Proposal for P802.1Qcc Control Flows

Norman Finn
Cisco Systems

Version 2

Apr. 15, 2014

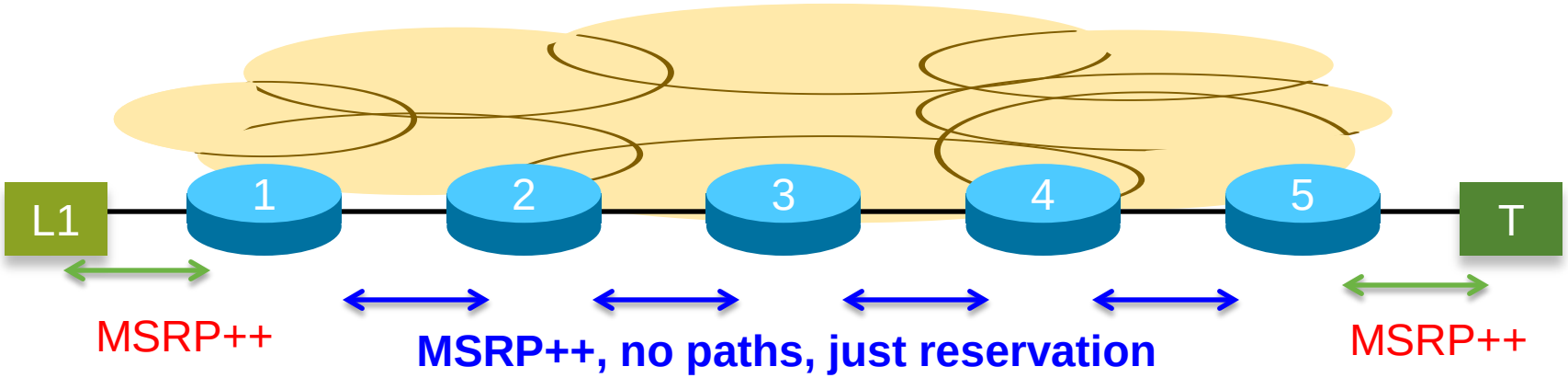
This presentation

- This is cc-nfinn-control-flows-0414-v02.
- It is based on cc-nfinn-Inputs-Outputs-0314-v02, cc-nf-rc-Information-Flows-04-14-v1, and comments received during its first presentation.

Principles (from cc-nf-rc-Information-Flows-04-14-v1)

- We need a Central Compute and Control (CCC) function to compute multiple paths, and to perform complex operations such as changing schedules or paths for some flows, while other flows are still running and being guaranteed service qualities.
- We have to be able to do without a CCC for backwards compatibility, and for use cases that simply don't need the features that a CCC can most efficiently supply.
- **We have to understand the flow of control information before we can pick protocols.**
- The control information flow is dictated by the QoS requirements, and is independent of whether we're doing L2 or L3.
- **An L2-only solution is absolutely required.**

P. Network built **without** Central Computation Control function (peer-to-peer)

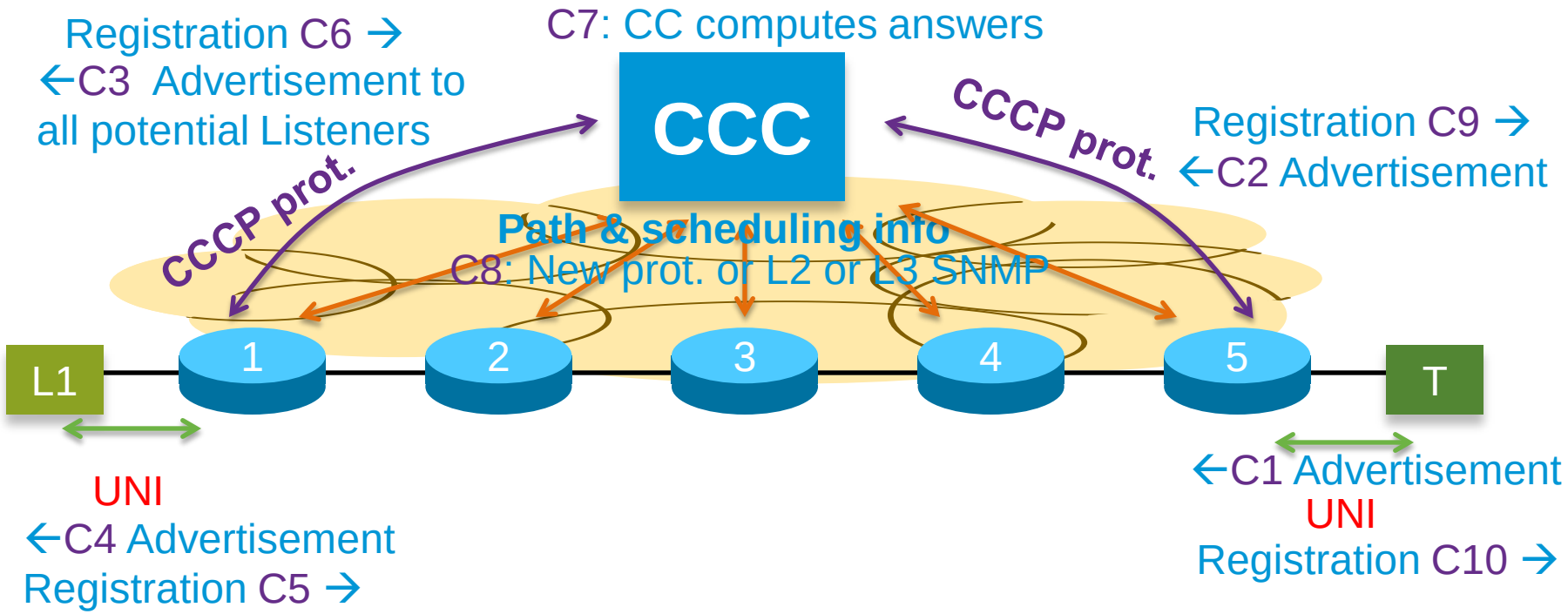


← P3 Advertisement Registration P4→

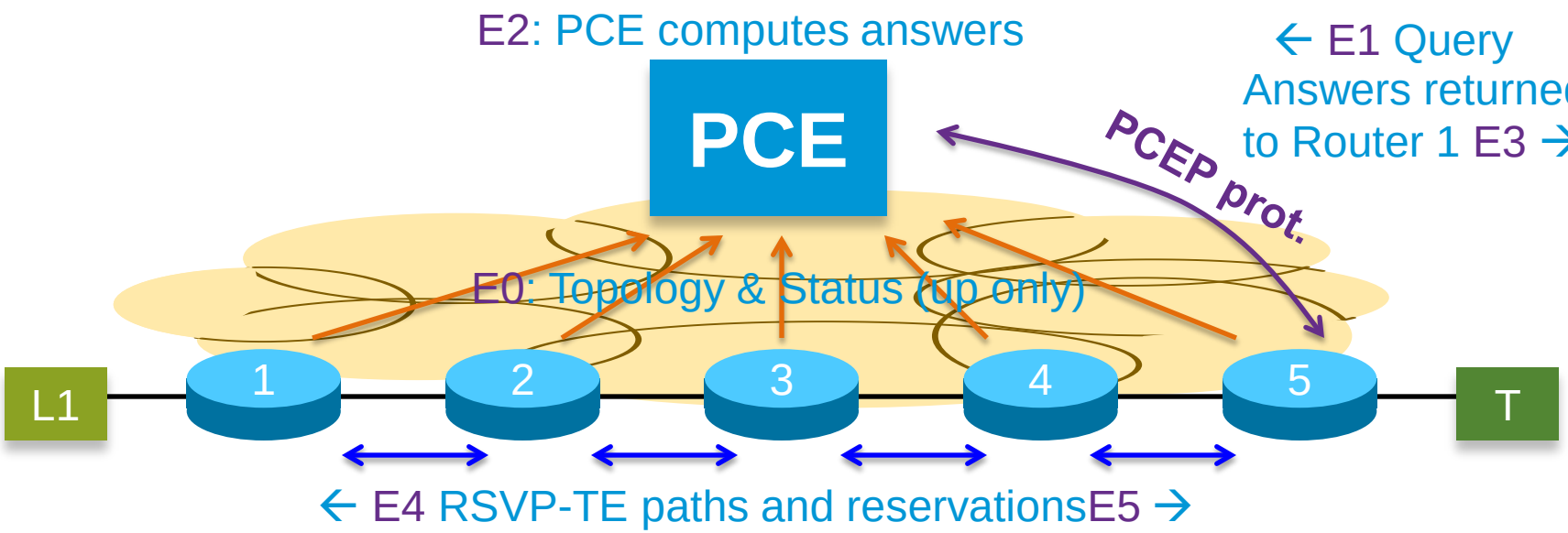
← P2 Advertisement (hop by hop) Registration P5 → (hop by hop)

←P1 Advertisement P6 Registration →

C. Network built **with** Central Computation and Control function (CCC)



E. Current PCE information flow



- Talkers and Listeners are not involved.
- Edge router(s) nearest Listener(s) obtains a path from the PCE via PCEP, and uses RSVP-TE to establish the path in the network.
- This information flow is optimum when paths are stable, and seamless bandwidth/path changes are not being made. Comparison of E and C information flows needs to be made.

List of information elements

This is a strawman!

- The rest of this deck is a strawman, at which people are free to throw sticks, stones, and/or lighted matches.
- It tries to gather the kinds of information that needs to flow.
- Arguments that more or fewer information elements are required are welcome.

Ispec (Identification specification)

- Meta information for functions to share information via protocols.

1. **iKey**: Database key to identify streams to control protocols
   - There may be no need for this, or we may need many. (TBD)
   - The Fspec will be sufficient for many purposes.
   - End-to-end stream ID, used only by layers above the TSN Layer (E.g. a URL for a TV program. Such items are usually carried in other (non-TSN) protocols.)
   - AVB v1 Stream ID based on Talker, for AVB v1 compatibility.
   - AVB v1 Stream ID based on some other role, e.g. based on a CCC, an edge network node, or a Listener.
2. **iReqTime**: Time of original request
   - Appears in protocols only to stabilize peer-to-peer (network node or CCC) decisions. Otherwise, this is an internal CCC variable.
3. **iRank**: Rank of request
   - Importance of flow, e.g.”911 call”.

Fspec (Flow specification)

- These parameters can change, e.g. for VLAN ID remapping or Network Address Translation (NAT).

1. **fDAddr**: The Listeners' address stack, from L1 up to the highest layer supported by the TSN Shim.
   - These are "native" addresses – those used to reach the Listener(s) through normal (non-TSN) networking processes.
   - Addresses are those used on the link on which the Fspec is being transmitted; addresses may or may not have end-to-end significance. VLAN-ID is included, but not L2 priority.
   - There is only one Listener address stack. It may be a unicast for a single Listener, or a multicast, for one or more Listeners.
2. **fSAddr**: The Talker address stack
   - One locally-significant, native address stack, as for 1.
3. **fEAddr**: Encapsulation parameter (address) stack
   - Specifies additions and/or substitutions made to the Talker and/or Listener address stacks at this point in the flow.

Tspec (Transmission specification)

- Characterizes the **Talker's promise** to the network.
- The Tspec can change at interworking functions.

1. **tIntvl**: Measurement interval
   - Let's think about this one. Is this number user friendly? Is it something the user can easily and reliably produce?
   - Perhaps what we need is an "ingress burst buffer size?"
2. **tPkts**: Max packets per interval
   - With max packet size, the bandwidth specification
3. **tSize**: Max packet size
4. **tBytes**: Bytes per measurement interval
   - Low priority for inclusion. This can make the packing of flows into resources a little bit more efficient.
5. **tEncTyp**: Encapsulation type (TSN, L3 pseudowire, HSR-like, etc.)
   - Type of encapsulation, not the parameters (e.g. VLAN IDs or labels).
   - Strong connection to Max packet size. AVB v1 uses TSN encaps.

Nspec (Network specification)

- Characterizes the **Network's promise** to the Talker and Listener(s).

1. **nMxLat**: Required end-to-end maximum latency
   - No default. This is the Traffic Class choice in AVB v1.
2. **nMnLat**: Required end-to-end minimum latency required
   - "Jitter" is usually a "±". Maximum allows no "+". Max + min make sense.
   - Default: 0.
3. **nLoss**: Required packet loss rate
   - Perhaps expressed logarithmically, to make it an integer.
   - Default: "best effort at zero congestion loss."

Aspec (Additional Tspec/Nspec items)

1. **aTsNs**: Alternative sets of Tspec + Nspec values
   - Network can pick an alternate set in the interest of global optimization.
   - Network must select a set; it cannot pick among individual parameters.
   - Default: no alternatives.

Pspec (Path specification)

1. **pPath**: Ordered list of {Network node ID, Port ID}
   - List may be incomplete—not all network nodes need be specified, and the Port ID may be left to the network node to determine.
2. **pConst**: Path constraints, e.g.:
   - A list of network node IDs to be avoided.
   - "What to optimize" choices (available non-TSN bandwidth, network node buffers, link and node reliability, …)
3. One can argue that the "required packet loss rate" in the Nspec belongs here.

Wspec (forWarding specification)

1. **wFdb**: Filtering/forwarding database entries required to forward the various flows.
2. **wParms**: Parameters required by transmission selection algorithms such as:
   - Per-priority per-port shaper parameters.
   - Per-queue output schedule.
   - Per-queue input schedule.

Rspec (Router/Bridge specification) (1/2)

- The context of these parameters may be a network node, a particular flow, an encapsulation, a set of input-output port pairs, some combination of those, or some other context, TBD. We have work to do, here.

1. **rMnFDel**, **rMxFDel**: Minimum and maximum forwarding delay
   - From arrival of first bit of packet (at lowest address level in Fspec) until packet is available for transmission selection from all output queues to which it can be placed.
2. **rMnTDel**, **rMxTDel**: Minimum and maximum transmission delay
   - From transmission selection until first bit of packet is placed on the transmission medium. See also gLkDel.
3. **rAccMnDel**, **rAccMxDel**: Accumulated minimum and maximum delay from Talker to Listener(s)

Rspec (Router/Bridge specification) (2/2)

4. **rCap**: Capabilities of a network node. We can expand this, but for the moment, it's a catch-all, including:
   - Supported encapsulation forwarding types, e.g. routing, bridging, IEC 62439-3, ITU-T Y.8032, TSN bridging, PBB-TE, etc.
   - Supported termination / proxy / interworking encapsulations
   - Supported transmission selection algorithms and options.
5. **rResv**: All established reservations in this network node
   - Allows backup CCC to collect state of network.
6. **rStatus**: Status of links, e.g. short-term history of non-TSN load.
   - This is the sort of thing sent to PCE via ISIS, at this time.
7. **rTimeAcc**: Time sync accuracy.

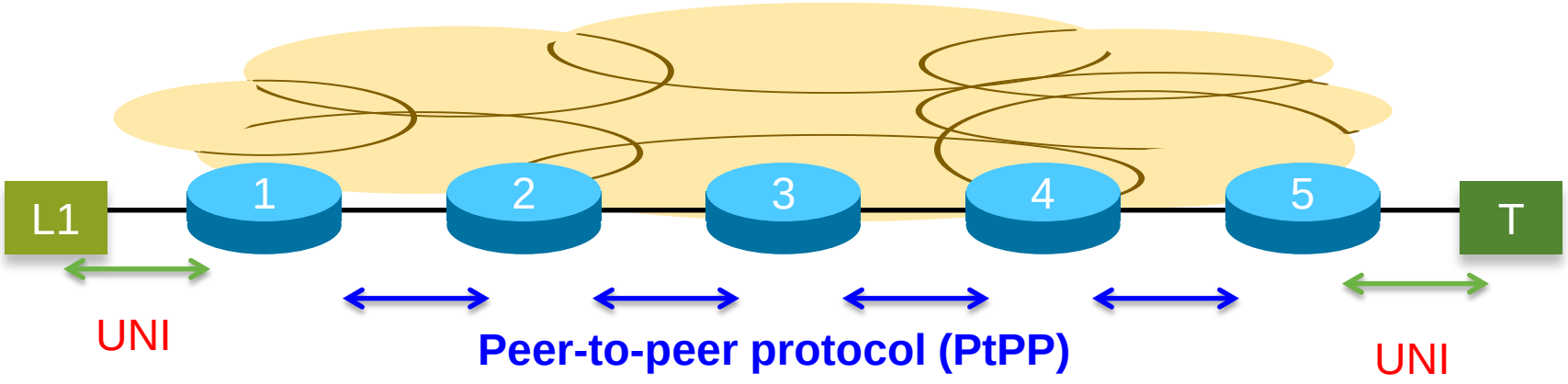
Gspec (topoloGy specification)

- Much of this is per port. This information is tied to, and we may decide to merge it with, the Rspec.

1. **gID**: Identify of this network node.
   - There may be more than one of these, because a system can have multiple functions (router, bridge) and multiple IDs (LLDP, routed ISIS)..
2. **gConn**: Connectivity
   - A list of Port IDs, and a list of the gIDs of the neighbors on each port.
3. **gLkDel**: Link delay.
   - This is a maximum, preferably measured.
   - Variation is handled by rMnTDel and rMxTDel.
4. **gLkParms**: Link parameters (speed, type, etc.)

Hop-by-hop information element requirements

P. Network built **without** Central Computation Control function (peer-to-peer)



← P3 Advertisement Registration P4→

← P2 Advertisement (hop by hop) Registration P5 → (hop by hop)

←P1 Advertisement P6 Registration →

Protocol choices: Peer-to-Peer

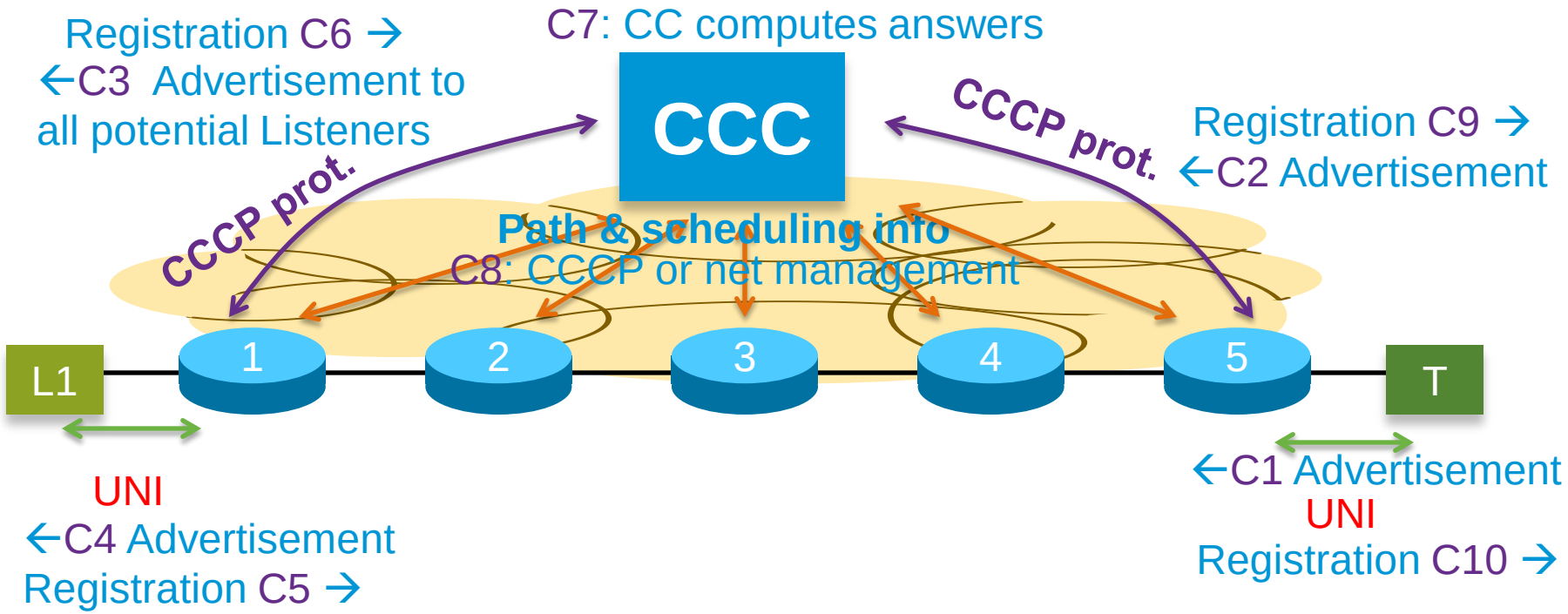
- "UNI" is, at least, P802.1Qcc MSRP++.
  - Required for backwards compatibility with AVB.
  - May be viable long-term with L3 additions.
  - Other protocols could carry the same information.
- "PtPP" is, at least, P802.1Qcc MSRP++.
  - The IETF, in cooperation with IEEE 802.1, may also may wish to add information elements from P802.1Qcc to some existing protocol.
- NOTE: It is not clear to this author whether we want the peer-to-peer model to offer a minimum latency guarantee.

Information elements: Peer-to-Peer

hop	Ispec Fspec	Tspec Nspec Aspec	Pspec† Wspec	Rspec Gspec
P1 Advert T to N5	iKey (fDAddr fSAddr)	tEncType (all others)*	all	none
P2 Advert N5 to … to N1	iKey (fDAddr fSAddr)	tEncType (all others)*	all	rAccDel
P3 Advert N1 to L	all	tEncType (all others)	none	none
P4 Reg L to N1	iKey (all others)	all	none	none
P5 Reg N1 to … to N5	all	all	none	none
P6 Reg N5 to T	all	all	none	none

* Full Tspec, Nspec required from Talker if Listener is to be told the Nspec.
† Pspec required only if we support source-routed path from Talker.
(Items in parentheses must be supplied by Talker, Listener, or both.)

C. Network built **with** Central Computation and Control function (CCC)



Protocol choices: CCC

- "UNI" is, at least, P802.1Qcc MSRP++.
  - Required for backwards compatibility with AVB.
  - May be viable long-term with L3 additions.
  - Other protocols could carry the same information.
- "Net management" is, at least, SNMP.
- "CCCP" could be PCEP, or something new.
  - IETF PCEP has many of the required elements, and fits the information flow model.
  - IEEE 802.1 can choose to invent something new.
  - IEEE 802.1 can define only information elements.

Information elements: CCC

hop	Ispec Fspec	Tspec Nspec Aspec	Pspec* Wspec	Rspec Gspec
C1, C2 Advert T to N5 to CCC	iKey (fDAddr fSAddr)	tEncType (all others)	Pspec only	none
C3, C4 Advert CCC to N1 to L	all	tEncType (all others)	none	none
C5, C6 Reg L to N1 to CCC	iKey (all others)	all	none	none
C8 Net mgt CCC to Nx	all	all	all	none
C8 info Nx to CCC	none	none	none	all
C9, C10 Reg CCC to N5 to T	all	all	none	none

* Pspec used T→CCC only if we support source-routed path from Talker.
(Items in parentheses must be supplied by Talker, Listener, or both.)
This slide, in particular, needs further work.

Summary – node/node/controller flows
(other suggestions welcome)

Information flow	+	–
"P" no CCC, all horizontal	What MSRP does today Can support SQF	Cannot make global tradeoffs
"C" CCC–to-node direct, no horizontal	CCC–node can be net mgmt Quickest way to requirements Should merge nicely with "E" Does global reallocation best	"E" more efficient sometimes
"E" talker Edge drives CCC and horizontal	What IETF PCE does today Stateful backup in progress Models controller-to-controller CCC-to-node in progress Easy for smart Talker to drive	"C" more efficient sometimes

- **All three flows will be a part of the ultimate solution.**

Summary – protocol choices
(other suggestions welcome)

- Central Computation and Control
  - New thing (defined by protocols), IETF PCE++
- Topology collection by CCC/PCE
  - ISIS (OSPF), report neighbors via CCC-to-node vertical
- UNI
  - MSRP++, RSVP-TE++
- Node-to-node horizontal
  - MSRP++, RSVP-TE++
- Edge node to CCC request/response
  - CCCP (a new protocol), PCEP++
- CCC-to-node vertical
  - CCCP, PCEP++, SNMP, NETCONF

Summary – the "shortcut"
(other suggestions welcome)

- Define MSRP++ in 802.1Qcc for flow "P" and for UNI in all flows.
- Define CCCP for edge node-to-CCC request/response for flow "C".
  - This can be as simple as defining TLVs and picking a transport protocol.
- For CCC-to-node exchanges for flow "C", including topology discovery, either:
  - Include data elements for this in CCCP, or
  - Do this via network management.

Thank you.