Automotive Software Systems
Complexity: Challenges and Opportunities

INCOSE International Workshop
MBSE Workshop
January 26th-28th, 2013

Christopher Davey
Senior Technical Leader
Software & Control Systems Engineering
Global EE Systems Engineering Group
Ford Motor Company
Presentation Outline:

1. Ford Motor Company & EE Systems

2. Automotive EE & Software Complexity Challenge and Opportunity

3. A Paradigm Shift: Fully Integrated Systems Engineering
   1. MBSE
   2. Model Based Architectures
   3. Why, When, Where & How

4. Model, Information, Intellectual property re-use: The need for enterprise wide - PLM/ALM Solution

5. Lessons Learned so far.....
Why do we need all these software Systems?

<table>
<thead>
<tr>
<th>Past</th>
<th>Present</th>
<th>Future</th>
</tr>
</thead>
<tbody>
<tr>
<td>Propulsion Systems</td>
<td>CARBURETTOR</td>
<td>HIGH EFF. FUEL INJ.</td>
</tr>
<tr>
<td>Safety &amp; Accident Prevention</td>
<td>AIRBAGS</td>
<td>LANE DETECTION</td>
</tr>
<tr>
<td>Human Machine Interface</td>
<td>RADIO</td>
<td>TOUCH SCREEN</td>
</tr>
</tbody>
</table>

- Technology Progression leverages Vehicle Software & Control Systems -> Proliferation
Properties of EE Software Systems

Enhanced Vehicle Software and Electrical Management

Capabilities are required to maintain the Robustness and Quality of Electrical Hardware Software and Control Systems as Feature Quantity & Complexity Grows

- > 80% of new features are Software enabled EE systems.
- > 70% of new systems are distributed in nature

CD Class Vehicle has approx.
- 50-70 Computers on board with multi-core SW process management
- > 10 MByte of Control Software
- > 15 Million LOC/Instructions
- > 5,000 Software Parameters
- > 50,000 functional requirements
- > 1,000,000 pages of specifications
- > 10,000 buildable Vehicle-Series-Variants (based on ECU component permutations per Vehicle line)

⇒ In some aspects a CD class vehicle has higher system, software and build complexity than a commercial aircraft....
General Trends in Vehicle Electronics

E/E Content

Electrical Design focus was on power Distribution and packaging-
Component Focused with local integrations

Electrical Design focus is now on Feature/Functional interaction
Complexity of software Relationships and Development of
Cross domain requirements – Feature focused

Higher Power Requirements

Software

Vehicle-to-Vehicle; Service; Web-Connectivity

Off-Board Communication & Connectedness

E-Connectivity

Consumer Electroic Devices Sync Module

Remote Diagnostics

Adaptive Headlamps

LSG

In Car PC

El. Water Pump

ACC

Infotainment

EPAS

Telematics

EM Valves

PTC Heater

IVDC

Displays

Steer-by-Wire

Blind Spot Detection

Brake-by-Wire

Fuel Cell

Adv. Restraints

ASM

Keyless Vehicle

Drive-by-Wire

Cluster

Body Elec.

ABS

P/T Electronics

Airbag

PA

General Trends in Vehicle Electronics

1880

1980

1990

2000

2010

CY
• Delivery of customer features requires the coordinated execution of multiple modules exchanging signals both wirelessly and over a number of multiplexed networks.

• >70% of all new customer features are software enabled and distributed in nature.

• Service Oriented Architectures (SOA) and well defined functional/software architectures are required to deliver efficient coordination between modules.
All supporting work products must be managed using a well defined Product Definition codification system and supporting naming conventions.
Overall automotive software complexity is growing rapidly

- Distributed software-system/controls solutions are growing rapidly
- System dependencies (coupling) are growing
- Functional safety related SW systems increasing
- Customer input signals are growing exponentially
- SW Lines of code (KLoC) is growing rapidly
• Mechanical Engineering requirements are typically less complex than EE-SW requirements and more stable
• EE-SW System requirement generate many decomposed levels of requirement-specifications that are more dynamic in nature.
• Software Systems Engineering bridges the gap between Traditional Systems Engineering and Software Engineering, it is an elaboration of the SE principles/practices.
• Release Configurations / Configurable Baselines

Key Process Area – Work Products

Versions

Requirements  Architecture  Implementation  Validation Verification  Calibration  ...

Variants

Releases: ⬝  ⬝  ⬝

Logical Schematic

Physical Schematic

EE Architecture

EE Physical Model

Courtesy of Reiner Busch, Ford Aachen Research Laboratories
In conventional engineering, we work in the linear zone where stress is linearly related to strain and the behaviour of the engineering system at ‘run-time’ is much more predictable. Software is in general, fully chaotic.

### Why do Model Based Systems Engineering?

#### Generic Software E/E Failure Mode Mapping:

<table>
<thead>
<tr>
<th>Work Product Objects</th>
<th>Failure Modes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Architectures</td>
<td>- Missing</td>
</tr>
<tr>
<td>Features</td>
<td>- Incomplete</td>
</tr>
<tr>
<td>Requirements</td>
<td>- Inconsistent</td>
</tr>
<tr>
<td>Interfaces</td>
<td>- Incorrect (content)</td>
</tr>
<tr>
<td>Parameters</td>
<td>- Incorrect Version / no versioning</td>
</tr>
<tr>
<td>Executable Models</td>
<td>- Inappropriate design</td>
</tr>
<tr>
<td>Test Vectors</td>
<td>- Unaligned (incorrect configuration)</td>
</tr>
<tr>
<td>Software Binaries</td>
<td>- Un-accessible (Global Engineering Centers)</td>
</tr>
<tr>
<td></td>
<td>- Non-Validated</td>
</tr>
<tr>
<td></td>
<td>- Duplicate and/or Conflicting</td>
</tr>
</tbody>
</table>

- Number of Critical-to-Quality work products and associated failure modes typically increase with: growth in SW-EE Feature content, growth of organizational interfaces, growth in global markets and re-use strategies.
Model Based Systems Engineering/Validation

Customer Requirement - Operational View – (MBA)

Vehicle Level Validation (Ford)

System Requirements - Logical View- (MBR)

Component Specifications - Physical View – (MBD)

Vehicle Level Testing

HiL System Testing

BreadBoard Testing

Model Based Component Testing

- Software Development/CMMi/SPICE
- Spiral
- Rapid development
- Agile
- Extreme

System Verification (Ford)

Component Creation/Testing (MBAutoCode/Test)

Escaped Software Defects Distribution

Number of Defects

Requirements Design and Implementation Integration Aftermarket

Requirements Design and Implementation Integration Aftermarket

Number of Defects

Requirements Design and Implementation Integration Aftermarket

Number of Defects
Traditional Textual Document Approach

- Paragraph
  - Requirement #1  (Text)
  - Requirement #2  (Text)
  - Requirement #3  (Text)

- Features
  - Visio Diagram

- Document
  - Levels
  - Requirement Objects
  - Manual Embedded Graphics
  - Manual Gen Test Cases

Use Case - Object Oriented Approach (UML)

(Example: Global Infotainment Systems)

- Use Case #1
- Use Case #2
- Use Case #3
  - Scenarios
  - Scenarios
  - Scenarios

Executable Models

MBSE – The MathWorks
Stateflow/Simulink

- Test Input
- Algorithm
- Expected Output
- Plant Dynamics Model
- Test Sequence/Harness

MBSE – Rhapsody - UML

- Test Input
- Algorithm
- Output
- Plant Dynamics Model
- Test Sequence/Harness

Hardware Interface

Platform Independent Model – PIM (Fully Re-usable)

Platform Dependent Model – PDM (Vehicle Specific)
Efficient Utilization of Model-in-the-Loop, PiL & Hardware-in-the-Loop Testing requires:

- Adoption of Platform Independent Models
- Platform Dependant Model
- Signal Abstraction Layer Mapping to Maximize Artifact Re-use
- Management of Test Artifacts: Test Environments, Test Vectors, Model Parameterization Data, Plant Models
When, What and How to Model - Considerations

- **Business Process Modelling**
  - How dependent are the Development, Production and Service Process-Objects?
  - What degree of Business/Engineering visibility do you want to provide?

- **Business Operational Models**
  - In-house Software System Development (SwSystDev)
  - Outsourced-Supplier SwSystDev
  - Hybrid SwSystDev (Internal/Supplier) (Feature/Function Co-linking)

- **Suitability of Feature/Subsystem for Executable modelling**
  - Degree of re-use
  - Feature Complexity
  - Feature Stability
  - Feature Distributed Nature
  - Degree of newness…
  - Domain Competency/skillset

- **Type of Model Framework**
  - Standards (Industry, Corporate, Domain, Organisation.).
  - Degree of Compliance/Leveraging of Standards
• Big picture processes and models...

**CORE**
*Product Development*
1. Business model
2. Application/Architecture model
3. Detailed Design
4. Code * test
5. System & Test
6. Issues Management

**APPLICATION**
*Product Implementation*
1. Detailed Implementation Design
2. Implementation Code and test
3. Systems Test
4. Acceptance Test
5. Issues Management

**Business Co-ordination**
1. Project management
2. Quality Management
3. Change Management
4. Cost Management

**Business Facilitation**
1. Configuration and Variant Management
2. Interface Management
3. Training
4. Quality Assurance
5. Metrics and Reporting

**Product Customer Usage/Dealerships**
1. Release Mngmnt
2. Performance Mngmt
3. Customer Support
4. Warranty Mngmt
5. Change Mngmt
# Architectural Views

<table>
<thead>
<tr>
<th>Decomposition</th>
<th>Module Views</th>
<th>C&amp;C Views</th>
<th>Allocation Views</th>
<th>Other documents</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Roles</th>
<th>D</th>
<th>S</th>
<th>O</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Project Manager</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Development team</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td></td>
</tr>
<tr>
<td>Test Engineers</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td></td>
</tr>
<tr>
<td>Integration Engineers</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td></td>
</tr>
<tr>
<td>Design Engineers</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td></td>
<td>S</td>
</tr>
<tr>
<td>Service Engineers</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
</tr>
<tr>
<td>Product-Line Engineer</td>
<td>D</td>
<td>D</td>
<td>S</td>
<td>O</td>
<td>S</td>
</tr>
<tr>
<td>Internal Customers</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>End Users/Customers</td>
<td>D</td>
<td>D</td>
<td>S</td>
<td>D</td>
<td>D</td>
</tr>
<tr>
<td>Analysts</td>
<td>D</td>
<td>D</td>
<td>S</td>
<td>D</td>
<td>D</td>
</tr>
<tr>
<td>Infrastructure/EcoSystem IT support</td>
<td>S</td>
<td>S</td>
<td>S</td>
<td></td>
<td></td>
</tr>
<tr>
<td>New Stakeholders</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Current/Future Architects</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
</tr>
</tbody>
</table>

Key: D - Detailed Information; S - Some Details; O - Overview Information; x - anything

- Which Architectural Perspectives are Critical to Business Success?
- What degree of interoperability do you need between the Architectural Views?
- What Degree of Executable Architecture Model do you need to implement?

Derived from Data within "Documenting Software Architectures, 2nd edition, Paul Clemens et al"
Effectiveness? Failures Models, Causal Factors & PCA Study

– Six Sigma Studies (DFSS & DMAIC)
  • Process Maps
  • Hidden Factories
  • Capability Assessments
  • Causal factor - Defect Correlation statistical studies
– The Critical Few – Critical to Business (CTB) Process, Methods, Tools and Information Management
  • CTB Metrics
– Hierarchical Decision Process to Prioritise the Critical Few

• DMAIC and DFSS based analysis identifies and prioritises the key areas that need to be addressed

<table>
<thead>
<tr>
<th>Alternative</th>
<th>Utility</th>
</tr>
</thead>
<tbody>
<tr>
<td>Powertrain Transmission Control</td>
<td>0.858</td>
</tr>
<tr>
<td>Sync Module - Customer Facing</td>
<td>0.852</td>
</tr>
<tr>
<td>ACC stopgo</td>
<td>0.827</td>
</tr>
<tr>
<td>Sync Module - Vehicle Facing</td>
<td>0.794</td>
</tr>
<tr>
<td>Powertrain Control Module</td>
<td>0.787</td>
</tr>
<tr>
<td>Low-speed CMHi (City Safety)</td>
<td>0.784</td>
</tr>
<tr>
<td>Terrain response</td>
<td>0.779</td>
</tr>
<tr>
<td>Active Safety</td>
<td>0.776</td>
</tr>
<tr>
<td>Powertrain Engine Control</td>
<td>0.777</td>
</tr>
<tr>
<td>Restraints Control Module</td>
<td>0.728</td>
</tr>
<tr>
<td>Restraints (Occupant &amp; Pedestrian)</td>
<td>0.719</td>
</tr>
<tr>
<td>Brake Controls</td>
<td>0.698</td>
</tr>
<tr>
<td>Driver Information and Warnings</td>
<td>0.661</td>
</tr>
<tr>
<td>MyKey</td>
<td>0.655</td>
</tr>
<tr>
<td>Infotainment</td>
<td>0.650</td>
</tr>
<tr>
<td>Audio System</td>
<td>0.650</td>
</tr>
<tr>
<td>Wiper/Washer Systems</td>
<td>0.641</td>
</tr>
<tr>
<td>Adjustable Accelerator/Brake Pedal</td>
<td>0.641</td>
</tr>
<tr>
<td>Tilt/Telescoping Steering Column</td>
<td>0.627</td>
</tr>
<tr>
<td>Steering Control</td>
<td>0.621</td>
</tr>
<tr>
<td>Personalization</td>
<td>0.620</td>
</tr>
<tr>
<td>Vehicle System Services</td>
<td>0.601</td>
</tr>
<tr>
<td>TPMS Feature</td>
<td>0.601</td>
</tr>
<tr>
<td>SPDJB - Vehicle Start</td>
<td>0.598</td>
</tr>
<tr>
<td>Airbag Control</td>
<td>0.585</td>
</tr>
<tr>
<td>Suspension/Ride Control</td>
<td>0.581</td>
</tr>
<tr>
<td>Power Supply</td>
<td>0.575</td>
</tr>
<tr>
<td>Reverse Park Aid Feature</td>
<td>0.574</td>
</tr>
<tr>
<td>Exterior Lighting</td>
<td>0.572</td>
</tr>
</tbody>
</table>
Fundamental capability is to have full Linkages, Dependency info and Version Visibility between:

- Requirements/Functional Specifications
- Test Vectors
- Executable Models
- Parameters/Signals
- Configuration Data Sets
- Calibration Data Sets
- Software Binaries and Build configurations
Different Domains (Powertrain, Chassis, Electrical) use different modeling tools. Fundamental object management and relationship tasks are similar.
System high level functional requirements are modeled in SysML which are then decomposed into lower level behavior models.

Behaviors are partitioned into a logical architecture and physical architecture.

The functional system requirements and derived requirements are linked to the SysML behaviors using a Requirements Diagram.

Requirements gaps were identified when modeling the behaviors which were not apparent from the textual requirements alone.

Courtesy of Kyle Post, Ford Motor Company
Managing Vehicle System Complexity

Vehicle Software & Electrical Management – VSEM

Create a global information platform with seamless information management of engineering data from function to ECU S/W & H/W inclusive to create a complete PLM/ALM solution for electronic/software area.

– Provide a fully traceable, object level work product framework

– Support a Model Based Systems Engineering EE development process

– Single source for all information E/E development data

– Support change, version, and configuration management with full traceability

– Based on standardised open tool framework to enable utilisation of “best-in-class” commercial state-of-the-art “plug-in” tools.

– Designed on industry engineering standards and aligned to the AUTOSAR architectural framework
Adoption of Single Corporate data backbone and authoring tools enables WP and relationship Meta data re-use across Corporate PD systems.

Must support data objects that require rapid iteration (ALM) Vs structured management (PLM).
VSEM Project Scope and Implementation

Product Development

- Systems Engineering,
- Software Systems Engineering
- Software Engineering

> Model Based Systems Engineering<

EE System Artifacts that derive a Released BOM
Requirements, Tests, Models, Parameters, Devices, Calibrations, Configurations, Architectures...
(As-Designed)

In-Vehicle Software (IVS)

E/E System BOMs

Produced E/E System BOMs (As-Released)

Assembly Plants

- Valencia
- Saarlouis
- Cologne
- Genk
- Auto Alliance
- Cuaautitian

In Plant Flash

Manufactured Vehicle Series Variants

BOMs 1-50

Global Manufacturing Plant Reprogramming

(As-Built)

Dealerships and Web Services

- FoE - Dealer Updates
- FAP - Dealer Updates
- FSA - Dealer Updates
- Customer Web Services Sync My Ride

Global Dealerships & Customer Updates

(As-Modified)
Systems Engineering MB Framework

- FMEA
- P-Diagrams
- Boundary Diagrams
- Interface Matrix
- Noise Factor Mangmt
- Strategy
- DVP
- Safety Plan
- Prelim Hazard Analysis
- Functional Safety Concept

Feature Dictionary

- Virtual, Model Based Analysis; Dependencies; Coupling; Cohesion & Arch Quality Attribute Studies

Functional Architecture (Implementation Independent)

- Commodity Specific Reference Architectures
  - Sync Gen 2 Ref Arch V1.0
  - BCM Ref Arch V1.x

Logical Architecture (Re-usable Reference Architectures)

Physical Architectures (Vehicle Program Specific)

Global Signal Database (GSDB)
- Signal Definitions - Interface Specs
- CAN Message relationship
- Global Device Transmittal Database (GDT)
- SW-HW interfaces
- EE Devices – Sensors - Actuators

Vehicle Program Specific Architectures

Veh 1
Veh 2
Model Comparisons in BoM View

- BoM based model comparisons or graphical model differencing can be performed.
PLM/ALM Based Model Management

- Unique ICONS to provide visual cues to identify and relate feature/model artifacts
• Modeling Tool Integration provides both a Bill of Material view and a Graphical Model Browser view

• All model object references; devices, input/output signals, connectors, requirements, test vectors, parameters, calibration & configurations are related and managed in a central database
Full Traceability into Impacted Work Products

- Environment supports full impact analysis to understand any change in it’s Vehicle/Domain/Subsystem and Feature context.
- Enables fully informed, high value decision making.
- Full re-use with context and underlying assumptions
1. Different domains select different modeling tools due to multiple factors
   1. Ability of their domain specific artifacts to be represented completely within a modeling schema/construct set
   2. Number of engineers with a specific modeling competency within the domain
   3. Amount of legacy models of a given modeling tool type within a domain
   4. Level of abstraction required within a domain and across domains
   5. Availability/cost of existing/new Licenses
   6. Lack of engineering staff with modeling competency or allocated resource/time.

2. Some features/commodities do not benefit from detailed modeling (non-changing features that are truly purchased commodities and do not benefit from executable modeling efforts (still may benefit from static architecture models)

3. These forcing functions lead to the need to support a HYBRID MBSE environment that consists of a set of modelling tools with associated style guides, maturity levels, completeness levels

4. For HYBRID environments it is important that all of the critical to quality/function artifacts/Meta data and associated objects/relationships are managed with supporting lifecycle management tools

5. Enterprise wide change is hard, painful, challenging and rewarding AND far better than the alternative……

"On Ko Chi Shin" – “Studying the old promotes a better understanding of the new”. Ancient Japanese proverb.
Thank you for listening.