Common Vehicle Interface Initiative
OEM Roundtable #2

24 September 2020

AGENDA

- Recap of previous roundtable
- Scoping and questions
- Next Steps
- AOB

Recap of Previous Roundtable

First GENIVI/W3C Joint OEM Roundtable on Common Vehicle Interface Initiative (CVII) held on 16 July w/ 10 OEMs present

Intent was to launch an industry-wide, OEM-led dialog on the benefits of joint development of common vehicle data models, access protocols and standard interfaces in the entire scope of vehicle plus cloud

Supportive statements for the initiative provided by BMW, JLR and Ford

Additional support voiced by Volvo Cars

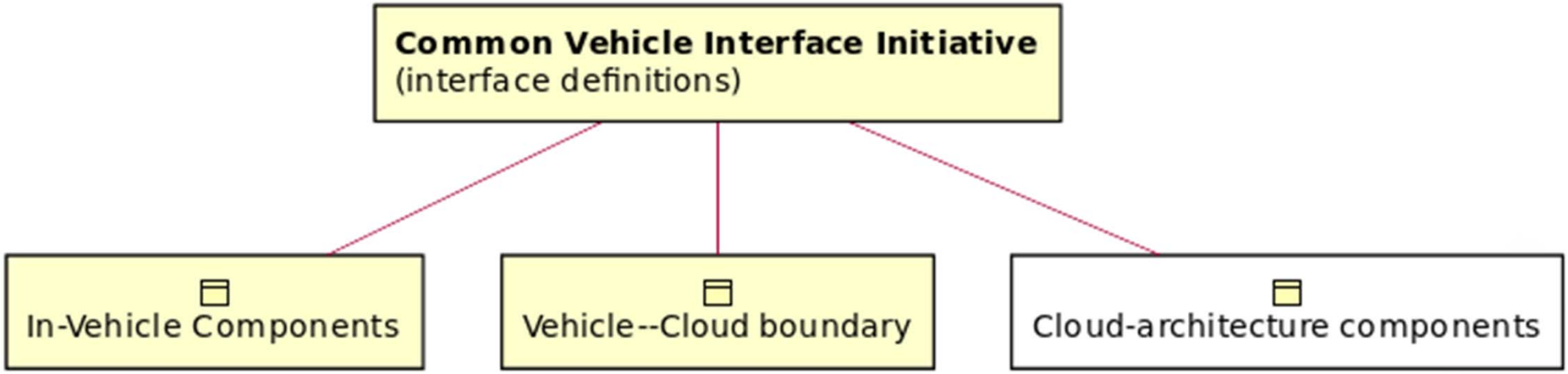
Questions raised about scope of the initiative which have been considered by GENIVI/W3C and by supporting OEMs

Scoping document and this presentation are the outcome of that additional scoping work

Combines illustrations with open questions for OEM feedback

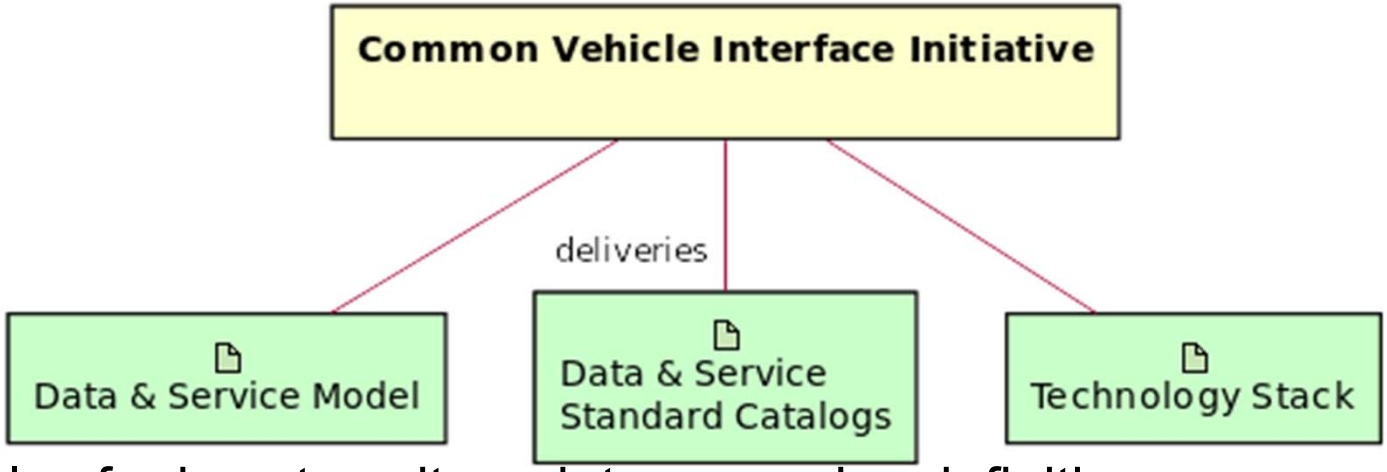
Does NOT highlight potential business value or opportunities

CVII Scopes



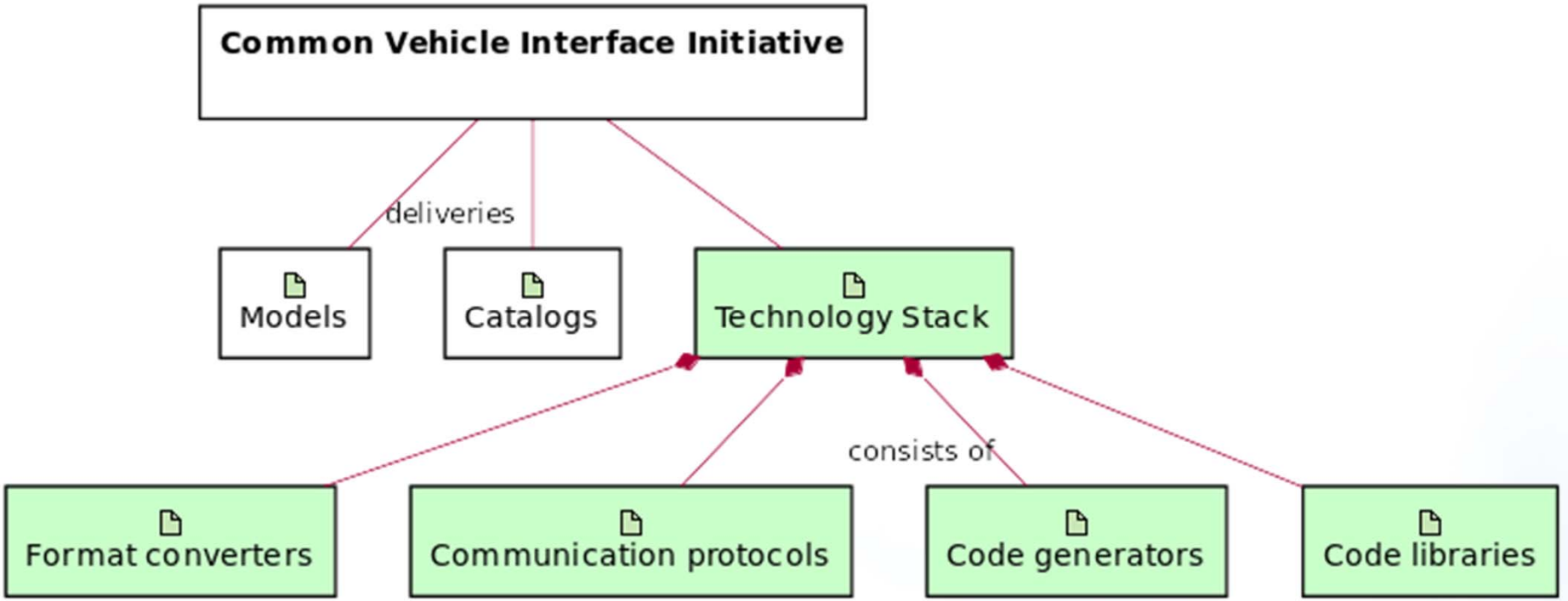
- Do the CVII participants agree that all these aspects are in scope?
- Which is the primary interest (if any) for your company?
- Is it better to explore the decomposition of cloud-architecture, including standard APIs, as a separate topic?

CVII Deliverables (recap and clarification)

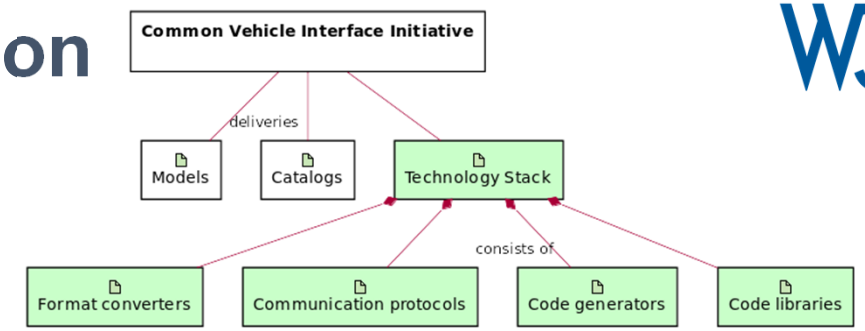


- **Model** = Rules for how to write a data or service definition.
  Defines the syntax/format of expressions and their meaning (expected behavior), including a specified set of metadata that is required, clarifiying the available datatypes to be used, and other rules.
- **Catalog** = Actual definitions. A collection of specified data items and Service APIs. The definition must follow the rules of the Model.
- **Standard Catalog** = Shared catalog of items expected to be provided by all implementations

Technology Stack – example of contents



Technology Stack – definition



By “Technology Stack”, we mean:

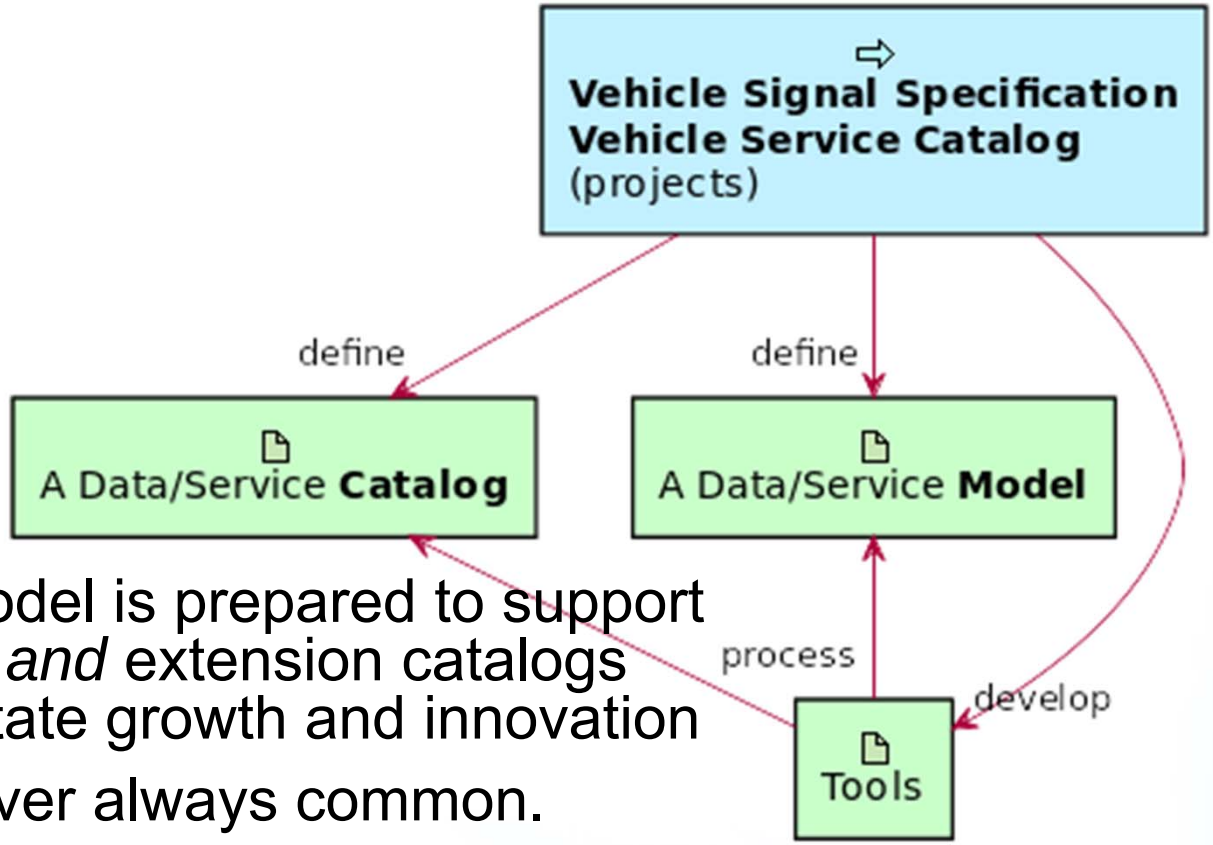
- Software (possibly some Hardware) involved in processing the agreed common data and interface models.

Including: translators, bindings, tools, protocols, components, code-libraries and other implementations

Definition of the CVII Technology Stack means to agree on the chosen technologies, and/or to develop those technologies.

- By *specification*?
- By *implementation*?

Model vs. Catalog



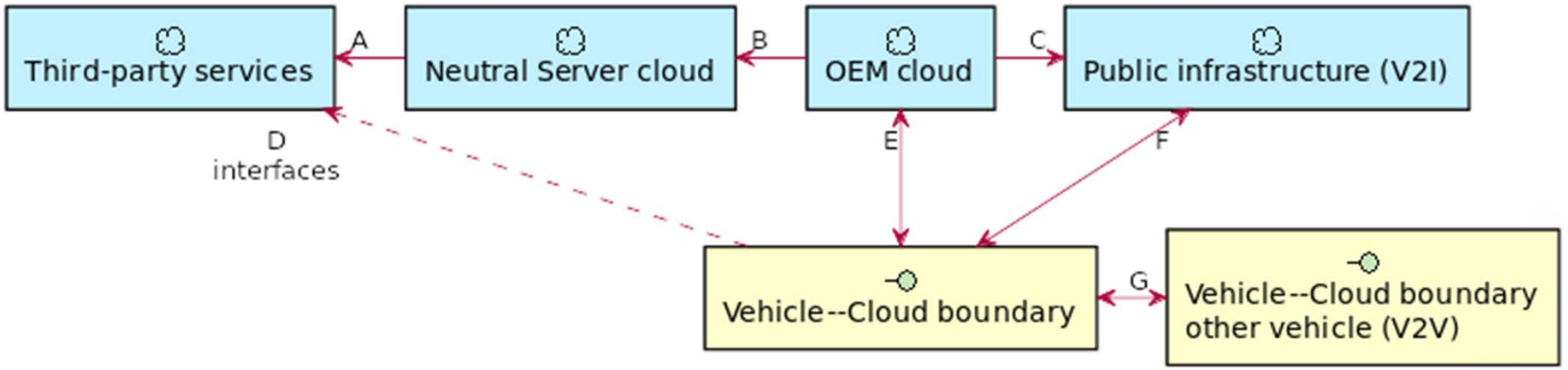
VSS* as common data model is prepared to support both the standard catalog *and* extension catalogs (even proprietary) to facilitate growth and innovation

The actual *model* is however always common.

→ This is required for the *technology stack* to be shared, and for a successful interconnection of parts across the whole system and industry. The common model guarantees that there is a common understanding among industry partners.

*(*the same applies for VSC of course)*

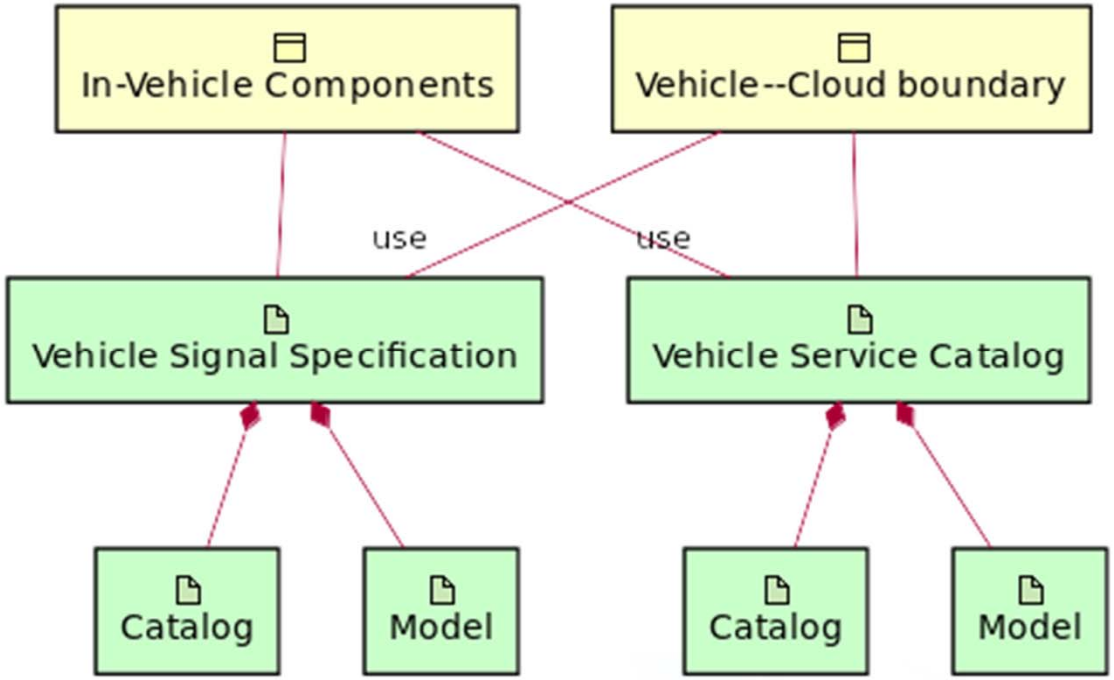
CVII – usage and impact (1)



"Typical" cloud architecture

Which interfaces are most appropriate for industry standards?
Which interfaces does your company want to focus on?
How do we best organize the responsibility between *projects & consortia*?

CVII – usage and impact (2)

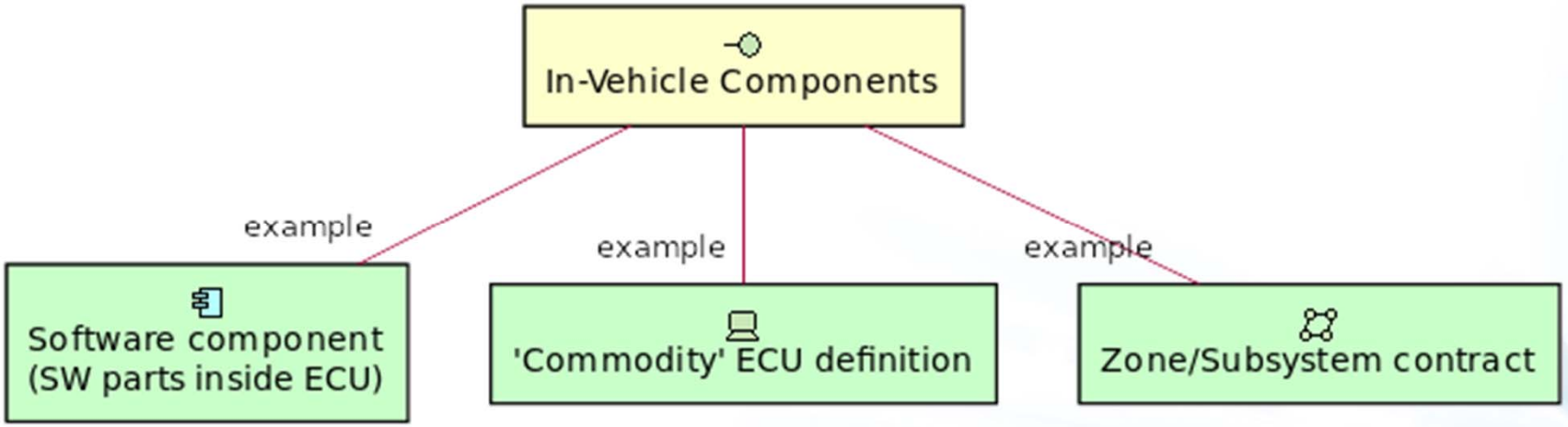


VSS and VSC have the most impact if used both inside and outside the vehicle.

CVII – usage and impact (3 – inside vehicle)

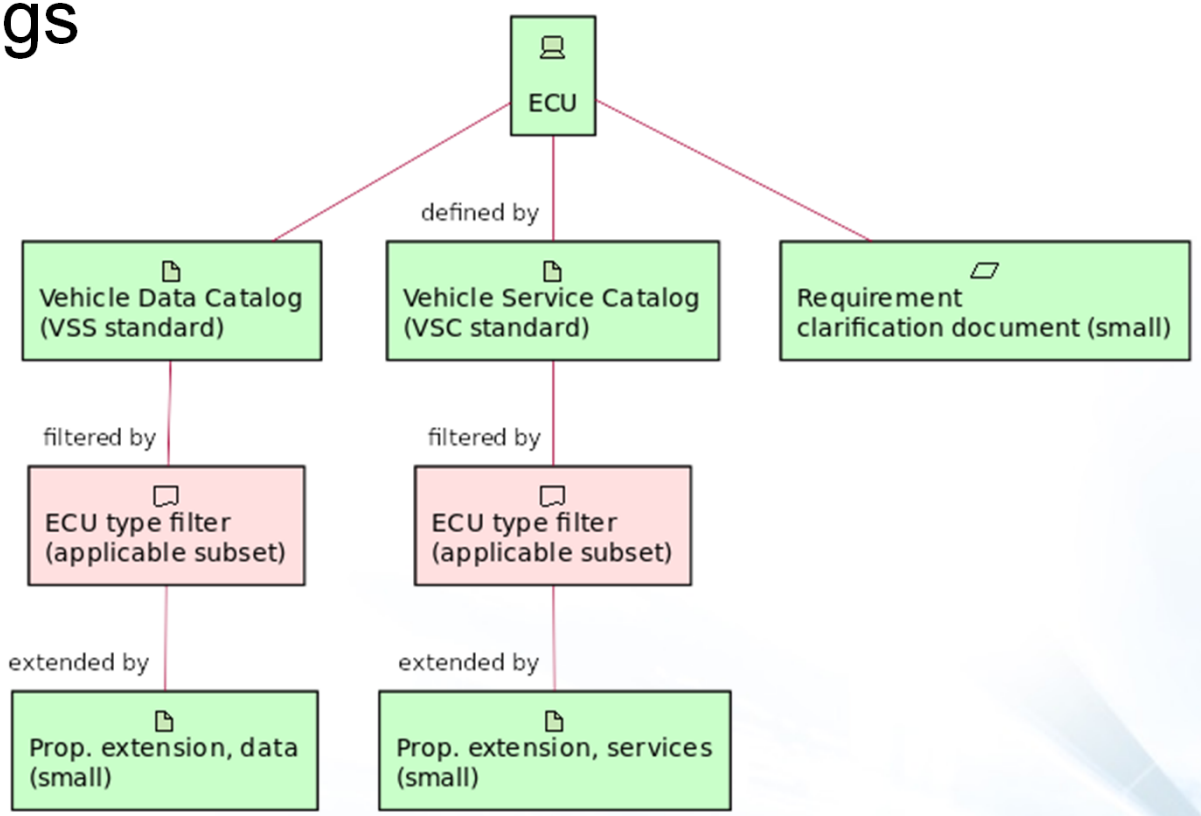
There is the potential to define the electrical architecture using “data” and “services”.

If a common data/service model is used → huge synergy across the development ecosystem (OEM/Tier1/TechSupplier/Innovator/)



CVII – usage and impact (backup)

Potential to define ECUs using standardized data and service catalogs



Next Steps

- Half day workshop is planned on 27 October (aligned with GENIVI virtual event)
  - Further scoping and work breakdown is the primary theme
  - All invited (including your in-vehicle and cloud experts) and expect an email from Steve with logistics next week
  - Outcome of workshop (and related W3C meetings) to be consolidated in a charter document describing scope, work ownership and execution plans
- Comments (e.g., answers to scoping questions or any other input) welcome to gandersson@genivi.org)
- Statements of support welcome today or to scrumb@genivi.org and ted@w3c.org

Thank you!

Visit GENIVI:

http://www.genivi.org

http://projects.genivi.org

Contact us:

help@genivi.org

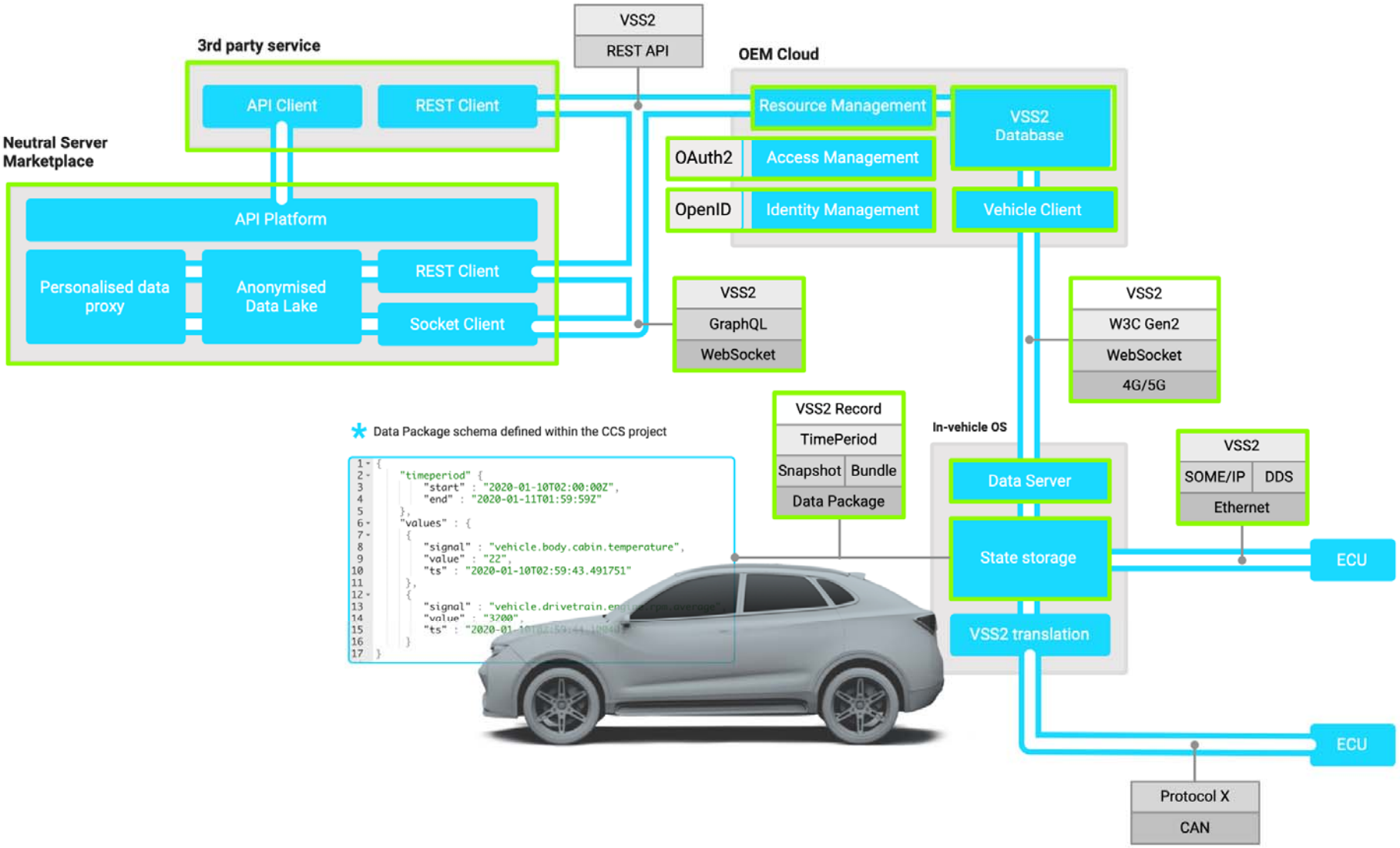
GENIVI®

BACKUP SLIDES

Why is VSS (VSC) the appropriate “source format”

- The selected format should be a plain-text, line-oriented format, like VSS:
  - Any text editor – no special tooling for editing
  - A line-oriented format reads like a normal logical hierarchical document would be written
  - Is easy to read and write, also by non-programmers. ← **MOST IMPORTANT FACTOR**!
  - A line-oriented format (YAML) promotes handling VSS like source code (version control in git repository, patches, change history, ...), unlike “syntax-oriented” (JSON, XML, …) which can be less convenient.
  - Still has formal meaning, machine-processable input, and easy to write and extend tools.
- **VSS** is plain-text as stated above, has *several years of development*, and includes:
  - Easy, obvious hierarchical model
  - Simple and recognizable data types
  - Reasonable modelling power for signals/data (e.g. instantiation)
- **VSS is extensible using VSS-layers.** Adds unique deployment information for different target environments. Adds your own (proprietary) signals and/or potentially local modification to common catalog when it is strictly required
- VSS is only the **source** and **model** of our information
  → Proven conversion to other formats (including GraphQL, JSON, XML, Franca IDL,)
- **W3C** develops official protocols for web-access to data with **VSS as the primary data model**

Communication Framework draft v5



Vehicle Signal Specification (VSS)

- Developed for around 4-5 years already
- **Common Data catalog** (std. signals, organization, taxonomy (hierarchy))
- **Common Data model** (description syntax, rules, datatypes, semantics)
- Related: **Partial Technology stack** (software, translations, protocols, bindings)
- **Best? choice for a common standard:**
  - Simple, text based, line-oriented (YAML)
  - Easy to read and write, easy to program
  - Simple datatypes and careful feature selection (no feature creep)
  - Open-source (spec, catalog and software). Open development process.
- What about alternatives?
- → The common VSS standard is only the source format (and defined behavior)
  ”Translations” are simply part of a technology stack setup.
- What other standards would otherwise apply?