes both the main and spare branches, so both branches are marked as “current”.

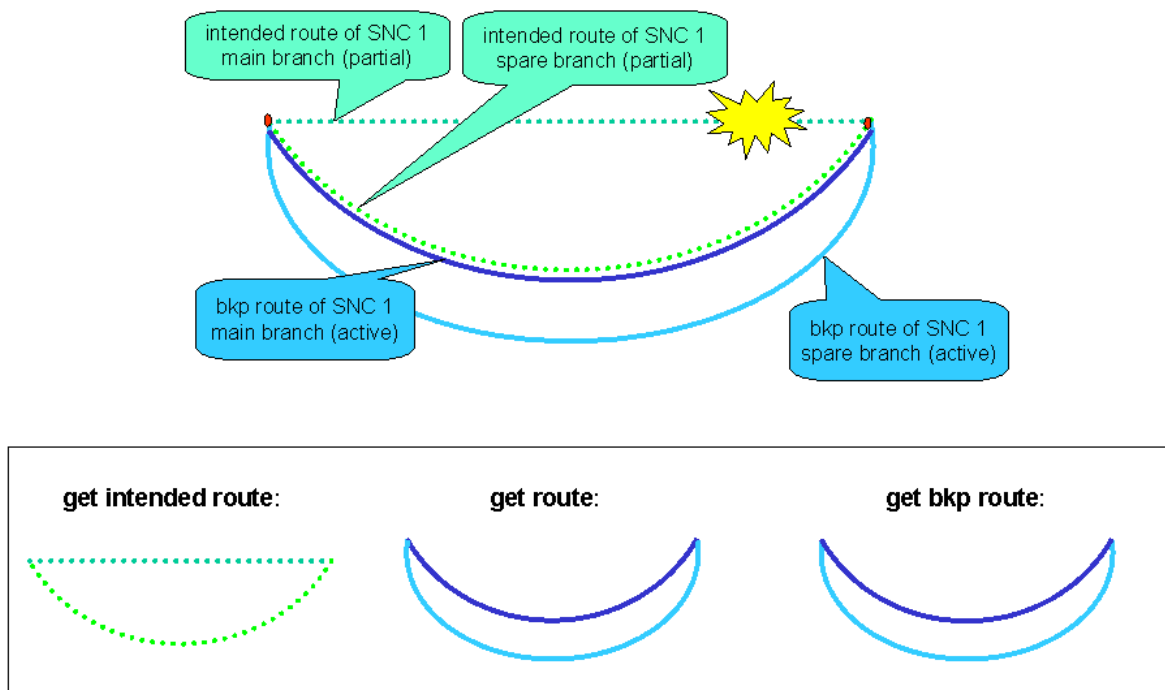


Figure 6: SNCP protected SNC with backup route, failure on main branch

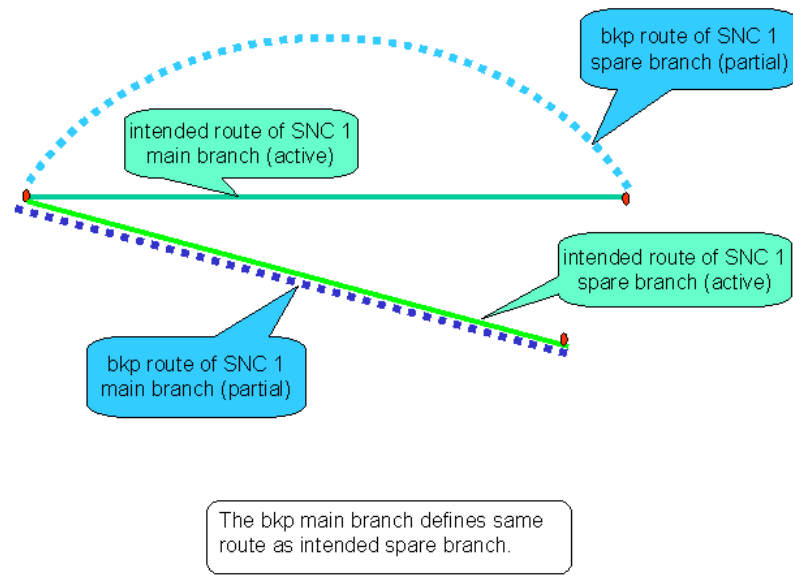


Figure 7: Add Drop SNC with backup route, no failures

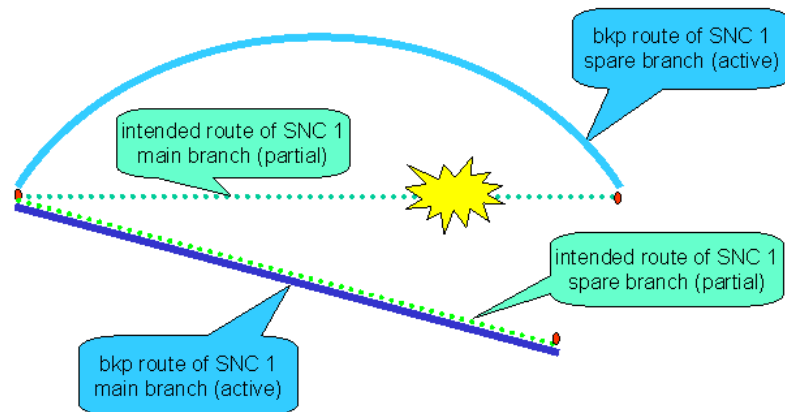


Figure 8: Add Drop SNC with backup route, failure on main branch

SUPPORTING DOCUMENT: SNC AND PROTECTION

In the following table the state changes related to the SNC modifications are shown, in case SNC has more routes.

Initial State of SNC to be modified	Initial State of route to be modified	Operation	Intermediate SNC State	Intermediate route State	Operation	Final State of SNC to be modified	Final State of route to be modified
PENDING	LOCKED	createModifiedSNC() addressing the intended or bkp route	PENDING	LOCKED	activateSNC()	ACTIVE	UNLOCKED
PARTIAL	UNLOCKED OR LOCKED	createModifiedSNC() addressing the intended or bkp route	PARTIAL	LOCKED	activateSNC()	ACTIVE <i>Restoration process may switch the routes so activating the just modified one</i>	UNLOCKED
ACTIVE	UNLOCKED or LOCKED	createModifiedSNC()	PARTIAL	LOCKED	activateSNC()	ACTIVE	UNLOCKED
ACTIVE	UNLOCKED or LOCKED	modifySNC()	N/A	N/A	N/A	ACTIVE	UNLOCKED

Table 6: state changes related to the createModifiedSNC(), modifySNC() and activateSNC() operations in case of successful activation of the SNC

In the following table the state changes related to the addition and removal of SNC routes are shown.

Initial State of SNC to be modified	Initial State of route to be added	Operation	Intermediate SNC State	Intermediate route State	Operation	Final State of SNC to be modified	Final State of route to be modified
PENDING	NOT EXISTING	addRoute()	PENDING	LOCKED	activateSNC() switchSNC() is refused	ACTIVE	UNLOCKED
PENDING	NOT EXISTING	addRoute()	PENDING	LOCKED	setAdminSt = UNLOCKED switchSNC() is refused	ACTIVE	UNLOCKED
PARTIAL	NOT EXISTING	addRoute()	PARTIAL	LOCKED	activateSNC() or setAdminSt = UNLOCKED	ACTIVE this time activation succeeded	UNLOCKED
ACTIVE	NOT EXISTING	addRoute()	ACTIVE	LOCKED	activateSNC() or setAdminSt = UNLOCKED	ACTIVE	UNLOCKED

Table 7: state changes related to the addRoute successful operation

Next table shows the meaning of route attribute “intended”:

route attribute	meaning	behavior
INTENDED = true	the route is the intended, or nominal, or default one. There can be only one intended route per SNC	In case the SNC is “revertive”, then the restoration process will always switch to this route if possible, i.e. if there are no failures or if the intended route is not locked.
INTENDED = false	the route is a backup one. There could be zero, one or more backup routes per SNC	The route is considered as protection route, available in case of failure. By means of setIntendedRoute() it is possible to set the addressed route as intended, and so the formerly intended route is now a backup route

Next table shows the meaning of ACTUAL route states, valid either the PENDING state of SNC is supported or not:

route ActualState	XCs
INACTIVE	None of its XCs is active in the network.
ACTIVE	all its XCs are active in the network. So it is the route where SNC traffic is currently carried. inUseBy shall be always false
PARTIAL	one or more, but not all the XCs are active in the network. <ul style="list-style-type: none">• If the route was unsuccessfully activated, then inUseBy shall be always false.• If the route was unsuccessfully deactivated, then inUseBy could be true.

Note: for efficiency reasons, in some implementations a route can be moved to PARTIAL actual state, so its activation, either by NMS or by restoration process, will be faster.

SUPPORTING DOCUMENT: SNC AND PROTECTION

Next table shows the meaning of ADMINISTRATIVE route states, valid either the PENDING state of SNC is supported or not:

route AdministrativeState	Meaning	Changeable by
LOCKED	The route is not allowed to be active.	<p>setRoutesAdminState()</p> <p>if all the routes of a given SNC are set to locked, the SNC transits in PENDING state.</p> <ul style="list-style-type: none"> - createSNC() creates one locked intended route - deactivateSNC() locks all routes - addRoute() creates one locked backup route - createModifySNC() modifies the addressed route, which transits to locked <p>Note for createModifySNC(): the SNC will transit to PARTIAL state, because the just modified route was never unlocked/activated before, and the old route is still ACTIVE in the network. So the SNC PARTIAL state means that an activateSNC or route unlock operation is needed.</p> <p>Moreover, it is not possible to determine the actual route of the SNC in the network once the route was modified.</p>
UNLOCKED	The route is allowed to be active	<p>setRoutesAdminState()</p> <p>If a route was modified by createModifySNC(), then setting it to unlocked implies the (possible) activation of the new route. It is up to the EMS/network to activate or not the just unlocked route, according to failure scenario, revertive behavior, etc.</p> <p>If the activation is successful, (route actual state is ACTIVE) then also the SNC will transit to ACTIVE.</p> <p>If a locked route of a PENDING SNC is set to UNLOCKED, the SNC will transit to either ACTIVE or PARTIAL.</p> <ul style="list-style-type: none"> - createAndActivateSNC() creates and unlocks the intended route - activateSNC() unlocks all routes - switchRoute() is a "manual" switch, so no route is locked or unlocked. - modifySNC() modifies and unlocks the addressed route - deleteSNC() fails if at least one route is unlocked

end of document