JSR xxx – A new telematics API?

hendrik.hoefer@microdoc.com
Connected Car: Some observations

• Telematics business models are quite successful in commercial environments (e.g. Logistics, Remote Maintenance), but have a slow adoption in the consumer space

• Most OEMs are building their own, mainly proprietary app – ecosystem to increase the value of their products

• OEMs consider Google and Apple to be their competitors

• OEMs are very concerned about safety/privacy/data ownership and want to/must control access to the safety critical components and private data in a vehicle
The amount of software in modern cars is astonishing

http://www.informationisbeautiful.net/visualizations/million-lines-of-code/
A typical Telematics Component Architecture

- **Trusted Domain**
  - Trusted Services
  - Java VM
  - OS (Linux/QNX/...)

- **Sandbox Domain 1**
  - Untrusted Services
  - Java VM
  - OS (Linux/QNX/...)

- **Sandbox Domain N**
  - Untrusted Services
  - OS (Linux/QNX/...)

- **Virtualization** (e.g., Hypervisor, Docker,...)

- **Realtime Domain**
  - Diagnostics
  - Message Buffering
  - Power Mgmt
  - ...

- **Realtime OS / CAN Stack**

Optional

Comm. Bus

IPC Protocol
“This specification defines methods for controlling and obtaining diagnostic information and conditions on various components built in to the vehicle.”

<table>
<thead>
<tr>
<th>Interface name</th>
<th>Component name</th>
<th>Possible Operations</th>
<th>States or Value [Unit]</th>
</tr>
</thead>
<tbody>
<tr>
<td>AntilockBrakingSystem</td>
<td>AntilockBrakingSystem</td>
<td>On/Off</td>
<td></td>
</tr>
<tr>
<td>Airbag</td>
<td>DriverAirbag, PassengerAirbag</td>
<td>- Set Mode</td>
<td>Enabled/Disabled, Fired off/Not Fired off</td>
</tr>
<tr>
<td>AirConditioning</td>
<td>AirConditioning</td>
<td>Turn on/off</td>
<td>On/Off</td>
</tr>
<tr>
<td>Antenna</td>
<td>Antenna</td>
<td>Extend/Retract</td>
<td>Extended/Retracted</td>
</tr>
<tr>
<td>Battery</td>
<td>Battery</td>
<td>-</td>
<td>Electricity Level [Voltage]</td>
</tr>
<tr>
<td>Brake</td>
<td>Brake</td>
<td>-</td>
<td>Engaged/Not Engaged</td>
</tr>
<tr>
<td>BrakeFluid</td>
<td>BrakeFluid</td>
<td>-</td>
<td>Fluid Level [Liter]</td>
</tr>
<tr>
<td>ChargingPlug</td>
<td>ChargingPlug</td>
<td>Plug Out/In</td>
<td>Plugged/Unplugged</td>
</tr>
<tr>
<td>DashboardIllumination</td>
<td>DashboardIllumination</td>
<td>Level set</td>
<td>Illumination Level [Level]</td>
</tr>
<tr>
<td>DifferentialOdometer</td>
<td>DifferentialOdometer</td>
<td>Reset</td>
<td>0 (Zero)</td>
</tr>
<tr>
<td></td>
<td>DriverDoor, PassengerDoor</td>
<td>Lock/Unlock</td>
<td>Locked/Unlocked</td>
</tr>
<tr>
<td></td>
<td>Engine</td>
<td>Start/Stop</td>
<td>Running/Stopped</td>
</tr>
</tbody>
</table>
• What might have hindered adoption also is
  – “For TCK we will charge a single one time fee of max $50000 USD and an annual maintenance fee, max $20000 USD/pa.”
• This JSR might be useful to build a telematics product within an organization, but as an API for 3rd party code it seems too “broad”
A very successful business model in the consumer space

- [http://www.octotelematics.com](http://www.octotelematics.com)
  - “Octo is the global brand leader in providing insurance telematics services as well as pioneering applications in motor rental and fleet management, car manufacturing, governmental sectors and a fast growing range of specialist applications.”
  - Octos solutions (hard- & software) run in several million vehicles
- Octo wants to help standardize (some) in-vehicle APIs
Technical scope for Version 1 of a new Telematics JSR

• APIs for consuming data
  – Derived from OBD, FMS standards
  – Speed, Accelerometer, GPS, Gyro, Ignition, etc.

• Other APIs
  – Lifecycle (startup/shutdown/suspend/reset)
  – Configuration
  – Persistence

• APIs for sending/receiving data to/from backend
  – Likely in version 1

• API HMI integration
  – Unlikely in version 1
A typical Telematics Component Architecture

- **Realtime Domain**
  - Diagnostics
  - Message Buffering
  - Power Mgmt
  - ...

- **Realtime OS / CAN Stack**

- **Application Domain**
  - Trusted Domain
    - Trusted Services
    - Java VM
    - OS (Linux/QNX/...)
  - Sandbox Domain 1
    - Untrusted Services
    - Java VM + JSR xxx
    - OS (Linux/QNX/...)
  - Sandbox Domain N
    - Untrusted Services
    - OS (Linux/QNX/...)

- **Virtualization** (eg Hypervisor, Docker, ...)

- **Optional**

- **Comm. Bus**

- **IPC Protocol**

- **MicroDoc**
What are we doing right now?

- Working through the legal framework (JSPA)
- Define the technical scope for this JSR
- Work on the the JSR proposal
- Find the right people for the expert group