The Struggle of SDVs with Traditional Automotive Architectures

Chaitanya Podalakuru
Andrew Steinman

Oct 2024

About Us

Chaitanya Podalakuru

- Vehicle Network Architect – In-Vehicle Networking & Architecture Tools at Ford
- Automotive engineer with expertise in embedded software development, integration, and architecture of ECU

Andrew Steinman

- Manager of In-Vehicle Networking & Architecture Tools for Ford Motor Company's Architecture and Platforms Organization
- System Engineer with expertise in vehicle electrical, software, and communications

Agenda

01 – SDV Characteristics and Enablers

02 – Overview of AUTOSAR Centric Workflows

03 – Challenges

04 – Future Outlook

01

Software Defined Vehicles (SDVs)

"SDVs are not simply cars with more software;
they are vehicles fundamentally defined and shaped by their software,
creating a new paradigm in automotive engineering and the user experience"
[1]

[1]Google Gemini Flash, Ford Large Language Model (LLM), 11 October 2024

Key Characteristics of a Software Defined Vehicle

- **Software First Approach**
  - Legacy architectures primarily focused on getting the hardware exactly right.
  - The focus in SDVs is more on the right software architecture that can enable ever changing experiences on similar or the same hardware
- **Data Availability**
  - Exposure of the atomic data and control across the vehicle to enable a wide range of current and future experiences rather than existing feature "verticals"
- **Updatability**
  - Over the Air Updates for bugfixes, security patches and new features
- **Scalability**
  - Easily being able to add or subtract pieces from various sources without impacting the whole system design
- **Decoupled Platforms**
  - Decouple lifecycles of Software platform and Hardware platform
- **Innovation on Wheels**
  - Open-up channel for customization with support for third-party apps.

SDVs generally have vastly different characteristics than that of EE architectures of the past

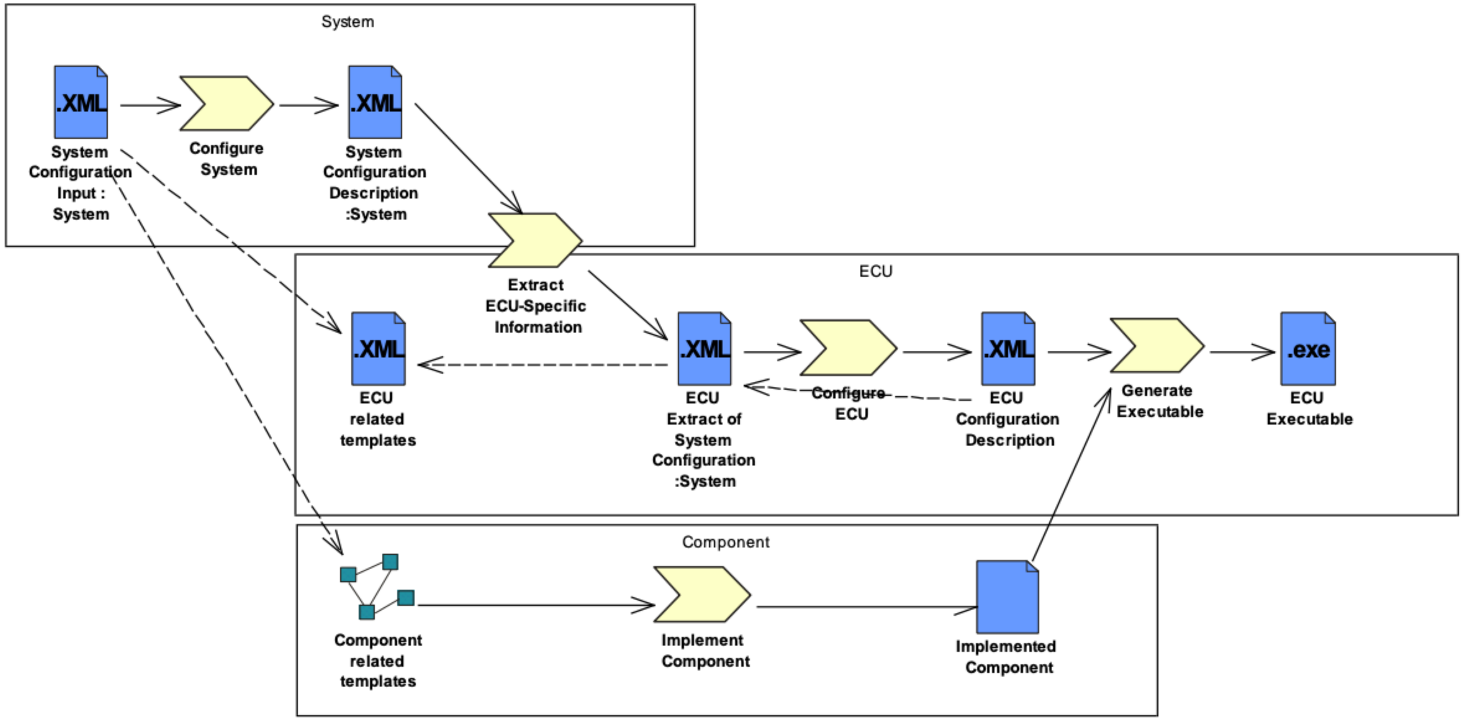
SDV Enablers	Details	Typical Embedded Systems Support?
Excess Performance Capacity	CPU (DMIPS, Hz, etc.) RAM / ROM Flash Communication bandwidth **Trade-off** = Cost	- Typically, highly constrained or optimized for Job1 - Each supplier or OEM may have multiple slightly different optimizations which cannot be easily applied in other systems
Full Stack Software Flexibility	How easy is it to move / add / delete? Does the whole system break if I touch one sub-system **Trade-off =** initial design time / cost	- AUTOSAR components may be portable w/ proper planning - Limited to Zero sharing across OEM / Supplier boundaries while systems are highly mixed
Communication Standard	Are all the ECUs, SWCs, Suppliers, OEMs following API standards facilitating re-use? **Trade-off =** cost to drive standard	- J1939 common in certain limited domains - SOME/IP (and similar) require somewhat static system specific definition (Classic) - Service implementations highly OEM specific
Back / Forward Compatibility	**Forward** – can my old system work with new components (anticipate change) **Backward** – can my new system work with old components (handle known change) **Trade-off =** compute, bandwidth, design cost	- High trade-off cost: Maintaining legacy systems vs. moving the industry forward - Highly cost sensitive systems, resistive to protecting for future "maybe situations"
Iteration & Continuous V&V	Can my system be iterated at the interface, component and whole system level to enable software developers? Does the toolchain support sustainable continuous testing to meet the increased iteration **Trade-off =** tool and talent investment	- Tool chains have limited often proprietary automation with convoluted APIs - Verification and Validation at the vehicle level often requires a large waterfall multi-week process (no-failing fast)

What can the Industry do to better address these?

02

Overview of AUTOSAR centric workflows

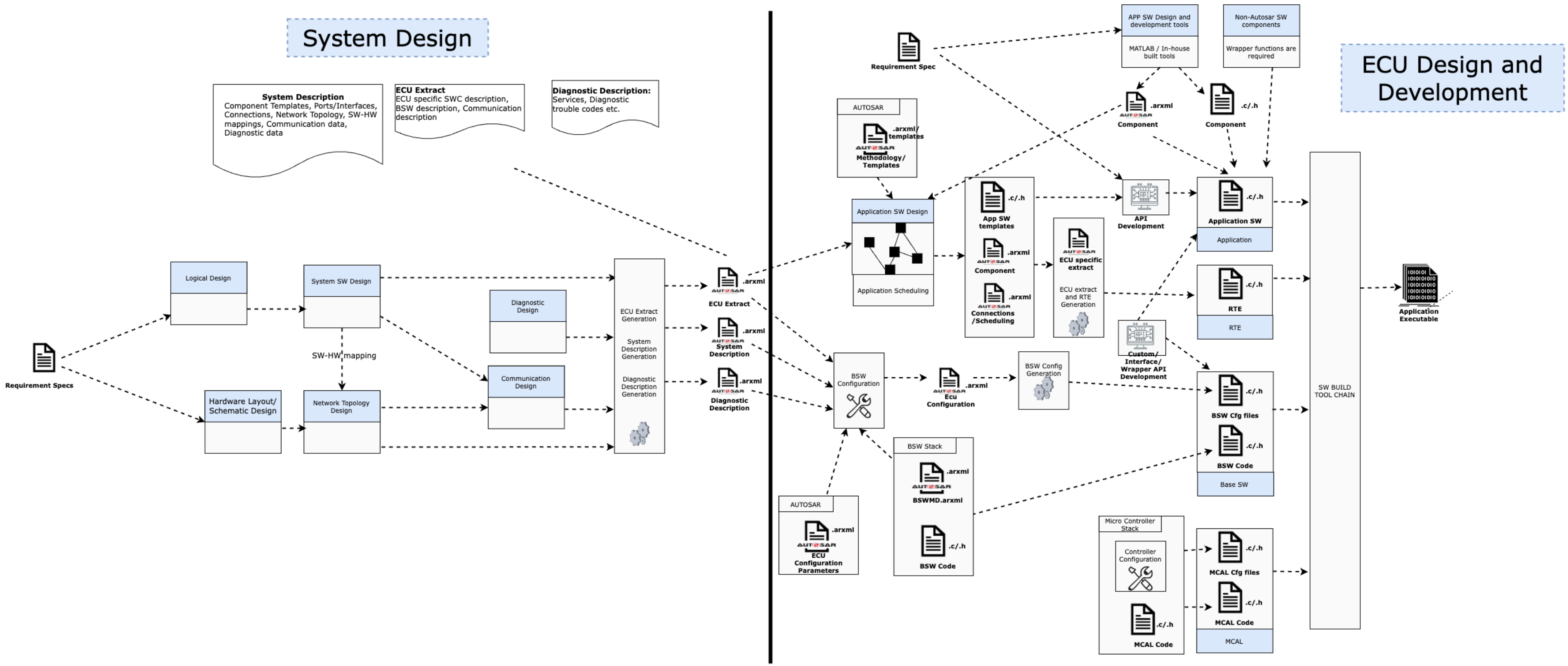
AUTOSAR Methodology and Industry Tools for Architecture Design



Reference: AUTOSAR_EXP_VFB.pdf from release AUTOSAR R21-11

GUI/ model-based vendor tools are commonly used by OEMs and suppliers to design and simulate system architectures.

Typical Workflow – Requirements to Binary



03

Challenges

Transition from network-centric to software-centric architectures

Electrical Architecture Design

- Re-architecture Complexity
- Complexity in normalizing hardware variants
- Difficulty in availability of architecture data to analyze performance or resource availability of ECUs

System Software Design

- Non-standardized software interfaces
- Software function knowledge is confined to suppliers.

Communication Design

- Bandwidth Limitations and Latency Constraints
- Resistance to update legacy Signal based PDU Packing
- Functional Safety and Cyber-Security Aspects
- Backward Compatibility

Vehicle Integration Testing

- Difficulty in alignment of ECU readiness for vehicle integration testing

Cost and Timing

- Longer OEM and Supplier integration cycles
- Development cost

Challenges in Traditional AUTOSAR Authoring Solutions

Configuration Management

- Heavily model based
- Vendor tool-specific

Modern Practices

- GUI-based tooling - inefficient and prone to user error
- Underutilization of modern software development practices

Integration Issues

- Late discovery of dependencies
- Limited automation and test capability

Cost Concerns

- Licensing cost
- Training cost

05

Future Outlook

Future Outlook in reshaping Architecture Design Approach to build SDVs

Hardware Design

- Focus on analyzing and reducing Classic ECUs by leveraging combined zonal and compute systems.

System Software Design

- Need for standardization of software interfaces and software functions.

Communication Design

- Standardize network interfaces to ECUs coupled with critical sensors and actuators.
- Revise traditional network rules by incorporating concepts such as signal deprecation policy and automated PDU-packer.

AUTOSAR Authoring Infrastructure and Data Management

- Pursue modern solutions for authoring AUTOSAR architectures
- Implement unified DevOps practices with OEMs and Suppliers in-loop for CI/CD.

Vehicle Integration Testing

- Adopt ECU virtualization to accelerate development and testing processes.

Creating a SDV not just about adding more software to a vehicle; it's about developing an entire ecosystem.

Q & A