On CNC/CUC

Contribution to IEEE 802.1Qdj

Juergen Wohlmuth, Astrit Ademaj

TTTech Industrial Automation AG Internal

January 22, 2021

In order to consider a broad range of different use-cases that should be supported, we need to:

- Generalize the given high-level concept
- Clearly describes the roles/responsibilities of CNC and CUC function
- Provide a concise separation of interfaces
- Provide a set of granularities (all based on the common understanding of *talker, listener,* and *stream*) and highlight their validity based on existing or foreseeable use-cases

Agenda

TTTech industrial

Contribution to 802.1Qdj:

1. Additional definition of responsibilities for CUC and CNC
2. Changes to the proposed definitions (in contribution dj-kehrer-P8021Qdj-d0-0-update-0520-v01.pdf)

- **CUC definitions and responsibilities**
- **CNC definitions and responsibilities**
- **CNC – CUC interface**

topics

Contribution

- **Add topics on the broad usability** of CUC-allocated functions in various use-cases, e.g.

1. Fully offline
   – Configuration generated offline, and distributed during system start-up using the CUC interface to talkers and listeners

2. Fully dynamic/online
   – Configuration generated online upon the stream configuration requests from talkers/listeners and distributed using the CUC interface to talkers and listeners

3. Mixed, incl. handover from (highly sophisticated) offline to dynamic/online handling
   – some configuration generated offline and distributed during system start-up

     - e.g., verified machine-setup

   – additional configuration is generated online

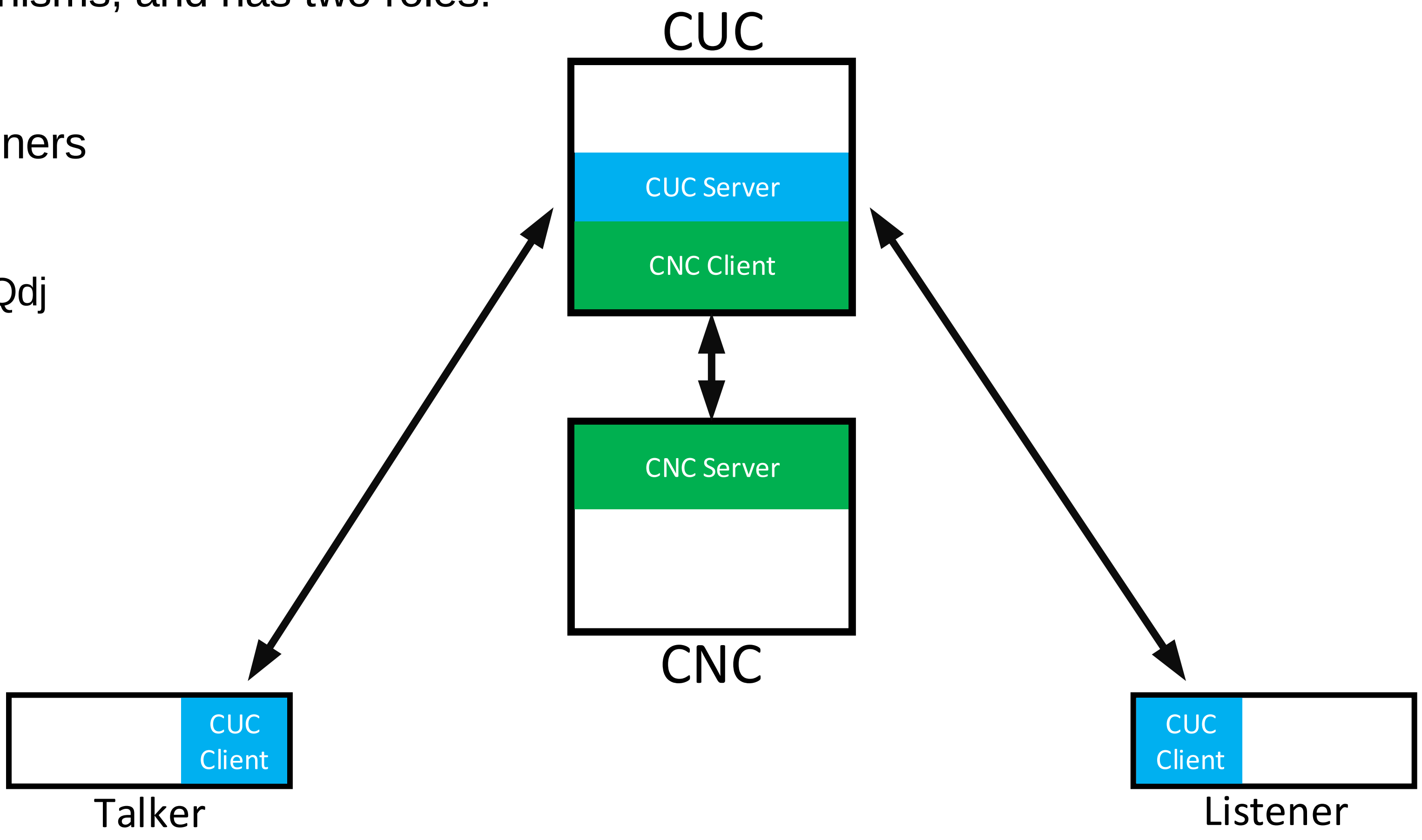
     - ad hoc extension (e.g., diagnostic equipment)

CUC Definition (2)

Proposed contribution

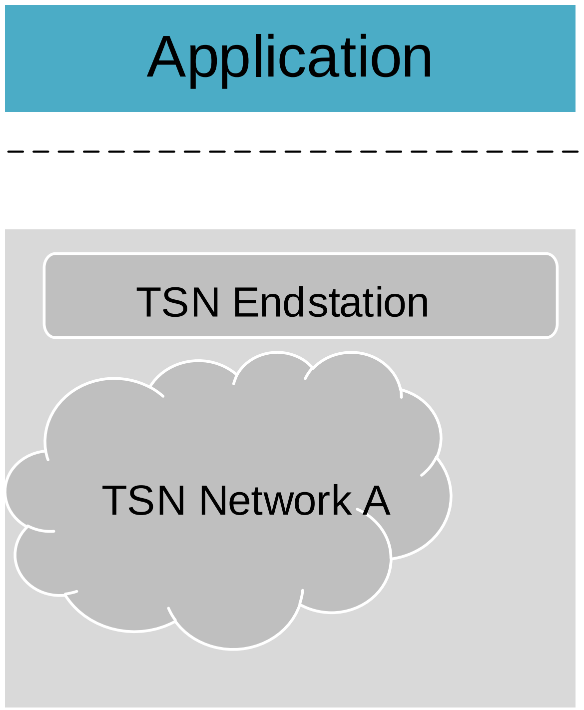
CUC is part of two client-server mechanisms, and has two roles:

1. **CUC is a server** to talkers and listeners
2. **CUC is a client** to the CNC
   - CUC - CNC interface is a subject of Qdj



StreamName (informative)

- StreamName is essential for the establishing the connections between talker and listeners in the ad-hoc operation



StreamName, LatencyReq,...

- StreamName is application specific (and configured by the application configuration mechanism) Identifier for a stream

StreamID, DMAC, VID,... MaxLatency,...

- StreamID is generated by the network

- **StreamName is used at the CUC to link the requests from a talker and listeners related to a specific stream**
- **CUC provides a StreamID to the CNC (CNC works with the StreamID)**
- It must be ensured that the StreamNames are unique within one TSN domain (assuming each domain has a separate CUC)
- Guidance: StreamName may be selected as a combination of the TSN DomainID and some application unique identifier within the TSN Domain

CUC Definition

TTTech industrial

Contribution at the IEEE 802.1 Interim Meeting, May 18-22

https://www.ieee802.org/1/files/public/docs2020/dj-kehrer-P8021Qdj-d0-0-update-0520-v01.pdf

- Suggested enhanced definition:
  - **3.x Centralized User Configuration (CUC):** A centralized entity that discovers end stations, retrieves end station capabilities and user requirements, and configures TSN features in end stations. It is a logical entity that can be located in any device of a network (e.g. a bridge, end-station, engineering tool, or network management system). The CUC is responsible for the following services:
    a. Collecting application level QoS requirements (e.g. application cycle time) for TSN streams from talkers and listeners.
    b. Translating the stream requirements from talkers and listeners to merged stream requirements.
    c. Communicating the merged stream requirements to the CNC.
    d. Retrieving the merged end-station communication-configuration from the CNC.
    e. Distributing the end-station communication-configuration to talkers and listeners.

    The protocols that the CUC uses for communication with end stations can either be specific to the user application (not specified in this standard) or a protocol specified by IEEE 802.1. A CUC exchanges information with a CNC in order to configure TSN features on behalf of its end stations. It communicates with the CNC through the Configuration-UNI defined in this standard.

CUC Definition

Contribution at the IEEE 802.1 Interim Meeting, May 18-22

https://www.ieee802.org/1/files/public/docs2020/dj-kehrer-P8021Qdj-d0-0-update-0520-v01.pdf

Translating the stream requirements from talkers and listeners to “*merged* stream requirements.”

Introduces high complexity to CUC

- Lessons learned from PTCC (OPC TSN) implementation
- **Merging** implies that the CUC must maintain the state of each stream (combine talker and at least one listener request)
- The same state needs to be maintained at the CNC level as well.
  - Increased complexity at the CUC function
  - Not needed. In many use cases the CUC and the CNC function are executed in the same host.
  - „merging“ only applies to talker and the first listener
  - every subsequently listener would have to join

Possible workflow in a „Merged“ “ mode

Legjend:

- **T1- Talker 1; L1: Listener 1; L2: Listener2**
- **SID1 – Network-Level Stream Identifier for Stream 1**

```
------------------------(CUC working in „merged“ mode)----------------
T1 (StreamName) -->  CUC
L1 (StreamName) -->  CUC
                 CUC {T1(SID1), L1(SID1} -->  CNC
                                         CNC  --> CUC {T1, L1}
                                                   CUC  --> T1(...)
                                                   CUC  --> L1(...)
```

- CUC translates the StreamName to StreamIDentifier (SID)

- CUC waits for a Talker and at least one Listener request before sending the first request to CNC

Possible workflow in a „Merged" " mode

Legjend:
- **T1- Talker 1; L1: Listener 1; L2: Listener2**
- **SID1 – Network-Level Stream Identifier for Stream 1**

```
-----------------------(CUC working in „merged" mode)---------------
T1 (StreamName) -->  CUC
L1 (StreamName) -->  CUC
                 CUC {T1(SID1), L1(SID1} -->  CNC
                                       CNC  --> CUC {T1, L1}
                                                 CUC  --> T1(...)
                                                 CUC  --> L1(...)
L2 (ASID1)-->  CUC
              CUC  {T1(SID1), L1(SID1), L2(SID1)} --> CNC
                                       CNC  --> CUC {T1, L1, L2}
                                                 CUC    --> T1(...)
                                                 CUC    --> L1(...)
                                                 CUC    --> L2(...)
```

- CUC translates the StreamName to StreamIDentifier (SID)

- CUC waits for a Talker and at least one Listener request before sending the first request to CNC (need to maintain the state)

- CUC need to store all the previous requests internally (lets denote it as "CUC-StreamObject")
-
- CUC sends the following requests with the complete stream information to CNC with every additional listener request

- CNC needs to compare the received "CUC-StreamObject" with the "CUC-StreamObject" from the previous request to check for the differences

Possible workflow in a „non-merged“ mode

Legjend:
- **T1- Talker 1; L1: Listener 1; L2: Listener2**
- **SID1 – Network-Level Stream Identifier for Stream 1**

```
----------------------------------(CUC working in „non merged“ mode)-----------------------------------
L1 (StreamName) -->  CUC
                 CUC  (L1(SID1)) -->  CNC

T1 (StreamName)  -->  CUC
                 CUC  (T1(SID1)) -->  CNC
                           CNC    --> CUC {T1, L1}
                                        CUC    --> T1(...)
                                        CUC    --> L1(...)

L2 (StreamName)  -->  CUC
         CUC  (L2(SID1)) -->CNC
                           CNC  --> CUC (L2)
                                        CUC    --> L2(...)  (CNC is working in incremental mode)
```

- CUC translates the StreamName to StreamIDentifier (SID)
- Forwards each Talker/Listener client request the CNC individually (no need to maintain the state)

- CUC sends only individual requests to the complete stream information with every additional listener request

- CNC has now the configuration data to generate the stream configuration

- CUC sends only individual requests to the complete stream information with every additional listener request

„Merged“ vs „non-merged“ mode

Legjend:
- **T1- Talker 1; L1: Listener 1; L2: Listener2**
- **SID1 – Network-Level Stream Identifier for Stream 1**

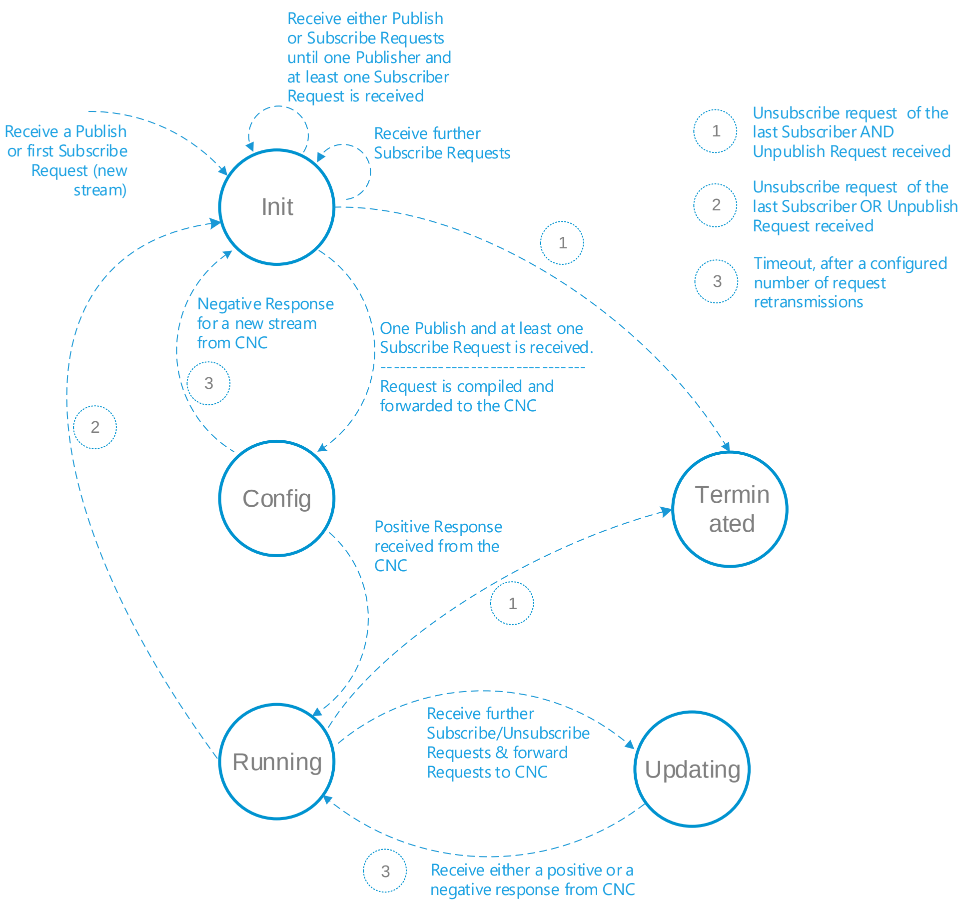
```
------------------------(CUC working in „merged“ mode)---------------
T1 (StreamName) -->  CUC
L1 (StreamName) -->  CUC
                  CUC {T1(SID1), L1(SID1} -->  CNC
                                 CNC  --> CUC {T1, L1}
                                            CUC  --> T1(...)
                                            CUC  --> L1(...)
L2 (StreamName)  -->  CUC
              CUC  {T1(SID1), L1(SID1), L2(SID1)} --> CNC
                                 CNC  --> CUC {T1, L1, L2}
                                            CUC    --> T1(...)
                                            CUC    --> L1(...)
                                            CUC    --> L2(...)
```

- CUC sends the complete stream information to CNC with every additional listener request

```
-----------------------------------(CUC working in „non merged“ mode)------------------------------------
L1 (StreamName) -->  CUC
                  CUC  (L1(SID1)) -->  CNC
T1 (StreamName)  -->  CUC
                  CUC  (T1(SID1)) -->  CNC
                            CNC   --> CUC {T1, L1}
                                         CUC   --> T1(...)
                                         CUC   --> L1(...)
L2 (StreamName)  -->  CUC
         CUC  (L2(SID1)) -->CNC
                            CNC  --> CUC (L2)
                                         CUC   --> L2(...)  (CNC is working in incremental mode)
```

- CUC sends only individual requests to the complete stream information with every additional listener request

State machine from PTCC *
PTCC = PubSub TSN Centralized Configuration



During the implementation we found that we even miss some state transitions, increased complexity

*PTCC = PubSub TSN Centralized Configuration, CUC-Endstation interface draft specification defined by the OPC TSN working group (not mandatory).

Merged vs Non-Merged Mode

Merged Mode

- CUC sends all request talker/listener requests parameters to CNC with every single request
- CUC needs to maintain the state
  - *increased complexity of CUC implementation*
- CNC need to maintain the same state as the CUC
- CNC need to compare the requests every time
  - *increased complexity of CNC implementation*

No significant advantages of this approach

Non-Merged Mode

- CUC sends individual request talker/listener requests parameters to CNC with every single request
- CUC can be implemented with a minimal or even state-less

Significant decrease of the CUC implementation complexity

CUC Responsibilities

- CUC translates protocol specifics to **the** CNC-terminology, i.e.,
  - it provides **following** services to its clients
    - TSN stream configuration functions to its clients
    - CUC discovery service to its clients
  - is a client to the CNC

- Some discussion in the other consortia (AVNU, OPC TSN WG) related to CUC topics

- Add a remark that:
  - *CUC is responsible for handling the TSN related stream configuration between the talker/listeners and the CNC.*
  - *CUC is not responsible for handling*
    - *Application Configuration Parameters*
    - *Middleware Configuration Parameters*
    - *Network Driver/Stack Configuration Parameters (IP address, etc,.).*

CUC Definition

Green text – original contribution
Red text – changes

TTTech industrial

Proposed definition of CUC:

3.x Centralized User Configuration (CUC): A centralized entity that

- ~~discovers end stations,~~
- receives TSN end system capabilities
- receives user requirements with respect to TSN configuration
- provides TSN configuration input for end systems
- It is a logical entity that can be located in any device of a network (e.g. a bridge, end-station, engineering tool, or network management system).

- Suggested enhanced definition:
  - **3.x Centralized User Configuration (CUC):** A centralized entity that discovers end stations, retrieves end station capabilities and user requirements, and configures TSN features in end stations. It is a logical entity that can be located in any device of a network (e.g. a bridge, end-station, engineering tool, or network management system). The CUC is responsible for the following services:
    a. Collecting application level QoS requirements (e.g. application cycle time) for TSN streams from talkers and listeners.
    b. Translating the stream requirements from talkers and listeners to merged stream requirements.
    c. Communicating the merged stream requirements to the CNC.
    d. Retrieving the merged end-station communication-configuration from the CNC.
    e. Distributing the end-station communication-configuration to talkers and listeners.

    The protocols that the CUC uses for communication with end stations can either be specific to the user application (not specified in this standard) or a protocol specified by IEEE 802.1. A CUC exchanges information with a CNC in order to configure TSN features on behalf of its end stations. It communicates with the CNC through the Configuration-UNI defined in this standard.

Proposed definition of CUC:

The CUC is responsible for the following services:

a. provide discovery service to end systems
b. provide a stream name service, in order to relate the talker and stream requests
c. receives QoS requirements (e.g. application cycle time) for TSN streams from talkers and listeners
d. Translating and forwarding the stream requirements from talkers and listeners to the CNC interface
e. receives the ~~merged~~ end-system stream-configuration input from the CNC for talkers and listeners
f. Distributing the end systems stream-configuration input to talkers and listeners (if needed)

The protocols that the CUC uses for communication with end systems can either be specific to the user application (not specified in this standard) or a protocol specified by IEEE 802.1. A CUC exchanges information with a CNC in order to configure TSN features on behalf of its end stations. It communicates with the CNC through the Configuration-UNI defined in this standard.

topics

Contributions

Actual Goal:

- **Broad usability** of CNC-allocated functions in various use-cases, e.g.
  - Fully offline
  - Fully dynamic/online
  - Mixed, incl. handover from highly sophisticated (offline) to dynamic (online) handling

- CNC-allocated functions shall be generic to be usable by **various** CUCs, e.g.,
  - OPC UA over TSN

Responsibilities - Topology

- CNC is responsible for configuring/controlling forwarding plane and shall be aware of the physical topology
- CNC must be aware to which bridges/bridge ports and application plane (i.e., and end station) is directly attached to - „connection points“.
- It shall not be in scope
  - how the CNC gets the topology information on the forwarding plane (bridge - bridge)
  - how the CNC gets the information on the „connection points“ of the application plane (bridge – end stations)
  - which additional tasks with respect to topology discovery the CNC shall perform
- Assumptions on the operation of a CUC or any higher layer must not be taken (*layering*).
- CUC shall be aware of the CNC operational mode

"Connection points" mark the boundary between application plane (end system, user application) and the control plane (CNC, CUC).

Multiple ways are possible, e.g., based on LLDP or preconfigured topology in CNC

For example, network verification or monitoring of streams (the latter requires Stream Identification).

Two types of requests in the CNC-CUC UNI

1. Client-based stream interface

- CNC receives talker and listeners requests (join/leave) independently.
  - Ad-hoc (dynamic) configuration

For highly dynamic scenarios that require independent development and deployment of talkers and listeners

2. Stream-set interface

- A set of streams: (talkers and its corresponding listeners) are provided to the CNC as an “atomic-set”
  - e.g., CNC receives the complete request on all scheduled streams (all talker and all listener requests)
  - e.g., CNC receives complete a single stream request, talker and all its listeners

For offline use where highly complex network requirements are necessary and thus planning algorithm needs „complete“ information.

For dynamic development and deployment of distributed applications (comprising talker and all listeners)

CNC shall expose its capabilities to indicate which planning paradigm(s) it supports

One or both of the above paradigms are possible

Two types of requests in the CNC-CUC UNI (2)

TTTech industrial

- Client-based stream interface
  - `Talker_Join_Requests`
  - `Listener_Join_Requests`
    - first `listener join` together with the `talker join` – create the basis for the stream configuration
  - `Talker_Leave_Requests`
    - causes a stream configuration to be removed from the network
  - `Listener_Leave_Requests`
    - if a last listener issues a `listener leave,` the corresponding stream configuration will be removed from the network
  - `Talker_Join_Response`
  - `Listener_Join_Response`
  - `Talker_Leave_Response`
  - `Listener_Leave_Response`

- Stream-set interface
  - `Add_StreamList[]`
  - `Add_StreamList_Response[]`
  - `Remove_StreamList[]`
  - `Remove_StreamList_Response[]`

CNC responsibilities (1)

- CNC can receive a Join Request from at east one Talker and one Listener in order to process/generate the stream configuration for a new stream
- CNC can process single requests coming from endpoints for an existing stream (already configured in the system), i.e.,
  - Join Request (from an additional Listener)
  - Leave Request (from an existing Listener)
  - Leave Request (from the Talker)

- A positive **“join” configuration response** will be sent from the CNC towards CUC including endpoint stream configuration parameters (DMAC, VLAN ID, transmit offset, …) for the endpoints, after the CNC has successfully
  - a) generated the configuration
  - b) distributed the configuration to the switches
  - c) received the confirmation from the switches that the configuration update is processed successfully
- A **positive “talker leave**” configuration response will be sent from the CNC towards CUC at the talker first. Leave response will be distributed to switches after it is ensured the frames of the particular stream are not in transit for the given path.

CNC responsibilities (4)

TTTech industrial

- CNC shall generate the configuration for the scheduled and non-scheduled streams
- CNC shall
  - consider the configuration for non-streams (traffic engineered)

- CNC operates in incremental configuration mode,
  - new requests do not change the configuration parameters of end devices which are not related to the new request
  - the QoS requirements of existing streams are still fulfilled

Example:
Max Latency = QoS Request
Accumulated Latency = Response from CNC

First request
S1: Config. MaxLatency_S1 = 10 ms
AccLatency_S1 = 5ms <10ms

Second request:
S2 Config. MaxLatency_S2 = 10ms
AccLatency_S2 = 6ms
AccLatency_S1 = 6ms <10ms