**Service Termination Points**

A Service Termination Point (STP) is an abstract reference to the “endpoint” of a connection. An abstract STP does not represent a specific physical object, it represents the notion of an endpoint of a connection. An STP can also represent an abstract point in a network topology where a connection *could* logically terminate. When we refer to an “STP” we are referring to the abstract notion of the endpoint, whatever it is and regardless of any physical association.

We can use symbols to represent STPs. And we can construct a set of operations and a grammar that allow us to manipulate Connections and their associated STPs in a symbolic manner. This is particularly useful in discussing PathFInding and describing operations a PathFinder might use to resolve and construct a path.

This use of these abstract objects and operations is an example of a “path algebra”. I.e. it allows us to manipulate paths symbolically without knowing all the specific topological constraints. This path algebra is one of the real values of abstracted STPs. And while it may not be of practical use in provisioning, it is useful in defining the topological operations used in pathfinding.

Implicitly, STPs do have certain characteristics – either stipulated by a service request, or drawn from a physical point they might represent in the topology. But in using STPs algebraically, we are more concerned that an STP represents a *particular set* of topological characteristics than what those particular characteristic might be.

The notion of an STP can also be *real* as well as abstract. A “real STP” is a specific (not general) point in the topology where a connection could be terminated. It *may* have a name. In practice, when an STP gets named, it will receive a full Endpoint Reference (a domain name/endpoint name tuple) and that entry in the Name Database will link to an element in the Topology Database. Topology elements will also need to link back to one or more endpoint names. So the NameDB and TopoDB are inter-related, and may even be implemented as a single integrated data structure.

Examples:

A connection X extends from STP “A” to STP “B”. Another connection Y extends from STP “B” to STP “C”. Since the egress point of the first connection is identified symbolically as the ingress point of the second connection, then we say connection Y is *concatenated* to connection X. Further, under some qualifying conditions on the connection constraints, we can say the concatenation is equivalent to a connection from A to C: If A>B & B>C; then A>C;

We define “>” as the “connects” operator. “A>B” is “A connects to B”. The > operator represents a set of constraints that when projected onto the topology database produces a set of resources that constitute a unidirectional path connecting STP A and STP B. There may be many resource sets (paths) that meet the constraints, but the A>B construct represents a single conformant but non-specific path from the set of valid paths. (Note: in the basic grammar, STPs are unidirectional objects, as are connections then by implication. A>B ≠ B>A. One could say: A>B = B<A, in the Grammar, these represent the same unidirectional connection.

The “==” is the “equivalence” operator. When two STPs are equivalenced, they represent the same specific but abstract endpoint – and they become symbolically interchangeable. For example: If, A>B1, B2>C, B1==B2 then A>B2 & B2>C = A>B2>C = A>B1>C;

For instance, the connection A>C could be the concatenation of two shorter connections. A>B, and B>C. These are abstract STPs. If however, we take the constraints in a service request and use those constraints to map these segments to real STPs, we may find that A was mapped apriori to the topological construct “VLAN=27, port=1/0, Node=F10e, domain=Allen”. And likewise C was mapped to construct “Port=0/4, node=E300x, domain=Charlie”, and B was mapped to topological construct “domain=Betty”; The implications here is that the segment A>B goes from a very tightly constrained real topological location, to a real STP that represents a much larger topological construct – a whole domain. While this may not be much use for a standalone connection request, in conjunction with the B>C segment, we have symbolically said “Build a connection from A to C that goes thru domain Betty. “ We have not made any other precondition on **how** or **where** the connections touch in Betty, just that they transit Betty by being concatenated anywhere that works within the constraints.

So from these simple examples, it should be clear that Service Termination Points (STPs) are abstract constructs that can be mapped to real topology in a myriad of useful ways, and that we can use these both to describe NSI connection processing in an abstract sense, and to represent real topological locations when appropriate.

Definitions:

**Service Termination Point** := 1. An abstract object that represents the ingress or egress point of a connection, or the abstract notion of a location in a topology where a connection could potentially originate or terminate. 2. A real point in a topology where a connection can originate or terminate.

**Endpoint** := In NSI, this is a location within a network that can be used as an endpoint for a connection.

**Endpoint Reference** := a two-tuple consisting of a {<network name>, <endpoint name>} . An “endpoint reference” is this tuple, the “endpoint” itself is the topological location it identifies.