NSI CS State Machine v2.0Oxford

Background

The current NSI CS v1.0SC state machine requires that **both a provision request message and the start time** trigger are needed to initiate the initial setup of a circuit (see Figure 1). A provision confirm message is not returned until the circuit has been successfully setup. In the current state machine, if the provision request message is sent well in advance (e.g. 1 week) of the start time, there can be a substantial amount of elapsed time before the provision confirm message is returned.

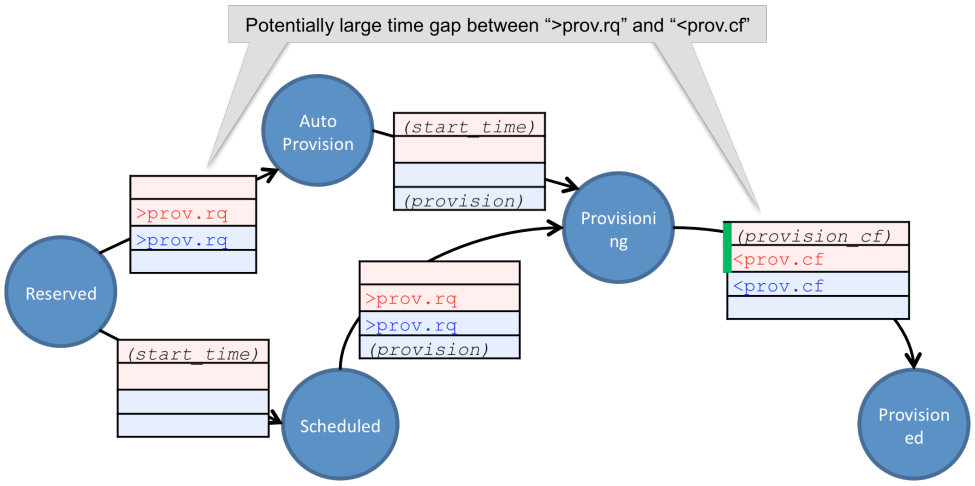

Figure 1. NSI CS v1.0SC State Transitions from "Reserved" to "Provisioned"

Oxford State Machine Proposals

During the OGF NSI workshop in Oxford (Mar 12-15, 2012) there were three core proposed changes; i. unsolicited messages from the PA to the RA as a form of notification, ii. the formalization of a Message Delivery Layer (MDL), and iii. the separation of the NSI CS v1.0SC state machine into two distinct state machines, one for the ultimate RA and Aggregator, and one for the ultimate provider.

Unsolicited Messages from PA to RA (Notify)

The use of unsolicited messages from the PA to the RA (or notification) was devised primarily to communicate to the RA a local event in the PA that resulted in a state transition in the state machine. An example of this is the "activate_ok.nt" and "activate_ng.nt" notify messages (see Figure 4 and 5) sent from the PA to the RA to indicate a success or failure of the circuit setup in the PA.

Message Delivery Layer (MDL)

Conceptually, MDL resides between the NSI protocol layer and the message transport layer and is responsible for:

i. The "determined" delivery of messages by using mechanisms such as timeout and re-tries.
ii. The aggregation of replies from child nodes to indicate if; i. all the children successfully received the message, or ii. if one or children failed to receive the message.

If the MDL is unable to confirm message receipt from all children, it returns a failure. This failure is considered fatal, and the NSA must initiate a state clean up.

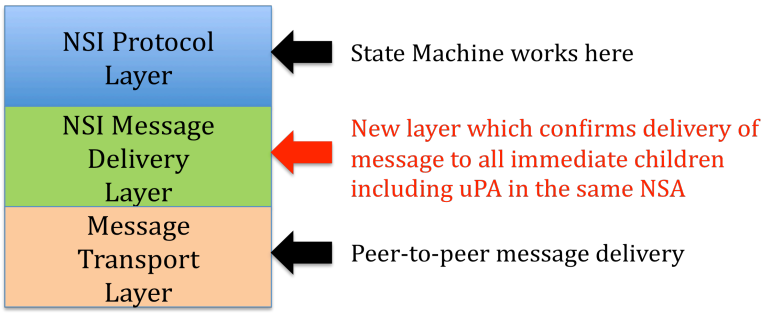

Figure 2. Conceptual layering in NSA

For consistency, the MDL conceptual separates the Aggregator and ultimate Provider functions within an NSA, as seen in the lower right NSA in Figure 3.

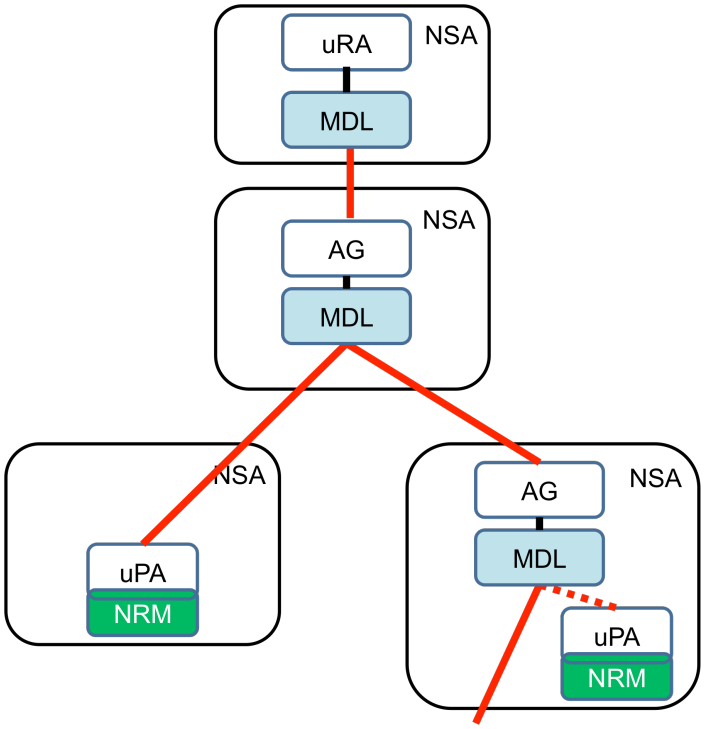

Figure 3. Functional placement of MDL

The following (see Figure 4) is an example workflow of the MDL.

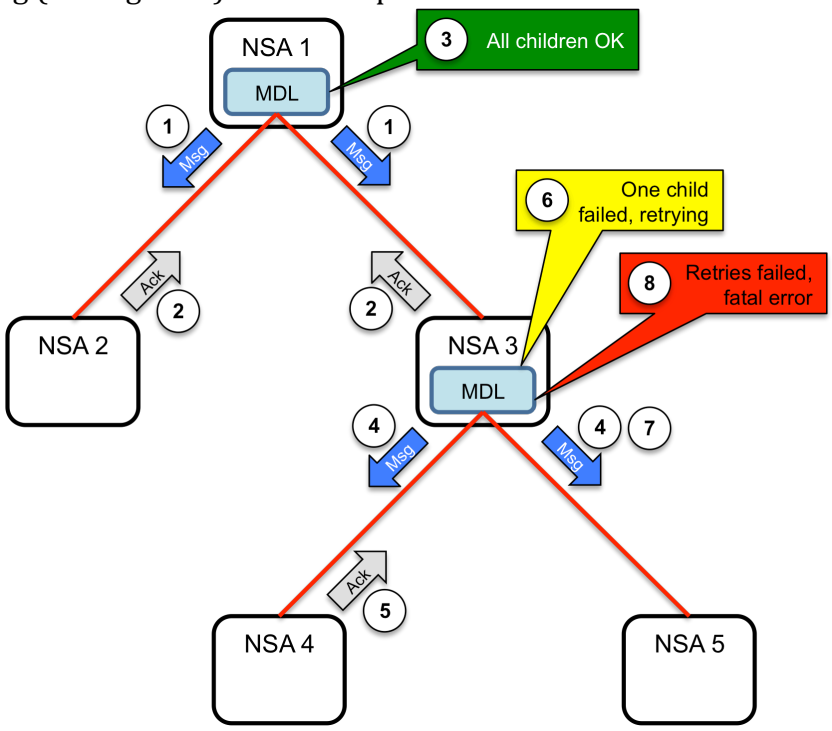

Figure 4. Example workflow of MDL

1. NSA 1 sends messages to NSA 2 and NSA 3 via the MDL.
2. NSA 2 and NSA 3 acknowledge receipt of the message.
3. The MDL in NSA 1 flags that both children (NSA 2 and NSA 3) received the message successfully.
4. NSA 3 sends messages to NSA 4 and NSA 5 via the MDL.
5. NSA 4 acknowledges receipt (NSA 5 does not).
6. The MDL in NSA 3 recognizes that NSA 5 did not acknowledge receipt of the message and retries sending the message.
7. NSA 3 resends the message to NSA 5 via the MDL (and may continue to do so for multiple times if it does not get an acknowledgement).
8. The MDL in NSA 3 flags that it did not get an acknowledgement from NSA 5 (after multiple attempts), gives up, and reports a fatal error.

Separation of uRA/Aggregator, and uPA State Machines (SM)

The motivation for separating the NSI CS v1.0SC state machine into two distinct state machines was done primarily for correctness. In particular, it was to address the issue of aggregator NSAs transitioning to the "Provisioned" state without knowing if the connection in its children NSAs were active. In addition, the notion of the control plane provisioning was decoupled from the data plane circuit setup by the introduction of the "Activating" and "Activated" states. In the NSI CS v1.0SC state machine, the "Provisioned" state indicated that the circuit was setup and ready for data to flow over it. In the v2.0Oxford state machine, the "Provisioned" state simply indicates that the provision confirm message (i.e. "prov.cf") has been received in acknowledgement of the provision request message (i.e. "prov.rq").

This represents the control plane workflow process. When the circuit is setup, a notification message (i.e. "activate_ok.nt") is sent from the PA to the RA to indicate and trigger a state change to the "Activated" state respectively. This represents the data plane workflow process. The table below (see Table 1.) is a summarization of the "Provision" and "Activated" states within the v1.0SC and v2.0Oxford state machines.

State	NSI CS v1.0SC	NSI CS v2.0Oxford
Provisioned	Control: The provision request message ("*prov.rq*") has been processed and acknowledged with a provision confirm message ("*prov.cf*") Data: **Circuit is ready for data movement**	Control: The provision request message ("*prov.rq*") has been processed and acknowledged with a provision confirm message ("*prov.cf*") Data: *No action taken*
Activated	Control: *Not applicable* Data: *Not applicable*	Control: An activate notification ("activate_ok.nt") was received Data: **Circuit is ready for data movement**

Table 1. Comparison of "Provisioned" state between v1.0SC and v2.0Oxford SM

Ultimate RA/Aggregator State Machine

The uRA/Aggregator v2.0Oxford state machine is represented below (see Figure 5). Several design decisions reflected in this state machine include:

- *Absence of "Activating" state*. The uRA and Aggregator do not manage any network resources and therefore are unaware when the activating process is initiated. Only when it receives an "activate_ok.nt" notify message does it know that the activation process has completed.
- *In the "Activated" state, when a provision request is received ("prov.rq"), it will return both a provision confirm ("prov.cf") as well as an activate notification ("activate_ok.nt").* Due to the decoupling of the control and data plane provision processes, both messages ("prov.fc, "activate_ok.nt") must be returned to indicate that the reservation is active and the circuit has been setup.
- *In any state, if a failure ("*.fl") or forced end ("fcd_end") is received, the state machine will transition to the "Terminated" state bypassing the "Terminating" state.* The "Terminating" state is to reflect normal clean up operations when a termination request ("term.rq") is received, and not part of failure scenarios or when a reservation has expired (i.e. reached it's end time).

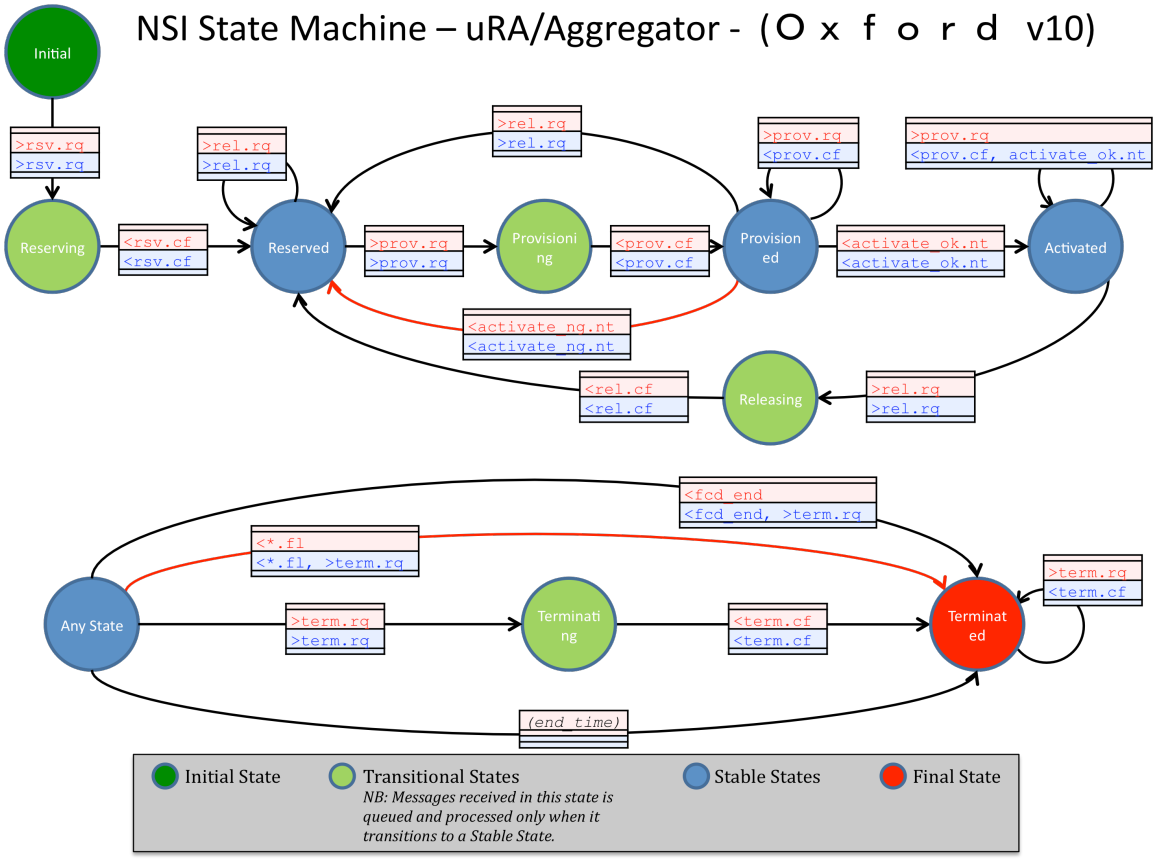

Figure 5. Ultimate RA and Aggregator NSI CS "Oxford" State Machine

Ultimate PA State Machine

The uPA v2.0Oxford state machine is represented below (see Figure 6). Several design decisions reflected in this state machine include:

- *In the "Auto Provisioning" state, a release request ("rel.rq") transitions the state machine back to the "Reserved" state*. This state transitions allows the user to stop a circuit from being activated even when it is already in the "Auto Provision" state.
- *In the "Activating" state, an activation failure transitions the state machine to the "Scheduled" state*. In the event of an activation failure, the reservation is not immediately canceled but goes to the "Scheduled" state if the time window is within the reserved period of the reservation. This is done to facilitate troubleshooting for subsequent re-provisioning. This is particularly useful for long lived reservations, where re-requesting a large number of network resources may be inconvenient.

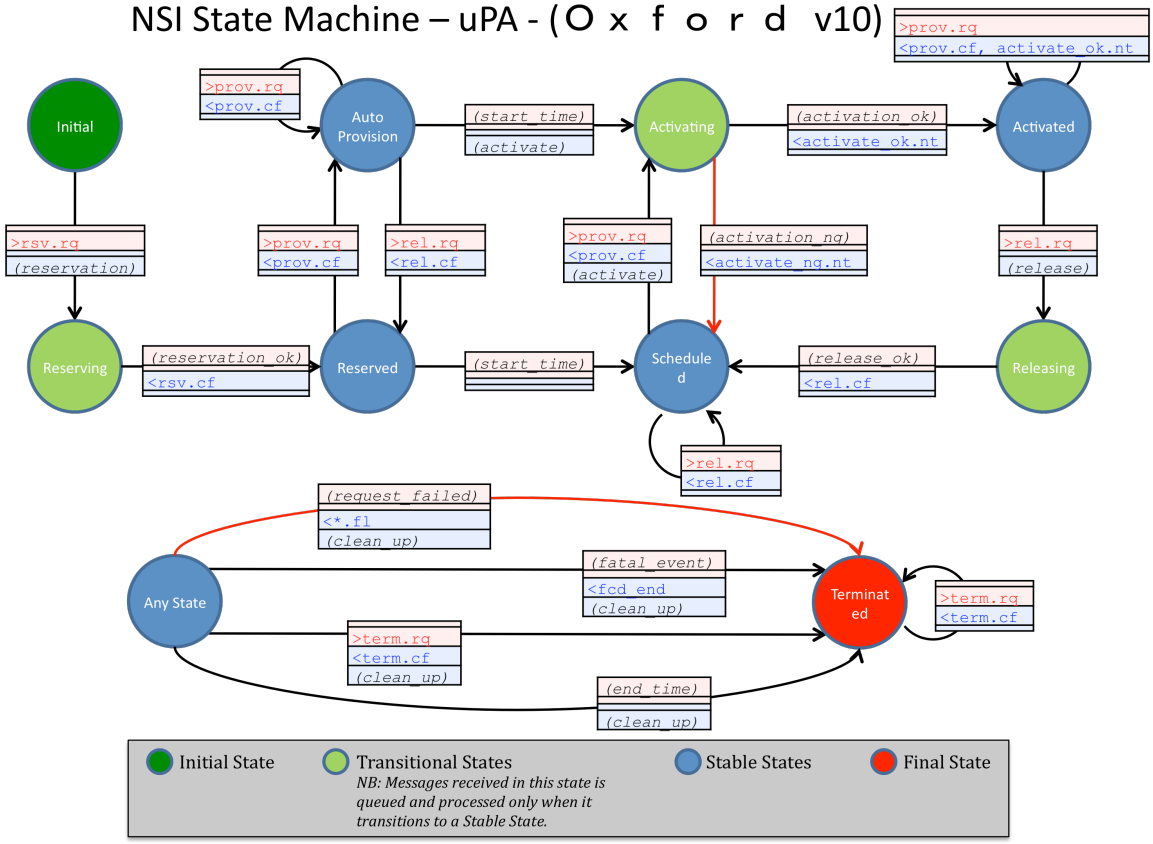

Figure 6. Ultimate Provider NSI CS “Oxford” State Machine