

# AUTOSAR E/E System Design VSx Tool Chain Overview

Kai Wu

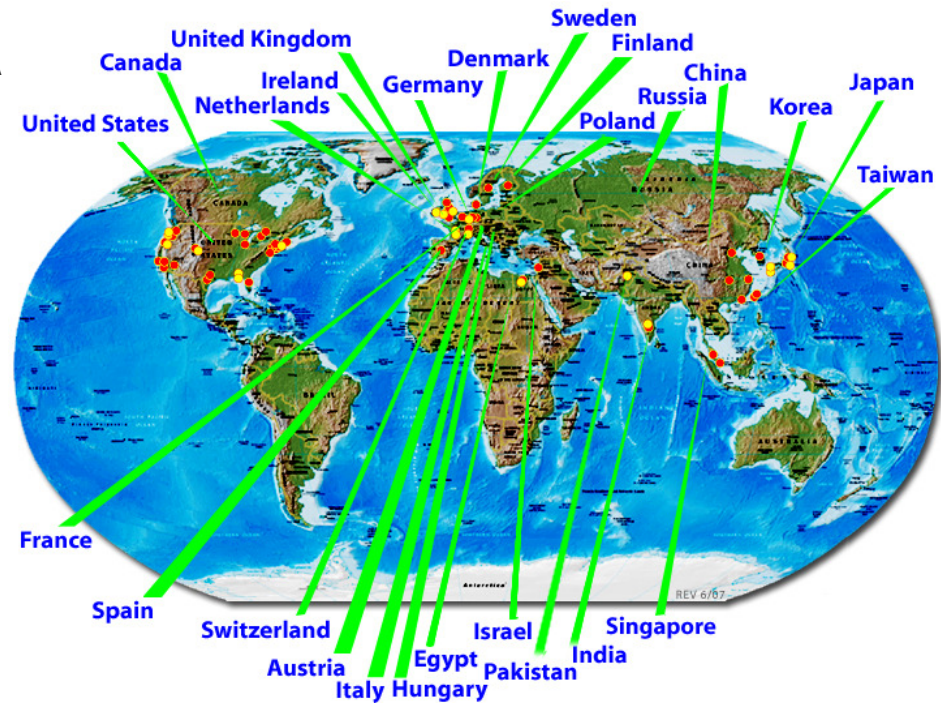
Technical Marketing Engineer

March 2010

**Mentor**  
**Graphics®**

# Mentor Graphics a Leading Technology Driver in Electronic Design Automation (EDA) since 1981

- Revenue of \$790M in 2008
- Market share ~23% of worldwide EDA market
- Largest ECAD supplier to the automotive electronics industry
- One of the largest SW companies in the world (66th)
- 4,500 employees worldwide
- Acquired Volcano (VCT) in May 2005
- Member of AUTOSAR since 2004

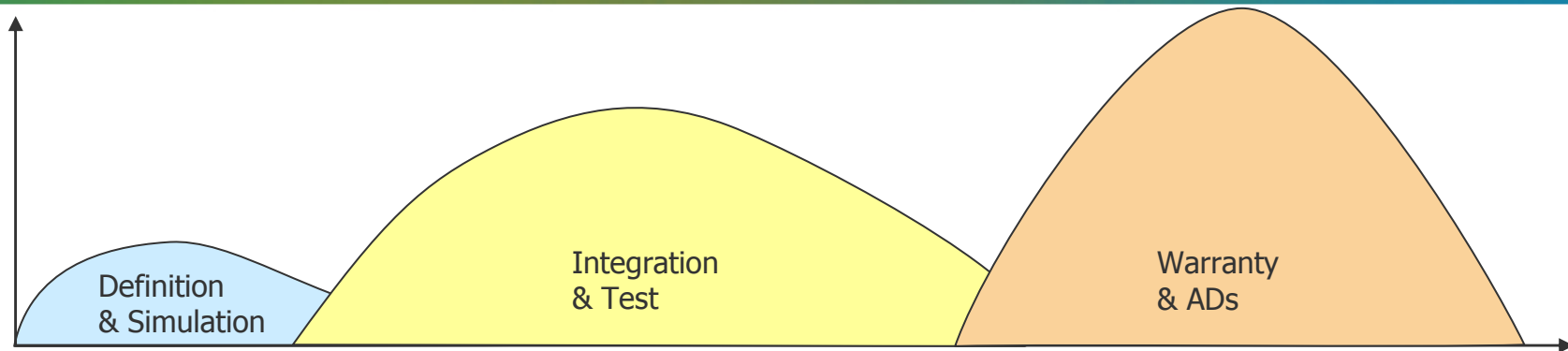


# Guiding principles for Mentor VSx tools

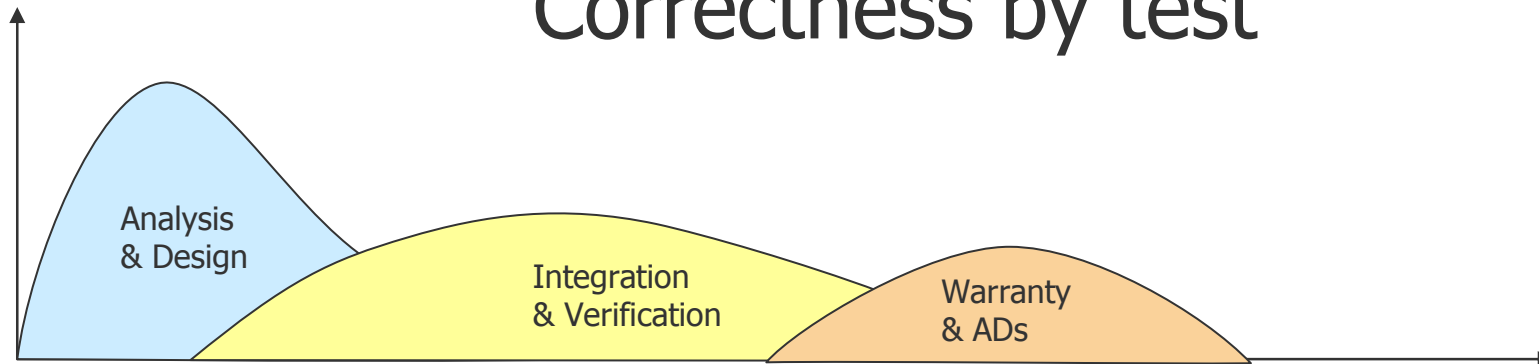
---

- Enable “front-loading” of development
- Enable shifting as much as possible of the validation effort to a virtual environment
- Use standard terminology and data exchange formats
  - AUTOSAR
  - Eclipse
  - EAST-ADL
- Cover the whole flow from requirements to SW and ECU implementation
- Enable customers to step-by step adapt individual parts of a complete solution

# Frontloading the E/E Development



Correctness by test

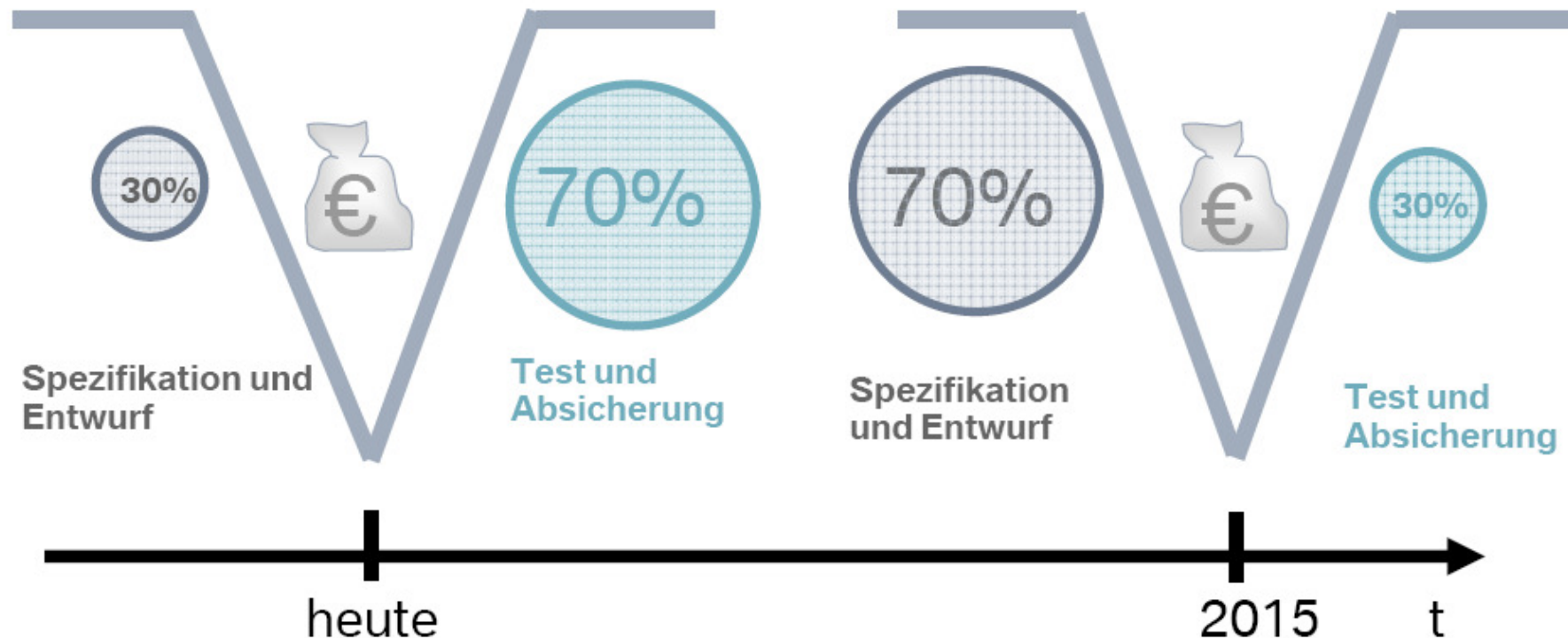


Correctness by design

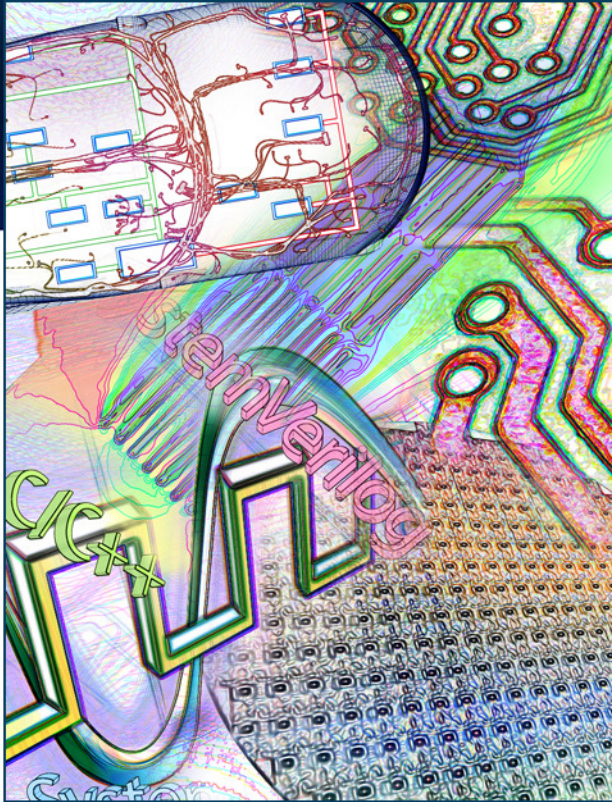


# BMW Example

AUTOSAR Tag  
@ BMW Group  
E. Frickenstein  
BMW Group  
16.01.2008

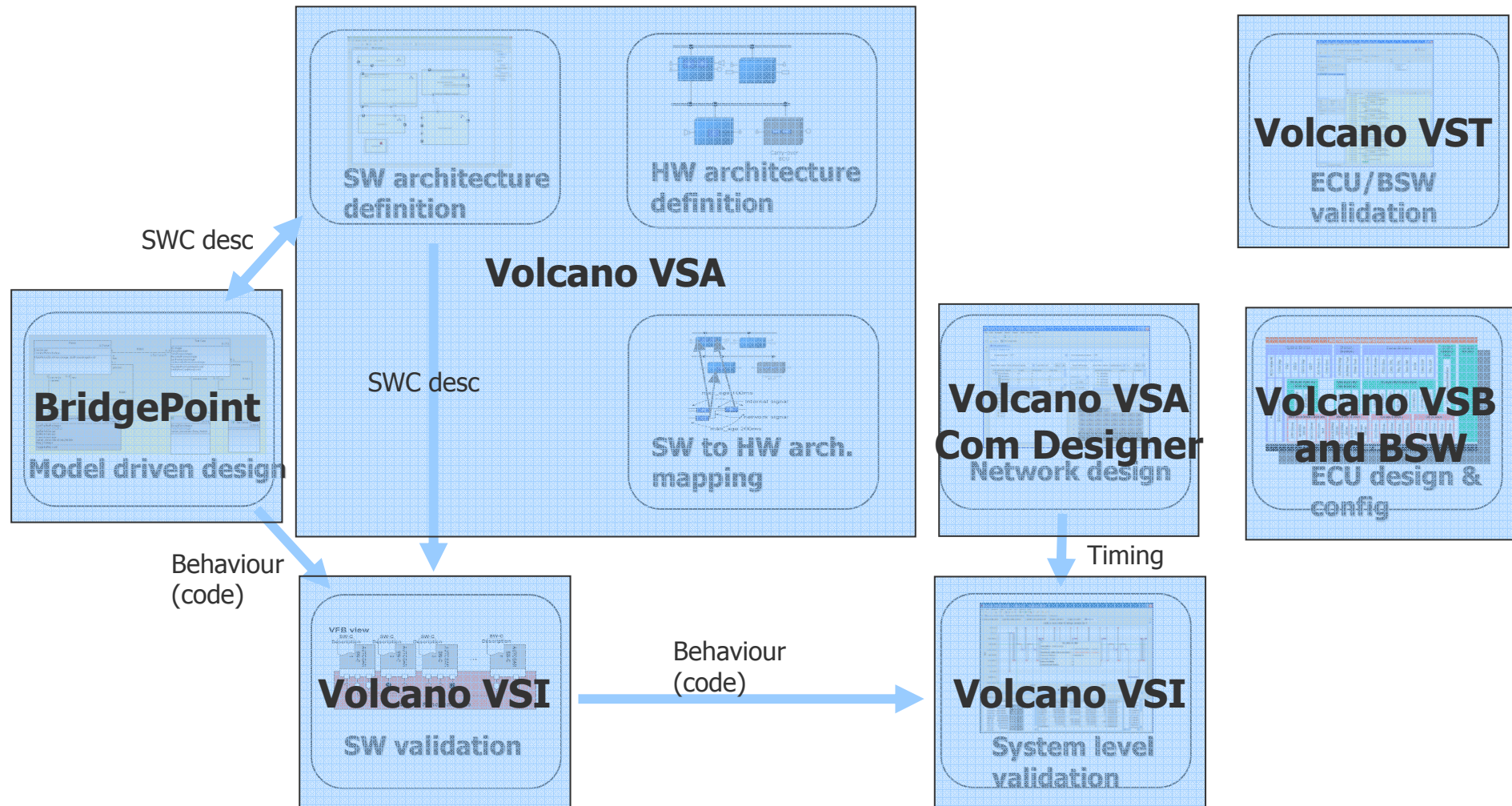


- Change from testing to correctness (impossible) to design to correctness!
- Frontloading is the main subject of Electronic Design Automation!
- Mentor is the company, which can take you there!



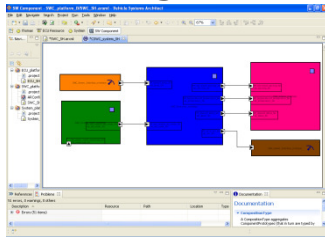
# Products overview

# Mentor VSx AUTOSAR SW development tools

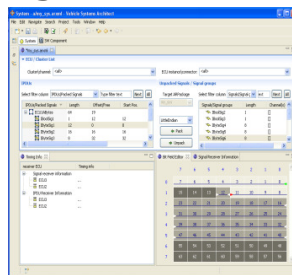


# VSx Toolchain

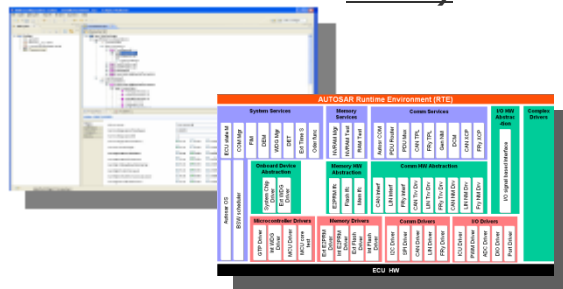
## E/E Architecture & Authoring



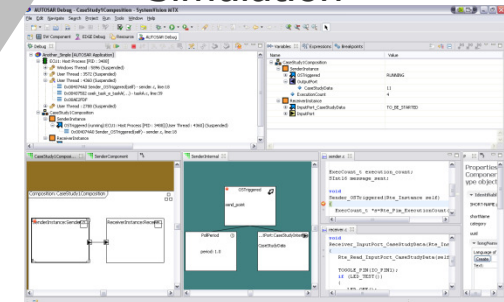
## System Design



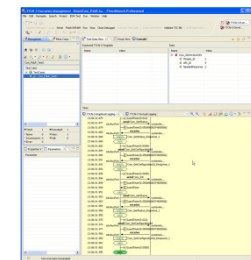
## Implementation (ECU Confia + BSW)



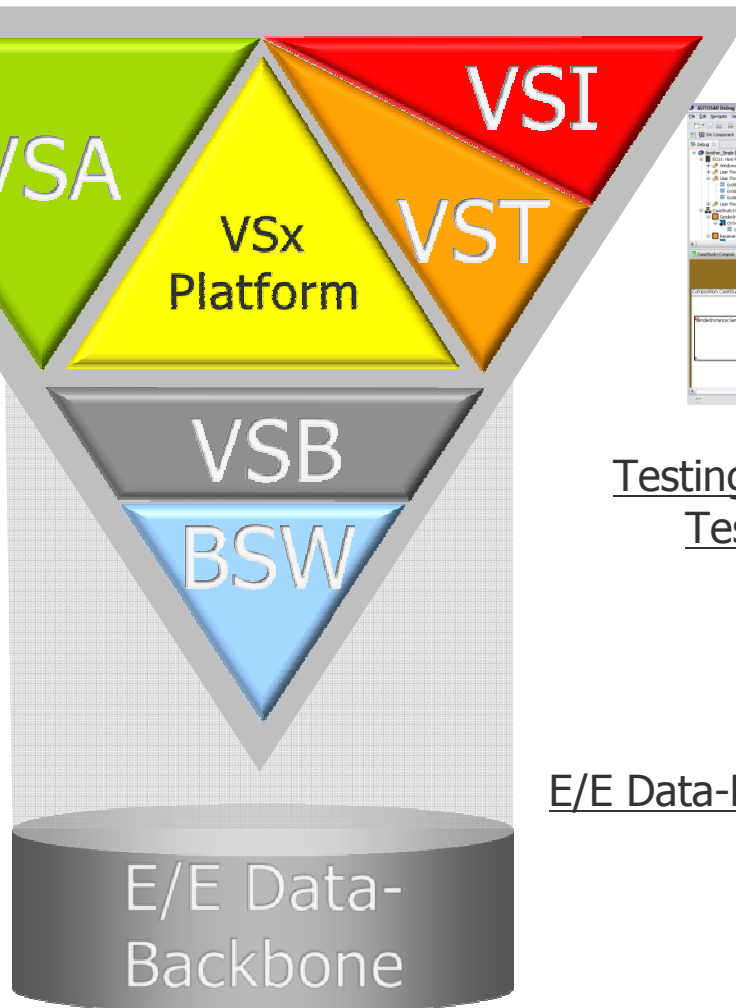
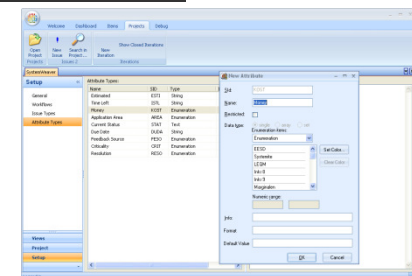
## Validation and Simulation



## Testing and Conformance Testing



## E/E Data-Backbone





# Tools Overview

## E/E Architecture & Authoring

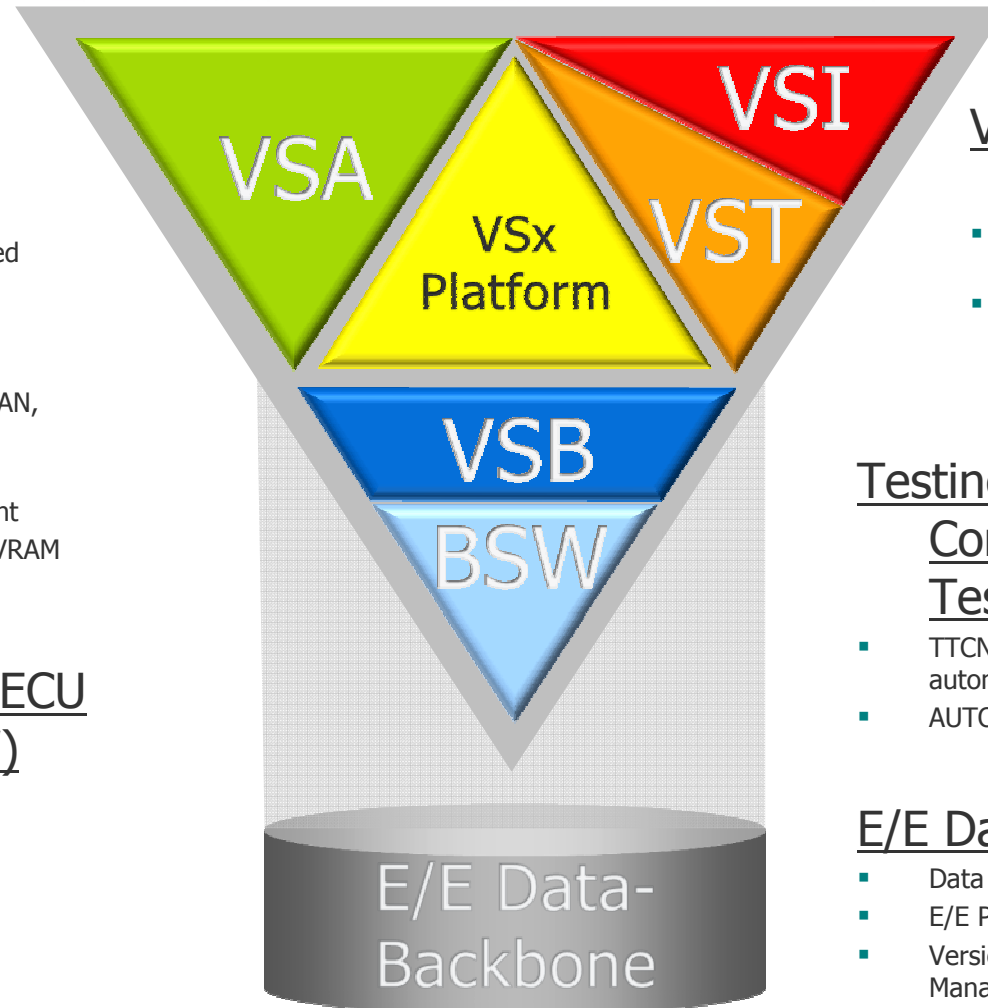
- E/E Architecture Exploration
- System and ECU Design
- SWC, CSWC and deployment
- AUTOSAR and EAST-ADL based
- Concurrent engineering

## System Design

- Network Design for (LIN, CAN, FlexRay)
- System and ECU Design
- SWC, CSWC and deployment
- RTE, Diagnostics (ODX), NVRAM

## Implementation (ECU Config + BSW)

- ECU Configuration
- AUTOSAR BSW



## Validation and Simulation

- Distributed simulation, debugging and validation of SWC
- xtUML and IDE based SW development

## Testing and Conformance Testing

- TTCN-3 based testing of automotive SW (on PC and Target)
- AUTOSAR conformance testing

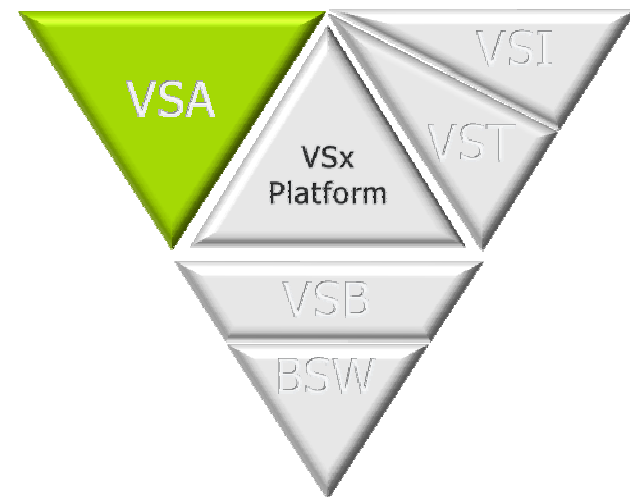
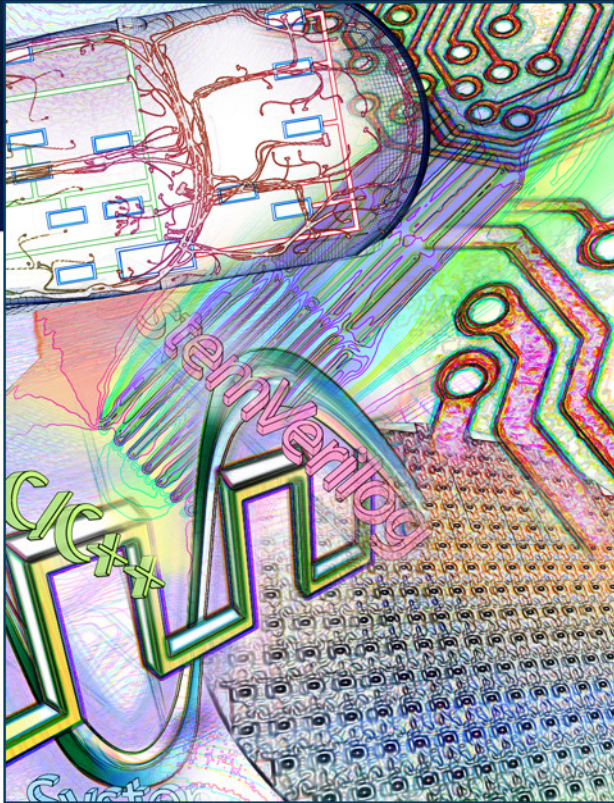
## E/E Data-Backbone

- Data management
- E/E PLM
- Version, Release and Variant Management



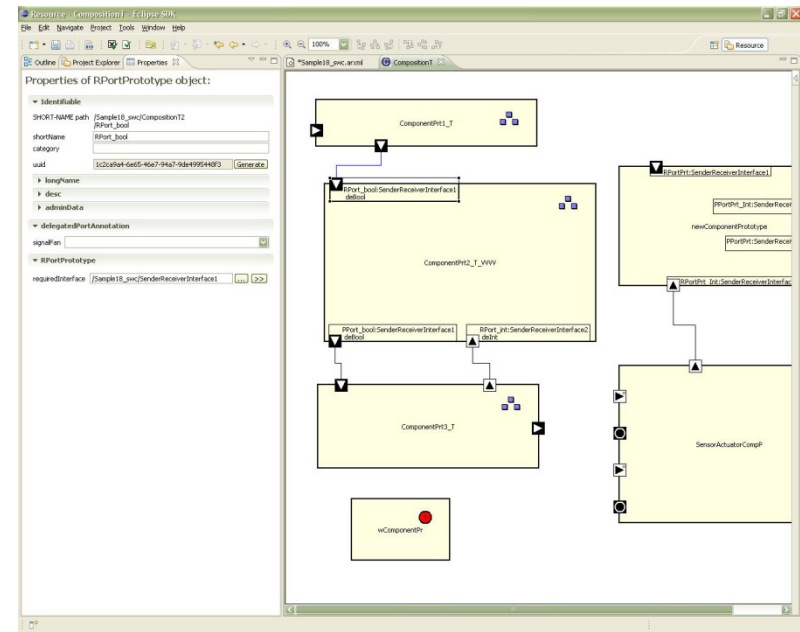
# VSA

## Vehicle Systems Architect

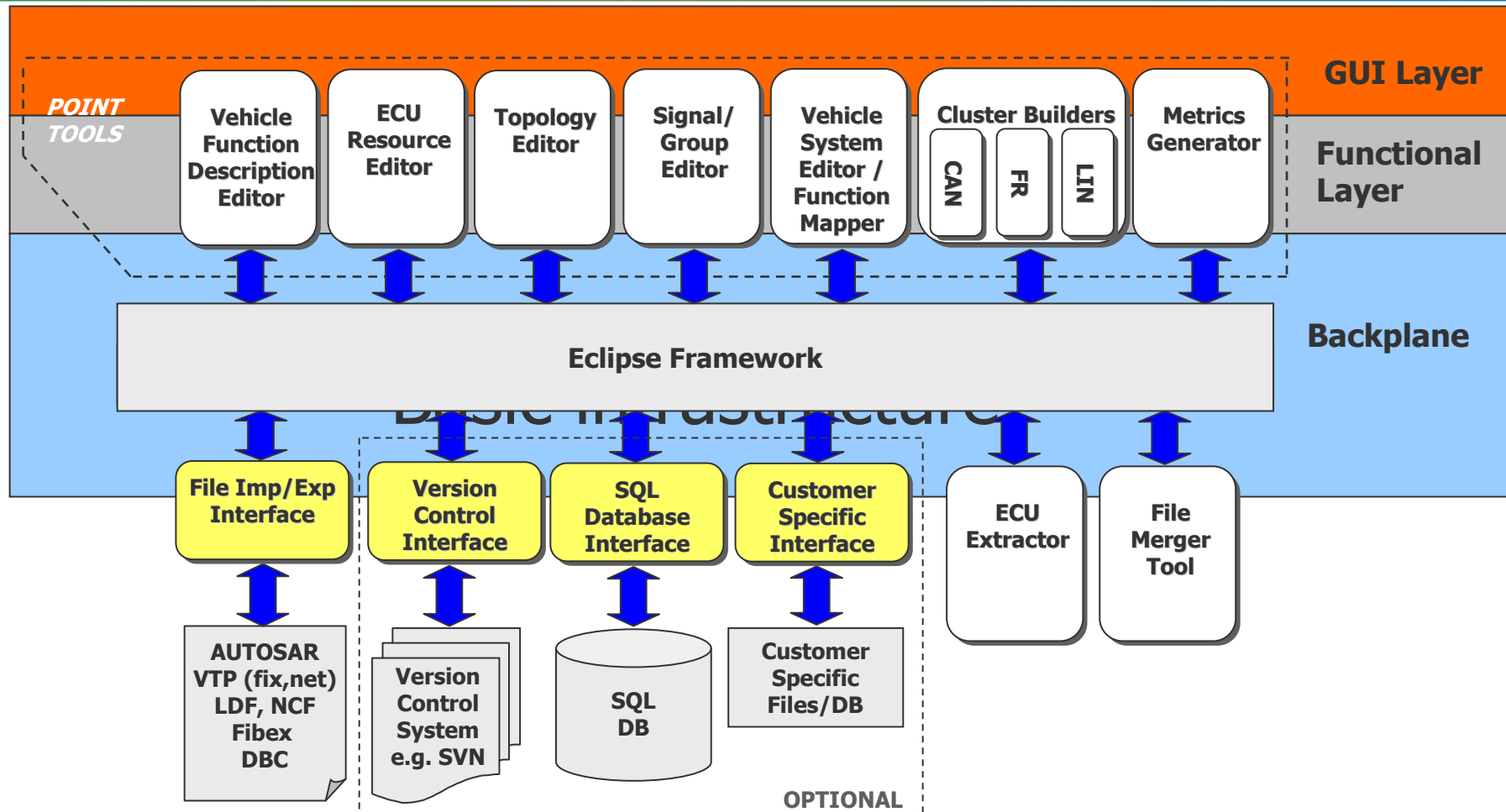


# Volcano VSA

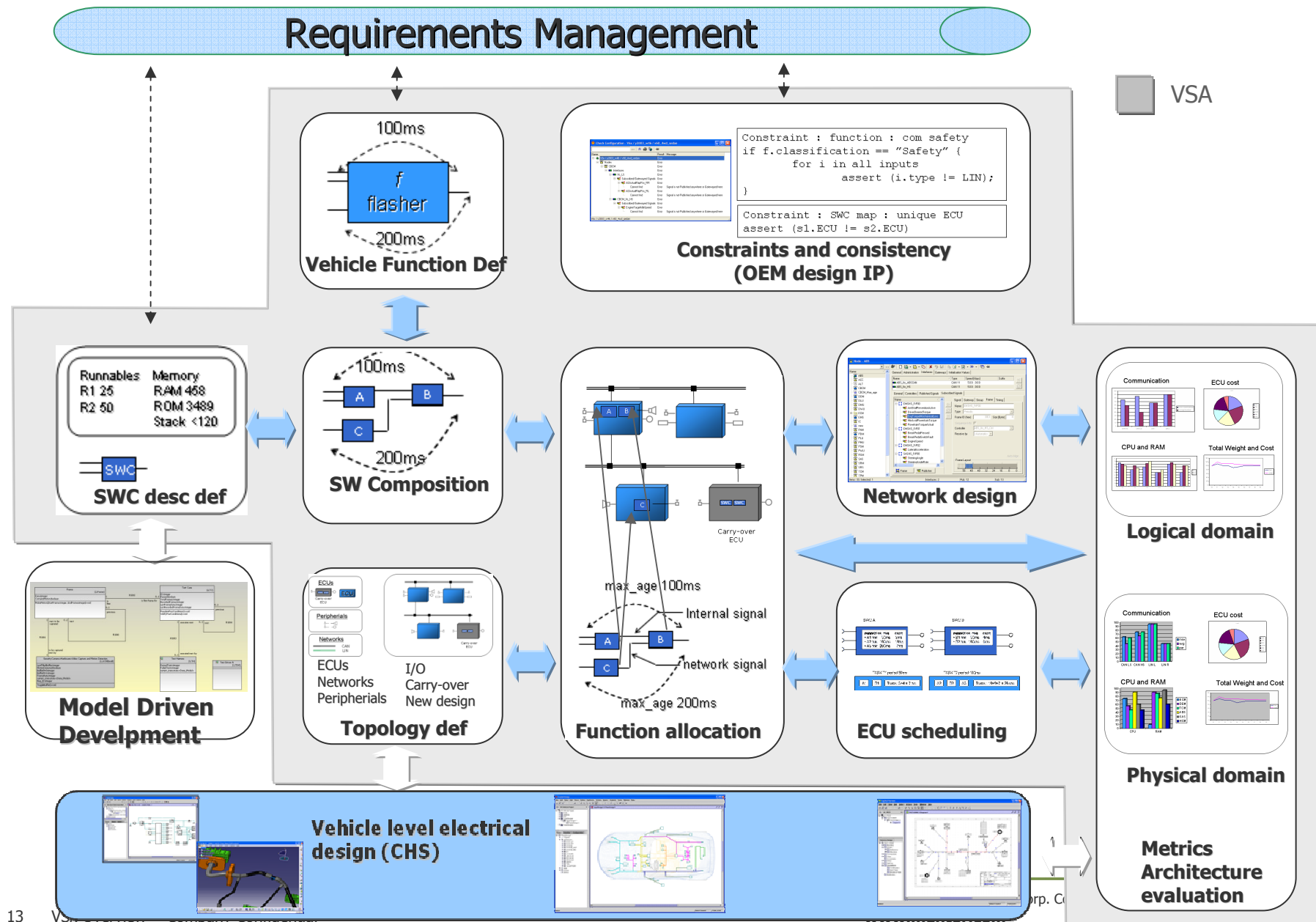
- VSA is a system level design tool for vehicle SW, electronic and communication systems
- Currently VSA is focused on development and implementation level
- VSA is being extended into various areas:
  - High-level function design
  - E/E Architecture design support
  - Design Data Management
  - Variability management



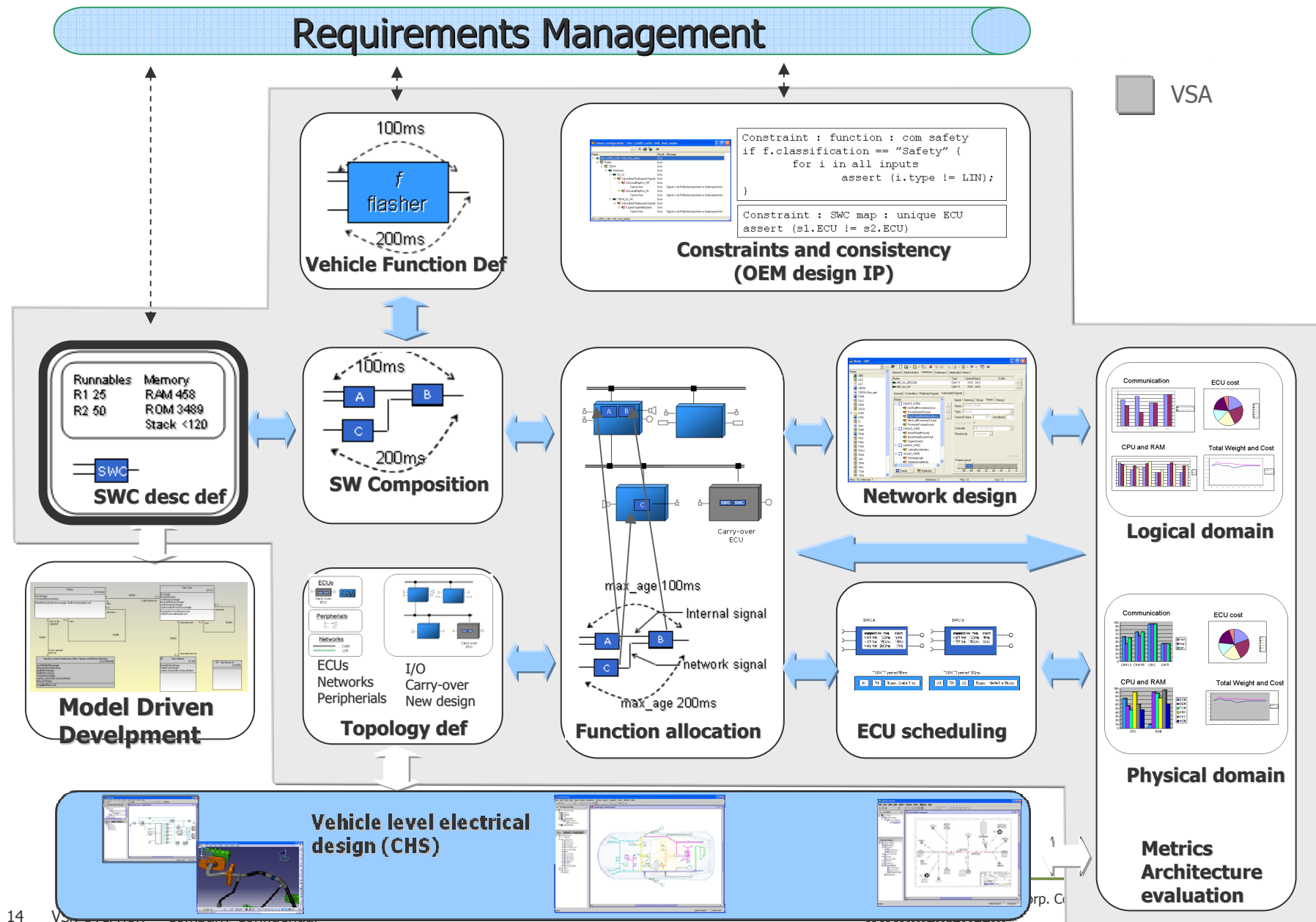
# VSA Technical Overview



# VSA – Activities



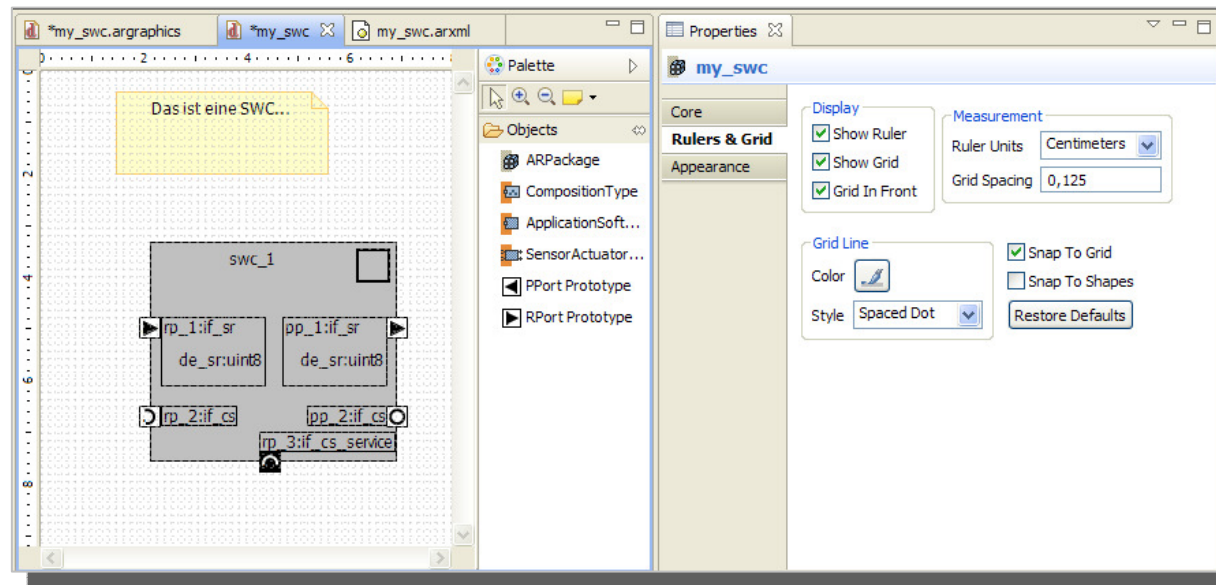
# VSA – Activities



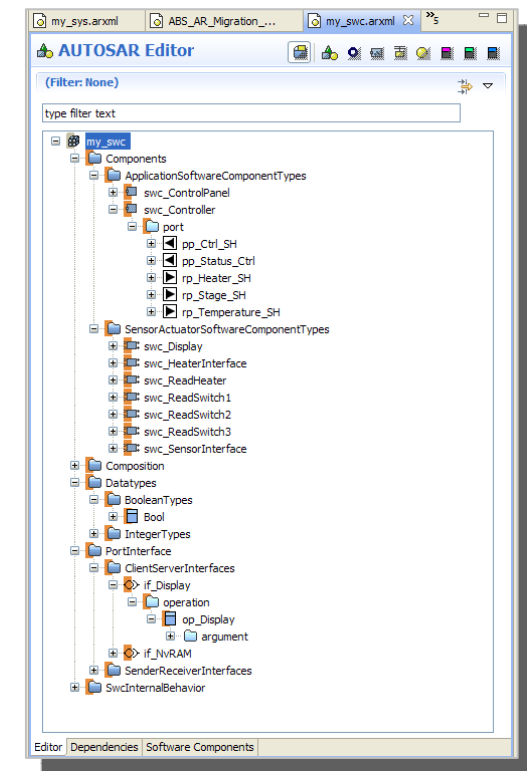


# SWC definition

- Define ports, interfaces, runnables
- Graphical or tree-like views



*Graphical SWC editor*



*AUTOSAR Editor*

# Compu-method definition

- Graphical editor to define relation between internal and physical values
  - Rational function
  - Linear
  - Piecewise linear
  - Constants
  - Texttable
  - ...

Software Components

PortInterface Mapper | Software Composition | Computation Method

Computation methods

Type text to find  Next

Name

- my\_sys
  - newCMT
  - my\_swic
  - ECU\_SH

Add Remove

unit

Scales

Phys2Int Int2Phys Basic Expert

Type text to find  Interval  Jex  Short label

Contents type Contents value Type Rational Formula

Rational Formula  $(6 + 3*x + 1.4*x^2) / (2 + 3*x)$

Lower  ( ) [ ]  $\infty$

Upper  ( ) [ ]  $\infty$

Value

	0	1	2	3
Num	6	3	1.4	
Den	2	3		

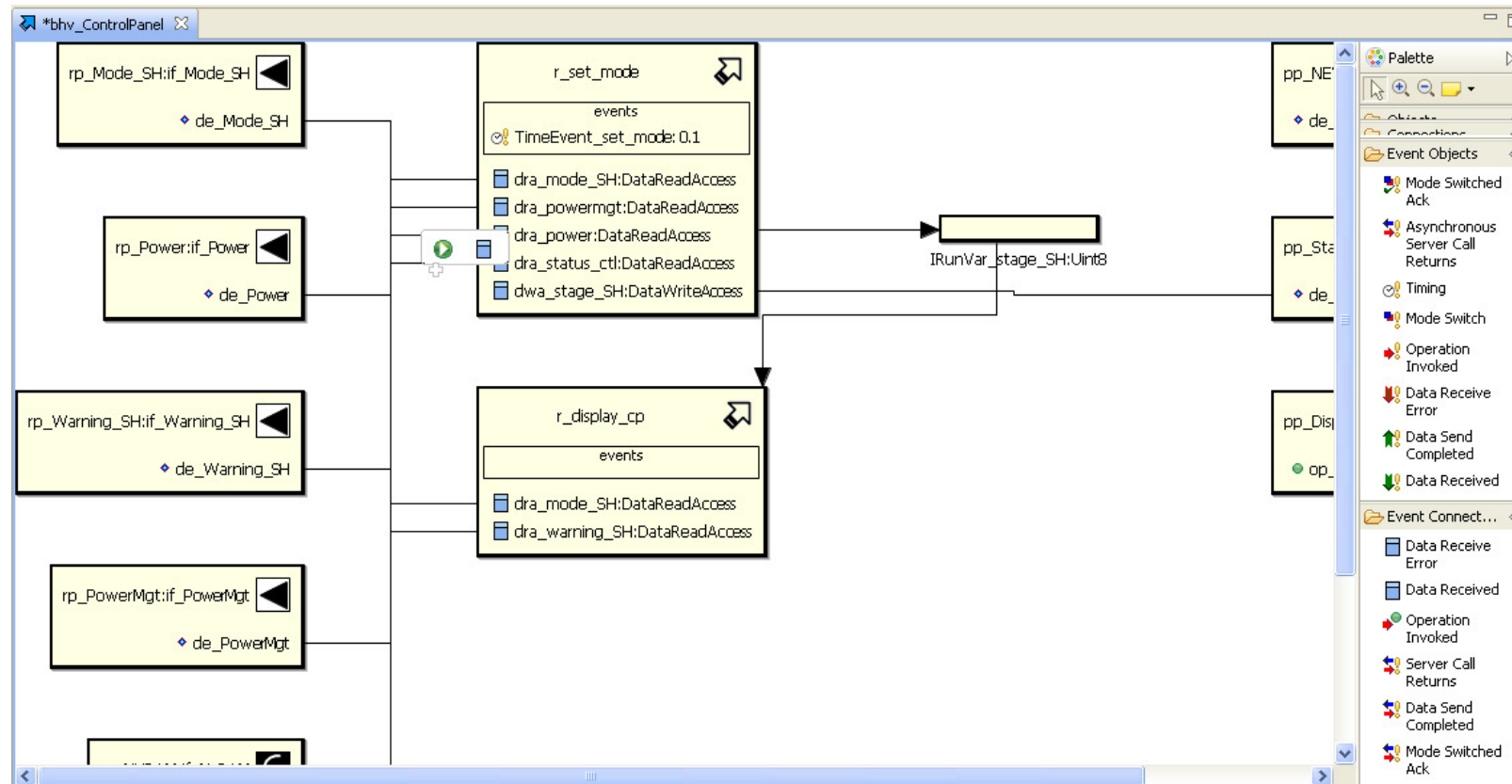
Add Remove Apply Discard

Visualization

Physical

Internal

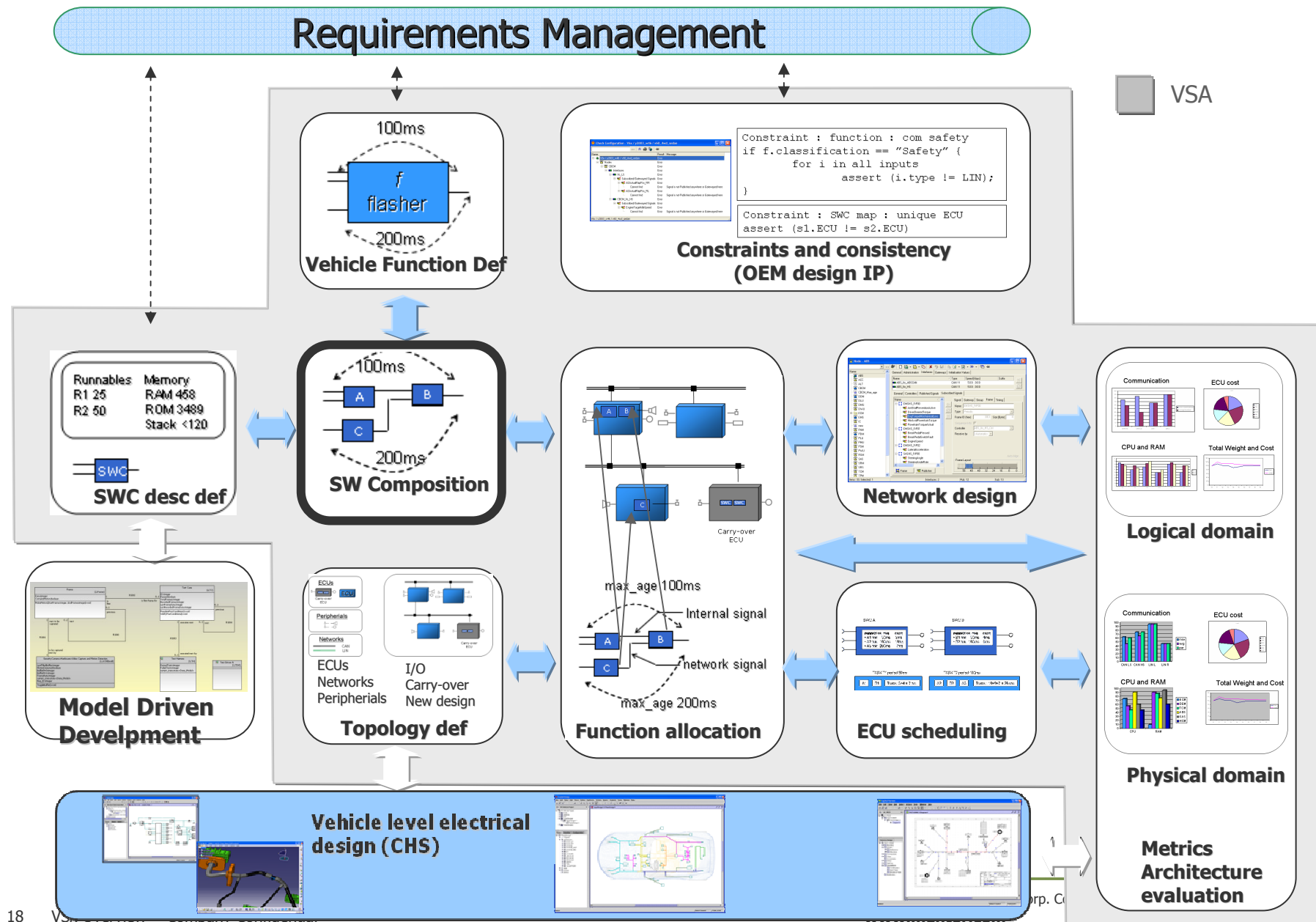
# Runnable Editor



- Runnables
- Interrunnable variables

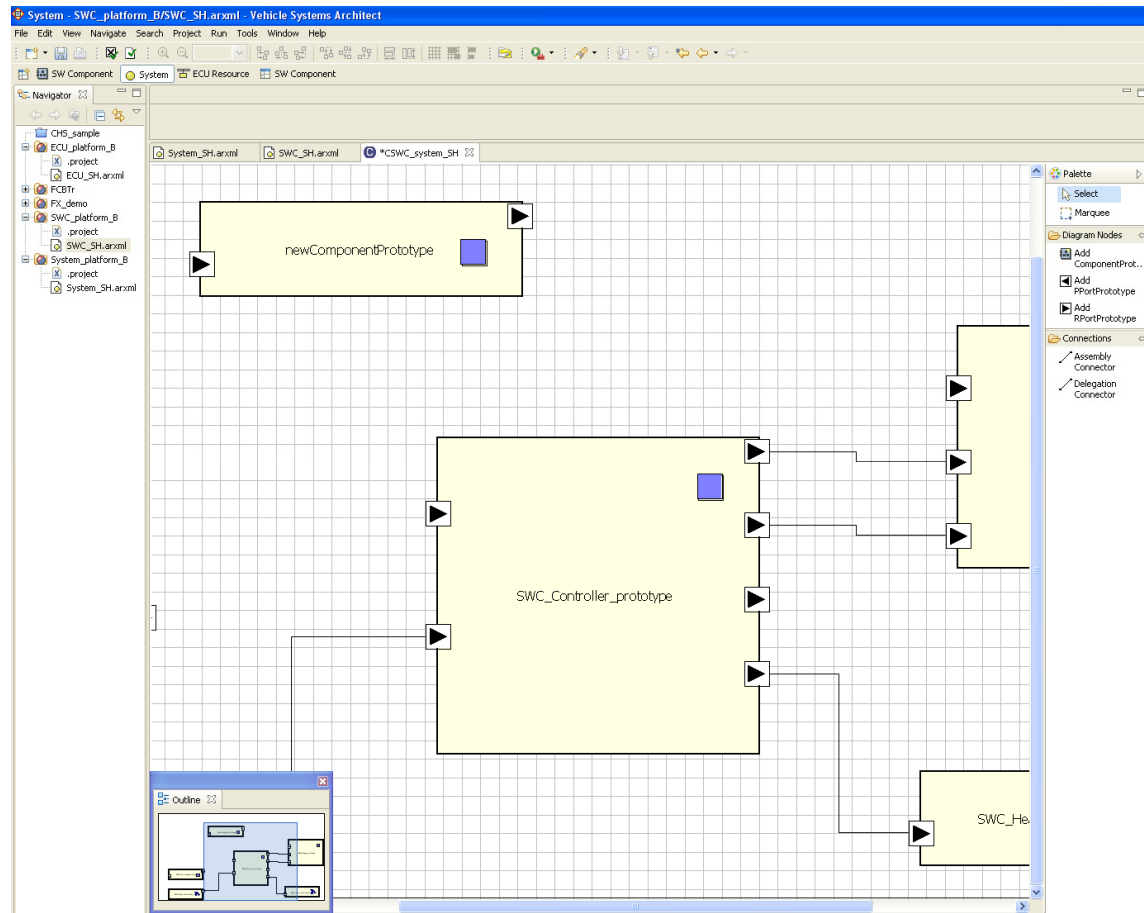
- RTE Events
- Data send/receive points

# VSA – Activities



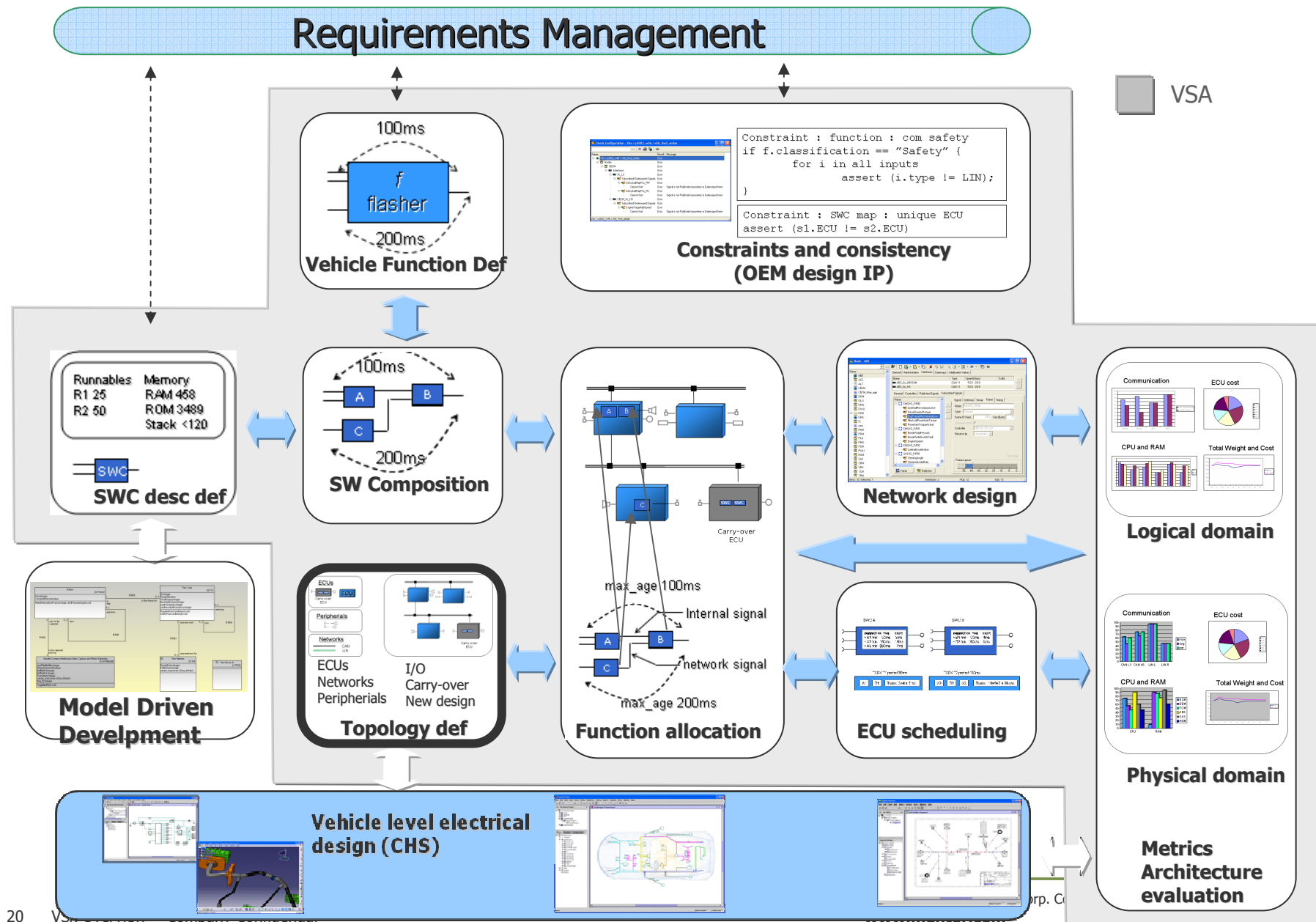
# SW composition definition

- Graphical design of SW compositions
- Zoom in/out
- Outline view
- Delegation ports

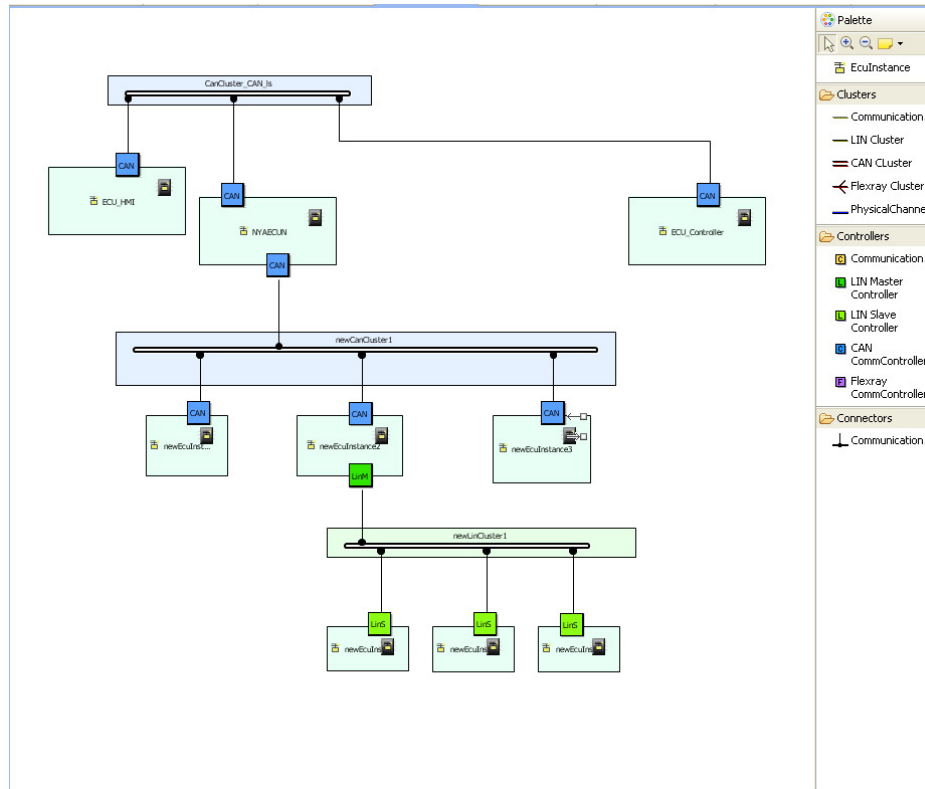




# VSA – Activities

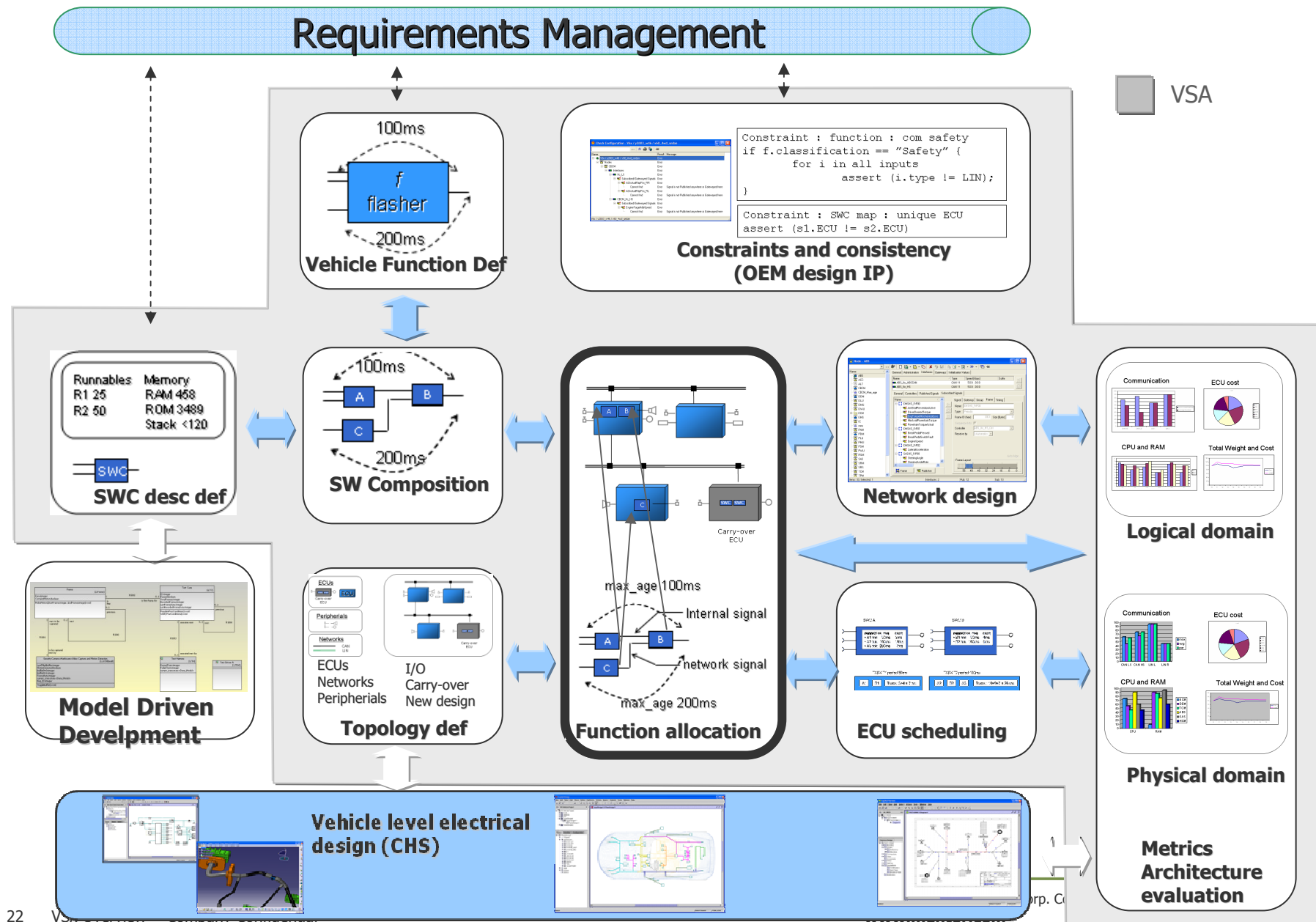


# Topology Definition



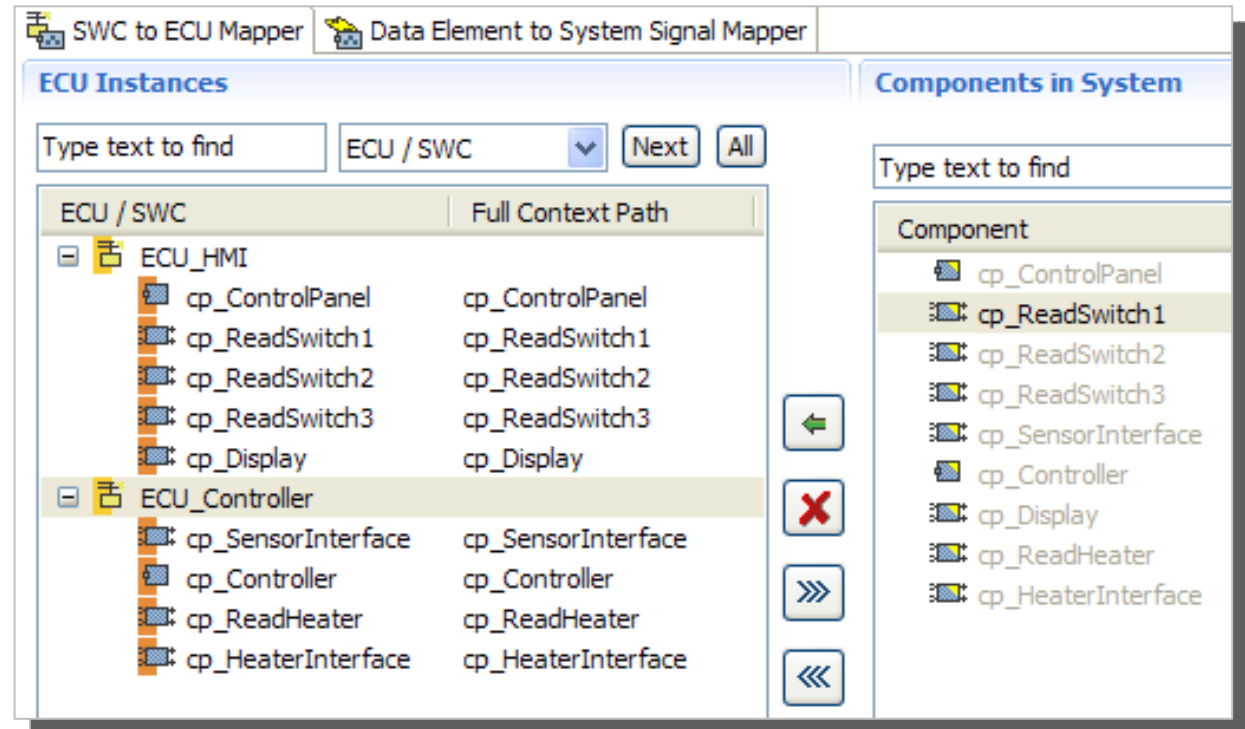
- ECUs
- NETworks
- Communication interfaces
- Communication Connectors

# VSA – Activities



# SWC to ECU mapping

- Map SWC to ECUs
- Search by typing



# ECU Resource Generator

- Generates ECU resource based on system design
- Allows user to define additional ports and pins
- Select external/internal property

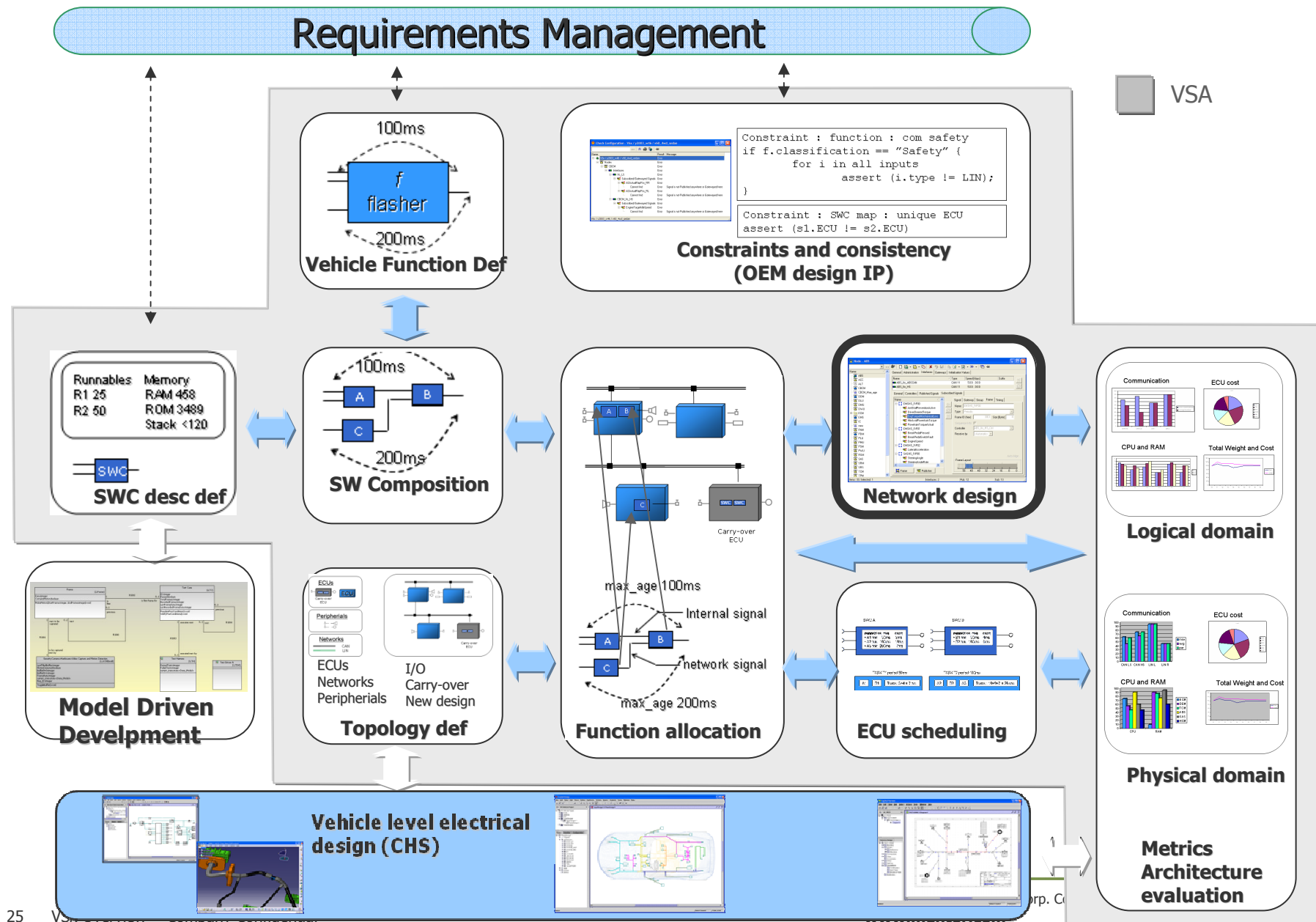
The screenshot shows the 'ECU Resource Generator' dialog box with the title 'Defining ECU HW Element'. The instructions state: 'The ECU Instance should be selected. Refine the name of the ECU and the related Sensors and Actuators.' Below the instructions, there are two dropdown menus: 'ECU Instance:' and 'ECU HW Element:', both set to 'ESC'. A table titled 'Sensors/Actuators:' lists various components with their types and internal/external status.

Name	Type	Internal	External
IHBT_SW	SENSOR	X	
ESC_WRNG_LP	SENSOR	X	
ABS_WRNG_LP	SENSOR	X	
WHL_SPD_OUT	SENSOR	X	
ESC_VLV1	SENSOR	X	
ESC_VLV2	SENSOR	X	
ESC_VLV3	SENSOR	X	
ESC_VLV4	SENSOR	X	

At the bottom of the dialog, there are four buttons: '< Back', 'Next >', 'Finish', and 'Cancel'. A help icon (?) is also present in the bottom left corner.

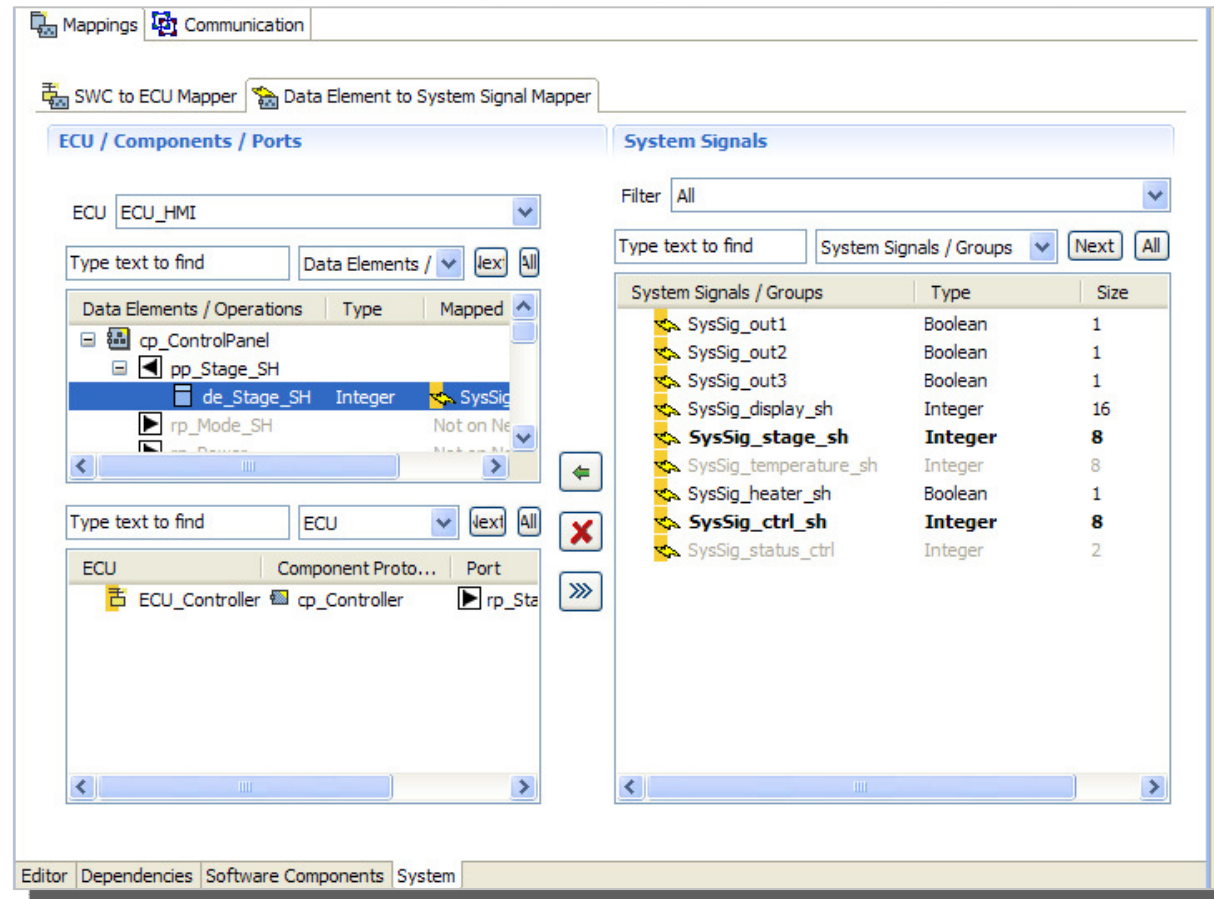


# VSA – Activities



# System Signal to Data element mapping

- Define the relation between SWC data elements and system signals

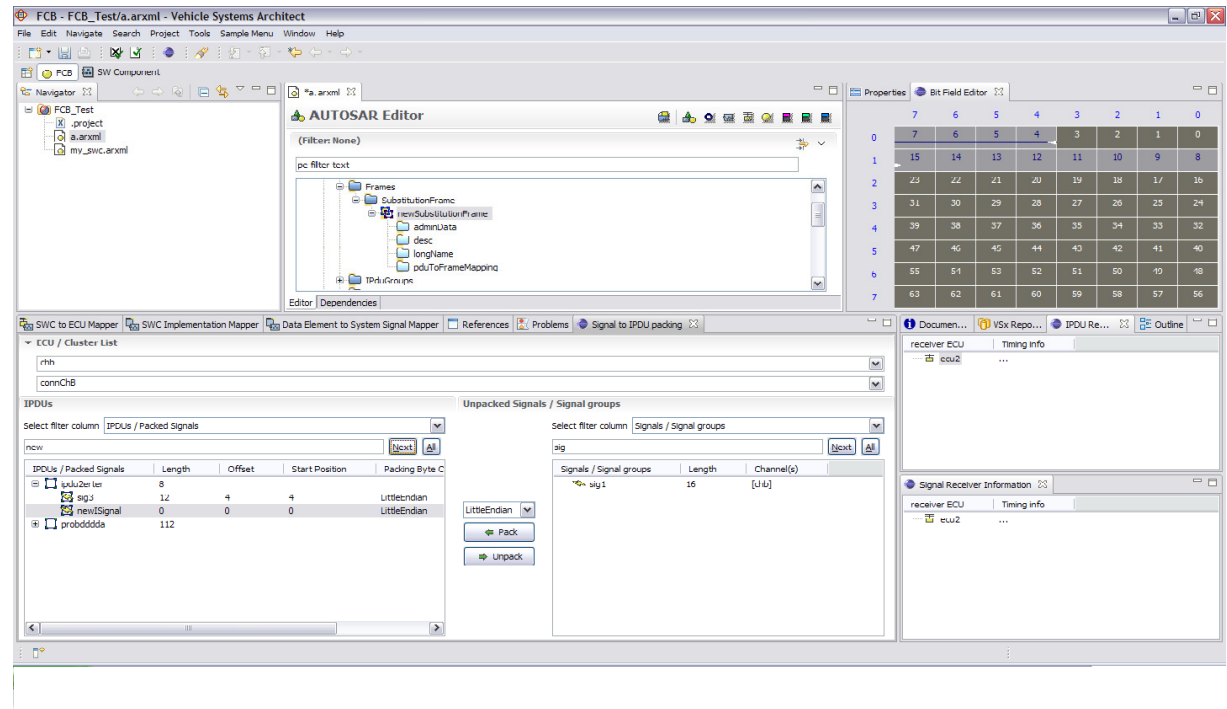


# Network Design

## VSA - Com Designer (option to VSA)

### ■ AUTOSAR based network design

- ISignal to IPDU mapping
- IPDU to frame mapping
- consistency check



- The Com Designer options are separate products for each protocol type (CAN, LIN, FlexRay)

# Manual FlexRay Scheduler

System - FCBTr/Mixed\_04.arxml - Vehicle Systems Architect

File Edit Navigate Search Project Tools Window Help

SW Component System ECU Resource SW Component

\*Mixed\_04.arxml

System

System fcb\_sys/newSystem

Target Package fcb\_sys

Mappings Communication

Cluster FlexrayCluster/A

ECU instance Ecu1

ISignal Flexray Protocol Configuration ISignal to IPDU packing IPDU to Frame packing Manual Scheduler

Slots

Static part Dynamic part

Type text to find Frames Next All

Frames	Channel	B.Cycle	Repetition	Calc. Rep. [ms]
slot 2				
slot 3				
slot 4				
slot 5				
newFrar A		8	8	120
slot 6				
slot 7				
slot 8				

Triggered frames

Type text to find Name Next All

Name	Channel	Deadline [ms]	Starting time [ms]	Repetition time...	Debounce time...
newFrame	A			200,00	400,00

Visualization

Cluster ECU

Time [ms] 0

Cycle/Slot	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
7 A	Tx																													
7 B																														
8 A					Tx																									
8 B																														
A	Tx																													

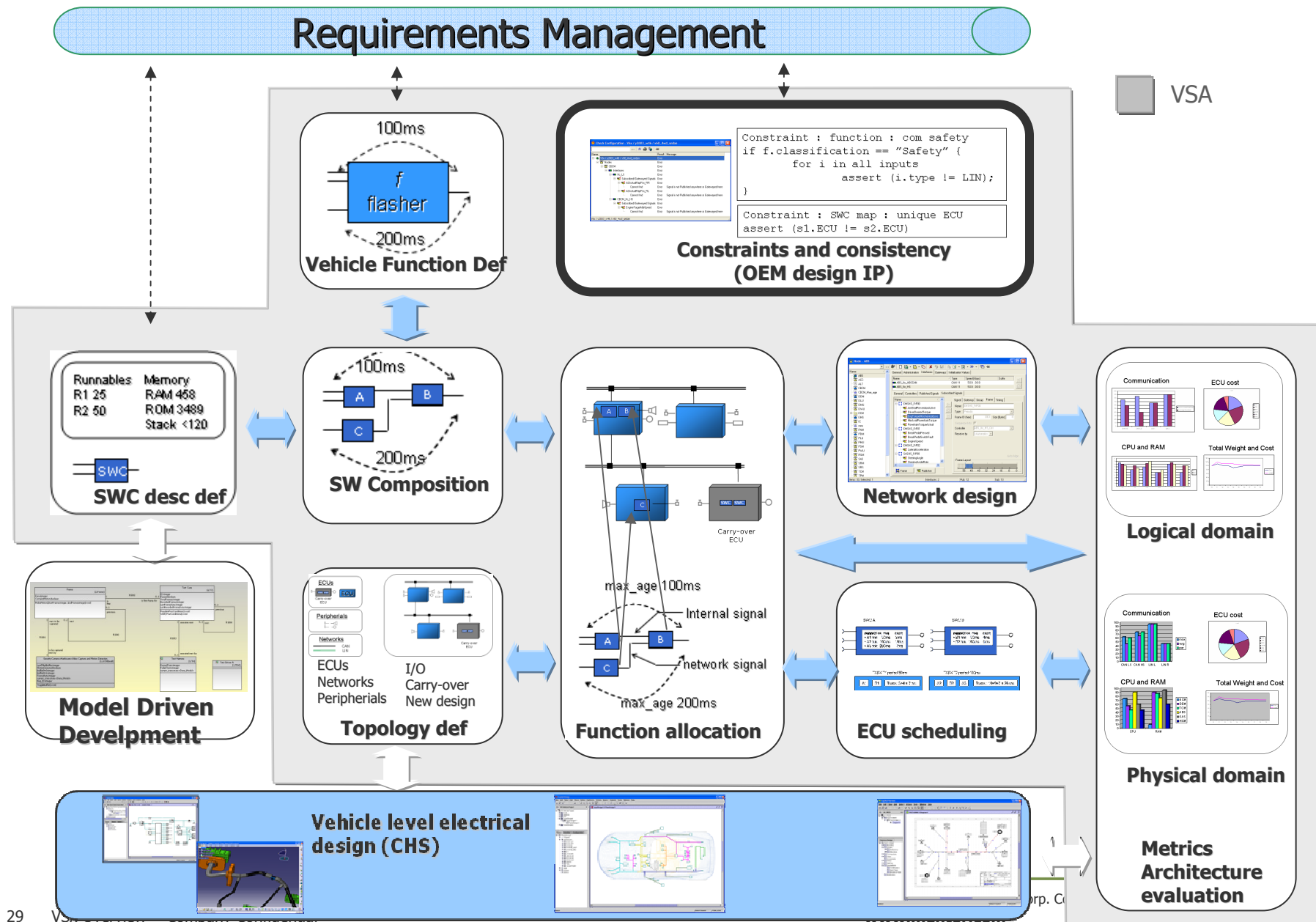
Frame name: newFrame  
Channel: A  
Slot Id: 5  
Basic cycle: 8  
Repetition: cycleRepetition8  
Debounce time: 400.0  
Repetition time: 200.0  
Tracking time: 120.0

Editor Dependencies Software Components System

startVSx Overview Company Confidential

architectur... vsa Presentation network de... 2 Microso... 4 Microso... System - F... untitled - P... 14:24

# VSA – Activities



# Script support

- Scripts can read/write from the VSA data model
- Implement custom consistency checks, reports, small features etc
- Can be used for almost any task, not only consistency checks

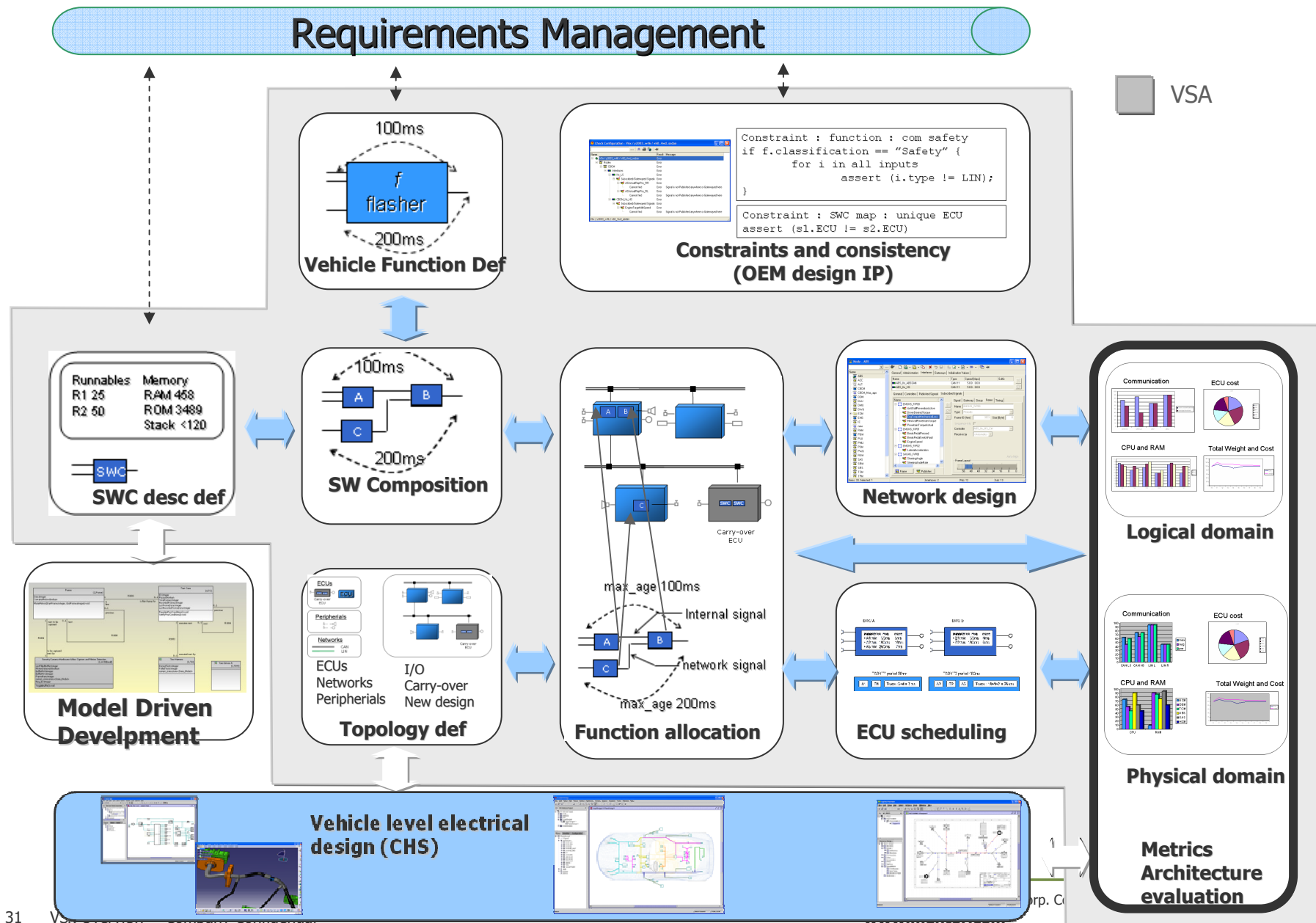
```
/**
 * Get list of SWCs for all packages
 * @return {[String, AtomicSoftwareComponentType, ARPackage]} Returns a list of SWCs
 */
function getSwcList()
{
    var root = ModelUtils.getModelRoot(am);
    // Get list of ARPackage
    var packages = root.getTopLevelPackage();
    var swcList = new Array();
    var total=0;

    // Loop all packages
    for(var i = 0; i < packages.size(); i++) {
        var pck = packages.get(i);
        var pckEl = pck.getElement();

        // Check each element in this package
        for(var j = 0; j < pckEl.size(); j++) {
            for (var c = 0; c < swcTempl.length; c++) {
                var element = pckEl.get(j);
                if(element instanceof swcTempl[c][1]) {
                    swcList[total] = [packElementBundle(pck,element), element, pck];
                    total++;
                }
            }
        }
    }
    return swcList;
}
```

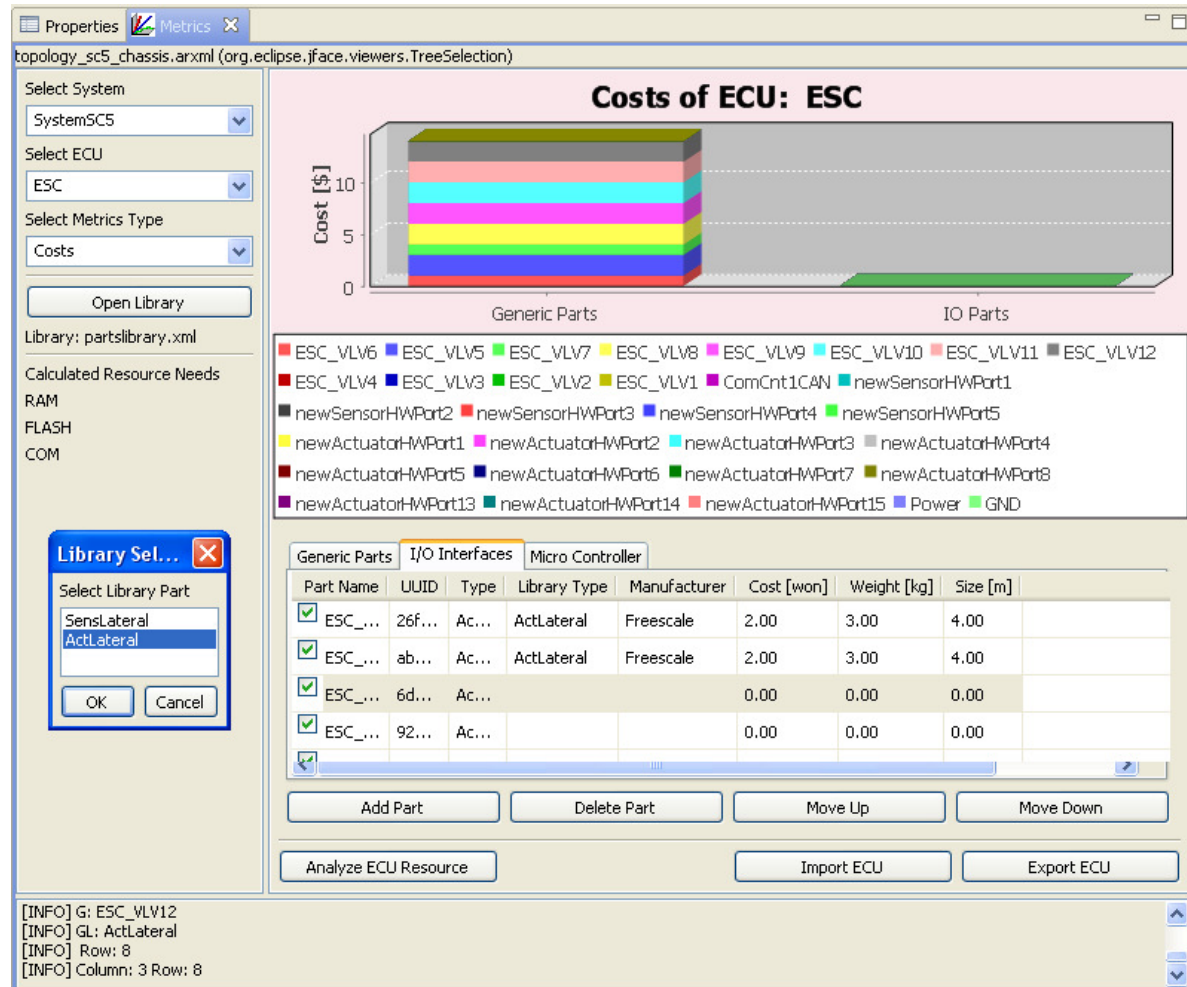


# VSA – Activities

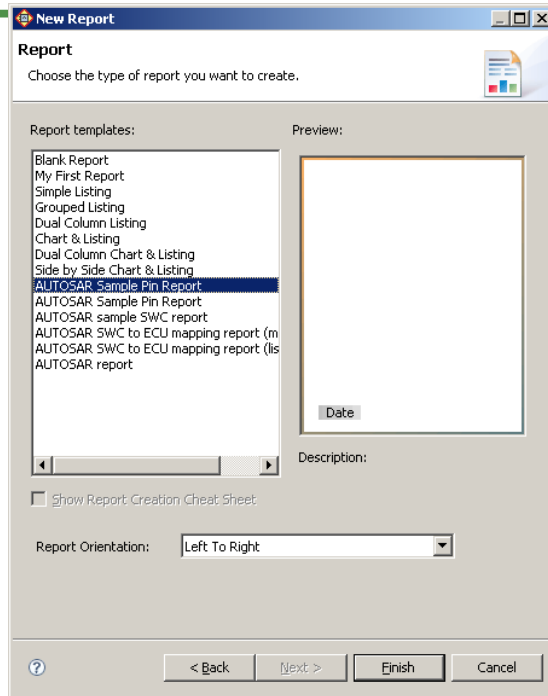


# Metrics Generator

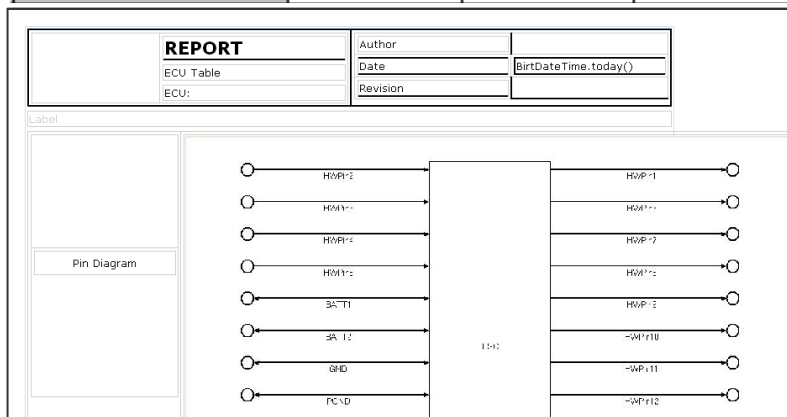
- Defined metrics concept
- Implemented metrics generator
- Parts library



# Configurable Report Engine

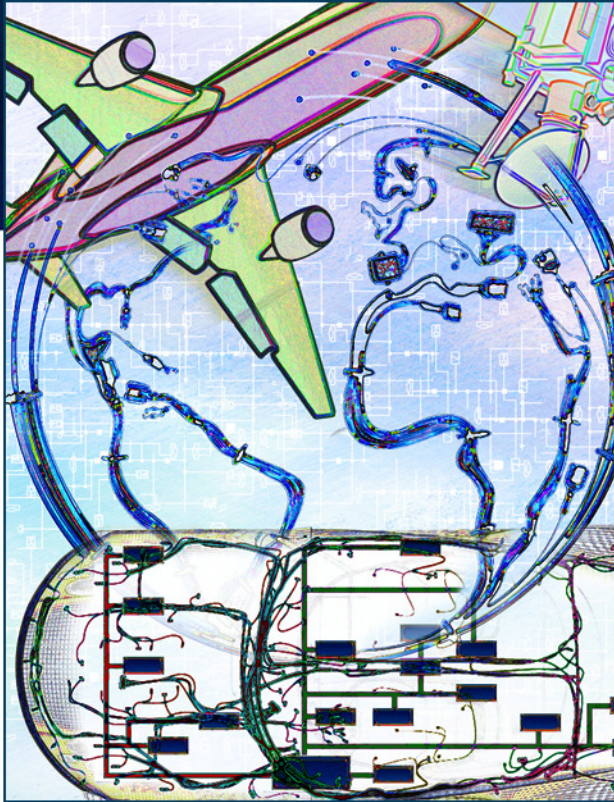


Vehicle	COST	WEIGHT	SIZE
SystemSC1	12.00	25.00	26.00
SystemSC2	11.00	27.00	25.00
SystemSC3	10.00	19.00	23.00
SystemSC4	5.00	13.00	11.00
SystemSC5	11.00	23.00	25.00
SteeringSystem	20.00	34.14	29.60
Target Value	44	99	



- User can configure content and layout
- Reads data from the VSA data model

- Present data in various formats
  - Table
  - Diagram
  - Pictures
  - Text
  - PDF, HTML, Word, Excel



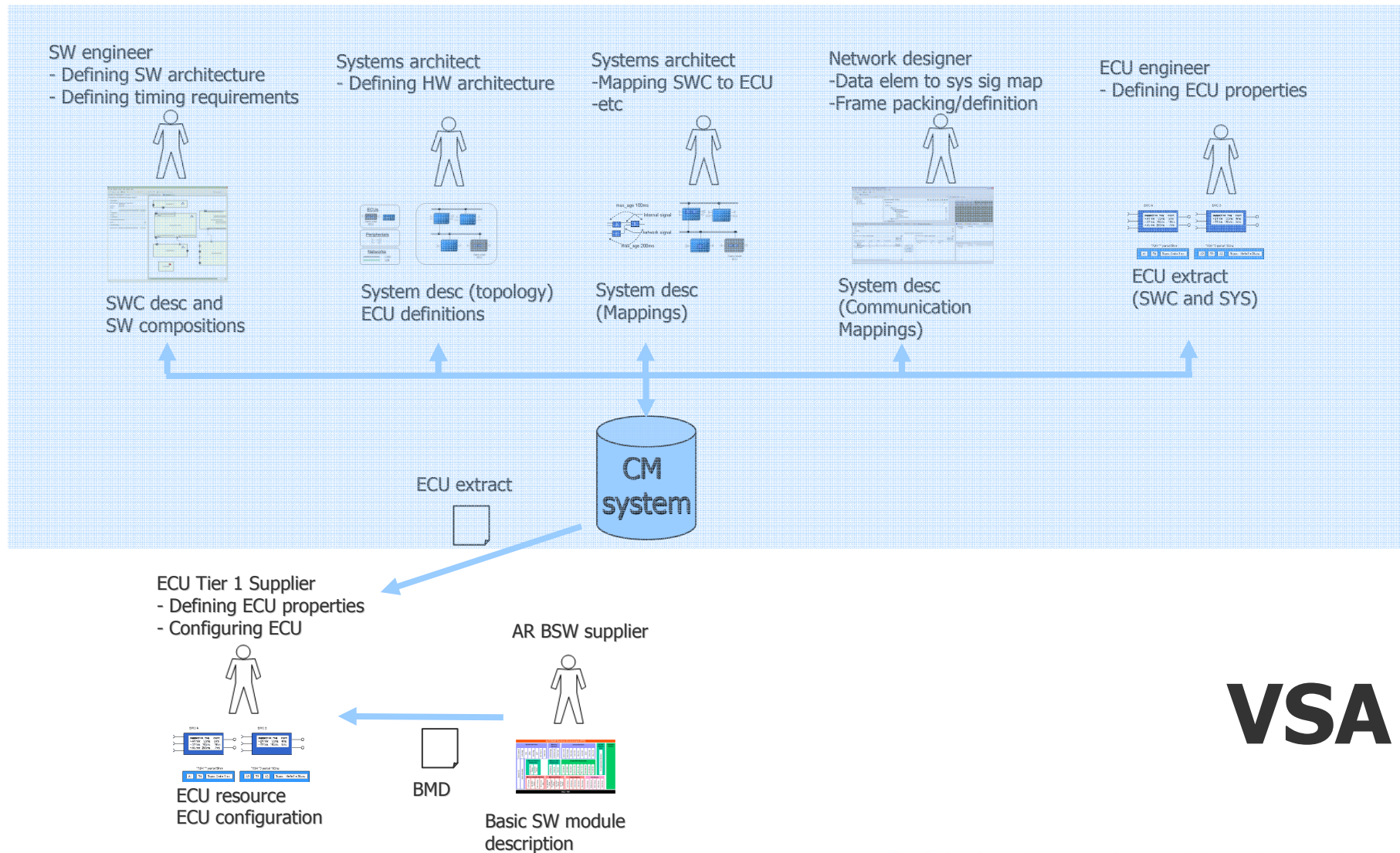
# Multi-user and other supporting features

Multi-user

Merge tool

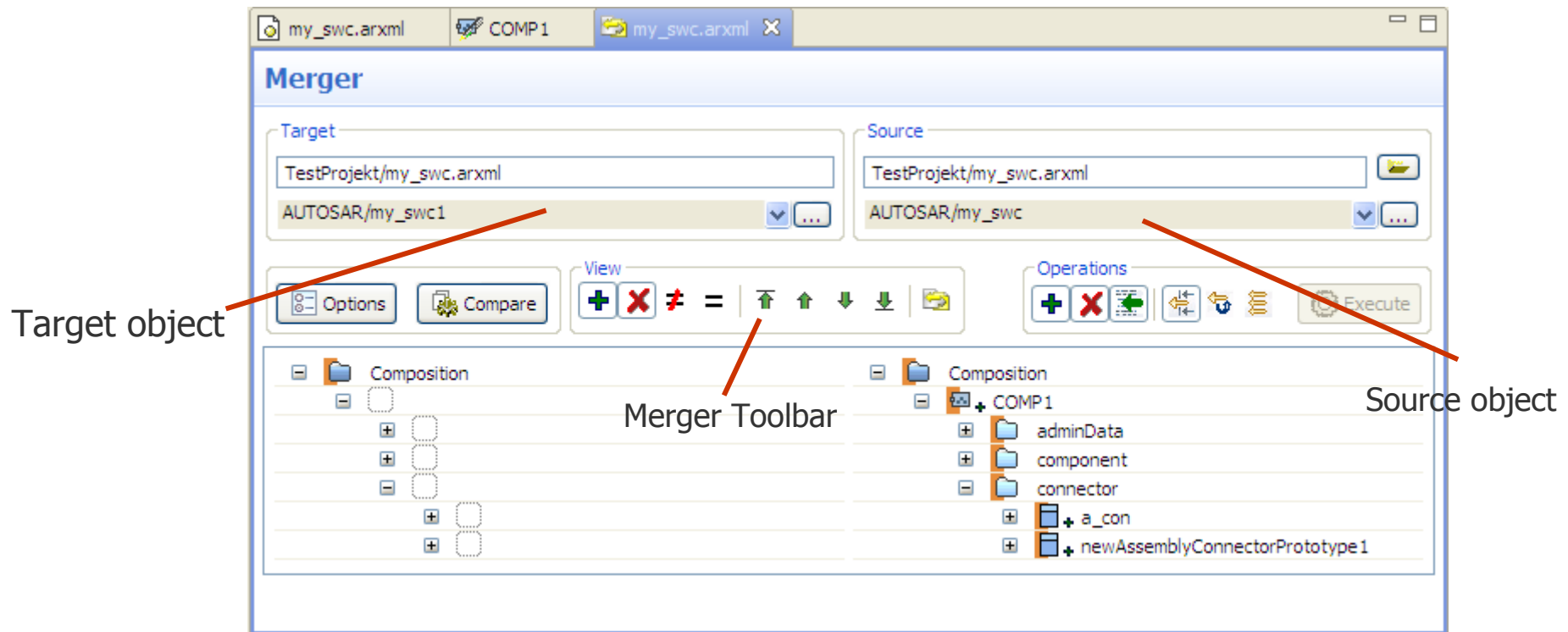
Timing model

# VSA and AUTOSAR – roles (simplified)



# VSA Merge Tool

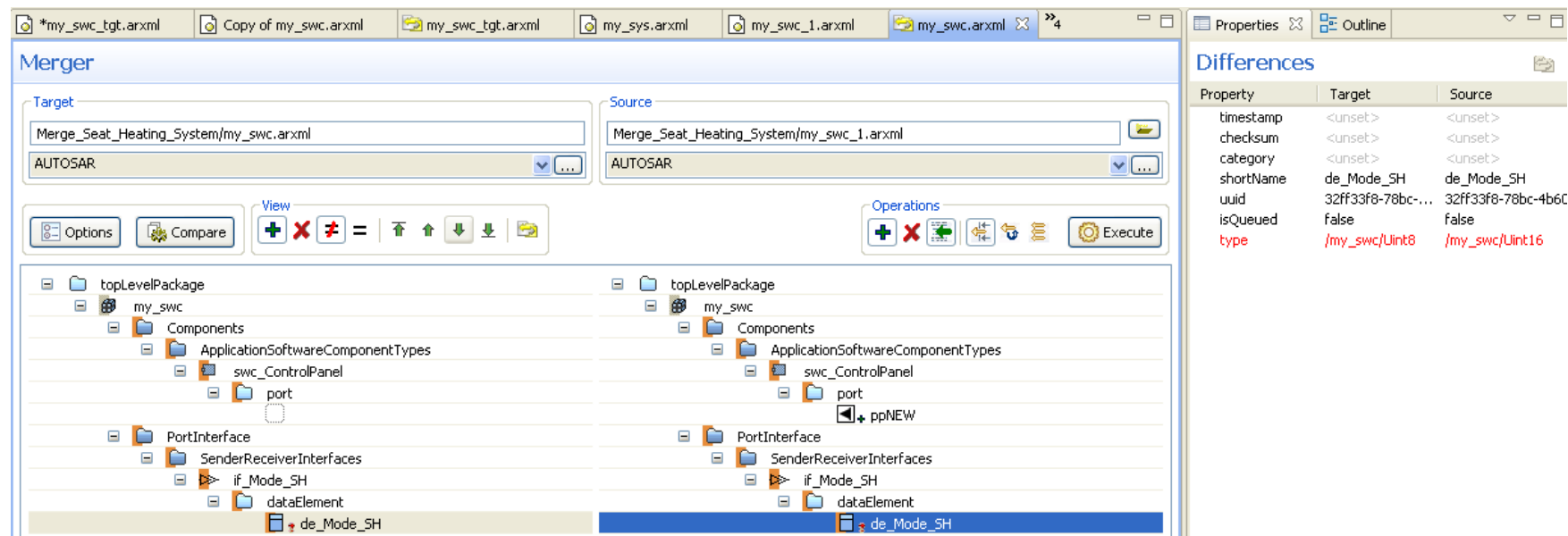
- The VSA merge tool allows merging objects that has been updated by different sources
  - Enables iterative development in separate tools
  - Enables merging data modified by supplier or OEM respectively



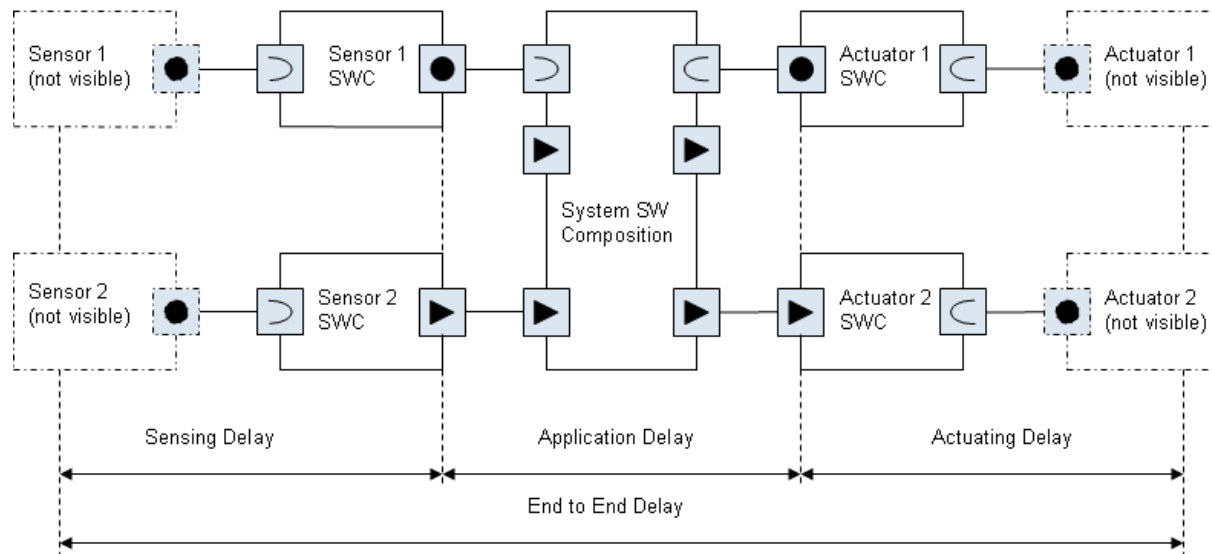


# VSA Merge Tool

- Displays difference of objects and allows merging selected elements and properties



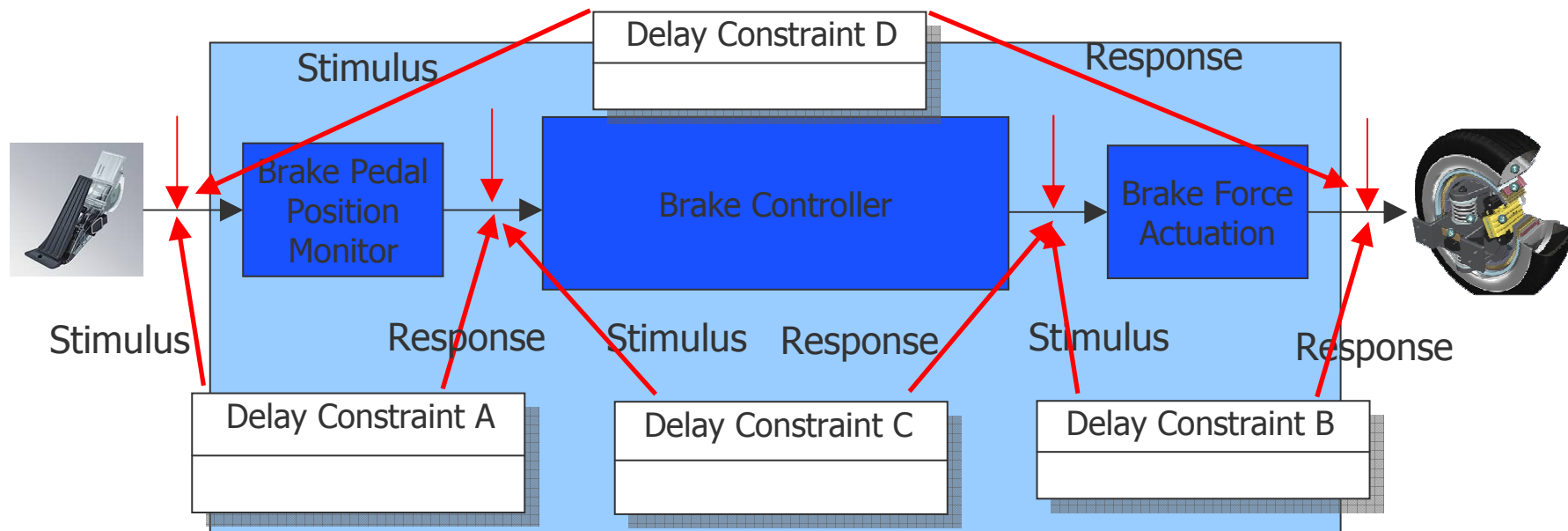
# VSx Timing Model

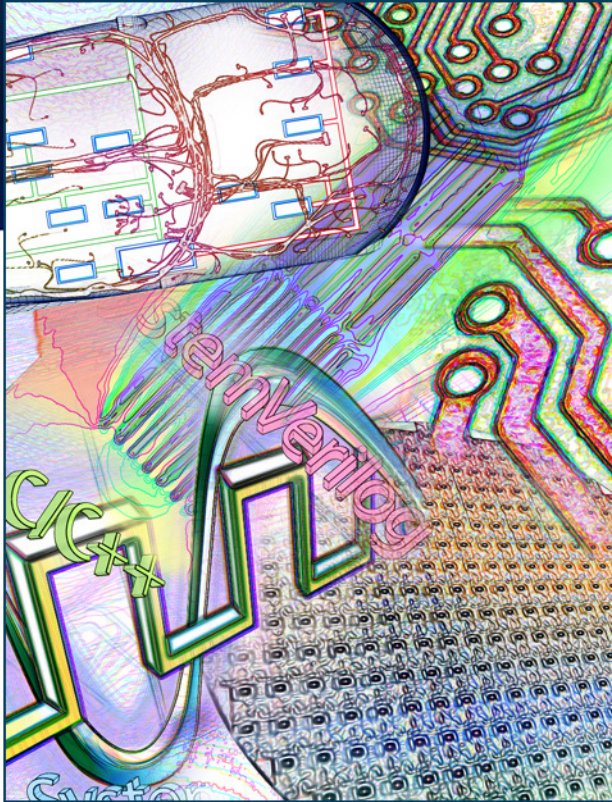


- Uses AUTOSAR and TIMMO (EAST ADL) timing model
- A full system timing model covering from sensor read to actuator effect
- Based on defining events and timing requirements between the events

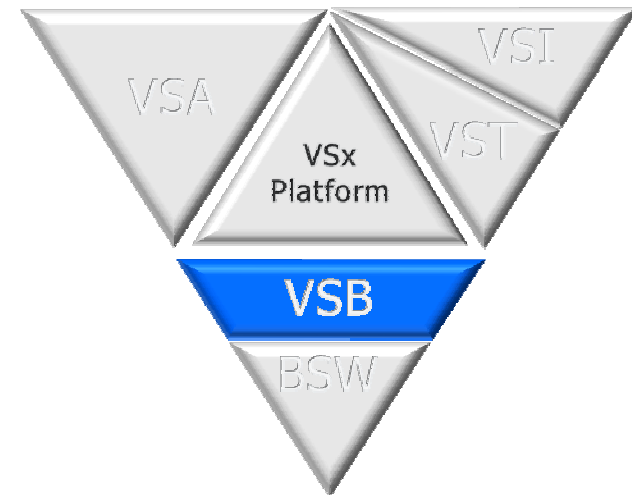
# VSx Timing Model

- Covers timing requirements of "Delay", "Synchronization" and "Repetition" type
- Event chains can be defined to cover true "end to end" timing requirements for a function





# ECU SW design and configuration - VSB

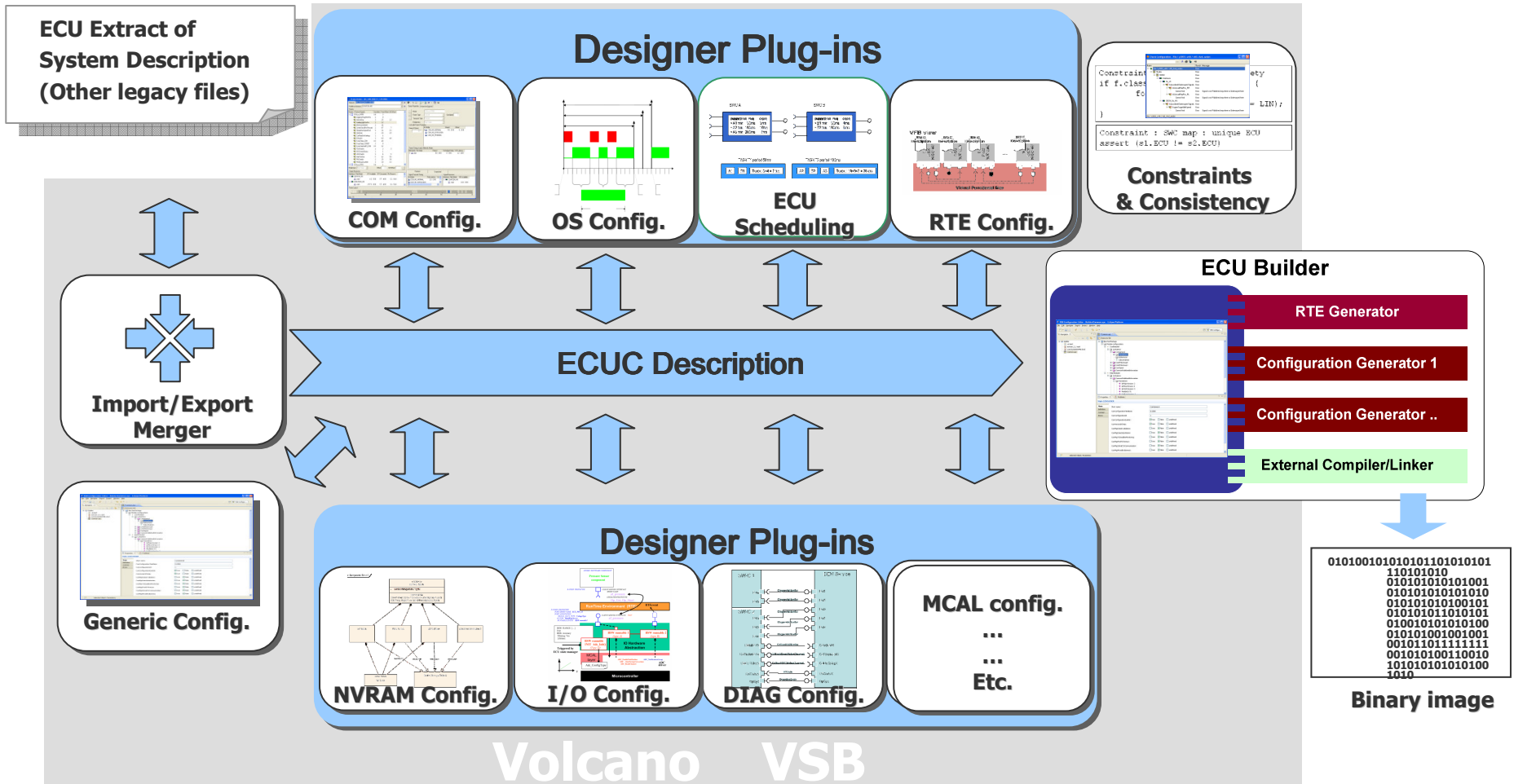


# Volcano Vehicle Systems Builder (VSB)

---

- Tool for ECU SW developers/integrators that need to:
  - Transform parameters from Extract of System description into ECUC parameters for a specific set of BSW modules
  - Configure BSW module ECUC parameters
  - Create BSW module descriptions
  - Integrate and build ECU software and BSW module software
  - Get design help for specific set of BSW modules
  
- VSB is containing several point tools, the most important are:
  - BSW module generic configuration editor
  - Designer Profile plug-ins (helps configuring a specific BSW module)
  - RTE generator
  - ECU project builder

# Volcano VSB





# VSB BSW Configuration Editor

## ■ Two types of Configuration Editor

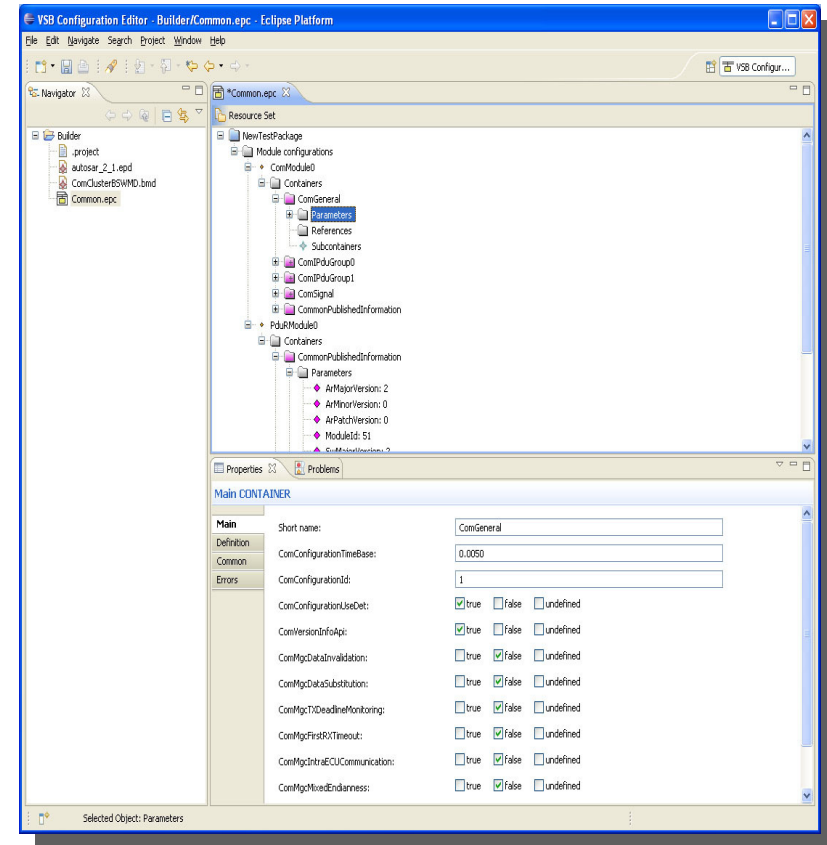
- Generic
  - A generic tree editor for advanced users
  - Supports configuring OEM specific modules
- Specialized (Designer Profile plug-ins)
  - Hides AUTOSAR complexity
  - Enhanced GUI
  - Targeting complex AUTOSAR modules:
    - Communication (CAN, LIN and FR)
    - NVRAM
    - OS, RTE, SchM
    - Diagnostics
    - I/O hardware abstraction
    - etc....

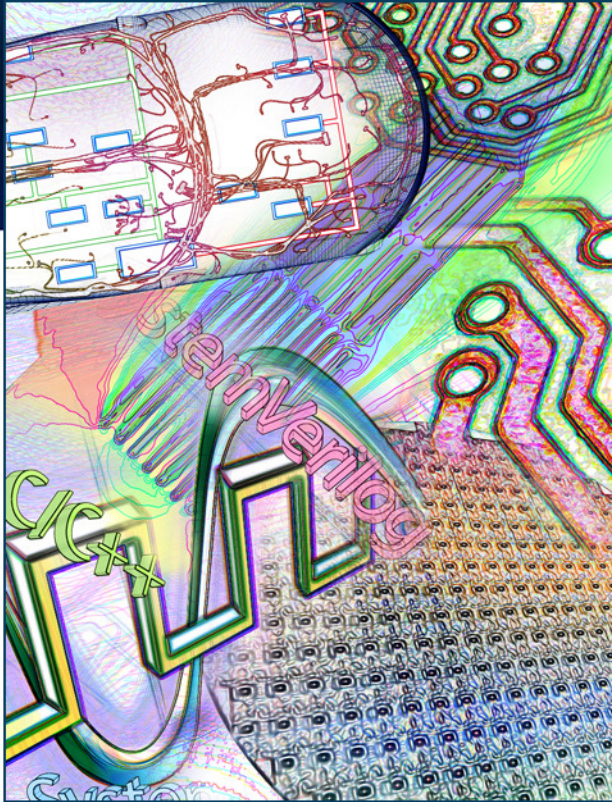
## ■ Consistency and constraints

- Automatic handling of AUTOSAR dependencies
- Supports User defined constraints

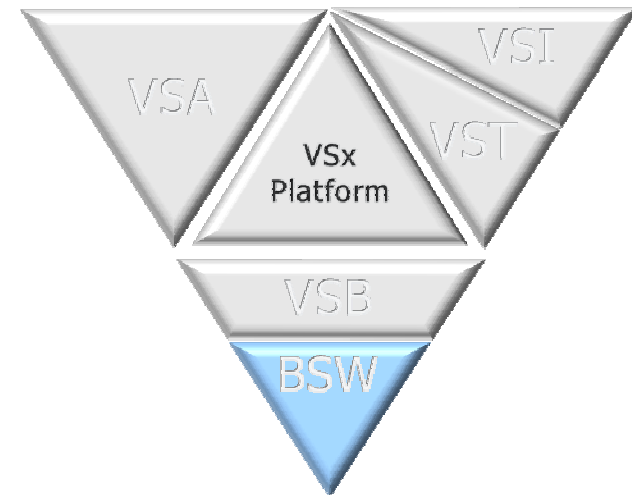
## ■ ECU builder plugin

- Integrated make system to minimize build effort





# Embedded Software or Basic Software - BSW



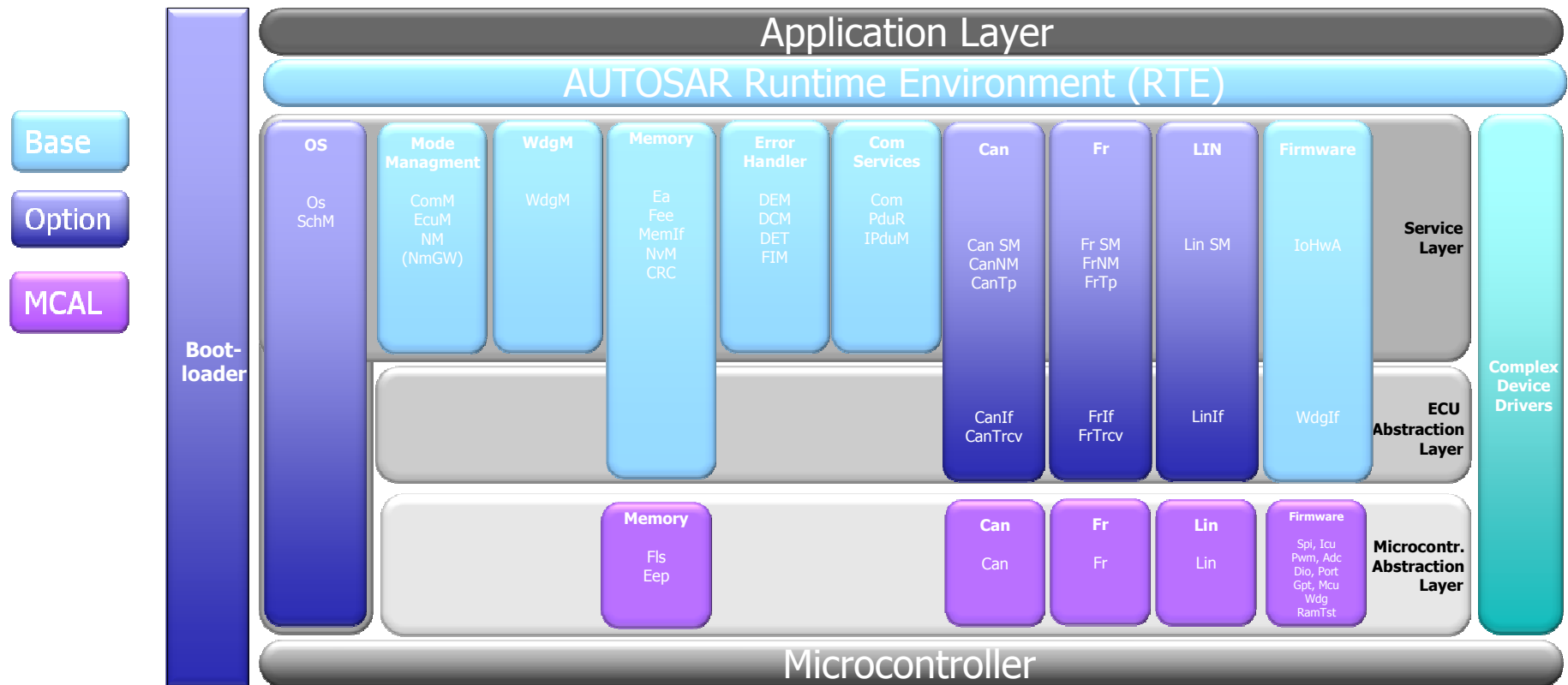
**Mentor**  
**Graphics®**

# AUTOSAR Basic Software Overview

---

- Currently at AUTOSAR version 3.0.2
  - Standard AUTOSAR BSW
    - Reference MCU: NEC V850 / GreenHills compiler
    - Complete stack available now
  - Optimized AUTOSAR BSW (Nano)
    - BSW stack footprint requirements: ROM < 130kB, RAM < 8kB
    - Reference MCU: S12XEP100 / Metrowerks CodeWarrior compiler
    - Release available Q1 2010
    - Major Nano configurations done = Minimized configuration effort
- Design/configuration tools
  - VSB – Configuration editor/design tool
  - VSC – configuration generators

# AUTOSAR BSW 3.0.2 stack

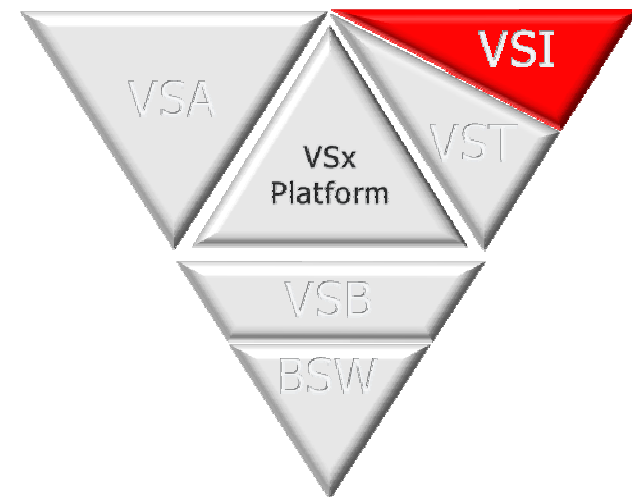
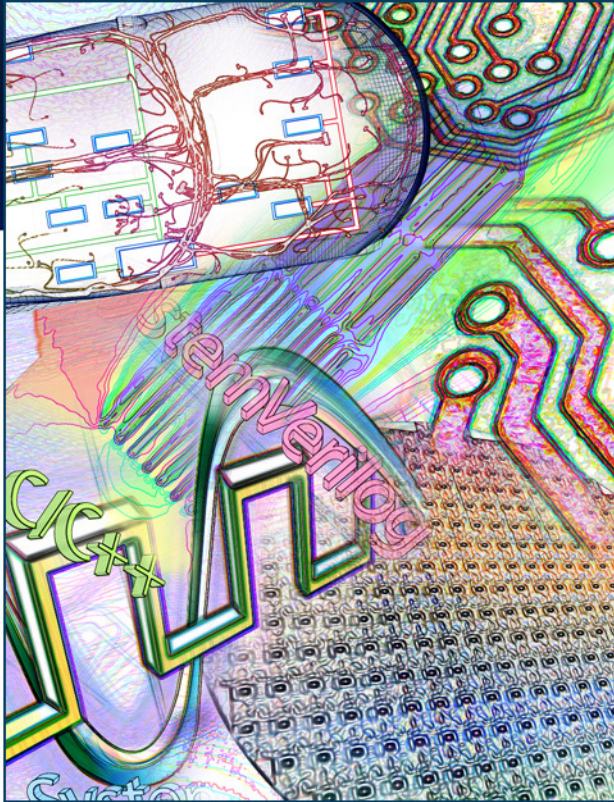


**BSW is ported to target hardware and delivered as fully validated object libraries**



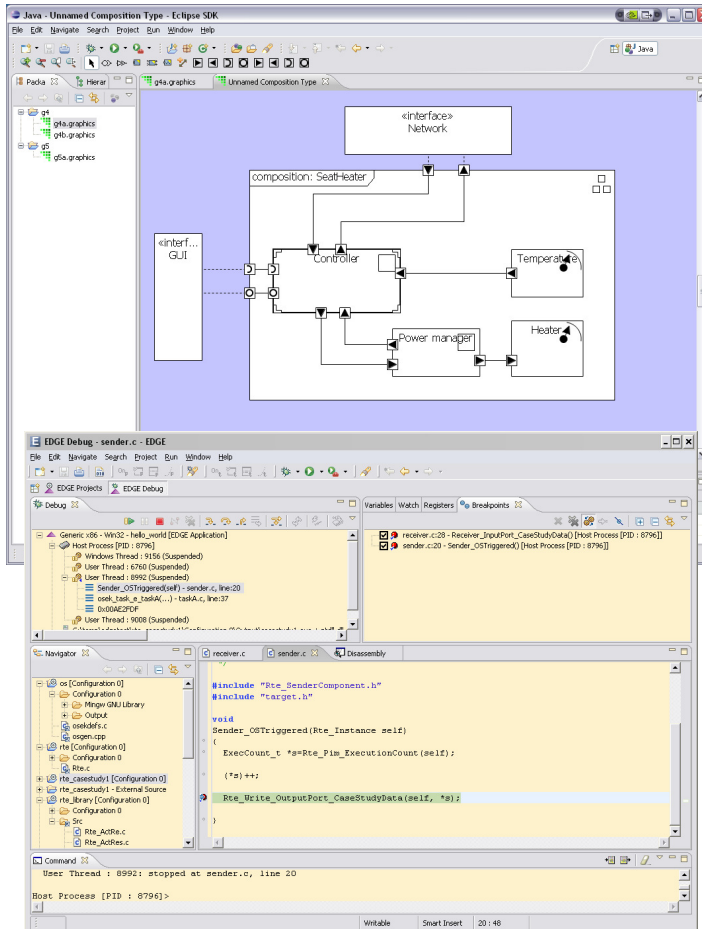
# VSI

## Vehicle Systems Integrator

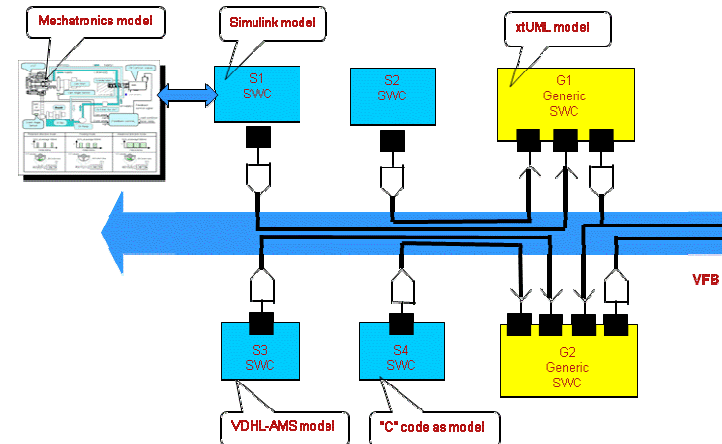


**Mentor**  
**Graphics®**

# Virtual Systems Integrator



- Multi-lingual multi-processor Model Driven Development environment
- Design verification and validation of embedded software in distributed systems
- Initial target: AUTOSAR
- Integrates C/C++, UML, Simulink, EDGE debug, etc



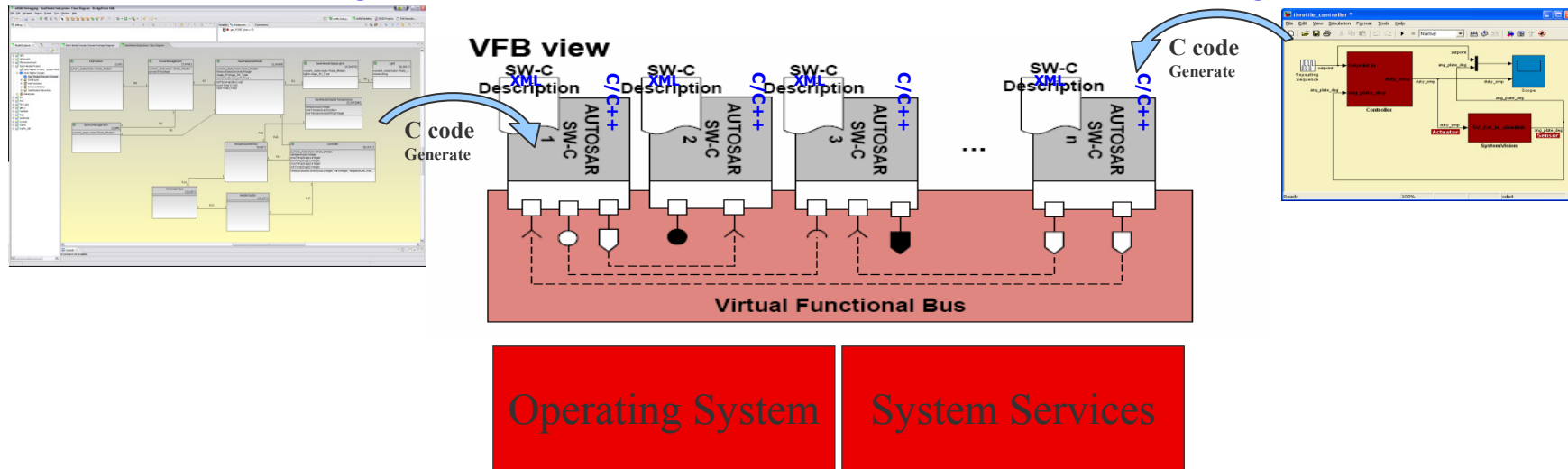


# VSI Application Software Development

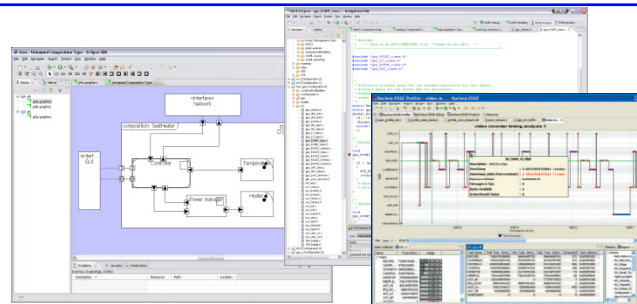
*Sourcing functionality*

Software Modeling

Algorithmic Modeling



Models use code generated from BridgePoint UML, Simulink, etc. or hand coding



# VSI Tool Suite

**Breakpoints**

**Profiler, Code and Data Trace**

**Software Component View**

**SWC and Composition**

**Runnable and Task**

**IDE, Debugger**

Name	Value
CaseStudy1Composition	
SenderInstance	
OSTriggered	RUNNING
OutputPort	
CaseStudyData	11
ExecutionCount	4
ReceiverInstance	
InputPort_CaseStudyData	TO_BE_STARTED
InputPort	

Name	Time (ms)	Usage
RECEIVER	1000000000	1000000000
SENDER	1000000000	1000000000
SYSTEM	1000000000	1000000000
CONTROL	1000000000	1000000000
MESSAGE	1000000000	1000000000
ACT_14	1000000000	1000000000
ACT_15	1000000000	1000000000
ACT_16	1000000000	1000000000
ACT_17	1000000000	1000000000
ACT_18	1000000000	1000000000
ACT_19	1000000000	1000000000
ACT_20	1000000000	1000000000
ACT_21	1000000000	1000000000
ACT_22	1000000000	1000000000
ACT_23	1000000000	1000000000
ACT_24	1000000000	1000000000
ACT_25	1000000000	1000000000
ACT_26	1000000000	1000000000
ACT_27	1000000000	1000000000
ACT_28	1000000000	1000000000
ACT_29	1000000000	1000000000
ACT_30	1000000000	1000000000
ACT_31	1000000000	1000000000
ACT_32	1000000000	1000000000
ACT_33	1000000000	1000000000
ACT_34	1000000000	1000000000
ACT_35	1000000000	1000000000
ACT_36	1000000000	1000000000
ACT_37	1000000000	1000000000
ACT_38	1000000000	1000000000
ACT_39	1000000000	1000000000
ACT_40	1000000000	1000000000
ACT_41	1000000000	1000000000
ACT_42	1000000000	1000000000
ACT_43	1000000000	1000000000
ACT_44	1000000000	1000000000
ACT_45	1000000000	1000000000
ACT_46	1000000000	1000000000
ACT_47	1000000000	1000000000
ACT_48	1000000000	1000000000
ACT_49	1000000000	1000000000
ACT_50	1000000000	1000000000
ACT_51	1000000000	1000000000
ACT_52	1000000000	1000000000
ACT_53	1000000000	1000000000
ACT_54	1000000000	1000000000
ACT_55	1000000000	1000000000
ACT_56	1000000000	1000000000
ACT_57	1000000000	1000000000
ACT_58	1000000000	1000000000
ACT_59	1000000000	1000000000
ACT_60	1000000000	1000000000
ACT_61	1000000000	1000000000
ACT_62	1000000000	1000000000
ACT_63	1000000000	1000000000
ACT_64	1000000000	1000000000
ACT_65	1000000000	1000000000
ACT_66	1000000000	1000000000
ACT_67	1000000000	1000000000
ACT_68	1000000000	1000000000
ACT_69	1000000000	1000000000
ACT_70	1000000000	1000000000
ACT_71	1000000000	1000000000
ACT_72	1000000000	1000000000
ACT_73	1000000000	1000000000
ACT_74	1000000000	1000000000
ACT_75	1000000000	1000000000
ACT_76	1000000000	1000000000
ACT_77	1000000000	1000000000
ACT_78	1000000000	1000000000
ACT_79	1000000000	1000000000
ACT_80	1000000000	1000000000
ACT_81	1000000000	1000000000
ACT_82	1000000000	1000000000
ACT_83	1000000000	1000000000
ACT_84	1000000000	1000000000
ACT_85	1000000000	1000000000
ACT_86	1000000000	1000000000
ACT_87	1000000000	1000000000
ACT_88	1000000000	1000000000
ACT_89	1000000000	1000000000
ACT_90	1000000000	1000000000
ACT_91	1000000000	1000000000
ACT_92	1000000000	1000000000
ACT_93	1000000000	1000000000
ACT_94	1000000000	1000000000
ACT_95	1000000000	1000000000
ACT_96	1000000000	1000000000
ACT_97	1000000000	1000000000
ACT_98	1000000000	1000000000
ACT_99	1000000000	1000000000
ACT_100	1000000000	1000000000

# Benefits of an AUTOSAR System Simulator

## ■ Excellent Collaboration

- OEM and Tier1 communicate around executable models
- Achieve early specifications
  - Required functionality is communicated unambiguously, before architectural decisions are made
- Achieve early integration and test
  - Specifications with integrated functions and verification suites increase chances of first-time success

## ■ Verification is fast and accurate

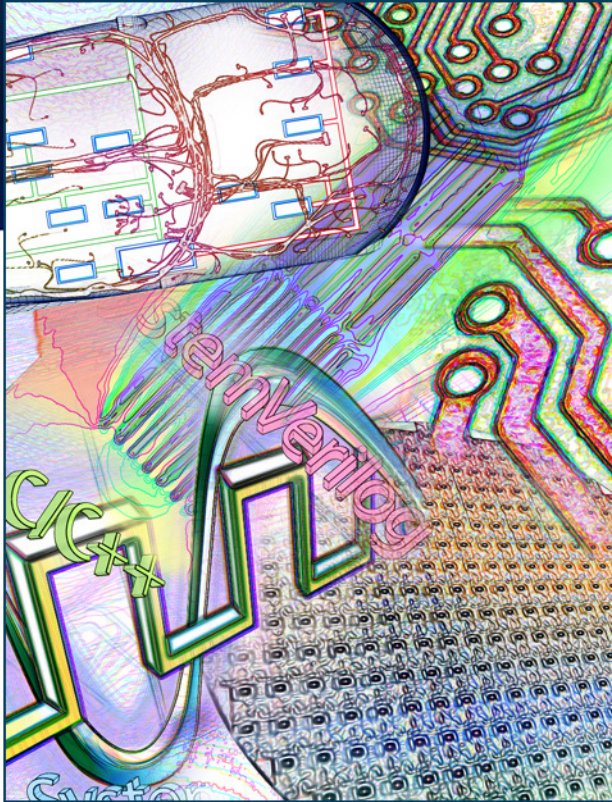
- Simulates at a high-level of abstraction

## ■ Integrate functions from multiple tools

- Functionality is best expressed in multiple domain-specific modeling languages

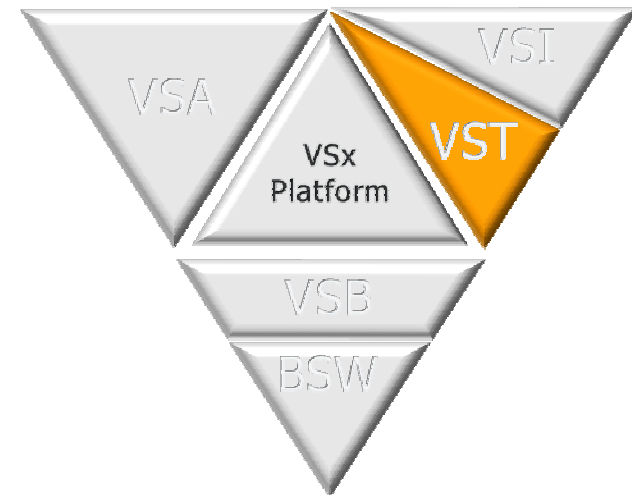
## ■ Use native modeling tools

- Developers remain in familiar tools, preserving investments into existing tools and libraries



# VST

## Vehicle Systems Test



**Mentor**  
**Graphics®**

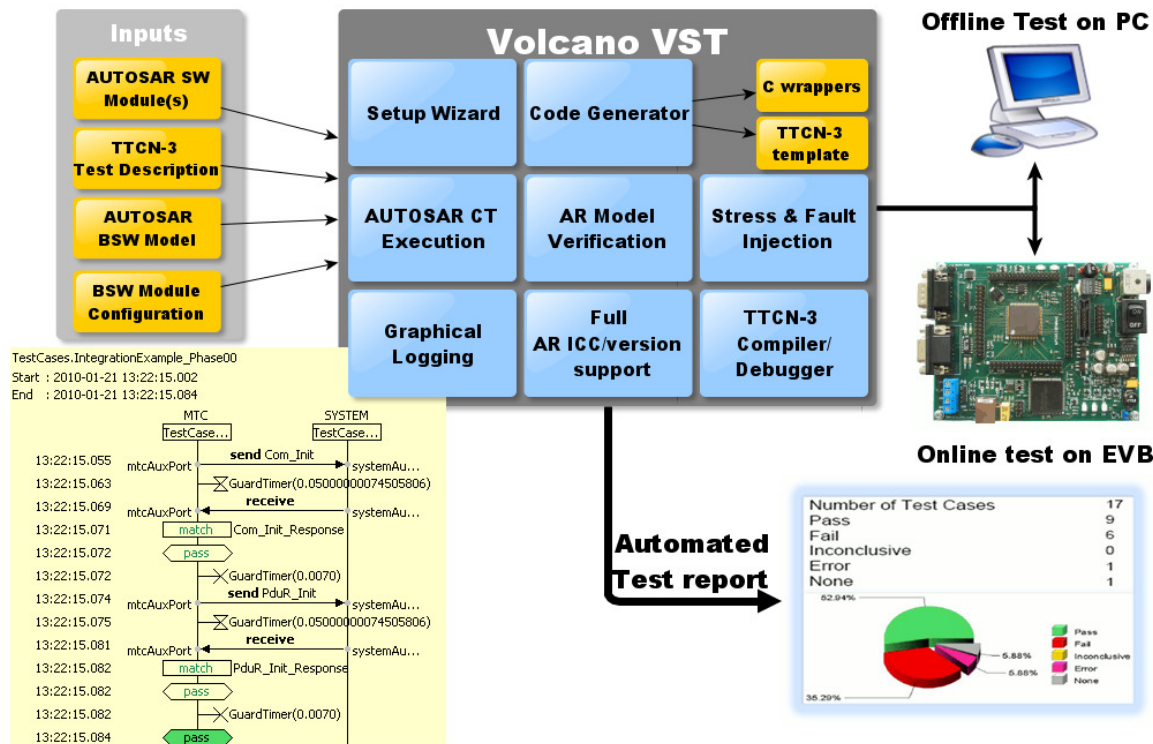
# VST

---

- Unified test environment for all AR BSW modules
- Full compliance with AR conformance tests (TTCN3 based)
- Minimized testing implementation effort
- Automatically generates
  - Upper/Lower layers stubs
  - TTCN-3 triggers/responses records
  - Test reports
- Minimized test execution time
- Achieves easy integration between BSW modules from different sources



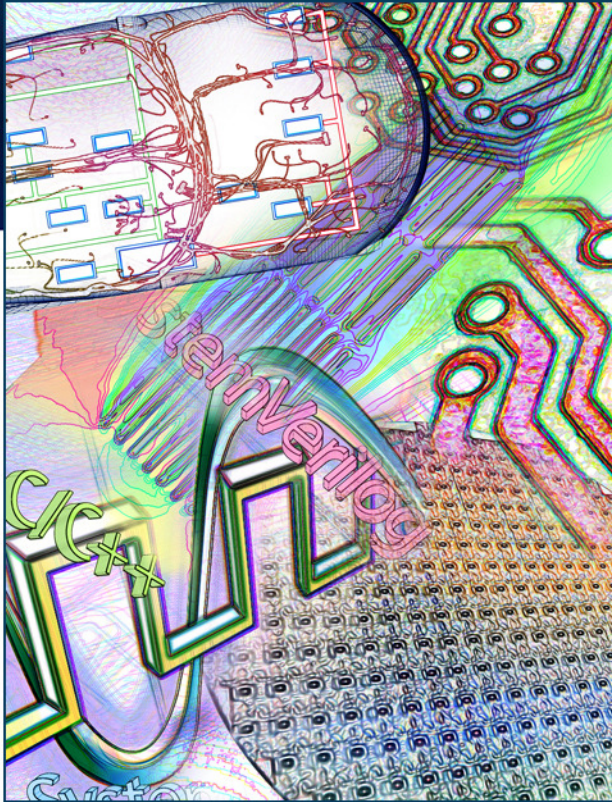
# VST – Vehicle Systems Tester



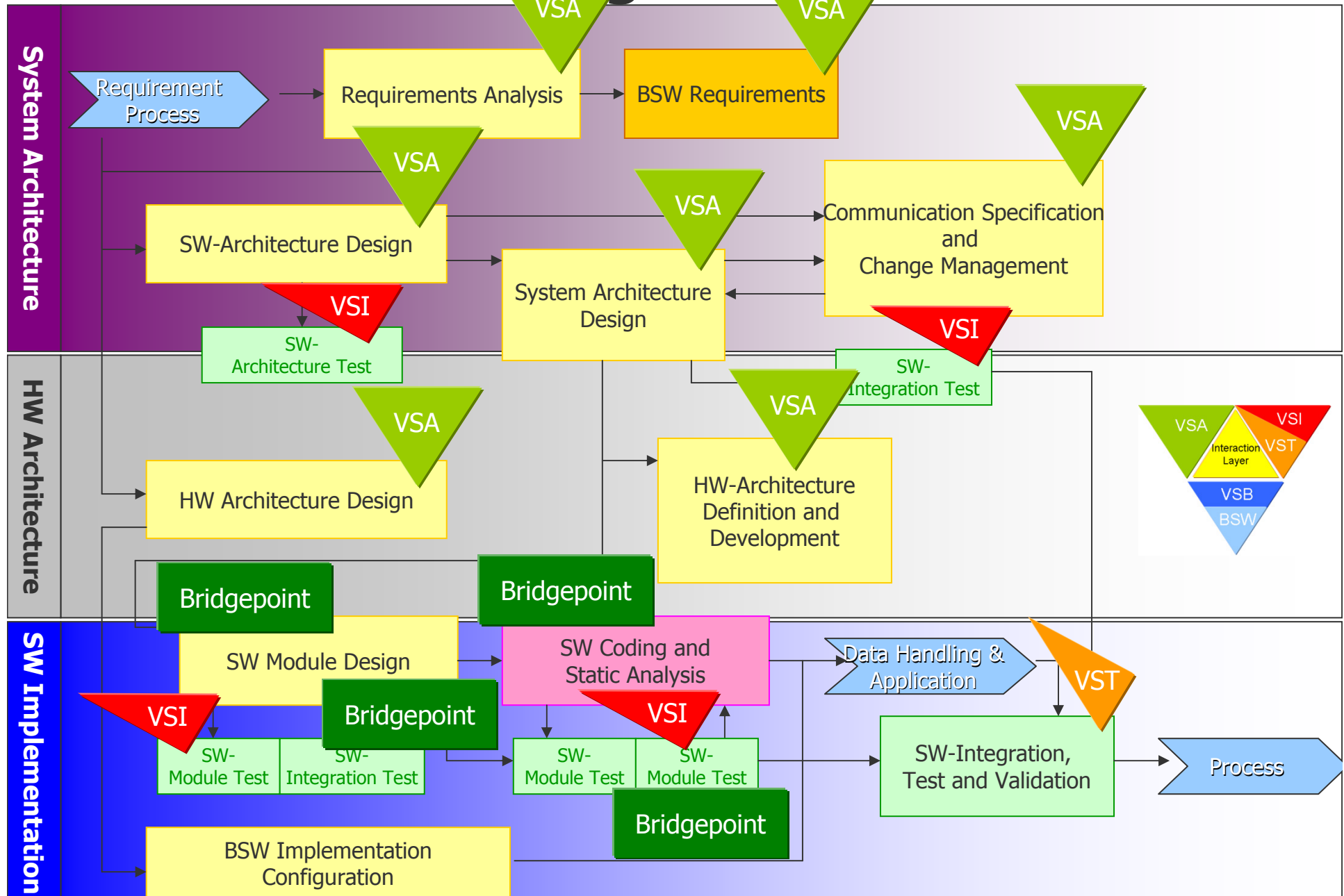
- One single environment for Test Development, Execution and Reporting.
- A generic test tool, not limited to AUTOSAR SW testing
- Direct execution of the AUTOSAR CTS
- Test Execution either on Target or on PC
- Based on well known and open Standard Language (TTCN-3) for Portability
- Automatic code generation of test object wrappers and TTCN-3 templates
- Automatic report generation



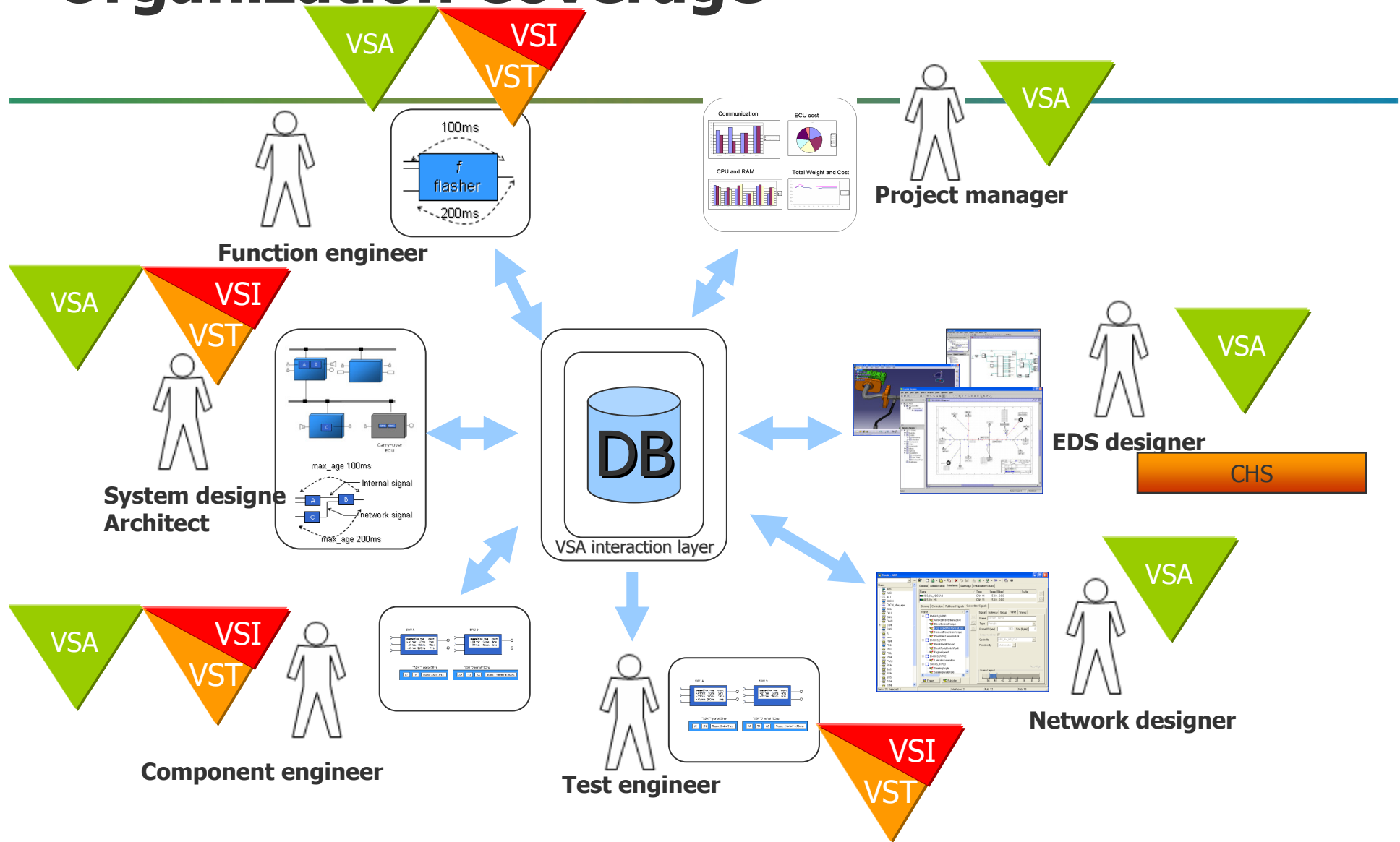
# Development Flow and Organizational coverage

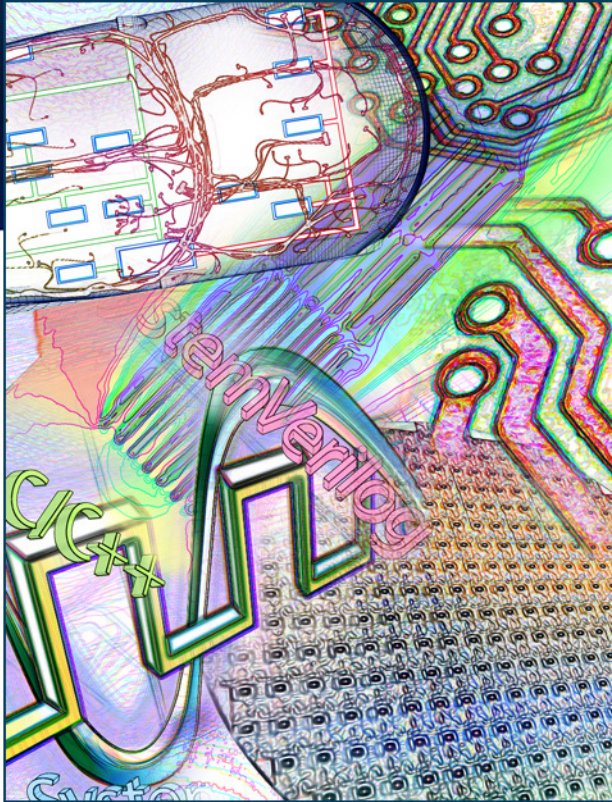


# EE Flow Coverage



# Organization Coverage





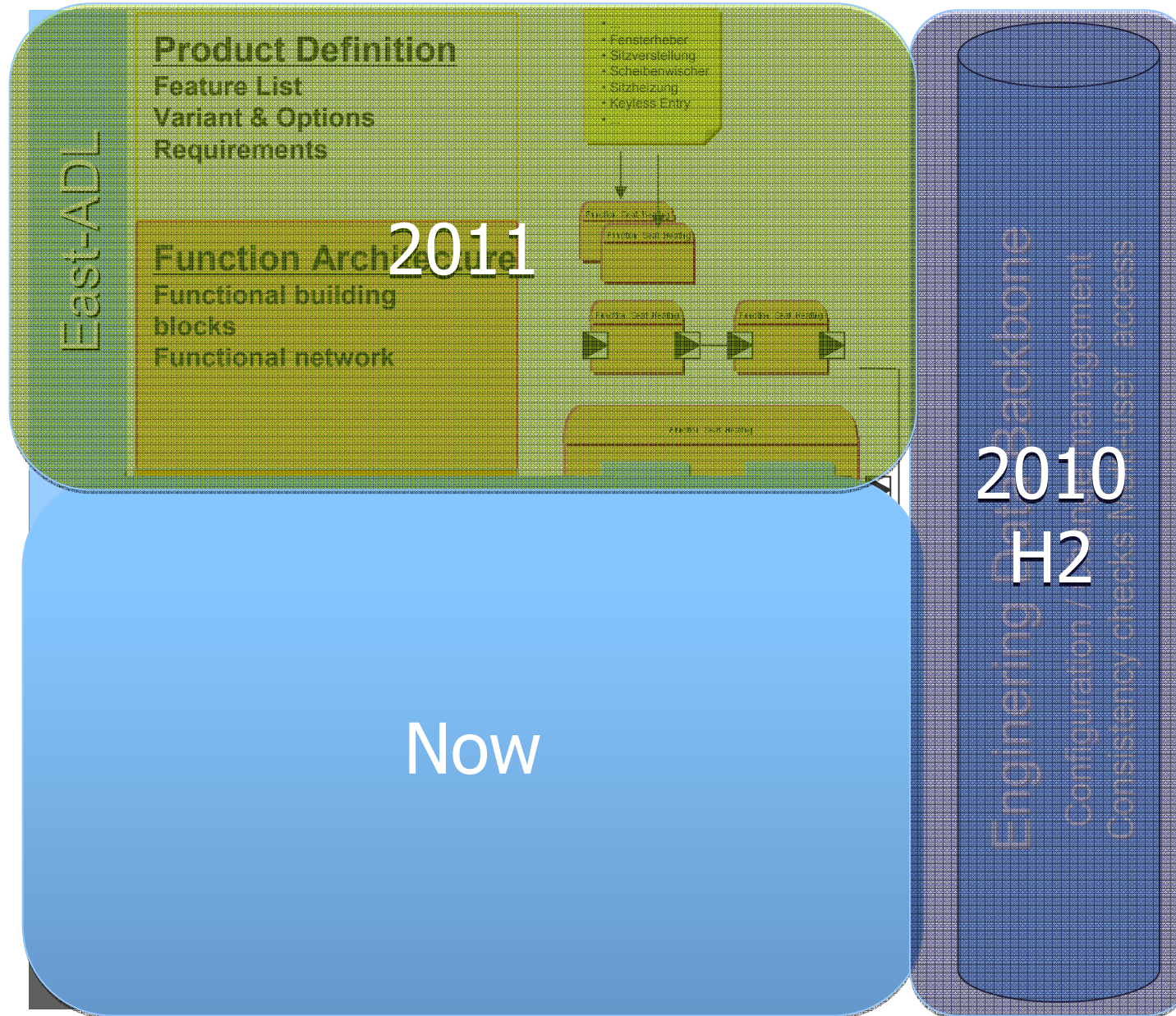
# Roadmap and direction



# Mentor Graphics Automotive Network Design – Short Term Direction

---

- Establish VSx Tool Chain with initial products VSA, VSA COM FlexRay, VST, VSI (Now)
- BSW and VSB AR 3.0 (Now)
  - FlexRay AR 3.0 Now
  - Rest (LIN, CAN, OS, RTE etc.) Q1 2010
- Add manual Network Design LIN, CAN, FlexRay, Ethernet (H1 2010)
- Introduce design automation
  - Network Design synthesis (H2 2010)
    - Algorithms for LIN and CAN from VNA
- Add support for massive multi-user capabilities with central data-repository (H2 2010)



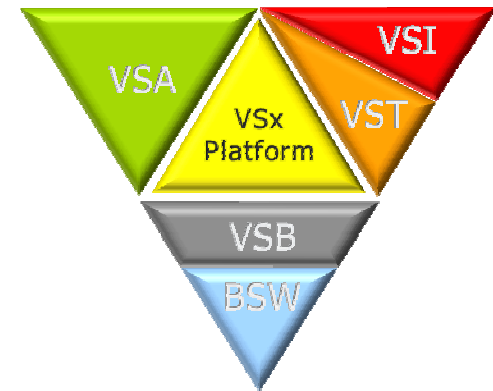


# Summary

---

## ■ Mentor Graphics Vehicle Systems Design Suite covers

- Architecture design
- Network Design
- ECU configuration, design and test
- VFB level simulation
- Implementation in ECUs



## ■ Overall goals

- Enable optimisation of the E/E architecture
- Enable (early) virtual verification of the system
- "Correctness by design"
- Digital Continuity from requirements to realisation



[www.mentor.com](http://www.mentor.com)