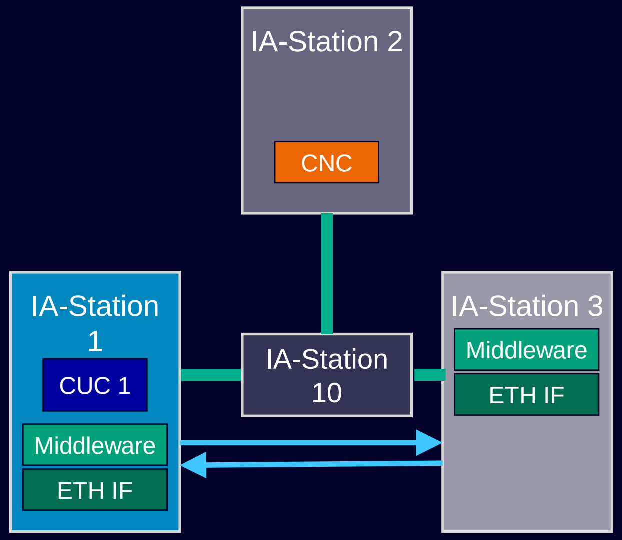
Stream Request and Response Dynamics for IA

Rodrigo Ferreira Coelho and Günter Steindl [Siemens AG]

SIEMENS

Expected behavior: establishing stream(s)



1. IA1 ON, IA2 ON, IA10 ON
2. IA3 ON
    1. CNC discovers IA3
    2. IA1 discovers IA3
    3. IA1 requests streams a, b via CUC1 (UNI)
    4. CNC computes streams a, b
    5. CNC signals CUC1 end of computation of a, b
        1. CNC provides StreamID and MAC-DA
    6. CNC configures network, talker and listeners with streams a, b
    7. CNC signals CUC1 end of configuration of a, b
    8. IA1 application exchange data via streams a, b

Bridge config objects

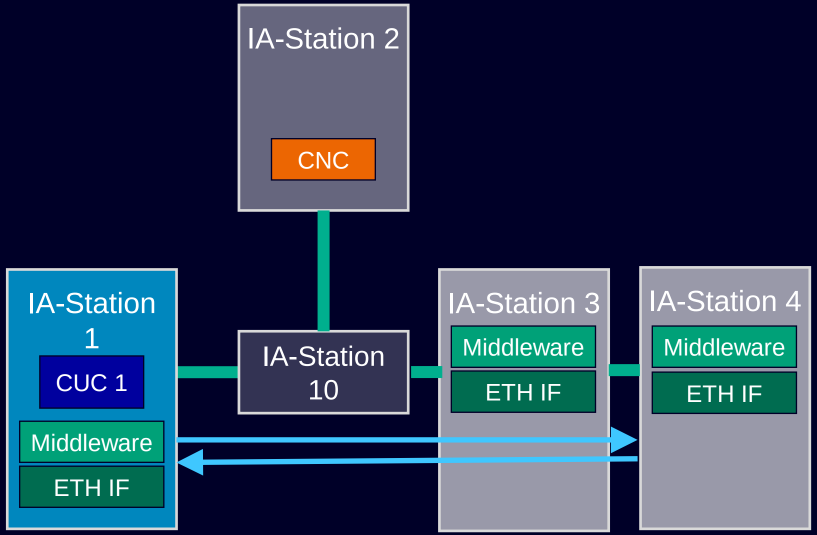
Stream objects (new YANG module)

- StreamID, DA-MAC, Time-Aware Offset, VID, PCP, …

Example 1: One CUC requests streams to one pair of Talker, Listener

SIEMENS

Expected behavior: establishing stream(s)

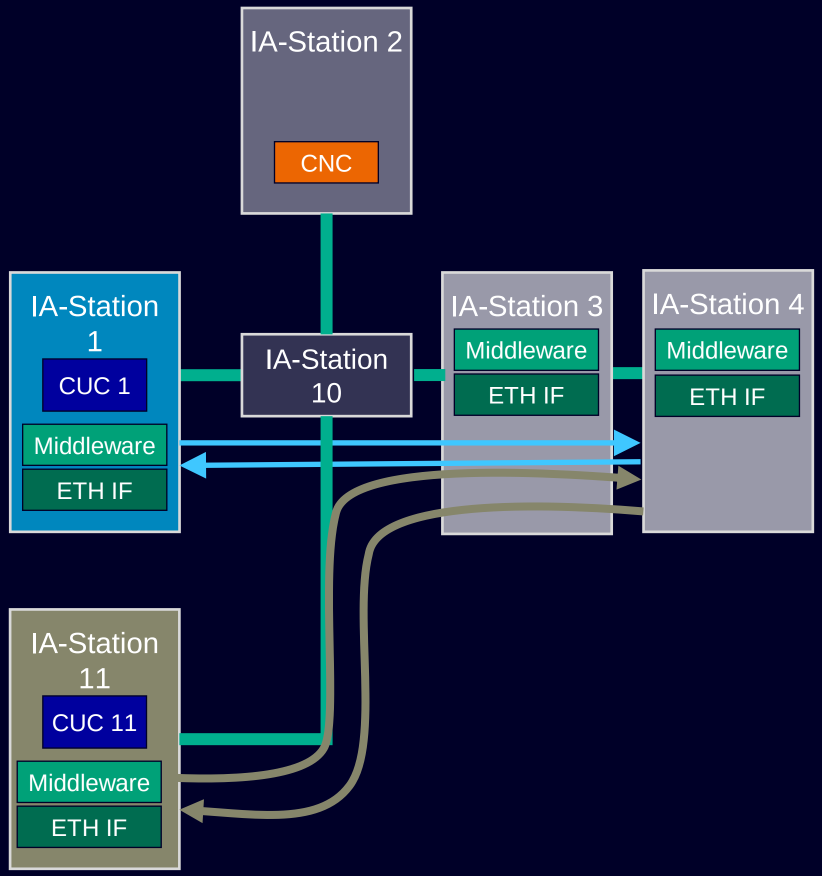


1. IA1 ON, IA2 ON, IA10 ON
2. IA3 ON
   1. *… as before*
3. IA4 ON
   1. CNC discovers IA4
   2. IA1 discovers IA4
   3. IA1 requests streams c, d via CUC1 (UNI)
   4. CNC computes streams c, d
   5. CNC signals CUC1 end of computation of c, d
      1. CNC provides StreamID and MAC-DA
   6. CNC configures network, talker and listeners with streams c, d
      1. Eventually reconfigures a, b
   7. CNC signals CUC1 end of configuration of c, d
   8. IA1 application exchange data via streams c, d

Example 2: One CUC requests streams to multiple pairs of Talker, Listener

SIEMENS

Expected behavior: establishing stream(s)



Example 3: Multiple CUCs request streams to multiple pairs of Talker, Listener

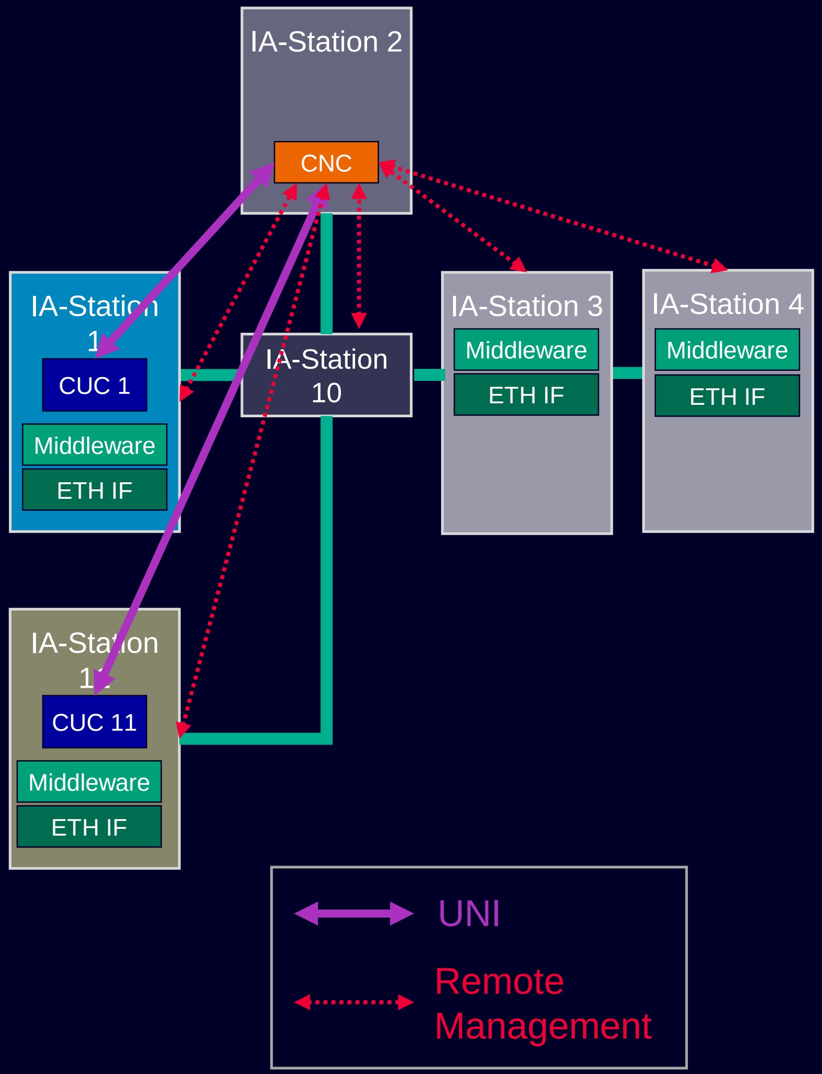
1. IA1 ON, IA2 ON, IA10 ON
2. IA3 ON
    1. *… as before*
3. IA4 ON
    1. CNC discovers IA4
    2. IA1 discovers IA4
    3. IA1 requests streams c, d via CUC1 (UNI)
4. IA11 ON
    1. CNC discovers IA11
    2. IA11 discovers IA4
    3. IA11 requests streams e, f via CUC11 (UNI)
    4. CNC signals CUC1 end of computation of c, d
        1. CNC provides StreamID and MAC-DA
    5. CNC computes streams e, f
    6. CNC signals CUC1 end of computation of e, f
        1. CNC provides StreamID and MAC-DA
    7. CNC configures network, talker and listeners with streams e, f
        1. Eventually reconfigures a, b, c, d
    8. CNC signals CUC1 end of configuration of e, f
    9. IA11 application exchange data via streams e, f

(concurrently to 5.)

1. CNC configures network, talker and listeners with streams c, d
    1. Eventually reconfigures a, b
2. CNC signals CUC1 end of configuration of c, d
3. IA1 application exchange data via streams c, d

SIEMENS

Expected behavior: establishing stream(s)



1. IA1 ON, IA2 ON, IA10 ON
2. IA3 ON
   1. *… as before*
3. IA4 ON
   1. CNC discovers IA4
   2. IA1 discovers IA4
   3. IA1 requests streams c, d via CUC1 (UNI)
4. IA11 ON
   1. CNC discovers IA11
   2. IA11 discovers IA4
   3. IA11 requests streams e, f via CUC11 (UNI)
   4. CNC signals CUC1 end of computation of c, d
      1. CNC provides StreamID and MAC-DA
   5. CNC computes streams e, f
   6. CNC signals CUC1 end of computation of e, f
      1. CNC provides StreamID and MAC-DA
   7. CNC configures network, talker and listeners with streams e, f
      1. Eventually reconfigures a, b, c, d
   8. CNC signals CUC1 end of configuration of e, f
   9. IA11 application exchange data via streams e, f

(concurrently to 5.)

1. CNC configures network, talker and listeners with streams c, d
   1. Eventually reconfigures a, b
2. CNC signals CUC1 end of configuration of c, d
3. IA1 application exchange data via streams c, d

SIEMENS

Expected behavior: establishing stream(s)

Conclusions

- Stream computation occurs sequentially (among requests of all CUCs)
- Stream configuration occurs sequentially in the same order as stream computation
- Configuration (resulted from stream requests) may occur concurrently to computation of other stream requests

1. IA1 ON, IA2 ON, IA10 ON
2. IA3 ON
   1. *… as before*
3. IA4 ON
   1. CNC discovers IA4
   2. IA1 discovers IA4
   3. IA1 requests streams c, d via CUC1 (UNI)
4. IA11 ON
   1. CNC discovers IA11
   2. IA11 discovers IA4
   3. IA11 requests streams e, f via CUC11 (UNI)
   4. CNC signals CUC1 end of computation of c, d
      1. CNC provides StreamID and MAC-DA
   5. CNC computes streams e, f
   6. CNC signals CUC1 end of computation of e, f
      1. CNC provides StreamID and MAC-DA
   7. CNC configures network, talker and listeners with streams e, f
      1. Eventually reconfigures a, b, c, d
   8. CNC signals CUC1 end of configuration of e, f
   9. IA11 application exchange data via streams e, f

(concurrently to 5.)

1. CNC configures network, talker and listeners with streams c, d
   1. Eventually reconfigures a, b
2. CNC signals CUC1 end of configuration of c, d
3. IA1 application exchange data via streams c, d

Example 3: Multiple CUCs request streams to multiple pairs of Talker, Listener

SIEMENS

Expectation from 802.1Qdj

SIEMENS

Using actions to request streams (instead of direct access to YANG module)

4.5. Pipelining

NETCONF <rpc> requests MUST be processed serially by the managed device. Additional <rpc> requests MAY be sent before previous ones have been completed. The managed device MUST send responses only in the order the requests were received.

Scenario: multiple CUCs requesting streams

- If actions/ RPCs are used to add streams
  - Requests are enqueued and sequentially processed @NETCONF server
  - FIFO order of process ensured
  - Needed: describing action input
    - Approx. 1 page of parameters (in Qdj)

- If no action/ RPC used to write into the YANG module
  - locking datastore is required to ensure consistency
  - response when trying to lock an already locked datastore is <rpc-error>
    - processing order of writing is unknown
      - CUCs must try to acquire the lock again
      - undefined delay, eventual starvation

SIEMENS

Read stream data from CUC

1.1 Terminology

state data: The additional data on a system that is not configuration data such as read-only status information and collected statistics.

5.1 Configuration Datastores

… The configuration datastore does not include state data or executive commands.

5.3. The Operational State Datastore (<operational>)

The operational state datastore (<operational>) is a read-only datastore that consists of all "config true" and "config false" nodes defined in the datastore's schema.

Define stream request/ response data as read-only/ config-false (state data)

- Written via actions (see previous slide)
- CUCs read stream data from operational datastore (<get>)
  - no locking needed

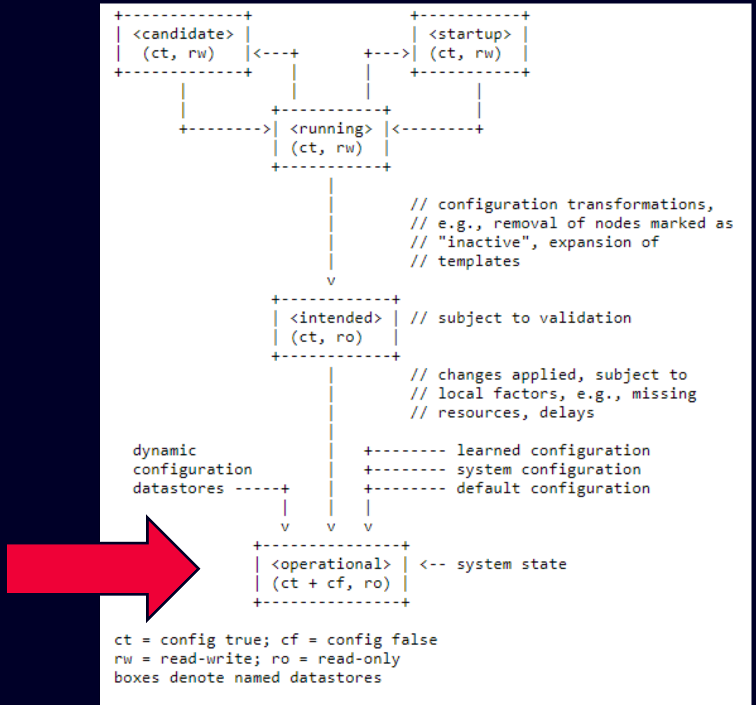


Figure 2

Further questions?

SIEMENS

Contact

Dr. Rodrigo Ferreira Coelho
System Architect
DI FA CTR ICO ARC
Siemenspromenade 1
91058 Erlangen
Deutschland

Phone: +49 9131 17-45546

E-mail: rodrigo.ferreira_coelho@siemens.com