What is Automotive Software Engineering?

- Today the engineering of software in cars is still in its infancy
- Lifecycle management of software in cars is in its early stage.
- Many suppliers and even some OEMs are not even at Capability maturity model level 2.
- Reuse of solutions from one car to the next is insufficient and only done in a consequent way in some limited areas.

In many sub-domains the functionality from one car generation to the next is only changed and enhanced by 10% while more than 90% of the software is rewritten.

- The reason is a low level, hardware specific implementation, which makes it difficult to change, adopt, and port existing code.
- Finally, the amount of automation in software production for software in cars is quite low.

What is Automotive Software Engineering?

- System: comprises a group of interacting components that is separated from its surroundings
  - status: a collection of properties at a point in time to describe the system
  - periphery: describes a grouping of components and their status that are not but part but influences the system status
  - boundary: border between system and periphery
  - interface:
    - input: inbound direction of data (signals)
    - output: outbound direction of data (signals)
  - subsystem: a system may be a component of a larger system and/or may contain subsystems itself

Rule of thumb:
- Systems with more than 7+2 = 9 components are considered complex
- Systems with less than 7-2 = 5 components are considered trivial
- Systems consisting of 5 to 9 components are manageable
What is Automotive Software Engineering?

- Process models:
  - Capability Maturity Model Integration® (CMMI)
  - Software Process Improvement and Capability Determination (SPICE)
  - V-Model (plaatje toevoegen)
- Standards:
  - ISO26262
  - MISRA-C standard
  - AUTOSAR

What is Automotive Software Engineering?

- Configuration management
  - Product and Life Cycle
    - development
    - production
    - operation and service
  - Variants and Scalability
    - components variants
    - scalable system architecture
  - Versions and Configurations
    - versions of a configuration evolve over time
    - separation of hardware characteristics from software

What is Automotive Software Engineering?

- Configuration management facilitates the administration of relations between systems and components
- Managed items:
  - Requirements
  - Specifications
  - Implementations, such program versions and data versions
  - Description files, such as for diagnostics, software updates, and software parameterization
  - Documentation

What is Automotive Software Engineering?

- Project management, because
  - Limitation in time
  - Limitation in resources
  - Fine-tuning of organizational structure to suit objectives
  - Cross references with other projects
- Objectives:
  - Quality
  - Time (deadlines/milestones)
  - Resources
  - Costs = Time + Resources
What is Automotive Software Engineering?

Project planning
- Phases:
  - definition
  - planning
  - implementation
  - completion
- Quality planning
- Cost planning
- Project scheduling

Project planning (continued)
- Development roles and Responsibilities

<table>
<thead>
<tr>
<th>Role</th>
<th>Responsibilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>Function development</td>
<td>Analysis of user requirements, specification of logical system architecture</td>
</tr>
<tr>
<td>System development</td>
<td>Analysis of logical system architecture</td>
</tr>
<tr>
<td>Software development</td>
<td>Analysis of software requirements, plus software specification, design, implementation and testing</td>
</tr>
<tr>
<td>Hardware development</td>
<td>Analysis of hardware requirements, plus hardware specification, design, physical implementation and testing</td>
</tr>
<tr>
<td>Peripheral development</td>
<td>Analysis of peripheral specific requirements, plus their specification, design, physical implementation and testing</td>
</tr>
<tr>
<td>Integration, testing and calibration</td>
<td>Integration, testing and calibration of onboard vehicle systems and their functions</td>
</tr>
</tbody>
</table>

Subcontractor management
- System and component responsibility
- Interfaces for specification and integration
- Defining the cross-corporation development process

Requirements management

Activities Common to Software Projects
- Requirements and specification
  - Domain analysis
  - Defining the problem
  - Requirements gathering
    - Obtaining input from as many sources as possible
  - Requirements analysis
    - Organizing the information
  - Requirements specification
    - Writing detailed instructions about how the software should behave
Activities Common to Software Projects

• Design
  • Deciding how the requirements should be implemented, using the available technology
  • Consists of:
    − Systems engineering: Deciding what should be in hardware and what in software
    − Software architecture: Dividing the system into subsystems and deciding how the subsystems will interact
    − Detailed design of the internals of a subsystem
    − User interface design (probably less important)
    − Design of databases (probably less important)

• Modeling
  • Creating representations of the domain or the software
    − Use case modeling
    − Structural modeling
    − Dynamic and behavioral modeling

• Programming
  • Quality assurance
  • Reviews and inspections
  • Testing
  • Deployment
  • Managing the process