Peoples,

I spent some time going through the specification in detail last week, but unfortunately, I used an older version of the document so last night I consolidated my comments against Guy’s version I pulled down on Monday. I colour coded for your reading pleasure. Red implies original text from the document.

Section 3.4 NSI Service Definitions

“A service request is fully specified when all parameters associated with that service have been determined either by explicit user specification or by implicit default values found in the Service Definition.”

Are service definition defaults a common global configuration or are these defaults a localized decision? If they are a localized decision then the requestor NSA should “fill in the blanks” so that all subsequent provider NSA contacted have the assumed default values filled in the service request.

And similarly,

Section 5.1.2 Service Definitions for Connection Services

“If a service parameter is not present in the service request, then the provider NSA should “fill in the blanks” from default values in the Service Definition. As the request is processed down the NSA service tree, default values adopted in one transit network may implicitly constrain the request in downstream networks. Therefore, in general, each NSA should use default values that provide the greatest leeway to the pathfinder in satisfying the request both within the local network and in external downstream networks.”

This mechanism is rather complex as described. If service parameters are left open ended by some NSA, then an additional visit to that NSA must be performed to finalize the actual negotiated parameters. In the tree model this would require a second pass to commit the final service definition negotiated across the network. In the chain model it would require the end terminating NSA in the chain to finalize the service definition and then every node returning up the chain would finalize their definition.

Section 3.6 Trust and authentication in NSI

The term “service handler” is used for the first time. Should this read, “message handler” as defined in figure 5?

“The second mode is to employ a more message based trust framework such as Web Services. This message based form is more appropriate for occasional messaging as might occur between an application agent and various provider NSAs.”

I believe this last statement is subjective and should be removed from the document. I am currently working on a production product that is processing over 400 SOAP messages a second with per message authentication while performing other more computationally heavy tasks. I think this exceeds the expectation of “occasional” :-)

Section 3.7 Error handling NSI

The term “Network errors” is ambiguous based on the topic at hand. Could we better qualify this to “Service plane (NSI protocol infrastructure) errors” to distinguish this from transport plane network errors? I was going to complain about our confusing use of the term “service” but have no valid alternative :-)

“A failure in the Service Plane should not result in an incomplete service.” should be restated as a goal objective. If we have started to provision an end-to-end transport connection and part way through we have a provider NSA or DCN failure we have no way of knowing if the sub-network connection was established or not, and therefore, we are in an incomplete state that we cannot recover from.

“For example, a user may request that if any NSA fails, all the NSAs handling the same service instance should tear down the Connection Service in the Transport Plane.” This is a very interesting error handling scenario. I had assumed that only the requesting NSA would need to listen for failure notifications from individual provider NSA against the services it instantiated, but with this example we imply that each NSA will listen for events from other NSA on services it may only tandem so that it might tear down the tandem during failures (a chain would only require adjacent NSA to be monitored as failures would be cascaded). Do we really want this additional complexity to protect against a double failure? I would recommend we keep it simple and have the requesting NSA decide when to tear down the transport resources through a cancel request, otherwise, all connection resources get cleaned up at end-time.

“Failures in the Service Plane during Reservation, Provisioning, Teardown, and Release phases can cause problems for the operation of the NSI.” Do we want to normalize these phases against the states described in Figure 15? Specifically, the phase “teardown” is not stated in Figure 15. In fact, is “teardown” not redundant with “releasing”?

“Figure 11: Local/Remote Failures” was a bit confusing for me. Does the rounded square represent a local NSA?

Should we expand this section, or add an appendix covering error use cases?

Section 3.8 Transport failure awareness

Detection of transport errors should be a local issue but the NSI protocol needs to specify a mechanism to notify other NSA of a local transport failure against a connection. The correlation of local transport error to impacted connections is a local matter.

Once again, should we expand this section, or add an appendix covering error use cases?

Section 5.1.3 The Connection Service States

“In the NSI, a connection goes through five phases: Reserving, Scheduled, Provisioning, In-Service, Releasing.” I think we could benefit from having a high level state machine in the document to capture additional information implied in the text. As I was trying to correlate the phases to the operations as defined in Figures 15 and 16, as well as include error handling, I formed the opinion that we need some additional state information beyond what we in Figure 15 to show the life cycle of a connection.

“When the Release has completed, the connection object is deleted from the Service Plane.” Given my previous statement I believe we do not want to delete the connection object after resources have been freed, but introduce additional end states that allow the object to exist after the scheduled end time. At the moment, if a cancelRequest is processed to completion the connection object would end up being deleted as soon as the transport resources have been released. Now I can no longer see the state of this connection object, and therefore, cannot determine any state information about the connection after the fact. If the originating user did not issue the cancel request, they would have no way to query their connection to see what happened.

I think an easy solution to this problem is to have a set of end states for a connection object and place a hold over timer on the connection object that would eventually remove it from the NSA, but only after a period of time (say 24 hours). We also need to clearly indicate if the connection was terminated due to error, a cancel request, or if the scheduled end-time occurred.

Section 5.1.4 Connection reservation messages

“If the connection request includes a valid start-time and an end-time then the request is considered to be an advance reservation request.” Does this preclude me from specifying a ”duration” instead of an “end-time” for an advance reservation?

“If the connection request has the start-time set to ‘asap’ and has a duration field rather than an end time field, the request is considered to be an immediate reservation request.” Why preclude an end-time as a possible field for the immediate reservation? We only need trigger off the “asap” to determine it is an immediate reservation.

Can we normalize the “de-provisioning” terminology to “releasing” as stated in previous sections?

“When operating in explicit mode, it is the responsibility of the requestor NSA to signal the reservation to begin provisioning and to begin de-provisioning of the connection. These signals are known as the ProvisionRequest and CancelRequest.” Based on the statement in section 5.1.5, paragraph 2, “The reservation end-time refers to the time at which the reservation is removed. (If the user has not yet sent a CancelRequest signal the connection is de-provisioned first)” can I assume that “signaling of de-provisioning” is optional and both automatic and explicit mode connections will be automatically torn down when end-time occurs?

Section 5.1.5 Connection reservation and timing parameters

I think we need to make the following two definitions consistent as they introduce conflicting code logic that really doesn’t need to be.

1. “For advance reservation with *automatic* provisioning, the start-time refers to the time at which the connection moves from provisioning state to in-service state.”
2. “For advance reservation with *explicit* provisioning, the start-time refers to the time at which the provider is able to accept a provision signal.”

In #1 the NSA must start provisioning the local connection segment at “start-time” – “guard-time” (this is my definition of guard-time and not the one in the document) so that the connection can be “in-service” by “start-time.” However, for #2 the “start-time” parameter represents the point at which the requesting NSA can request provisioning of the connection to start. In the case of #2, the actual “in-service” state is achieved at “start-time” + “guard-time” and not “start-time” as in #1. I think we should try to avoid this type of confusion in the document, as it will also imply two separate definitions in an NSA implementation.

May I suggest that the behavior of the “ProvisionRequest” operation changes the state of an “advance reservation with explicit provisioning” to an “automatic provisioning” state. This would be beneficial for two reasons:

1. A Requestor NSA can send down a “ProvisioningRequest” operation before “start-time” without receiving an error for issuing the request too early. The Provider NSA would then transition the explicit reservation to an automatic state and start provisioning the connection at “start-time” – “guard-time”.
2. If the Requestor NSA is made aware of the connection provisioning “guard-time” it can issue the “ProvisioningRequest” operation at “start-time” – “guard-time” to get the same behavior as the automatic provisioning case.

In both cases the “ProvisionConfirmation” notification, or perhaps a new ActivationConfirmation notification (if we want to keep the other one as ack to the operation itself), would be sent back when the connection is provisioned in the transport plane.

“The reservation end-time refers to the time at which the reservation is removed. (If the user has not yet sent a CancelRequest signal the connection is de-provisioned first).” Could we not also use “start-time + duration” to imply “end-time”?

““Infinite” can be used as an end time. In this case, resources are reserved forever (i.e. until a release request is received or may be overwritten by policy limits). ” If we get an infinite duration request and there is an NSA policy specifying a maximum connection duration, why would we not reject the connection request with an appropriate service definition policy error providing the policy maximum? This would allow the requesting NSA to find an alternative route that could support an infinite duration, or at least adjust expectations in a subsequent request.

“It takes some time to process a request. Possible maximum time required to process a request and make resources ready for provisioning is called “guard time”.” I thought that “guard-time” also included the transport provisioning/activation overhead as well? This definition seems to only cover the reservation overhead, so a requesting NSA must utilize a different value for the time to provision.

“This system is designed to be compatible with systems based on 2PC.” I do not think this statement is 100% true given the example provided. When we modified DRAC to support the Phosphorus/Harmony interface model they also had a two phase commit (reserve/hold resources and commit resources) which also included an explicit activation operation as well. I always questioned the value of commit operation, as it didn’t save us anything in DRAC as the reserve costs the most. I can now see the value if we are trying to do a “start-now” operation but want to make sure an end-to-end path is available before provisioning connections.

Section 5.1.7 Tree and Chain Connection modes for inter-domain pathfinding

I found this section interesting in that even though I understand general routing, tree, and chain path finding I had to read the section three times and made a ton of notes before concluding that I did understand what was written. I think restructuring this section a bit would solve my problem.

This section would benefit from a reference network topology diagram and an example course grained inter-domain path computation somewhere before tree and chain path finding are introduced. Figures 17 and 18 can then reference NSA and nodes from the example diagram to show how a path through the network could be reserved.

The general description of chaining needs to be expanded to provide some additional details. I would have expected head-end path computation as the first step to determine a rough path through the network, and to guide the “next hop” to receive the request in the chain. Although a similar statement was made for tree based path finding it was not stated for chain based, although, it may have been implied in the statements around reservation.

I am also concerned with this statement: “Alternatively, if the local NSA does not have sufficient topology information or authorization credentials to identify and interact directly with all the downstream networks, the local NSA can simply choose a neighbor network as the next hop, and using the interconnect STP as the ingress point, forward a request to that next hop NSA for handling.” This statement implies to me that I do not need to do head-end path computation and I can just throw the request to any adjacent node and it would magically reach its destination. In a highly connected network this might be a feasible plan, but in other cases there could be a lot of dead end computations before a viable path it found.

Lastly, the statement “It is highly distributed, scales well and is robust” could be brought into question given the description of chaining in this section ;-)