LET’S CO-CREATE THE: Software Defined Vehicle

Why vehicles need an event based system?

Thomas Spreckley
SW consultant - Bosch Connected Mobility Solutions
Storytime....... Wake-Up!
Gandhi’s Shoes

Q: What would you do in this exact situation?
Setting the context in Automotive

why bother?

• Focus is on few centralized, powerful vehicle computers... *(Zonal Arch.)* ...
  ... that implement & integrate higher-level vehicle functions

• General shift from mechanical complexity to software complexity

✓ Software becomes the majority-asset in the product value chain

1

How it’s going

- No space 4 QM
- No carry-over
- Integration is hell!
- Hardware dependence

2

What’s next?

+ Re-usable code
+ Standardized data
+ Hardware de-coupled
+ Faster development cycles

Centralized compute architectures
Event-Driven Architecture (EDA)
so what is it exactly?

Typically Software solutions today have to handle the challenges created by always-on, multi-user, asynchronous distributed apps.

- **Loosely coupled** ...
- **Scalable** ...
- **Extensible** ...
- **Reactive** ...
- **Fault tolerance** ...
- **Recovery** ...

---

**Basic concept**

- **Producer 1**
  - A
  - Event broker
  - **Consumer 1** subscribed to A

- **Producer 2**
  - B
  - Event broker
  - **Consumer 2** subscribed to A + B
  - **Consumer 3** subscribed to B

---

**Micro-Service implementation**

Event-driven systems are often implemented following a Micro-service design.

---

**Distributed Data**

Event-driven systems can handle this extremely effectively as they can rely upon multiple & distributed “sources of truth”

---

**Distributed Systems**

This means that the system components can be distributed & deployed across networked computers/VCU’s/ECU’s
Event-Driven Architecture (EDA)

theory in practice

Event Storming Technique
EDA lends itself very well to achieving complex “multi-node” operations that often occur asynchronously & depend on multiple databases across multiple deployments for example: (eg VCU 1 <-> VCU 2 <-> OPS cloud <-> 3rd Party cloud)
Event-Driven Architecture (EDA)

who is using it... "Everyone in IT!"

Challenges of multiple platforms Support
How to support multiple OS’s & hardware configurations with a simple Micro-service design & repeatable app behavior

Netflix API & architecture re-structure to enable scaled growth

Typical Monolithic application

- All services have equal Priority
- Products are simply recipes of services

Everything as a Service (EaaS)
Event-Driven Architecture (EDA)

Example project

IoT-event-analytics is a stream processing and complex event processing (CEP) engine that can be used with default transport protocol of MQTT & various data models

- Core Engine written in NodeJS
- Containerized deployment
- Standardized vehicle data model
- Complex rule-sets analytics
- Event History functionality
- Scalable throughput & performance
- Transport Protocol abstraction

Data model + Runtime-core + SDK support + Container tech

- Open Source
- BOSCH
A: One shoe is no use to anyone – why do things alone!?
A portfolio of open source components and activities, eg:

- w3c/GENIVI IoT Event Analytics: [https://github.com/GENIVI/iot-event-analytics](https://github.com/GENIVI/iot-event-analytics)
- w3c/GENIVI IoT Vehicle Edge: [https://github.com/GENIVI/vehicle-edge](https://github.com/GENIVI/vehicle-edge)
- NPM package: @genivi/iotea-js-sdk
- pypi package coming soon!
- Pre-built RPi4 & Ubuntu & Apertis images coming soon!
- Complete VScode dev-env. integration coming soon!
NEXT: CVII Session 3
Alignment and Adoption
SOTA and Insurance