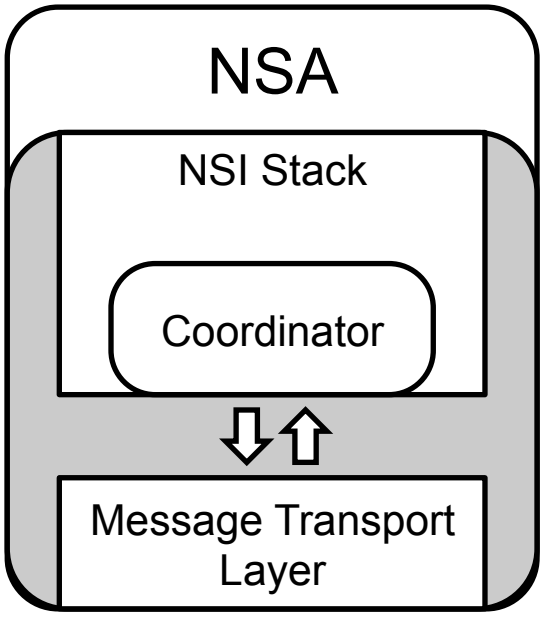
OpenGridForum

OPEN FORUM | OPEN STANDARDS

NSI CS Protocol State Machine Message Handling

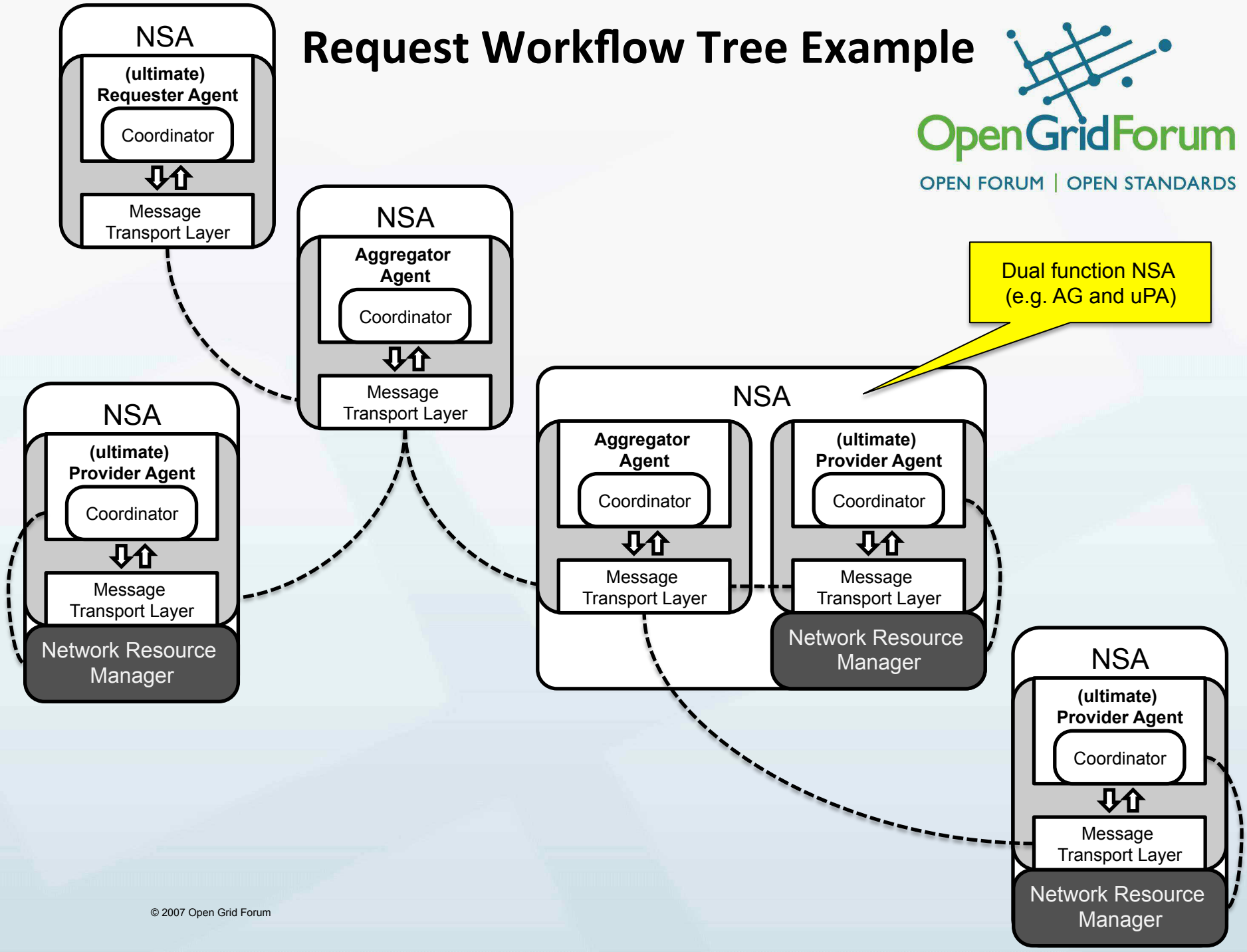
OGF 37

Coordinator and Message Transport Layer (MTL)

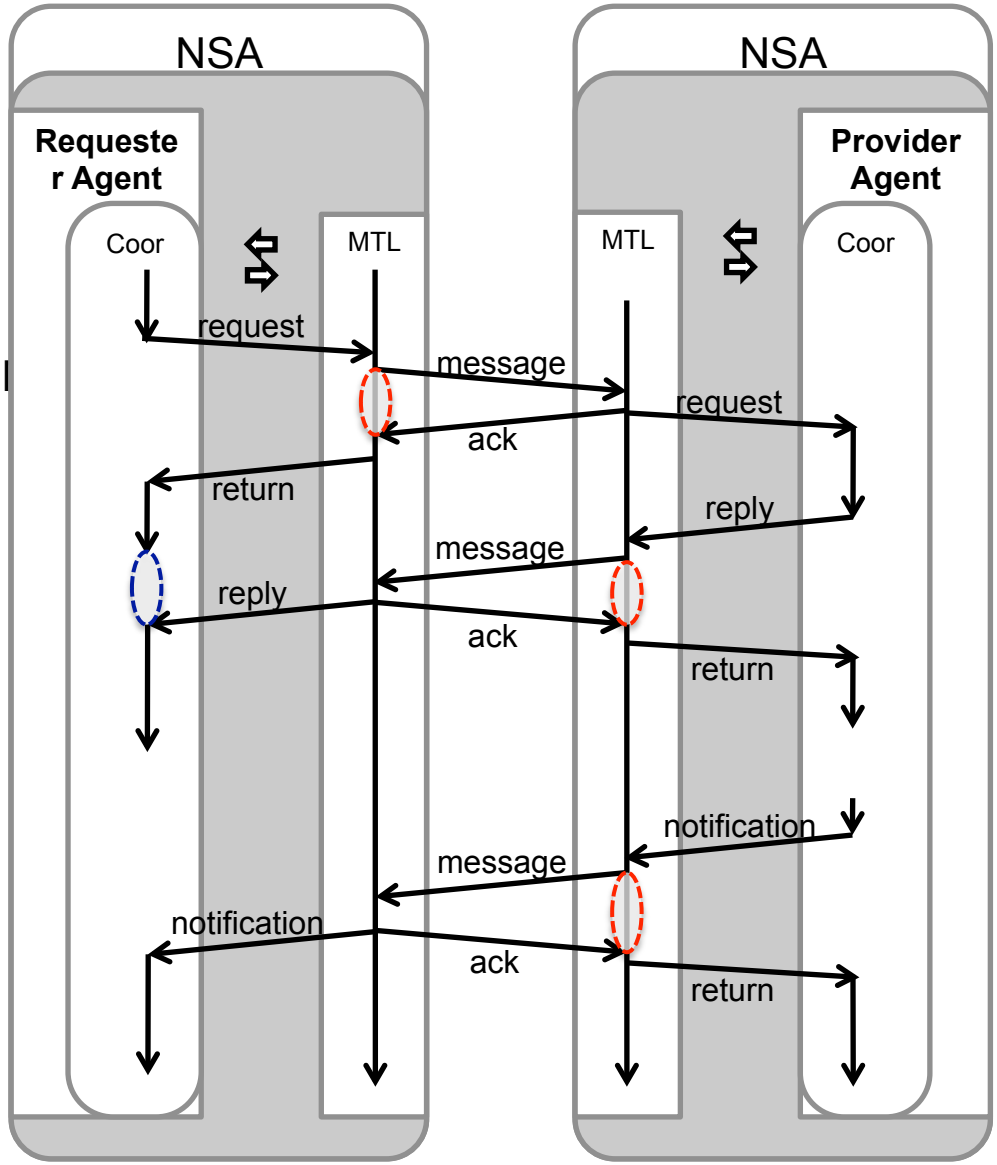


- Coor is a part of NSI stack, and uses MTL to send/receive messages
- Coor is primarily responsible for keeping track of messaging state, e.g.
  - Who was the message sent to
  - Was the message received (i.e. ack'ed or MTL timeout)
  - Who has not replied to the message (e.g. *.cf, *.fl, etc)
- MTL is primarily responsible for sending and receiving messages, and notifying Coor if the message was received, or if a (MTL) timeout occurs
- MTL interface (to Coor) has 2 simple operations:
  - Send: waits for ack to be returned by destination MTL, or timeout happens. Timeout value is implementation dependent. NB: The MTL may be implemented to retry sending messages, but this is opaque to the Coor
  - Receive: a thread in Coor is invoked when a message is received

Request Workflow Tree Example



Message ack, reply and timeouts



: MTL timeout may happen

: Coor timeout may happen

- Ack is sent by MTL for each message
  - If ack is not returned in a certain period of time, MTL timeout occurs
- Reply is sent by Coor (via MTL) and is either confirm, fail or not_applicable
  - Coor can timeout if expected reply is not received from a child

Timeouts

- Message transport layer (MTL) timeout
  - Underlying MTL (http/tcp) initiates a MTL timeout
  - Happens when an ack is not returned for a message.
- Coordinator timeout
  - Coor can timeout if a reply message is not returned in a certain period of time
- Coor notifies both MTL and Coor timeouts to the parent RA
- When a MTL/Coor timeout is notified, uRA can either retry or terminate the connection.
  - Retry is requested by NSI_messageRetry.rq, which has the original request message's id (correlation id) as a parameter
  - Coor keeps not-yet-replied requests in a table, so that it can re-send the request.

Notifications: Activation related

- There are no activateComplete.nt nor deactivateComplete.nt
- A general error message is used to notify following events. Those error are sent up the tree to uRA immediately
  - activateFailed: Activation failed at the time when uPA should activate its data plane
  - deactivateFailed: Deactivation failed at the time when uPA should deactivate its data plane
  - dataplaneError: Data plane is deactivate when deactivation is not expected. The error is recoverable.
  - forcedEnd: Something unrecoverable is happened in uPA/NRM

Notifications: modify timeout and MTL failure

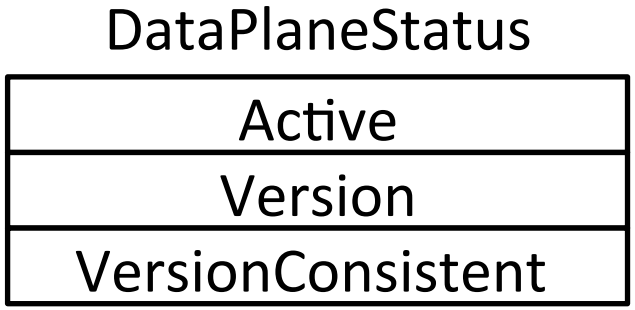
- NSI_modifyTimeout.nt
- NSI_genericEvent.nt
  - Message delivery failure will be notified by this message (to be defined)
- When a MTL/Coor timeout is notified, uRA can either retry or terminate the connection.
  - Retry is requested by NSI_messageRetry.rq, which has the original request message's id (correlation id) as a parameter

Data plane activation

- Data plane should be activated if the PSM is in “Provisioned” state AND start_time < current_time < end_time
- Activation is done at the timing of following events (if the above condition is met), using the latest reservation information
  - PSM transits to “Provisioned”
  - At the start_time
  - Reservation is updated (by commit of modify)
  - Data plane is recovered from an error
- Data plane activation/deactivation are notified by DataPlaneStateChange.nt notification messages.
- Errors are notified by a generic error message

DataPlaneStateChage.nt (1)

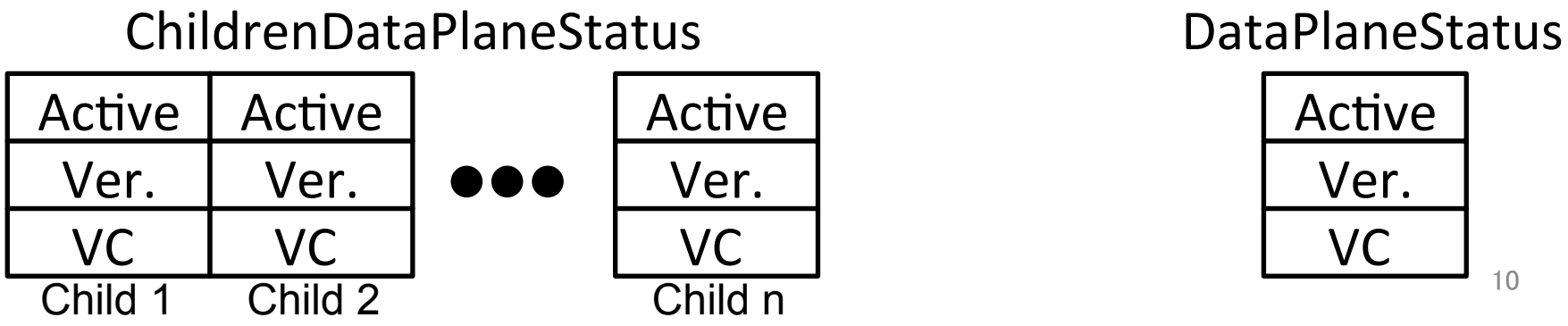
- PA and aggregator has DataPlaneStatus information
  - (Boolean) Active: True if data plane is active. For an aggregator, this flag is true when data plane is activated in all participating children
  - (Int) Version: For a uPA, current (latest) reservation version number. For an aggregator, the largest version number of the participating children. This field is valid when Active is true.
  - (Boolean) VersionConsistent: Always true for uPA. For an aggregator, If version numbers of all children are the same, This flag is true. This field is valid when Active is true.
- When a valid filed of DataPlaneStatus is changed, DataPlaneStatusChange.nt is sent up.



DataPlaneStateChange.nt(2)

- An aggregator keeps an array of statuses of its children, ChildrenDataPlaneStatus[1..n]
- Aggregator's DataPlaneStatus is determined by the following rule

```
if all of ChildrenDataPlaneStatus[1..n].Active are true
        then
        {
        DataPlaneStatus.Active = true
        DataPlaneStatus.Version =
            maximum_of(ChildrenDataPlaneStatus[1..n].Version)
        If all ChildrenDataPlaneStatus[1..n].Version are the same, and all
            of ChildrenDataPlaneStatus[1..n].VersionCosistent are true
                then DataPlaneStatus.VersionConsistent = true
                else DataPlaneStatus.VersionConsistent = false
        }
```



Information tracked by Coordinator