

Management Protocol for 1722.1

Rodney Cummings
National Instruments

Agenda

- Context
 - Focus of this presentation, relative to 'New Work' (slide 12) of http://grouper.ieee.org/groups/1722/contributions/2015/P1722_1-Industrial_Potential_Alignment.pdf
- Technical background
 - Overview of technologies we can choose from
- Recommendations

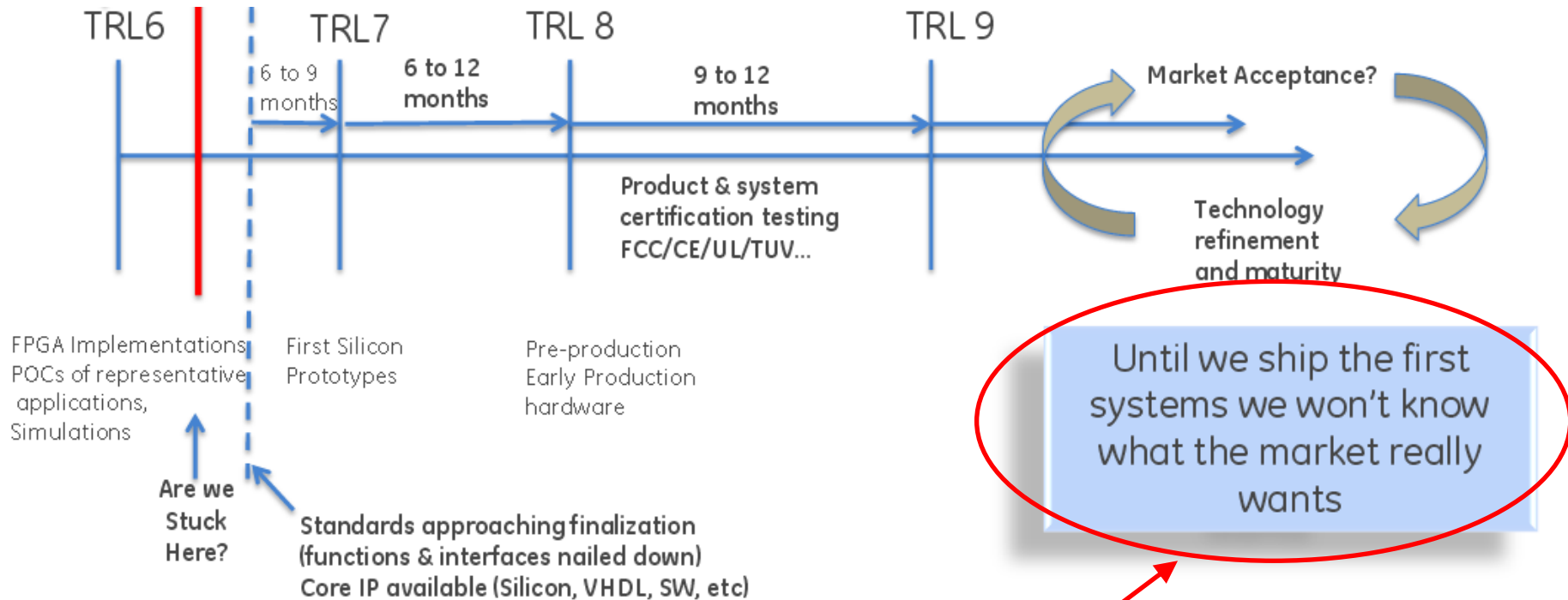
Context

Using slides taken from AVnu Industrial presentation
at May 802.1 TSN meeting:

<http://www.ieee802.org/1/files/public/docs2015/tsn-sexton-feature-priority-request.pdf>

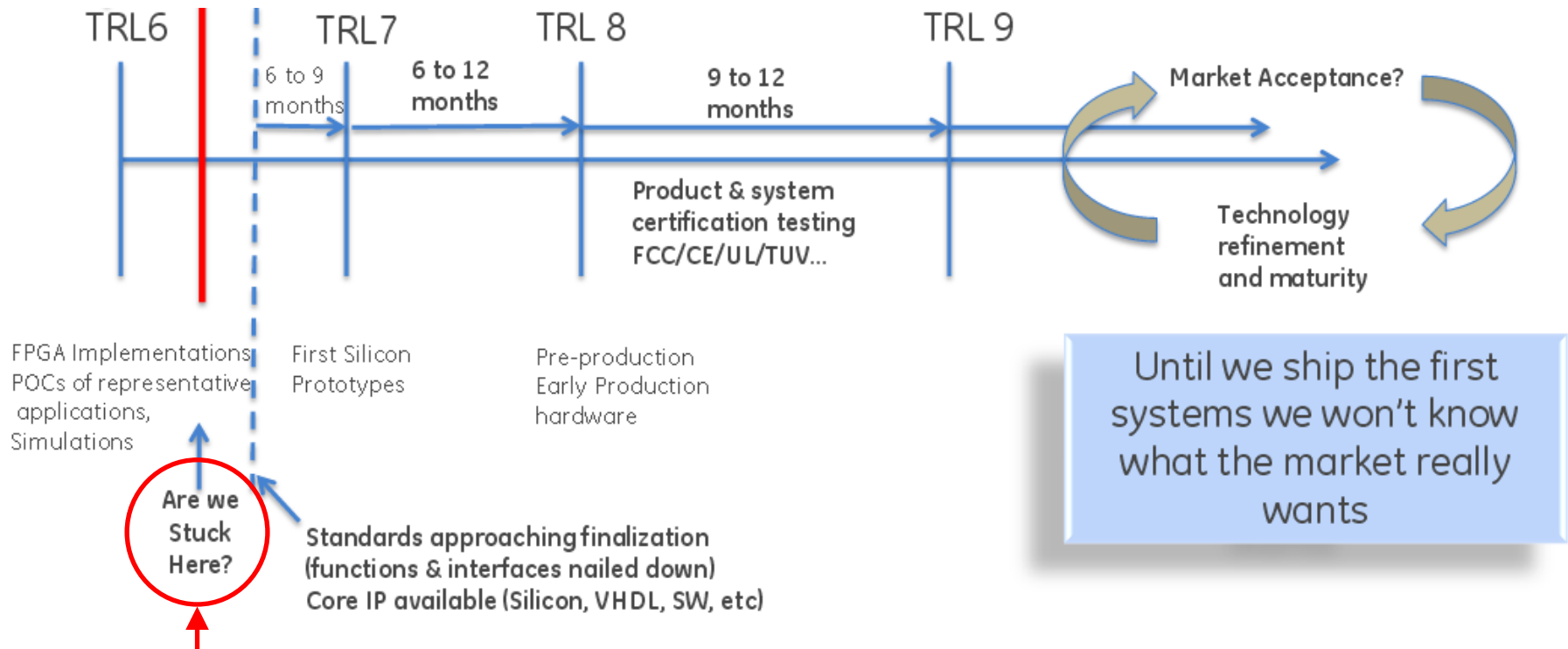
Red text is new

Moving Forward with Industrial (1 of 2)



Select minimum feature set viable for real applications.

Moving Forward with Industrial (2 of 2)



Get unstuck by focusing on

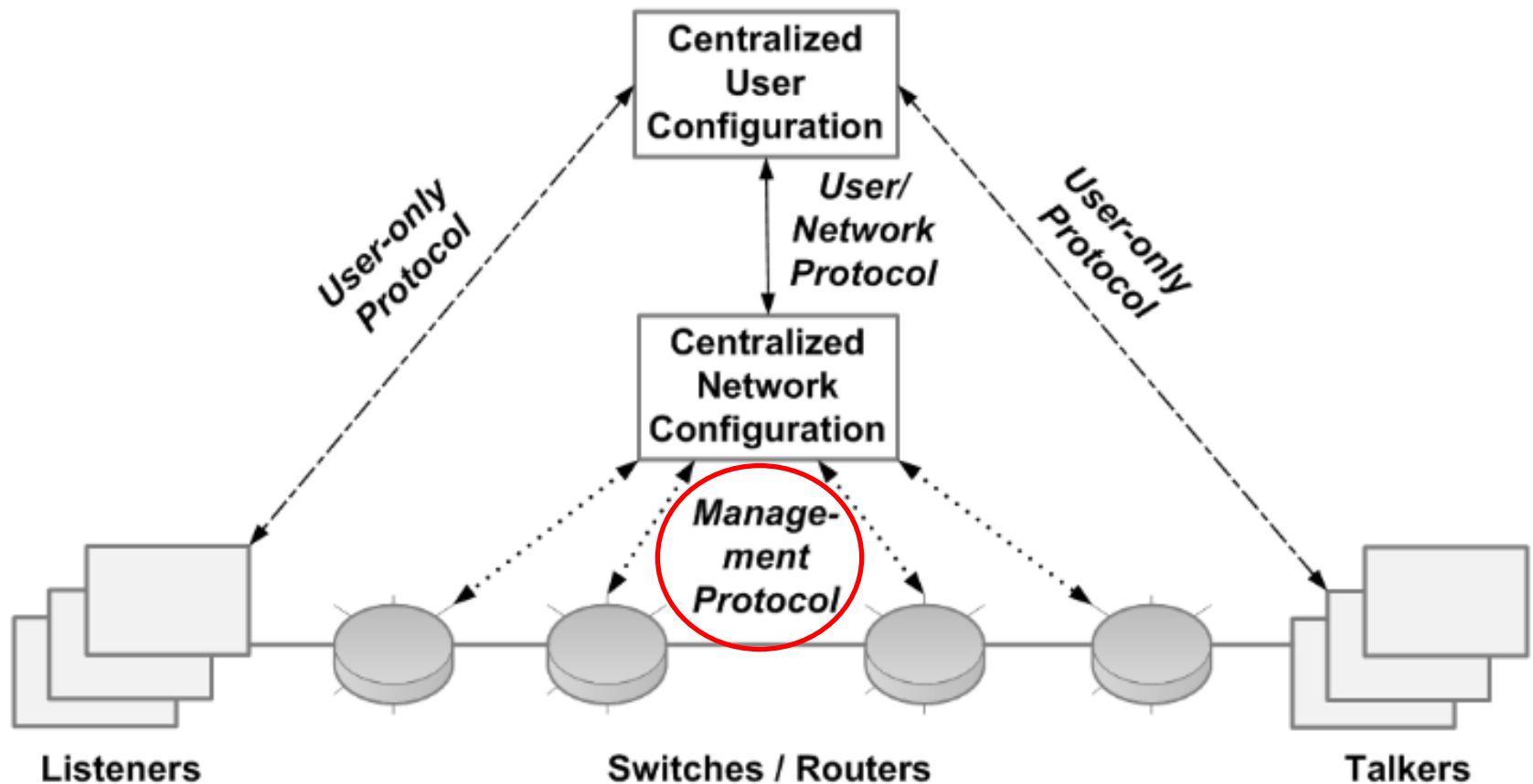
- Standards at-or-near publication; silicon at-or-near shipping
- Existing protocols with open source code (preferably C)

Minimum Viable for Industrial

Time to kickoff (dotted line)

Priority	Requirement	Project(s)	2015	2016	2017
1	Network time synch with static config	ASrev	Y	Y	Y
1	Scheduling	Qbv	Y	Y	Y
1	Centralized config	Qcc, Restconf/Netconf 1722?	Y	Y	Y
2	Seamless redundancy including time synch	CB, ASrev	N	Y	Y
2	Ingress policing including BE limiting	Qci	Y/N	Y	Y
2	Frame preemption	Qbu	N	N	Y
2	L3 support		N	N	Y
3	Cyclic schedule	Qch	N	N	TBD
3	Credit based shaper	Qav	N	N	TBD
3	Stream management (SRP)	Qat	N	N	TBD
-	ISIS	Qca	N	N	TBD

This Presentation: Management (1 of 2)



This Presentation: Management (2 of 2)

- Assumptions for Management Protocol
 - Required for any CNC use case
 - Users can be centralized or distributed
 - Server (bridge) can be constrained ([RFC 7252](#))
 - e.g. Industrial sensor with two external Ethernet ports, 10 KiB RAM
 - Configuration is non-volatile
 - Some use cases cannot rely on CNC to be continuously available
 - Configuration is automated
 - No human is involved
 - In-band
 - No out-of-band IT network for management

Technical Background

History: MIB and SNMP

- IETF specifies management protocols and models
- MIB = data model
 - Text specification of hierarchical variables in objects
 - Cross-vendor interoperability; Independent of protocol
 - Used by standards like 802
- SNMP = protocol and information model
 - Information model is on-the-wire encoding of data
 - Typically UDP, but layer-2 specified in [RFC 4789](#)
- [RFC 3535](#): SNMP/MIB great for 'read', bad for 'write'
 - Initiated creation of new management protocol(s)
 - TSN's CNC needs 'write', so it requires these new protocols

Management Protocol Creation

- Steps to create a new one (not necessarily sequential)
 1. Specify data modeling language
 2. Use #1 to specify data modules for features
 3. Transport: Specify protocol to carry mgmt messages
 4. Info: Specify information model (on-the-wire data value)
 5. Message: Specify mgmt message: request, reply, notification
 6. ID: Specify identifier of object to read/write (maps to #1)
 7. Transport Code: Open source code for #3
 8. Info Code: Open source code for #4
 9. Message Code: Open source code for #5 and #6
 10. Conformance: Create code to test/certify implementations

Step 1: Data Modeling Language

- Done: Published as YANG ([RFC 6020](#))

- Example

```
container system {
  leaf host-name {
    type string;
    description "Hostname for this system";
  }

  leaf-list domain-search {
    type string;
    description "List of domain names to search";
  }

  container login {
    leaf message {
      type string;
      description
        "Message given at start of login session";
    }

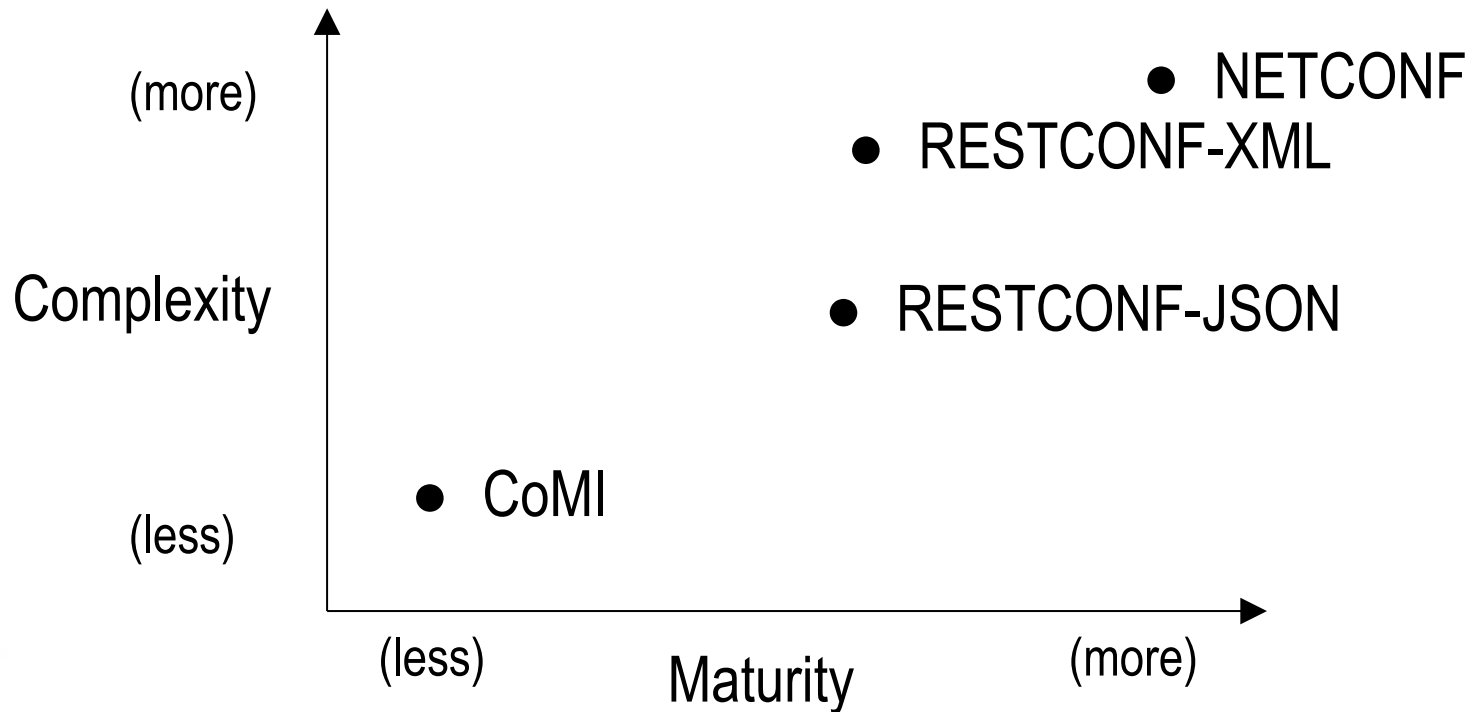
    list user {
      key "name";
      leaf name {
        type string;
      }
      leaf full-name {
        type string;
      }
      leaf class {
        type string;
      }
    }
  }
}
```

Step 2: YANG for TSN Features

- In-work
- All 802.1 features specify Managed Objects (e.g. .1Qbv)
 - Management data for that feature, independent of data model
 - In 802.1Q, this is Clause 12
 - Clause 17 is the corresponding MIB
- [802.1Qcl](#) is specifying YANG for core 802.1Q features
 - Including overall structure of 802.1 YANG modules
- With structure decided, filling in features is simple
 - Map clause 12 of feature in a manner analogous to .1Qcl









Steps 3-10: Multiple Options

- Summary
 - Complexity: How difficult to implement and execute?
 - Maturity: How complete is spec and source code?











NETCONF

- Pro: Most mature
 - IETF proposed standard ([RFC 6241](#))
- Pro: Full-featured
 - NMS focused (IT)
- Con: Heavyweight

	Step	Description
	3. Transport	SSH mandatory; on TCP
	4. Info	XML
	5. Message	NETCONF-specific RPC
	6. ID	Uses XML to select / search
	7. Transport Code	Many for SSH
	8. Info Code	Many for XML
	9. Message Code	At least two (e.g. libnetconf)
	10. Conformance	None that I know of

RESTCONF

- Con: Moderately mature
 - Published as draft of IETF NETCONF Working Group
 - Not yet RPC: <https://tools.ietf.org/html/draft-ietf-netconf-restconf-05>
- Pro: Lean features
 - For web servers
- Pro: Edits non-volatile
- Con: XML mandatory
 - JSON optional
 - JSON perceived to be leaner than XML









	Step	Description
	3. Transport	HTTPS (HTTP, TLS, TCP)
	4. Info	XML or JSON
	5. Message	HTTP methods (REST)
	6. ID	RESTCONF api-path string
	7. Transport Code	Many for HTTPS
	8. Info Code	Many for XML and JSON
	9. Message Code	Some in OpenDaylight
	10. Conformance	None that I know of

HTTP/REST in Industrial

- Web servers have become common in industrial
 - Search web for “<industrial company> web server”
 - Easy way to expose proprietary/custom features
 - Secure: Enables some degree of IT/OT convergence
- JSON ‘marketed’ as more lean and modern than XML
- HTTPS/REST/JSON server is ~80% of the way to a RESTCONF-JSON server
- Proposal in NETCONF working group: Change conformance to allow a JSON-only RESTCONF server
 - Rejected

CoMI: CoAP Management Interface

- Con: Least mature
 - Published as draft of IETF CoRE Working Group (next slide)
 - Not yet RPC: <https://tools.ietf.org/html/draft-vanderstok-core-comi-06>
 - Not ready for source code
- Pro: Lightweight
 - i.e. Constrained
 - For IoT devices









	Step	Description
	3. Transport	CoAP (DTLS, UDP, IPv6)
	4. Info	CBOR
	5. Message	REST methods in CoAP
	6. ID	CoMI-specific hash ID
	7. Transport Code	Many for CoAP
	8. Info Code	Many for CBOR
	9. Message Code	None that I know of
	10. Conformance	None that I know of

CoRE: Constrained RESTful Environments

- IETF [CoRE Working Group](#) started in 2010
 - Arose from [6LowApp](#) (wireless sensors); [6TiSCH](#) also
 - IoT like TSN/DetNet (building automation, smart grid, etc)
- CoAP: Constrained Application Protocol ([RFC 7252](#))
 - REST for constrained IoT devices; Mature [implementations](#)
- CBOR: Concise Binary Object Rep ([RFC 7049](#))
 - Typed binary encoding; Mature [implementations](#)
 - Based on JSON, so translation is loss-less
- CoMI: DTLS/CoAP/CBOR is ~80% of the way to CoMI
 - Remaining 20% is less mature than RESTCONF

Idea: 1722.1-specific Management

- Con: IETF is the go-to standards body for management of switches/routers
- Con: Reinvents too much wheel
 - Can mitigate by re-using CBOR and/or CoAP
 - Then the question becomes... why not just use CoMI?
 - If we want changes to CoMI, we can ask
 - e.g. layer-2

	Step	Description
	3. Transport	1722.1 AVDECC
	4. Info	? (translatable from YANG)
	5. Message	? (REST-like)
	6. ID	?
	7. Transport Code	
	8. Info Code	?
	9. Message Code	?
	10. Conformance	?

Recommendations

Assumptions

- Industrial devices often embed a switch
- Two classes of industrial device
 - Non-constrained: Often run web server today
 - e.g. controller
 - Constrained: Some are “bare metal” (no operating system)
 - e.g. low-cost sensor
- RESTCONF-JSON is best fit for non-constrained
 - CoMI’s feature set may be too limited (need to explore)
- CoMI is best fit for constrained
 - Aligned with JSON
- Centralized Network Config (CNC) is non-constrained

Recommended 1722.1 Management

- Applies to new 1722.1 conformance class for CNC
 - Including a bridge that is managed by CNC
- 1722.1 CNC shall support both
 - RESTCONF-JSON client
 - CoMI client
- 1722.1 bridge shall support at least one of
 - RESTCONF-JSON server
 - CoMI server

RESTCONF-JSON Roadmap

- “RESTCONF-JSON” uses the RESTCONF spec, but 1722.1 changes
 - XML “MUST” to “MAY”
 - JSON “MAY” to “MUST”
- This is non-conformant to RESTCONF
 - RESTCONF client (e.g. NMS) fails with JSON-only server
 - Not a key IoT use case (e.g. NMS also assumes out-of-band mgmt)
 - Nevertheless, conformance for each standard must be clear
- Recommend: Coordinate with NETCONF WG on name
 - If “RESTCONF-JSON” is not sufficient, discuss alternatives

CoMI Roadmap

- CoMI needs more work
 - Unlikely to be ready for 2015 kickoff
- Recommend: Engagement between TSN and CoRE
 - TSN implies 1722, AVnu, and 802.1
 - Engagement implies software investment
- Goal: Prototype to see what we like and don't like
 - If needed, make suggestions to improve I-D
 - Prototypes can lead to open-source

Thank You