COVESA

MORITZ NEUKICHNER

Navigating the Intersection of Technology, Open-Source, and Business for the Software-Defined Vehicle

#WhatTheSDV

Is this software-defined?

This is software-defined!

Software-defined means that you can change the key functions or nature of device by changing its software.

You can re-define the value of the device.

Bring value to the customer as quickly as possible

No matter how often you sell a new vehicle

Making vehicles software-defined

Decouple lifecycles of software and hardware

The software must evolve independently of the hardware. Updating software cannot mean upgrading hardware.

Then speed-up software delivery as much as possible.

Deliver new value in existing vehicle

Software updates provide new value. Software-defined vehicles are not only about easing maintenance but about providing new functions and services to existing devices.

Capture value of software

When software updates provide new value independent of hardware sales, software-defined vehicles must enable to monetize the value delivery.

All three go hand-in-hand

Decouple lifecycles of software and hardware

Would Apple be able to maintain the iOS ecosystem if each app would be bound to an iPhone generation?

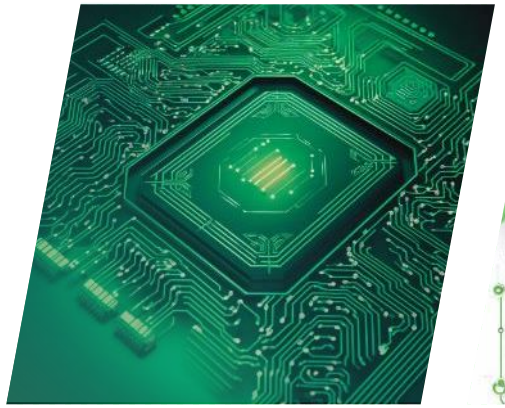
Deliver new value in existing vehicle

Would you buy an iPhone today, if it did not have an app store?

Capture value of software

Would Google have developed Android without the possibility to monetize on data or the app store?

Think beyond existing markets…





The **in-vehicle software** market

- Traditional market
- Transformation for Tier 1s and Tier 2s to long-term development and maintenance partnerships

The **software tools** market

- Traditional market
- Tools and infra-structure for software develop-ment transforming towards collaboration and virtual integration

The ecosystem of **in-vehicle app-stores**

- Replicate the app-store success of iOS and Android in the car
- Monetize on function sales and customization

The ecosystem **around the vehicle**

- Open APIs and data of the vehicle to build services around it
- E.g. pay-per-use vehicle insurance, delivery-to-trunk services

The ecosystem of **vehicle fleets**

- Manage vehicle fleets, fleet data
- E.g. Remote diagnostics, fleet supervision, fleet data aggregation and interpretation

Your value proposition must define what you want to control

We farmed out all the modules that control the vehicles to our suppliers because we could bid them against each other. [...] The problem is that the software is all written by 150 different companies and they don't talk to each other. [...] One-hundred-fifty completely different software programming languages. All the structure of the software is different. It's millions [of lines] of code.

That's why [..] we've decided [...] to completely in-source electric architecture. [...] To do that you need to write all the software yourself.

Jim Farley (CEO Ford) in Fully Charged, 2023


The problem is not ownership.
It is the ability and right to change and steer.

What are your priorities?

- Do you care about intellectual property?
- Is public ownership sufficient not to rely on a single external partner?
- Is proprietary ownership of a group of companies sufficient?
- Is an open interface specification sufficient?

- What is the governance of the software?
- Do I need to decide on the features?
- Do I only need to prevent exclusive governance of a single party?
- Is the right to fork sufficient to ensure freedom to change in the future?

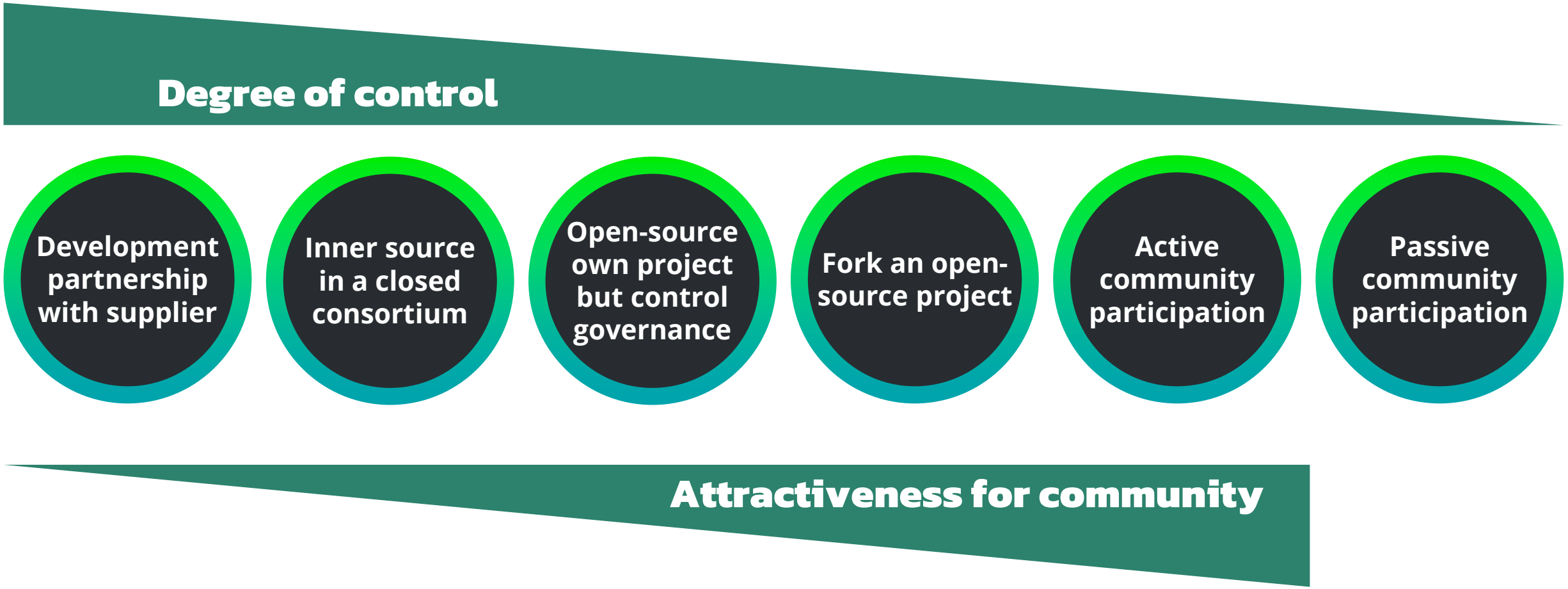
- Do I need to contribute code with my own teams?
- Is it sufficient to have a supplier making contributions in my name?
- Does my organization require the capabilities to change the software?

Let's play a thought game

- I am an OEM and want to develop a software platform and genuinely want to collaborate

- I want to
  - Deliver new differentiating functions to existing vehicles – exclusive control of an app ecosystem is not my goal
  - Bring the software into production and therefore have
    - Quality requirements (e.g. HIS, MISRA)
    - Process requirements (e.g. ASPICE)
    - Maintenance requirements (e.g. UNECE R.155 – 15 years security maintenance)
    - License requirements (e.g. No GPL 3.0)
  - participate in a community as I value other's contributions
  - Want to prevent a lock-in of a single vendor as I will operate complete fleets of vehicles that I want to be able to upgrade for 5-10 years

Modes for collaboration



Your contributing.md – example from eclipse sdv

Contributing

Welcome to the Ankaios community. Start here for info on how to contribute and help improve our project.

How to Contribute

[No Title]

This project welcomes contributions and suggestions. You'll also need to create an Eclipse Foundation account and agree to the Eclipse Contributor Agreement. See more info at https://www.eclipse.org/projects/handbook/#contributing-contributors.

If you have a bug to report or a feature to suggest, please use the New Issue button on the Issues page to access templates for these items.

Code contributions are to be submitted via pull requests. For this fork this repository, apply the suggested changes and create a pull request to integrate them. Before creating the request, please ensure the following which we will check besides a technical review:

- **No breaks**:
  - *Builds:* All builds pass (GitHub actions).
  - *Tests:* The unit tests still pass.
- **Docs updated**: Make sure any changes and additions are appropriately included into the documentation
- **Requirements**: Make sure that requirements are created for new features and those are traced in the code and tests.

Communication

Please join our developer mailing list for up to date information.

Your contributing.md – example from eclipse sdv

Contributing

Welcome to the Ankaios community. Start here for info on how to contribute and help improve our projec

How to Contribute

[No Title]

This project welcomes contributions and suggestions. You'll also need to create an Eclipse Foundation acc
Contributor Agreement. See more info at https://www.eclipse.org/projects/handbook/#contributing-contr

If you have a bug to report or a feature to suggest, please use the New Issue button on the Issues page to

Code contributions are to be submitted via pull requests. For this fork this repository, apply the suggested
integrate them. Before creating the request, please ensure the following which we will check besides a tech

- **No breaks**:
  - *Builds:* All builds pass (GitHub actions).
  - *Tests:* The unit tests still pass.
- **Docs updated**: Make sure any changes and additions are appropriately included into the documentati
- **Requirements**: Make sure that requirements are created for new features and those are traced in the

Communication

Please join our developer mailing list for up to date information.

This means:

- Open test assets
- Tests executable by anyone
  i.e. no proprietary tooling or open access to CI/CD infrastructure
- For HIS or MISRA compliance this means meeting the metrics

Alternative:

- Branch and retrofit compliance for community contributions running the risk of diverging solutions

Your contributing.md – example from eclipse sdv

Contributing

Welcome to the Ankaios community. Start here for info on how to contribute and help improve our projec

How to Contribute

[No Title]

This project welcomes contributions and suggestions. You'll also need to create an Eclipse Foundation acc
Contributor Agreement. See more info at https://www.eclipse.org/projects/handbook/#contributing-contr

If you have a bug to report or a feature to suggest, please use the New Issue button on the Issues page to

Code contributions are to be submitted via pull requests. For this fork this repository, apply the suggested
integrate them. Before creating the request, please ensure the following which we will check besides a tech

- **No breaks**:
  - *Builds:* All builds pass (GitHub actions).
  - *Tests:* The unit tests still pass.
- **Docs updated**: Make sure any changes and additions are appropriately included into the documentati
- **Requirements**: Make sure that requirements are created for new features and those are traced in the

Communication

Please join our developer mailing list for up to date information.

This means:

- Open design, requirements and documentation assets
- No proprietary tooling for writing architecture, requirements, and design
  (e.g. OpenFastTrace, ArchE)
- Community reviews must check for compliance

Alternative:

- Branch and retrofit design running the risk of diverging solutions

Your LICENSE.md – example from Apache License 2.0

5. Submission of Contributions. Unless You explicitly state otherwise,
   any Contribution intentionally submitted for inclusion in the Work
   by You to the Licensor shall be under the terms and conditions of
   this License, without any additional terms or conditions.

8. Limitation of Liability. In no event and under no legal theory,
   whether in tort (including negligence), contract, or otherwise,
   unless required by applicable law (such as deliberate and grossly
   negligent acts) or agreed to in writing, shall any Contributor be
   liable to You for damages, including any direct, indirect, special,

This means:

- We are solely responsible for the software: technically, IP-wise, commercially, regulatory, ...

Alternative:

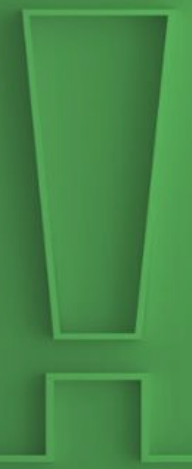
- Buy maintenance with corresponding liability for the duration of the lifecycle of your product
  e.g. Extended Security Maintenance

Most important.

Agree on project goal and guidelines.

You can have governance and license setup
and fail because you do not share a common goal.

e.g. “We do not break userspace!”



The software-defined vehicle

… is not about

Just adding over-the-air updates

Just about adding connectivity

Just about have more software in a vehicle

It is about

Time-to-market for software innovation

Decoupling lifecycles of parts that inherently have different lifetimes

Changing the understanding of ownership and collaboration

Changing development practices

Changing business models