2021-02-09 TAPI Meeting Notes

Date

09 Feb 2021

Attendees

- Andrea Mazzini
- Ronald Zabaleta
- Karthik Sethuraman
- Ramon Casellas
- Leo Nederlof
- Xiang YUN
- Hing-Kam Lam
- Victor Lopez

Goals

- Admin
- Review the updated TR-547-TAPI Reference Implementation Agreement_v1.1.docx
    - *New use cases*: 0d, 1g, 1h, 2a, 2b, 2c, 3d, 3e, 3f, 5d, 11a, 11b, 13b, 13c, 16a, 16b.
        - Continue review of new Telefonica contribution otcc2021_usecase0d_Plug_id.pptx
        - Continue review of 2021-01-26 TAPI Additional Call - Meeting Notes

Agreed Items & Priority

- Below the list of the agreed items and related priority for the next TAPI & RIA versions.
    - An item is blocking when its resolution is necessary precondition for the next delivery.

1. OTU(+ODUCn) CEP/CSEP as single point for OTU/OTSiA ConnectivityService provisioning (blocking, 1)
    a. 3R
    b. ENNI/INNI Asymmetric service provisioning for multi-domain scenarios, agree UCs.
2. OTS and OMS model (blocking, 2)
3. Lifecycle management of ConnectivityService at every layer, necessary to identify UCs (blocking, 3)
    a. Lifecycle management of single ConnectivityService, necessary to identify UCs
4. MEP/MIP model vs. direct inclusion of OAM parameters in the CEP (blocking, 4)
    a. ODU OAM
    b. Photonic OAM
    c. TCA provisioning
5. Elementary alarm (e.g. ITU-T cZZZ fault causes), including TCA related notif), current and history (blocking, 5)
6. Photonic model capability (blocking, 6)
7. UNI Client interfaces modelling. DSR/ODU multiplexing over ODU (not blocking)
8. RESTCONF Response codes for use cases (not blocking)
9. TAPI OAS, action points to be assigned (not blocking)
10. Routing Constraints (not blocking)
11. Physical Route (not blocking)

Discussion items

15 mins	Administrative	Andrea Mazzini	16 Feb 2021 TAPI Call: **2 hours**
			• Review the updated TR-547-TAPI Reference Implementation Agreement_v1.1.docx
			  • *New use cases*: 0d, 1g, 1h, 2a, 2b, 2c, 3d, 3e, 3f, 5d, 11a, 11b, 13b, 13c, 16a, 16b.
			  • Finalize agreements on UC 1g/2b (server layer constraints).
			  • *Consider to suspend the discussion on constraints to proceed with the other new use cases.*
			☑ **Ramon Casellas has taken the role of RIA Editor.**
			• Malcolm Betts will be reducing his level of participation in ONF TAPI.

15 mins	Review the updated TR-547-TAPI Reference Implementation Agreement_v1.1.docx • Review of new Telefonica contribution otcc2021_usecase0d_Plug_id.pptx	Ramon Casellas	Agreed to specify the encoding of SAPI/DAPI in the RIA (UML type is *string*). • Ramon Casellas the string should be a 16 hex characters, lower case. ☐ Ramon Casellas, Andrea Mazzini to check the reference standard for the format. • Agreed to add the "inter-domain-plug-id" attribute of ENNI-N NEP as name-value pair in **2.1.3 version**, specifying the separator between SAPI and D • Delegate to the orchestrator the management of SAPI/DAPI combinations at the two ENNI terminations, e.g.: • East ENNI-N it is SAPI-East/DAPI-West • West ENNI-N is it SAPI-West/DAPI-East
120 mins	Review the updated TR-547-TAPI Reference Implementation Agreement_v1.1.docx		Andrea Mazzini shows the 2021-01-26 TAPI Additional Call - Meeting Notes, use case 1g. • Last week Ramon Casellas indicated that it could be possible to *stretch* the Restconf by specifying in the same *POST* more distinct roots. • In case, the *POST* response should include n x HTTP Header *Locations.* • Reaffirmed that Restconf prescribes that identifiers must be defined by the Client. It is not foreseen Server assignment of identifiers. • More general discussion on how the ConnectivityService creation is coded in the "Minimum subset required of TAPI RESTCONF Data API" table of • Eventually agreed that the current POST is correct with respect to Restconf specifications: • /tapi-common:context/tapi-connectivity:connectivity-context/ • Considering REST, the above expression would indicate a modification of ALL the content of "connectivity-context", i.e. a complete replacement of tl indicates an ADDITION of the specified node to the "connectivity-service" list. See https://tools.ietf.org/html/rfc8040#appendix-B.2, where "artist" is the new "connectivity-service" and "library" is the "connectivity-context": **B.2.1. Create New Data Resources** To create a new "artist" resource within the "library" resource, the client might send the following request:

- Use case 1g: Unconstrained PHOTONIC_MEDIA/OTSiMC over MCService Provisioning

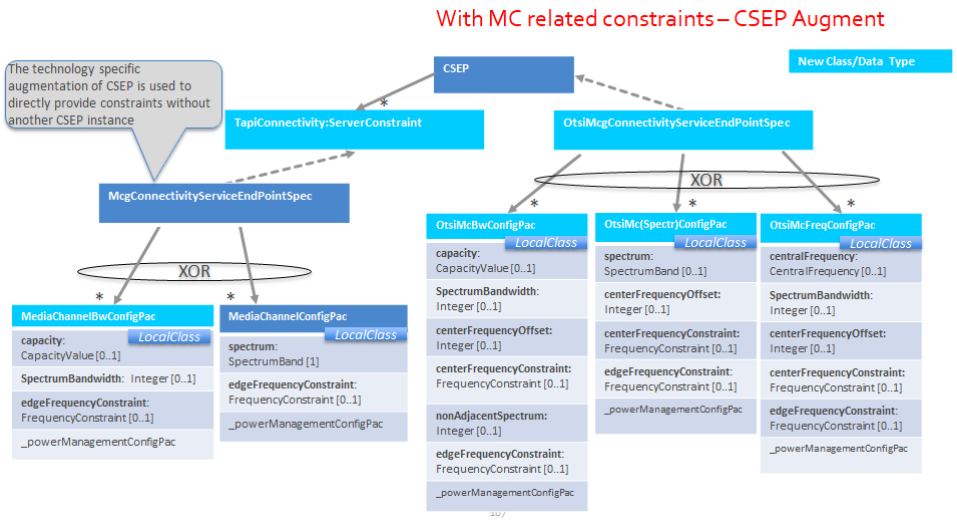
Andrea Mazzini

Nigel Davis

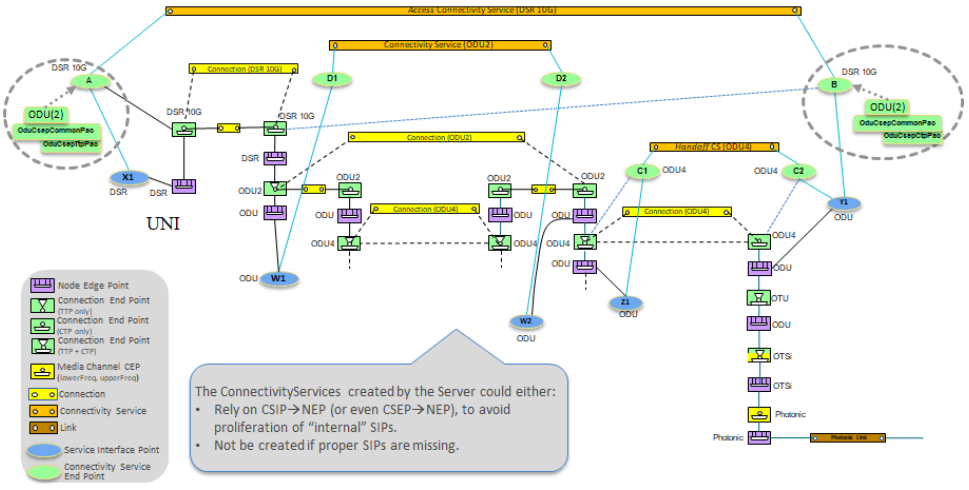
Ramon Casellas

```
POST /restconf/data/example-jukebox:jukebox/library HTTP/1.1
Host: example.com
      Content-Type: application/yang-data+json

      {
        "example-jukebox:artist" : [
          {
            "name" : "Foo Fighters"
          }
        ]
      }

   If the resource is created, the server might respond as follows:

      HTTP/1.1 201 Created
      Date: Thu, 26 Jan 2017 20:56:30 GMT
      Server: example-server
      Location: https://example.com/restconf/data/\
          example-jukebox:jukebox/library/artist=Foo%20Fighters
      Last-Modified: Thu, 26 Jan 2017 20:56:30 GMT
      ETag: "b3830f23a4c"
```

- Then long discussion on the best way to provide server constraints of a ConnectivityService.
- Andrea Mazzini presents a possible simplification of CSEP provisioning, where a new class, "ServerConstraint" is composed by CSEP:

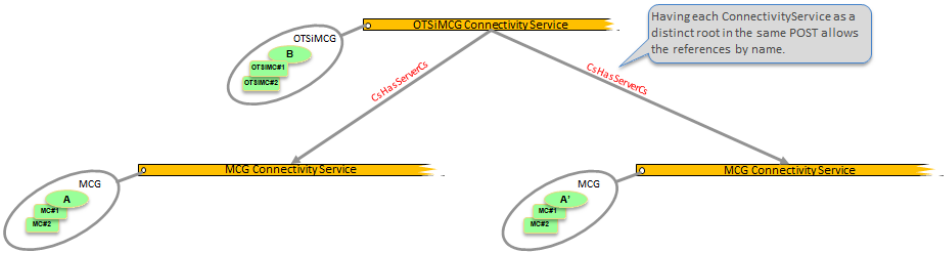


- In this way it is possible to recycle the technology specific augmentations of the generic CSEP without the need of additional CSEP instances, e.g.:

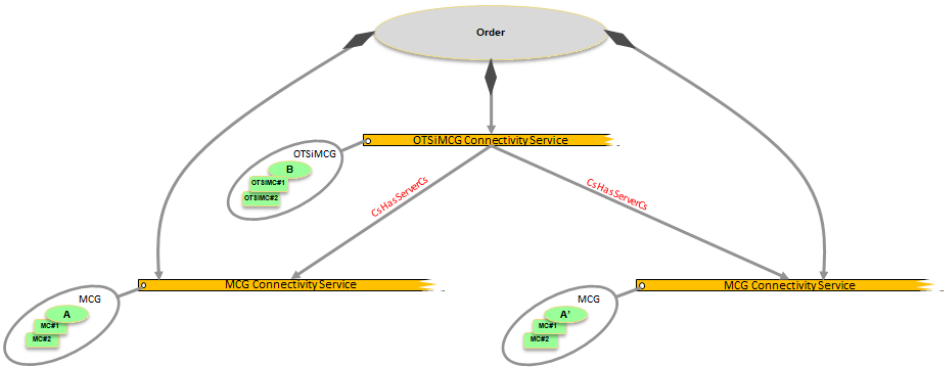


- Karthik Sethuramanand Ramon Casellas highlight that the proposed approach is limited to the constraints expressed by CSEP augments. There are layer of the ConnectivityService under provisioning.
- Karthik Sethuramanand Ramon Casellas agree that the bottom-up provisioning is still preferable for complex multi-layer provisioning scenarios.
- Andrea Mazzini recalls that regarding top-down provisioning, there is a requirement to create also server ConnectivityServices because:
  1. Alignment to bottom-up provisioning - if a given layer protocol foresees the ConnectivityService, then the ConnectivityService shall be instantia
  2. Editing capability of automatically created server layer connectivities.
- Karthik Sethuraman points out that the best way to fulfil the above requirements is to provision all the ConnectivityServices, client and server one(s)
  - Using only "constraints" to drive the creation of server ConnectivityServices leads to two ways to edit a server ConnectivityService:
    - by constraints associated to client ConnectivityService, and
    - directly on the server ConnectivityService once it is created.
- Andrea Mazzini post-meeting note: as already discussed on previous calls, it is not possible to provision a tree of ConnectivityServices, because the ConnectivityService(s). In other words, the tree structure does not work for ConnectivityServices.
- Some conclusions:
  1. For Use Cases 1g and 2b check whether TAPI 2.1.3 can support them through simple amendments (or even just name-value pairs)
     - If not, i.e. the modifications to 2.1.3 appear complex, then remove these Use Cases from 2.1.3 scope.
  2. In general, for simple constraints it may be preferable to use mechanisms like the CSEP "ServerConstraint" presented above.
     - *Post meeting note:* Two-stage constraining of a server ConnectivityService:
       - first stage at provisioning time of client ConnectivityService (few details),
       - second stage at editing time of created server ConnectivityService (full details).
  3. Explore the provisioning of more ConnectivityServices in one operation, currently there are two possible solutions:
     a. combine more ConnectivityServices / "roots" in same *Post,*
     b. introduce the *Order* model.

- Option a:

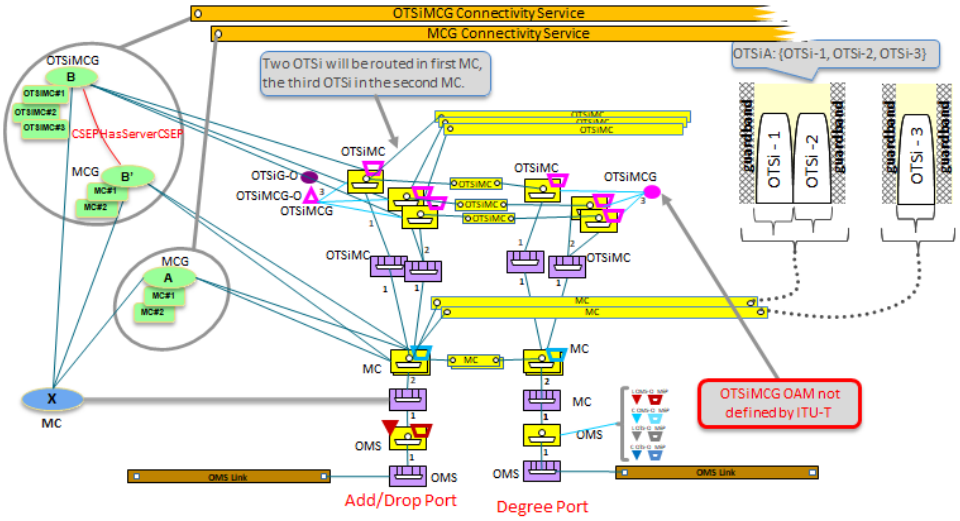


- Option b:



Additional notes:

- Allowing two distinct MCG CSEP instances may be a temporary solution for 2.1.3, but to be evaluated the forward compatibility.



- Defining dedicated data structures for server layer constraints may be more forward compatible but to be evaluated the detailed impact on 2.1.3 vers