GENERAL MOTORS

ROBERT WIRTHLIN, PHD

EMBEDDED SOFTWARE IN PRODUCTS: THE CONVERGENCE OF ALM WITH SYSTEMS ENGINEERING

2018 Spring Meeting, PLM Center of Excellence, Purdue University
Exploring Application Lifecycle Management and Its Role in PLM

BOTTOM LINE UP FRONT

- Application Lifecycle Management will not succeed in today's complex environment without Systems Engineering
  - Model Based Systems Engineering is one of the key enablers for successful ALM execution
  - The Automotive industry has some unique attributes compared to other industries and is illustrative of the SE and ALM paradigm

INTRODUCTION

Model Based Systems Engineering (MBSE) has held the promise of both improved quality and efficiency in the engineering of large complex systems

It has rarely been a reality

Ease of use and accessibility of the modeled data have long been weakness in model based tool chains

GM is using large scale model based methods today, and we have been tackling the challenges of usability and data access and usage.

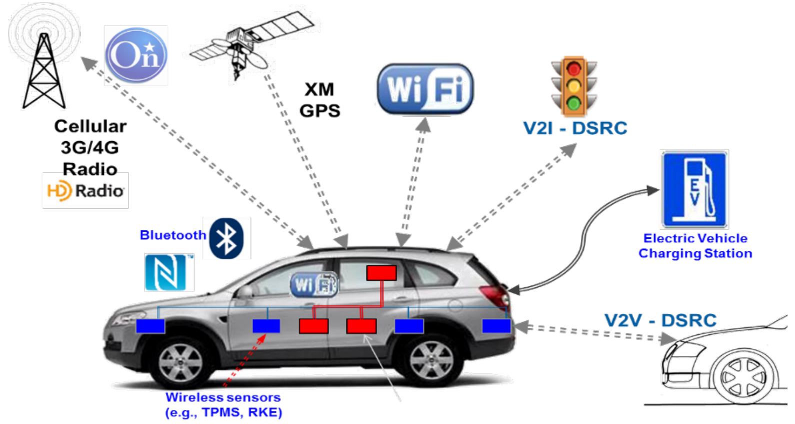
Today we will introduce what GM has accomplished in the ALM space and what we see as the next step to realize the promise of a Model Based methodology supporting the full engineering lifecycle

The Complex Automotive Problem Space

Variant Complexity

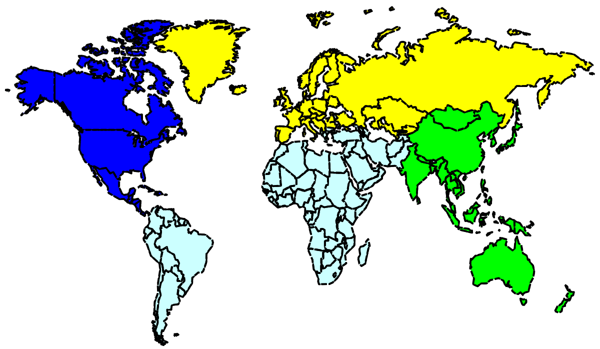
X

Increasing technical complexity



X

Increasingly stringent regulatory and diverse global market environment



Automotive Industry momentum toward Systems Engineering
. . . to deliver a safe, high quality product portfolio

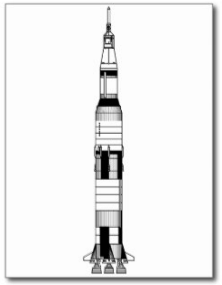
The more diverse the organization and the technology, the greater the value of a Systems Engineering approach

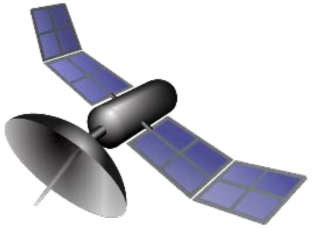
Systems Engineering Tailored to Automotive Industry

Systems Engineering in Automotive Industries is highly complex. It is not a simple textbook application.

Traditional SE applications in industries are characterized by:

- Large complex systems
- Relatively long life cycles
- Small production volumes





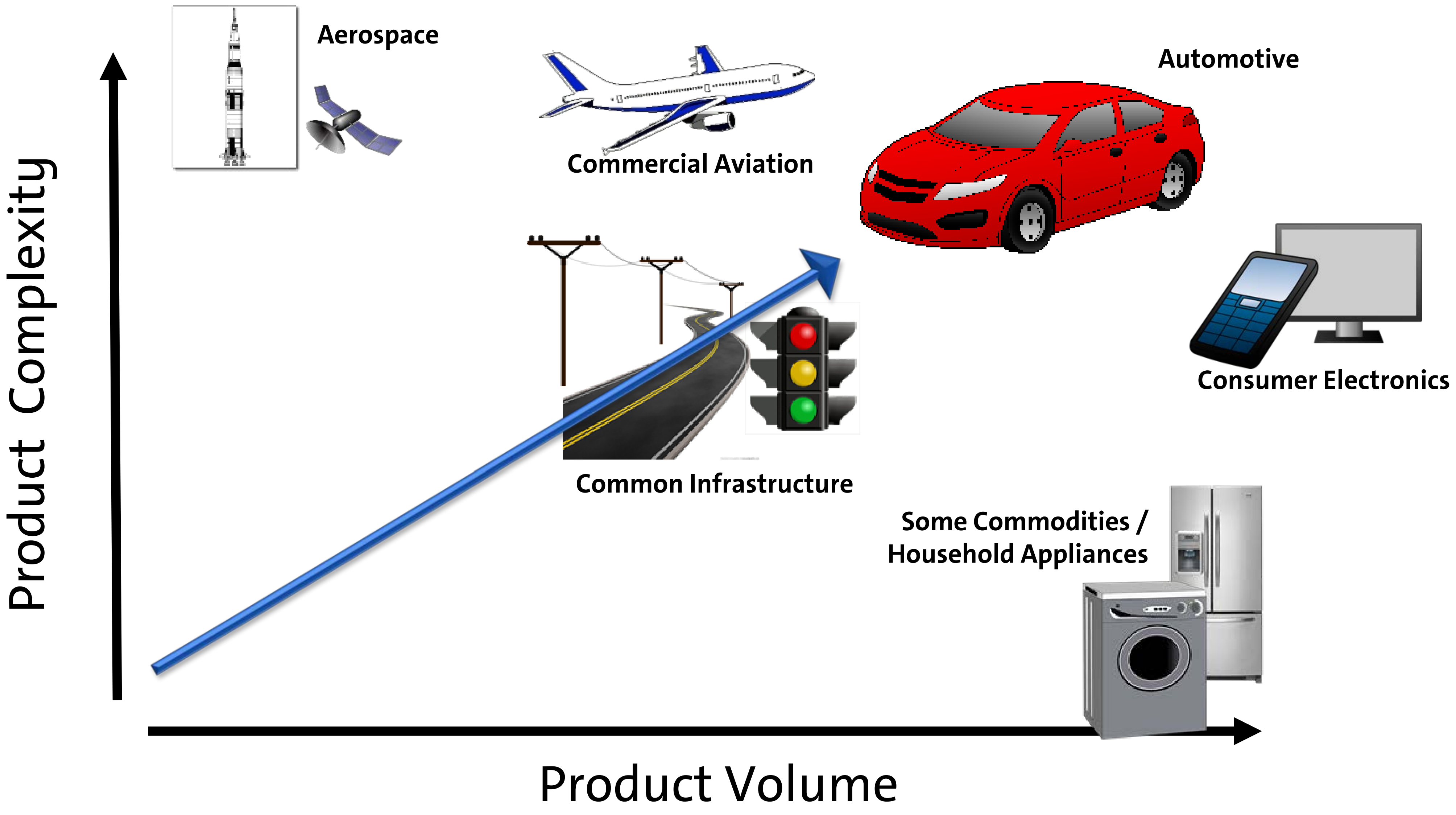
Automotive characteristics:

- High production volumes
- Increasing complexity
- Life cycle varies considerably within product
  - Example: Transmissions vs. Infotainment

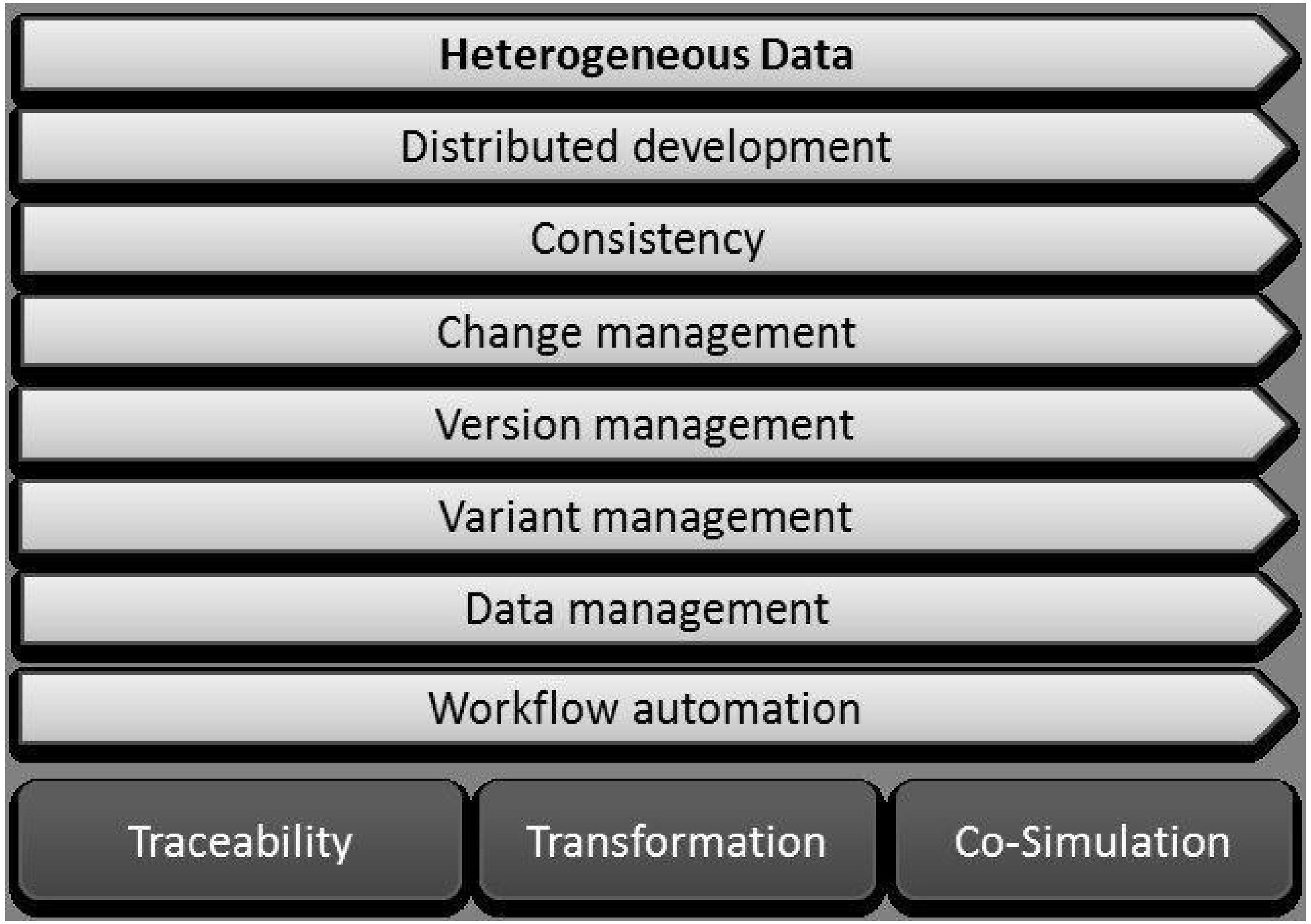
Automotive industry further characterized by:

- Extensive reuse of legacy engineering artifacts (requirements, architectures, validation plans)
- Management of huge product lines (at vehicle, system, and part levels)
- Ultra-competitive, high-capital business environment,

COMPARING INDUSTRIES AND SYSTEMS ENGINEERING



Harmonized Interoperability Challenges of the Automotive Domain



Traceability*, *Transformation*, and *Co-Simulation

From V3 Automotive Use Cases, CRYSTAL

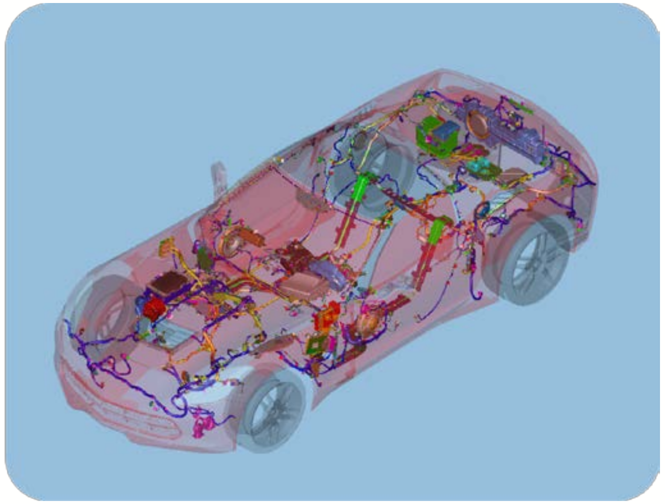
The Importance of Systems Engineering

"...the engineers had made all the parts function excellently, but when put together the whole was seldom satisfactory. So from here onwards, there was a continual struggle to improve the vehicle as a whole."

Maurice Olley, Chevrolet Director of Research and Development
February 16, 1957

- Speaking about conditions in the 1930s

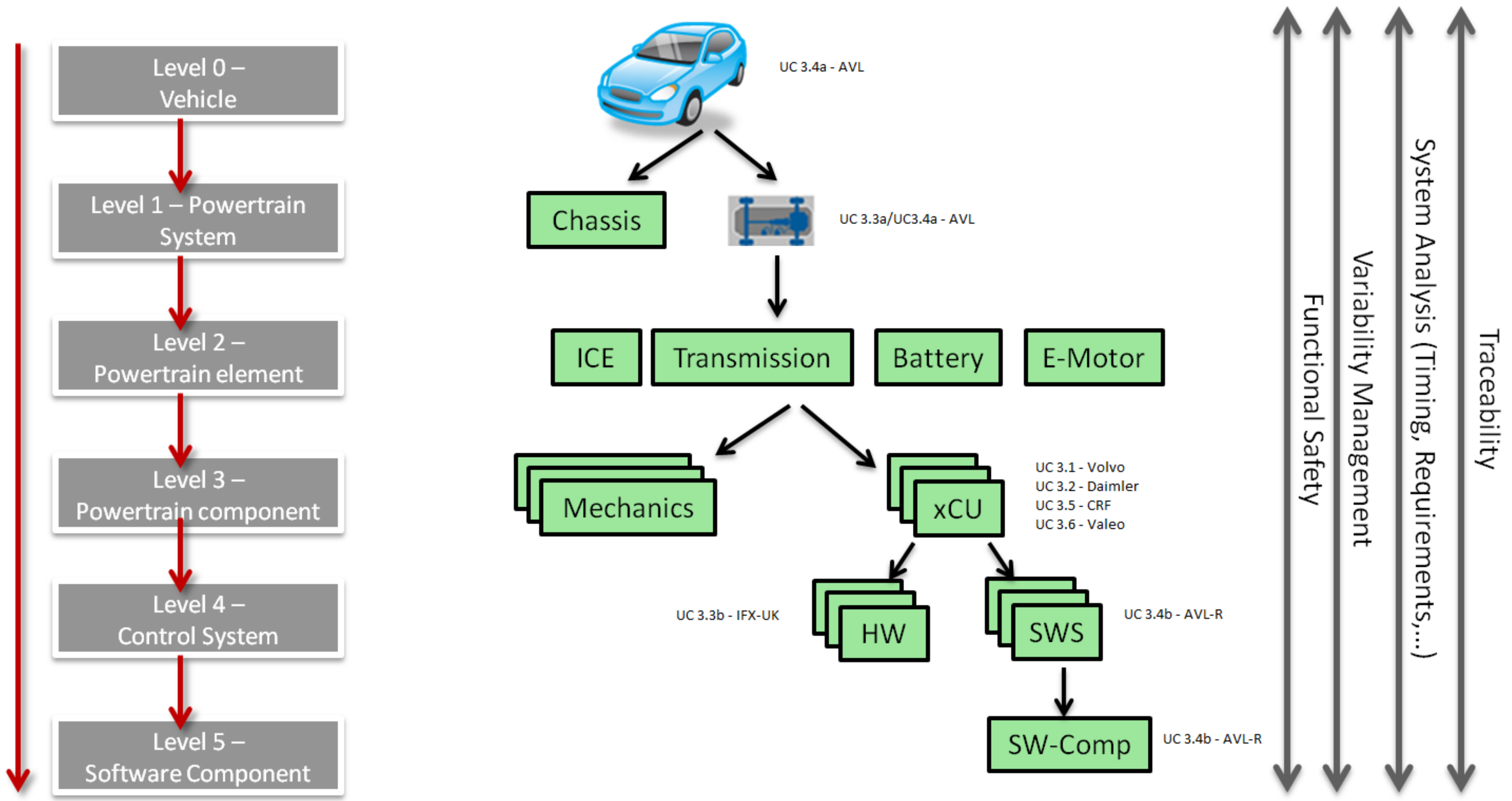
GENERAL MOTORS ENGINEERING CONTEXT



Modern Automobile

1,000,000s of product instances
100,000s of product variants
10,000s parts
100s Electronic Control Units
10,000,000s Lines of Code
10s Networks
10,000s Network Signals
10,000s Engineers

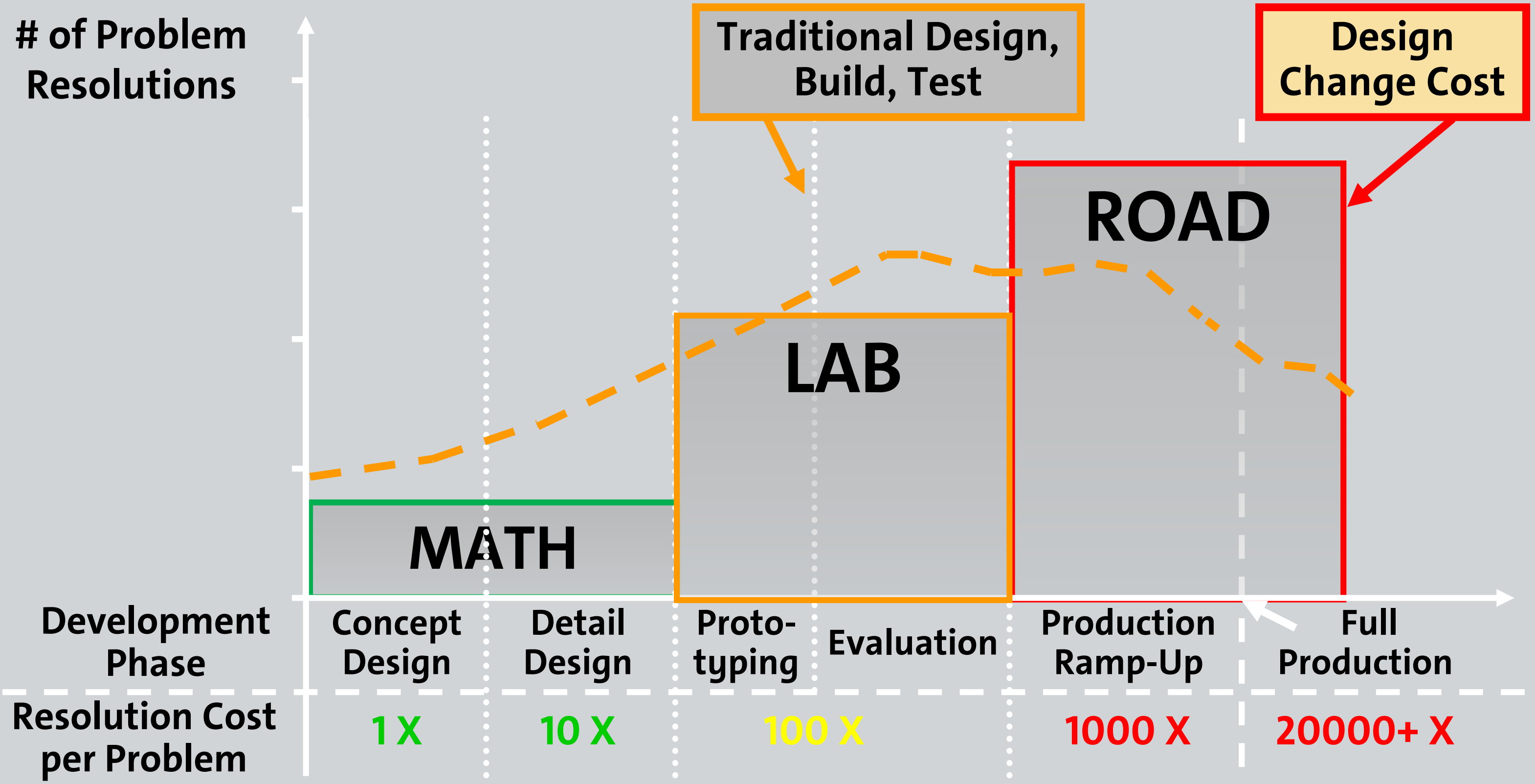
Vehicle decomposition in engineering and design



***From V3 Automotive Use Cases, CRYSTAL**

ROAD TO LAB TO MATH TRANSFORMS PROCESS

Frontloading Engineering with
First Time Capable lab and math processes
reduces Engineering costs



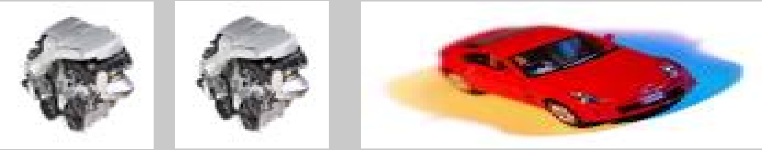
Source: General Electric 2003

RLM improves product development time

"RLM reduced the time necessary to create our initial calibrations by nearly 40%, saving us **<u>10 weeks</u>** of critical development time in our programs."

- Results come from the following improvements:
  - Automations of test procedures
  - Statistical and CAE methods
  - High quality data checking on the dynamometer

8 weeks dyno+18 wks veh=26wks



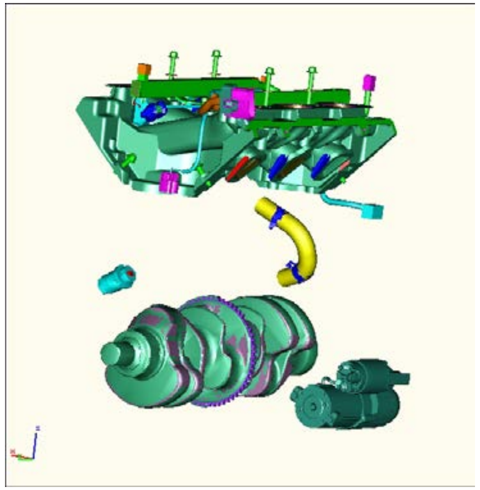
16 wks in dyno and analysis
minimizing vehicle testing

- Savings will increase as we shift more work away from physical validation

Model Types used

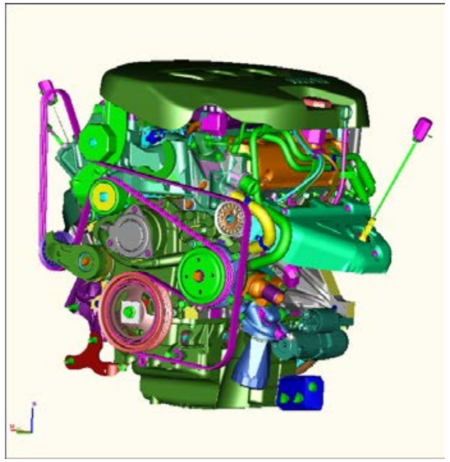
Component CAD Models

- Represents individual parts or sub assemblies in the as-manufactured state.
- Includes tolerance information in models in addition to nominal geometry.
- May also include alternate representations of the model to depict as-installed condition.

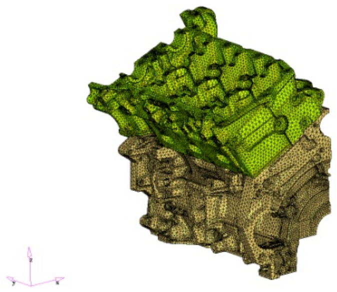


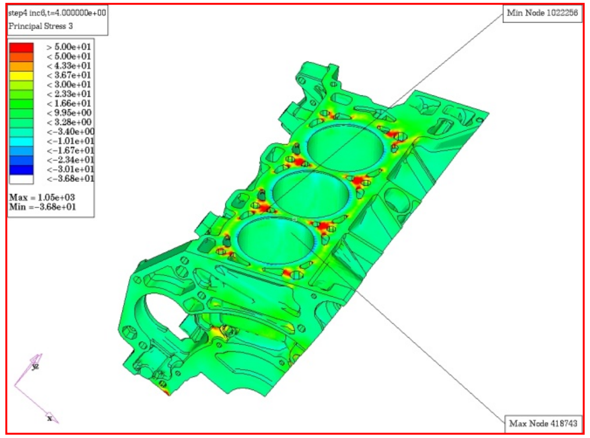
Engine Installation Assemblies

- Are packaging models for an engine application.
- Are assembled from Component CAD Models.



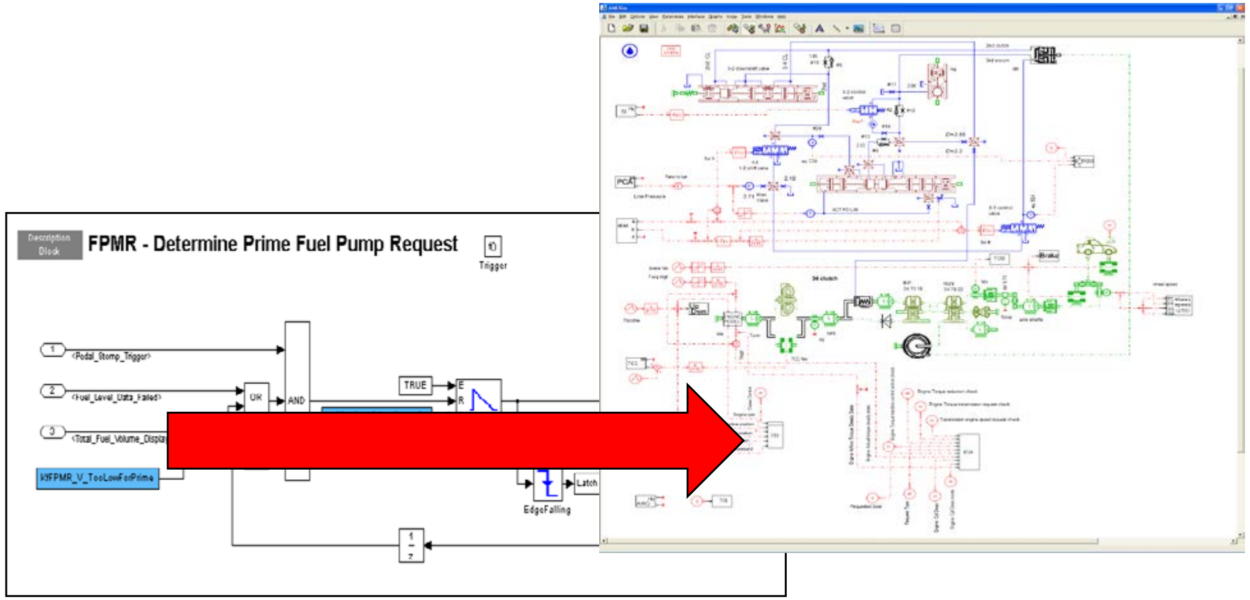
MODEL TYPES USED





CAE Models

- Represent systems, sub-assemblies and individual parts
- Are created and revised by the analysts as needed from CAD Models.

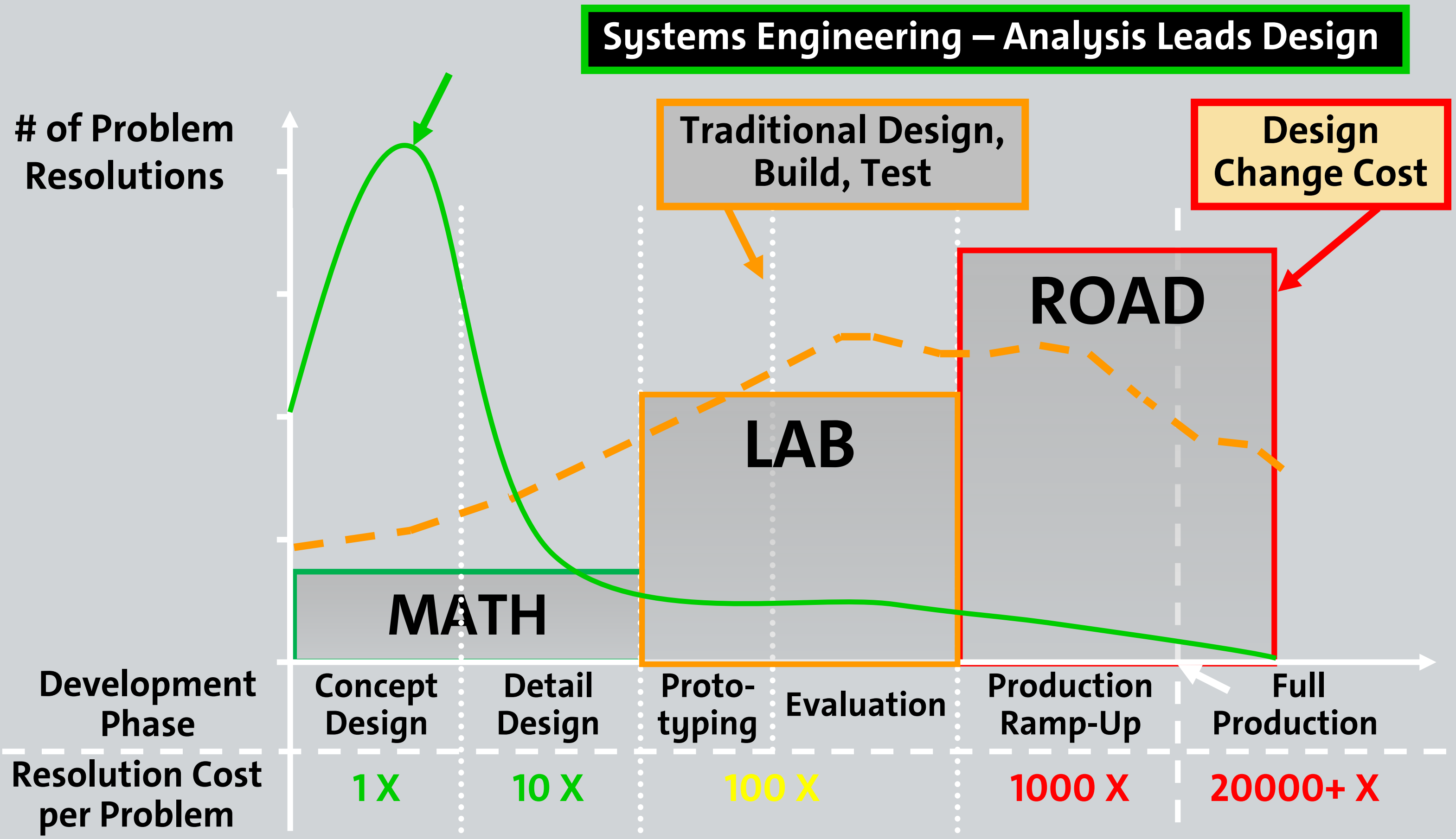


Controls Algorithm Models

- Models that Support Controls System Design
- Signal Delivery/Variation Analysis
- Hydraulics Models for Transmissions & Engine Actuator Systems
- Engine Performance

ROAD TO LAB TO MATH TRANSFORMS PROCESS

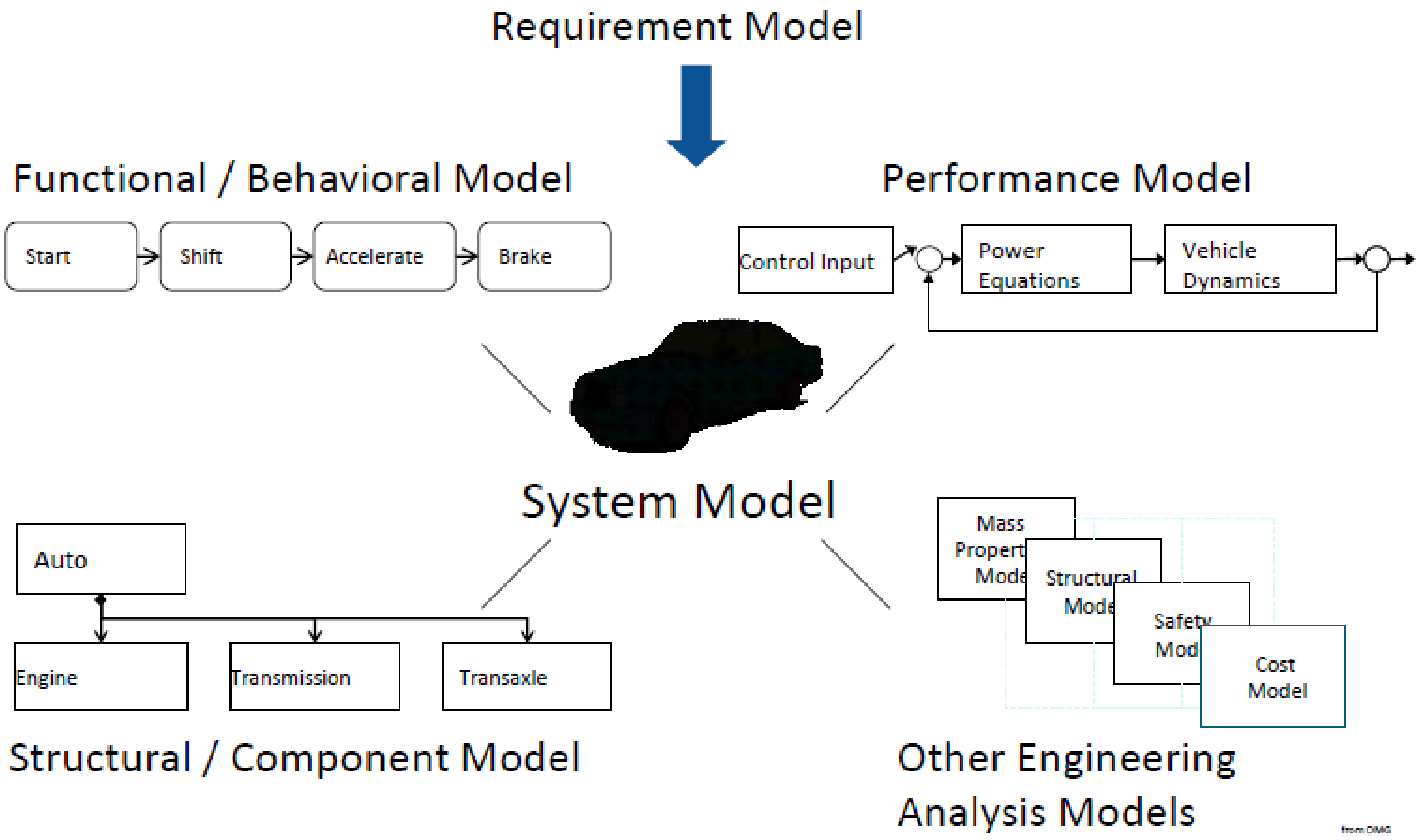
Frontloading Engineering with
First Time Capable lab and math processes
reduces Engineering costs



Source: General Electric 2003

How much more development time can be saved through Systems Engineering?

SYSTEM MODELING



Integrated System Model Must Address Multiple Aspects of a System

Delaware Valley INCOSE Chapter Presentation, Introduction to Model-Based Systems Engineering (MBSE) and SysML, June 30, 2015

THE DEFINING MODEL OF MBSE – THE SYSTEM MODEL

The System Model is the central connecting model of a MBSE methodology

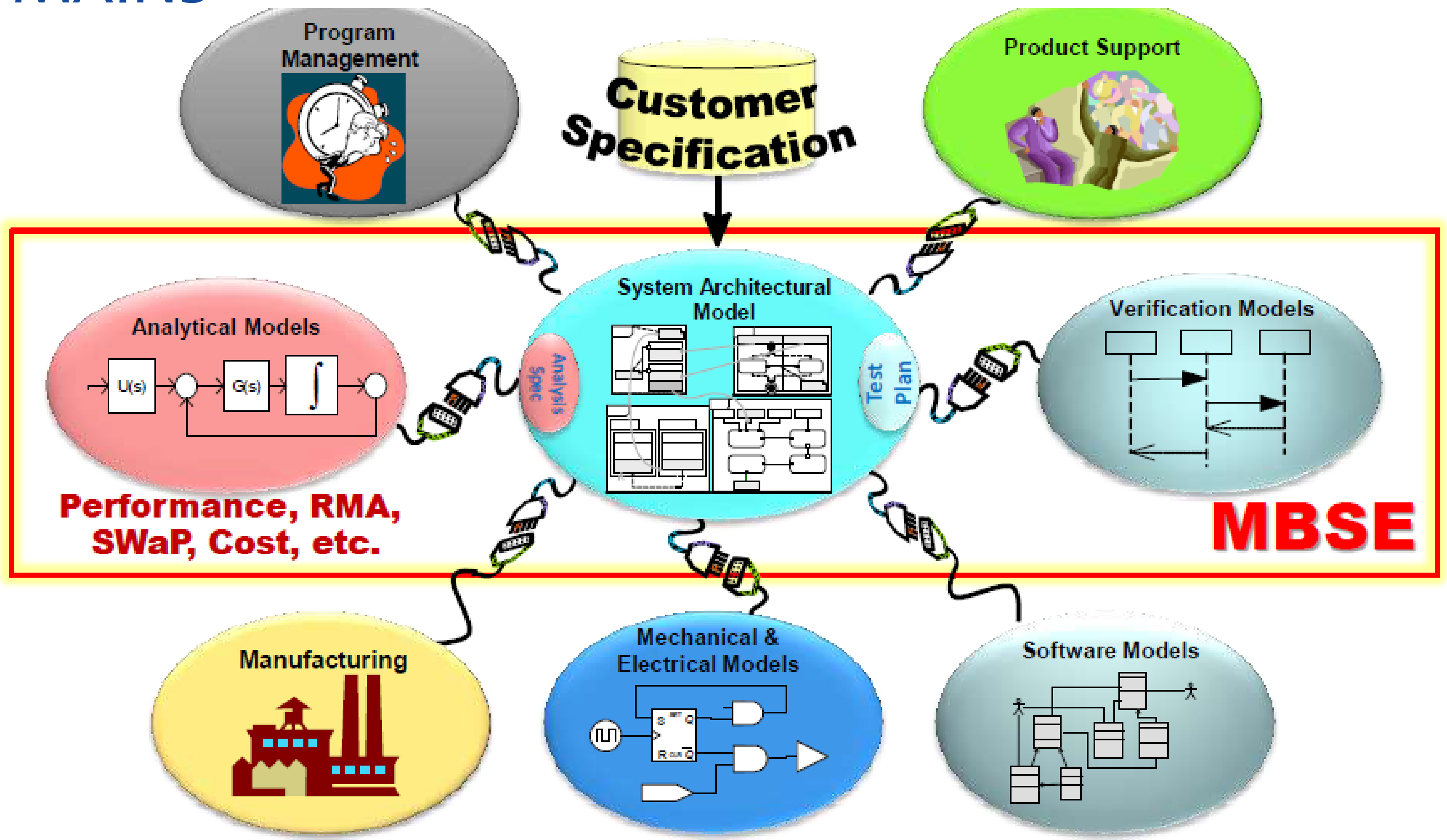
In general it represents the decomposition structure of the engineered system (functional, logical, physical) and the relationships between engineering information and entities

KEY FACETS OF MBSE

- The elicitation, allocation and traceability of Requirements.
- Enables the development of so-called Functional modeling, which is aimed at including the operational, functional and logical analyses.
- Enables so-called Physical modeling, based on a mathematical modeling of systems and a numerical analysis, which supports a simulation environment where the system behavior is investigated.
- Coordinates the overall interoperability of those models, with a overview on heterogeneous simulation techniques.
- Provides a preliminary description of the verification and validation of the systems.
- Helps frame some outlines about the strategic issues related to the integration between design and production, namely Application Lifecycle Management (ALM) and Product Lifecycle Management (PLM).
- Illustrates challenges related to the Configuration Change Management as it is currently implemented through the available tools.

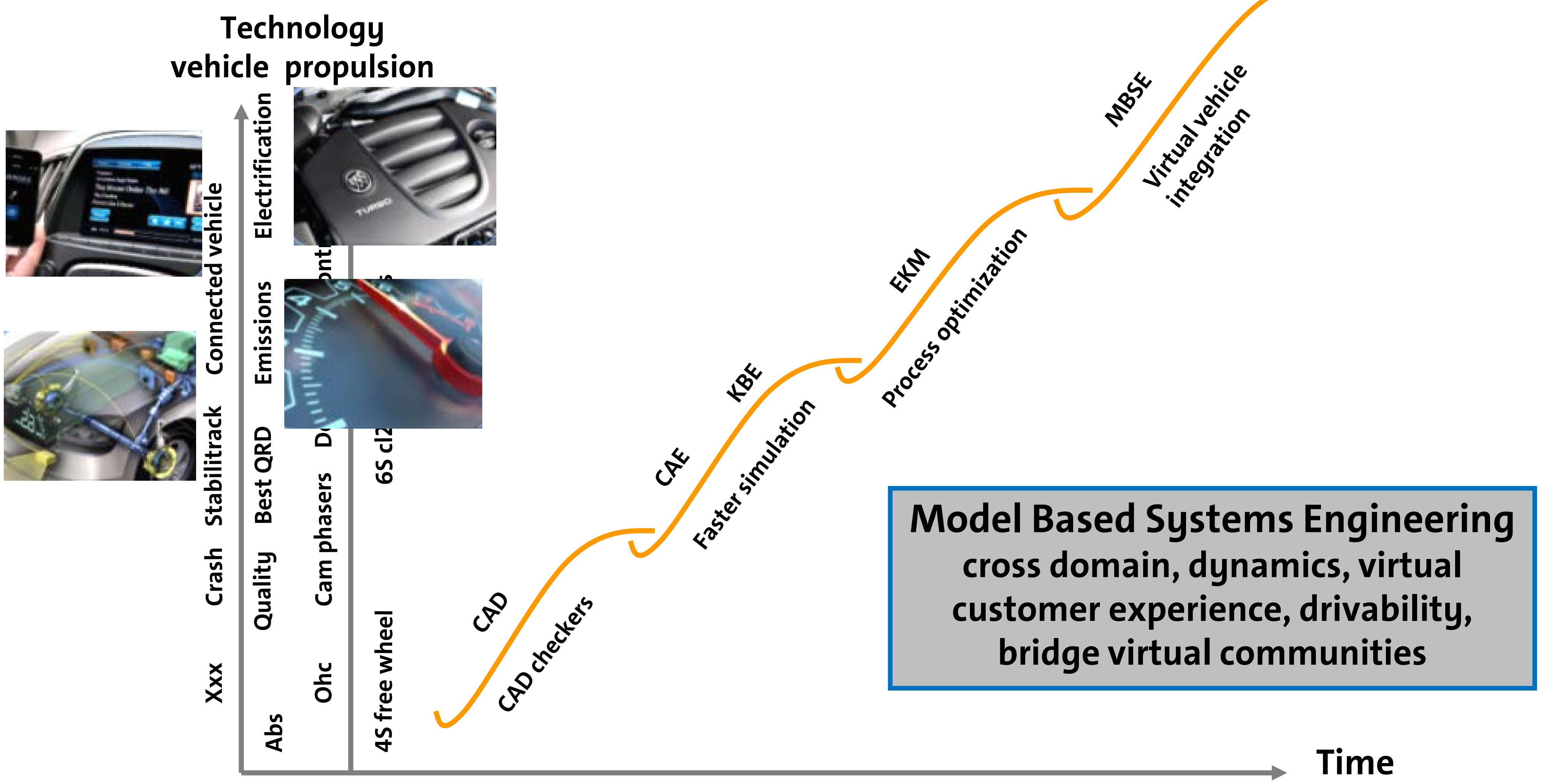
Adapted from: Systems Engineering and Its Application to Industrial Product Development

MBSE INTEGRATION ACROSS DOMAINS

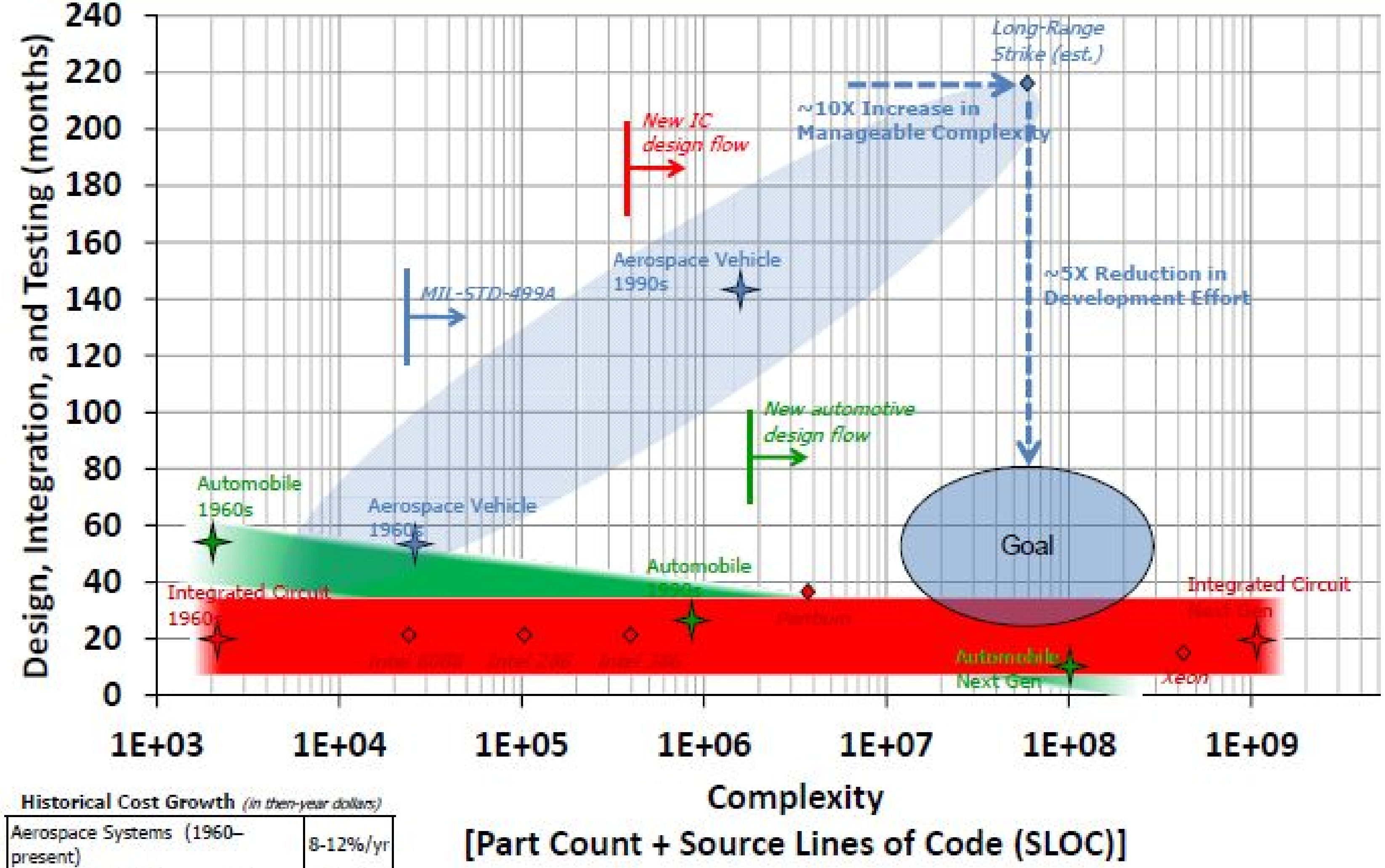


Delaware Valley INCOSE Chapter Presentation, Introduction to Model-Based Systems Engineering (MBSE) and SysML, June 30, 2015

PRODUCT COMPLEXITY DRIVES MATH TOOL TECHNOLOGIES



Dealing with complexity



Historical Cost Growth *(in then-year dollars)*

Aerospace Systems (1960–present)	8-12%/yr
Automobiles (1960–present)	4%/yr
Integrated Circuits (1970–present)	~0%/yr

Electrical, controls and software

- GM has one of the most complex systems and software product line engineering challenges in the world
  - 3000 contributing engineers
  - 300 hierarchical subsystems
  - Thousands of variant features
  - Millions of product instances per year
  - Tens-of-thousands of unique product variants
  - Dramatic increase in variation due to new propulsion systems and active safety
  - Global diversity in legislative regulations
  - Extreme economic and competitive pressures
  - Product line and feature set evolves annually
  - 15 concurrent development streams








GM AND MBSE IN THE ALM SPACE

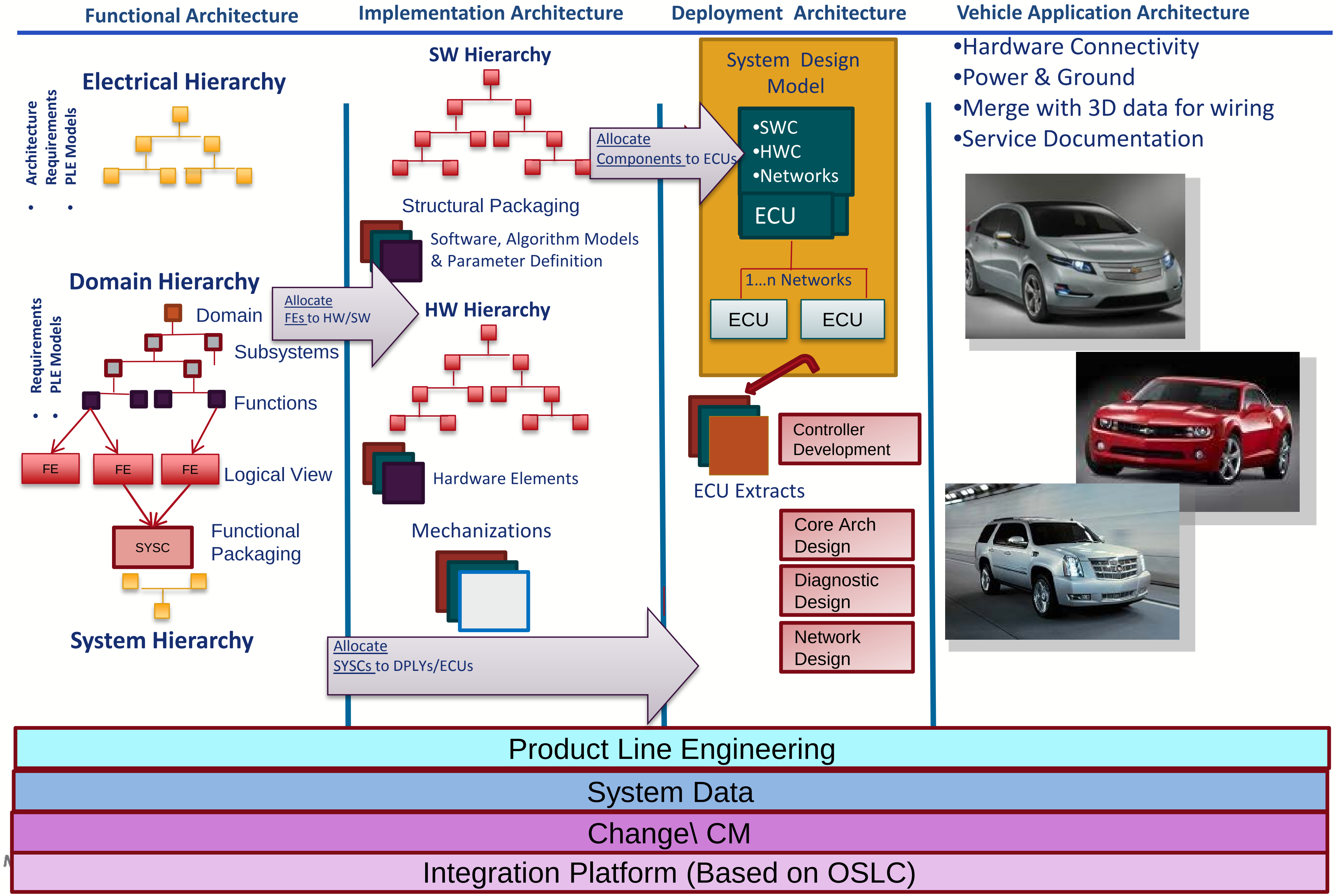
At GM we have a full system model of our electrical systems and are working to expand that practice to all disciplines

Some activities performed with our Model Based methodologies:

- Multiple allocation patterns of reusable requirements to different design solutions
- Derivation of required interface and network content based on deployment pattern
- Connection of software level detailed designs to system level architecture designs

SYSTEM ENGINEERING DEVELOPMENT LIFECYCLE - ELECTRICAL

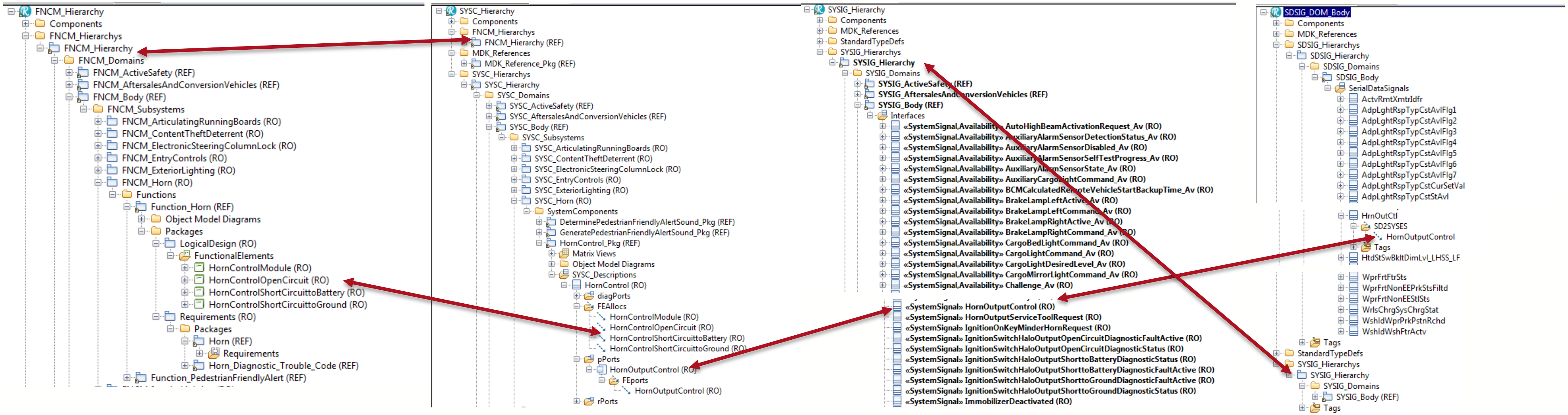
Traceability Relates the Elements



FUNCTIONAL ARCHITECTURE

A FUNCTIONAL DECOMPOSITION; REQUIREMENTS ALLOCATED TO SYSTEM ELEMENTS USED TO DRIVE INTERFACES

Functions | System Components | System Signals | Serial Data Signals



- Key Model Elements:
  - Functional Elements (FE)
  - Functional Interfaces
- Defines Requirement Allocation

- Key Model Elements:
  - System Components
  - System Interfaces
- Defines FE Allocation

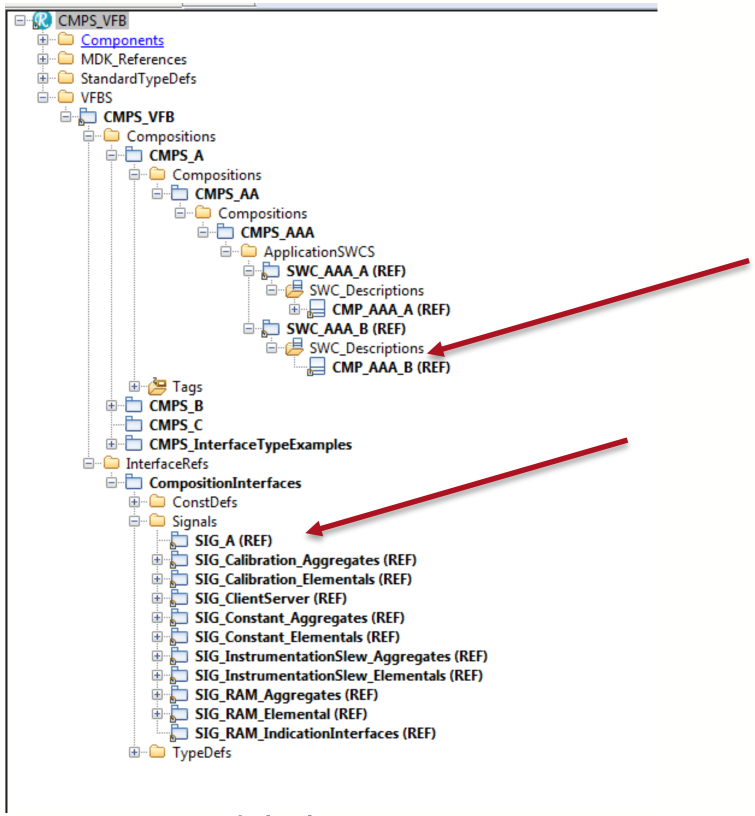
- Key Model Elements:
  - System Signals
- Abstract interface contract
- Bridge to other Architectures

- Key Model Elements:
  - Serial Data Signals
- Binds to System Signals

IMPLEMENTATION ARCHITECTURE

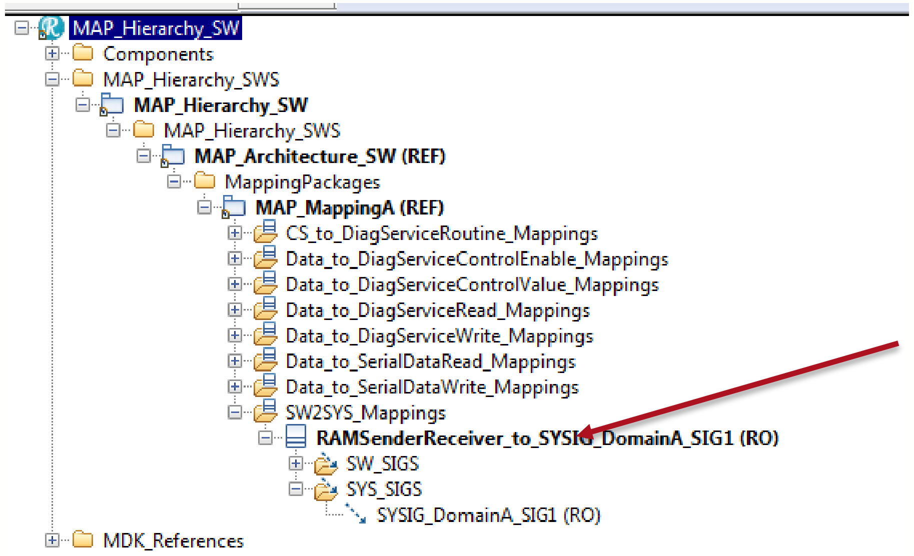
SOFTWARE DETAILED DESIGN INFORMATION CONNECTED TO SYSTEM LEVEL DESIGNS

Software Components and Signals



- Key Model Elements:
  - Software Components (SWC)
  - Software Interfaces (SW Signals)

Software Mapping

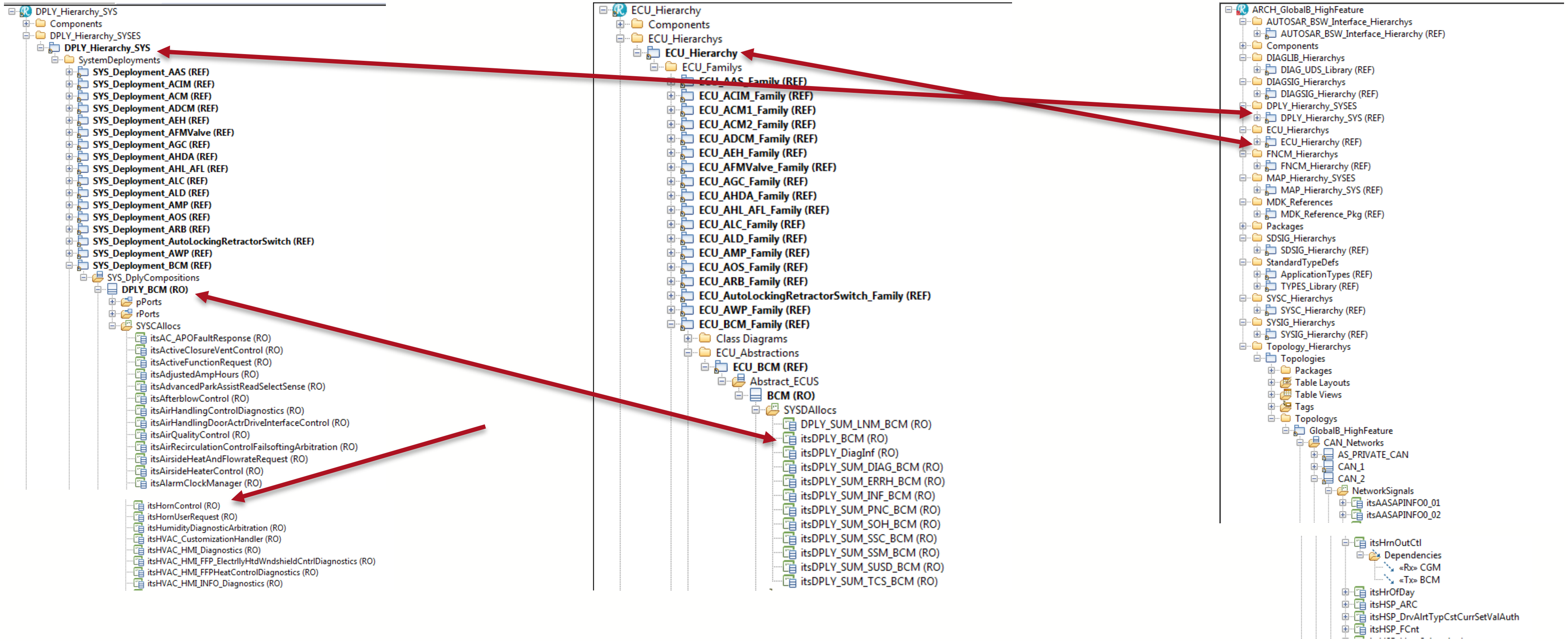


- Defines SW Signal to System Signal Mapping

DEPLOYMENT ARCHITECTURE

REUSABLE ELEMENTS IN DIFFERENT DESIGN PATTERNS

Deployment Compositions | ECUs | Network Topology



- Key Model Elements:
  - Deployment Compositions
- Defines System Component Allocation
- Defines Software Component Allocation

- Key Model Elements:
  - ECUs
- Defines Deployment Composition Allocation

- Network Signals between ECUs determined by…
- System Component contracts to System Signals
- System Signal binding to Serial Data Signals

CURRENT ACHIEVEMENTS

The previous slides provide a very brief overview of the capabilities GM has put in place. With them GM has achieved:

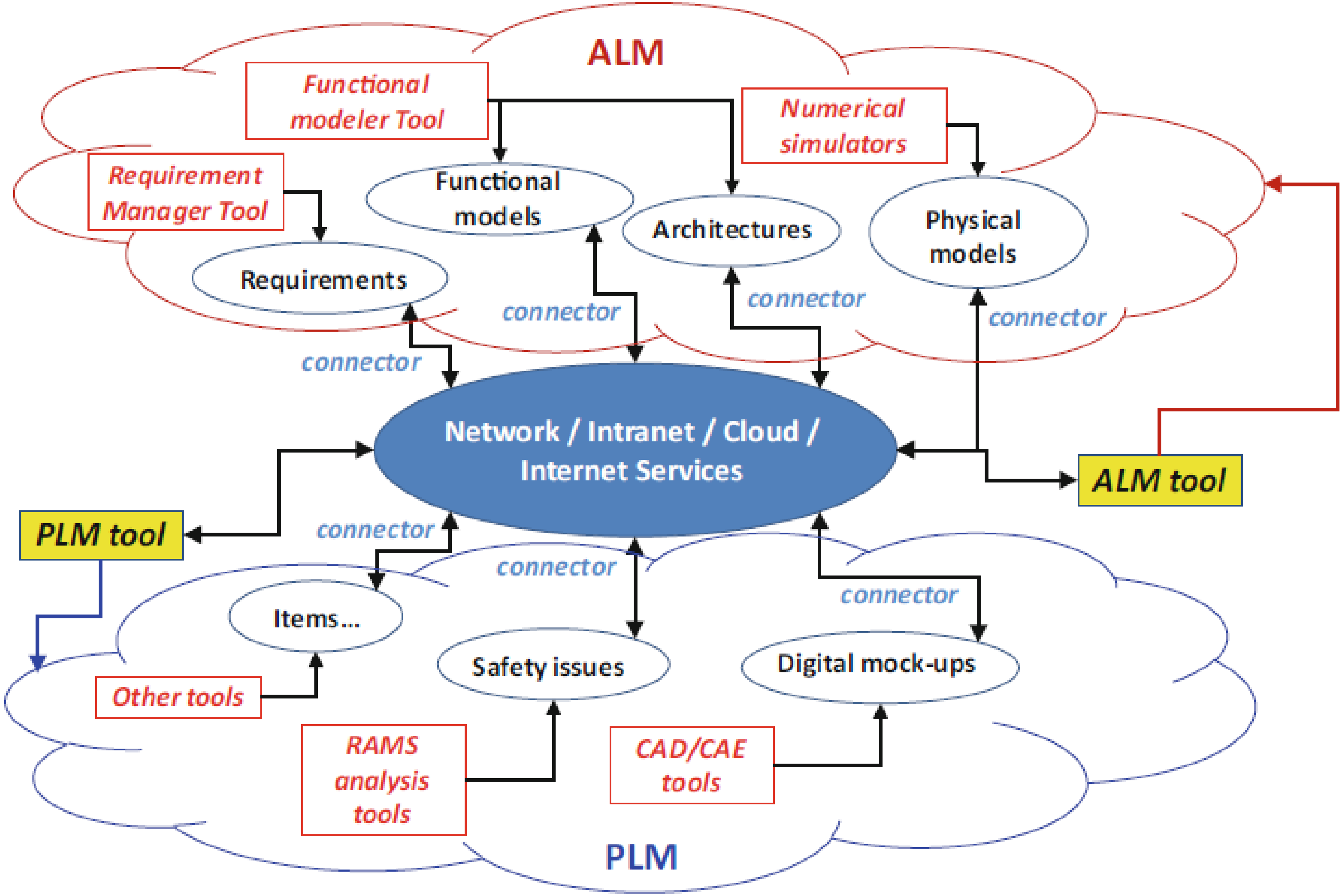
- Ability to **generate** network and architecture configurations
- Ability to **generate** middleware software
- Ability to **generate** allocation based requirements specifications
- Ability to **generate** full architecture analysis and reports

With tailored user experience to our specific product and language

But…

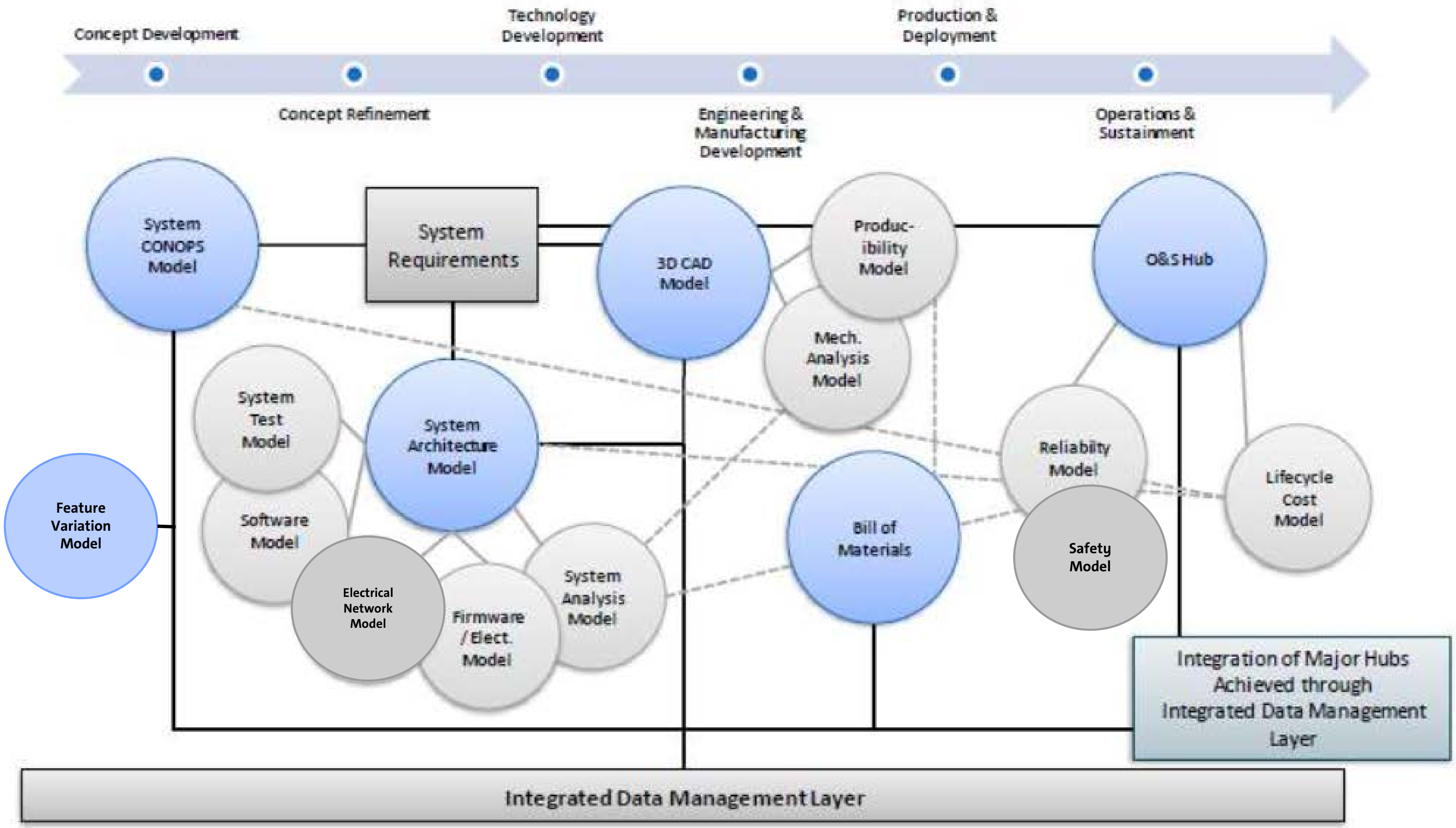
FUTURE STATE AND CHALLENGES AHEAD

Integration of the ALM and PLM processes and overview on the data, analyses, connections and tools involved

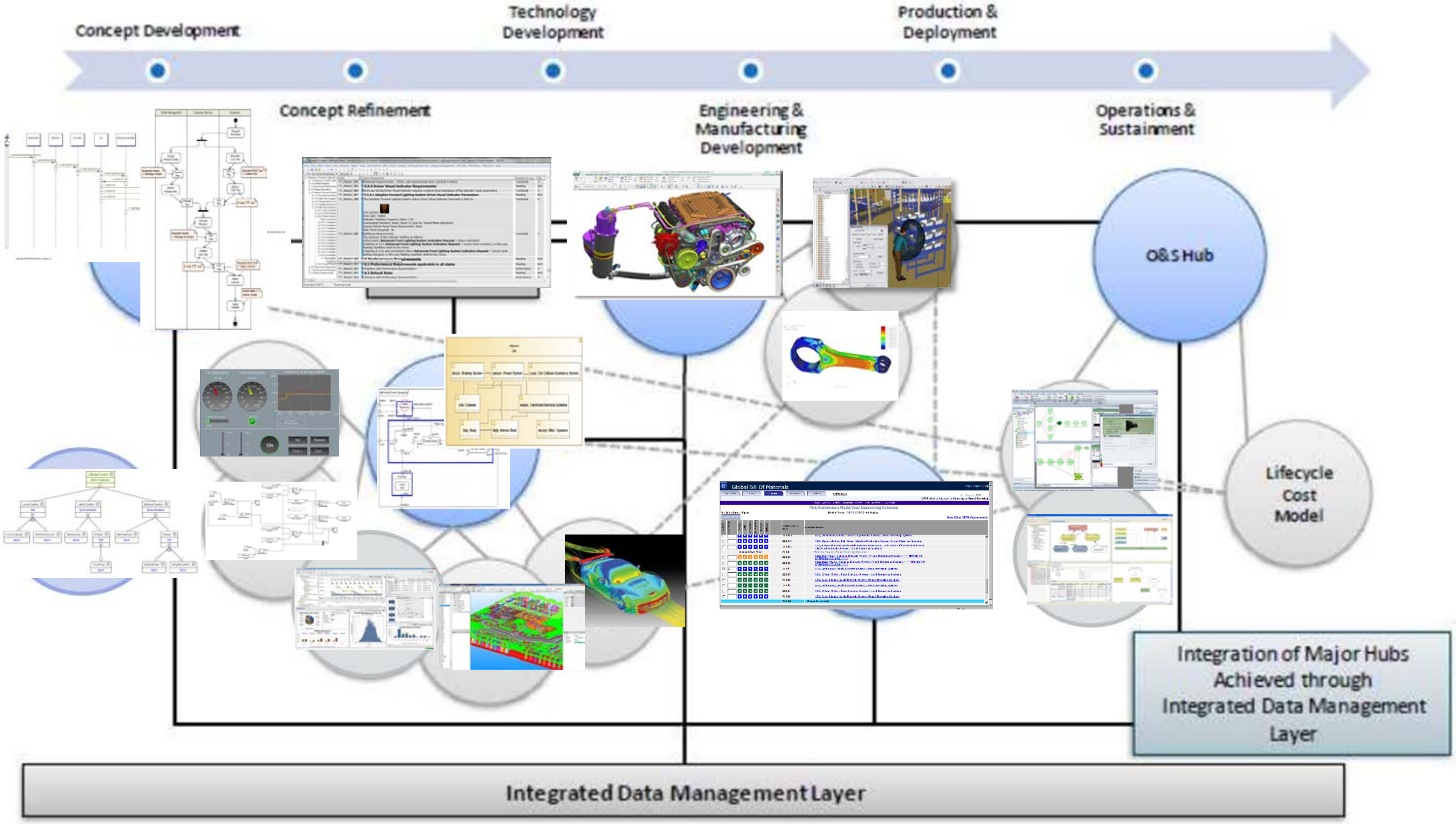


Source: Systems Engineering and Its Application to Industrial Product Development

Model Based Engineering (MBE) Conceptual Framework



Model Based Engineering (MBE) Conceptual Framework



Forward looking:
Model Based System Engineering and linked data Systems

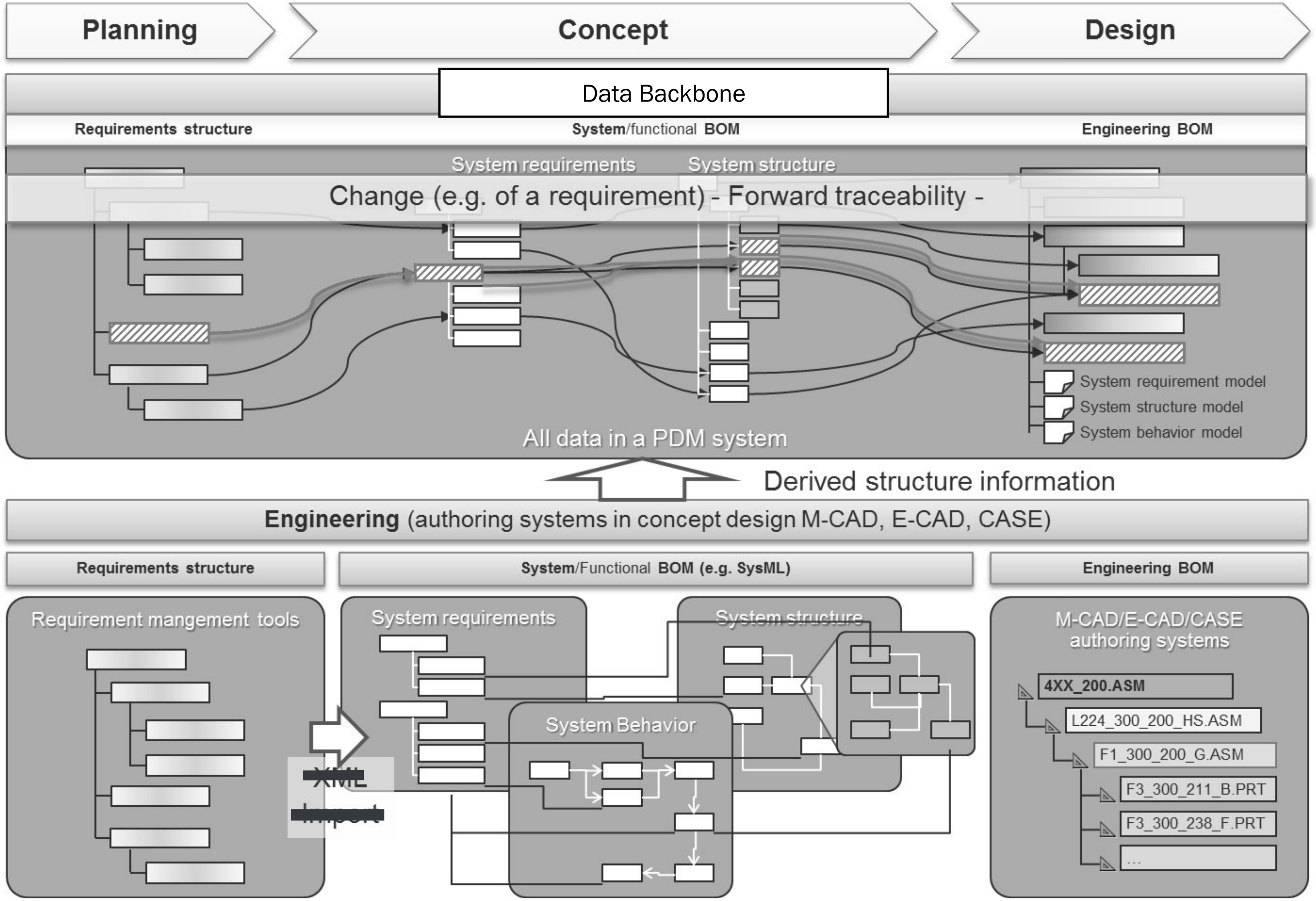
Objective: **Linked** engineering data system; i.e. traceability

Why: With our system complexity driving requirement counts into the millions and peer relationships into the hundreds of thousands, people talking can not keep track of the interactions.

Data Systems have to help:

Less reliant upon manual connectivity

Integrated Requirements, DFMEA and System Safety tools.

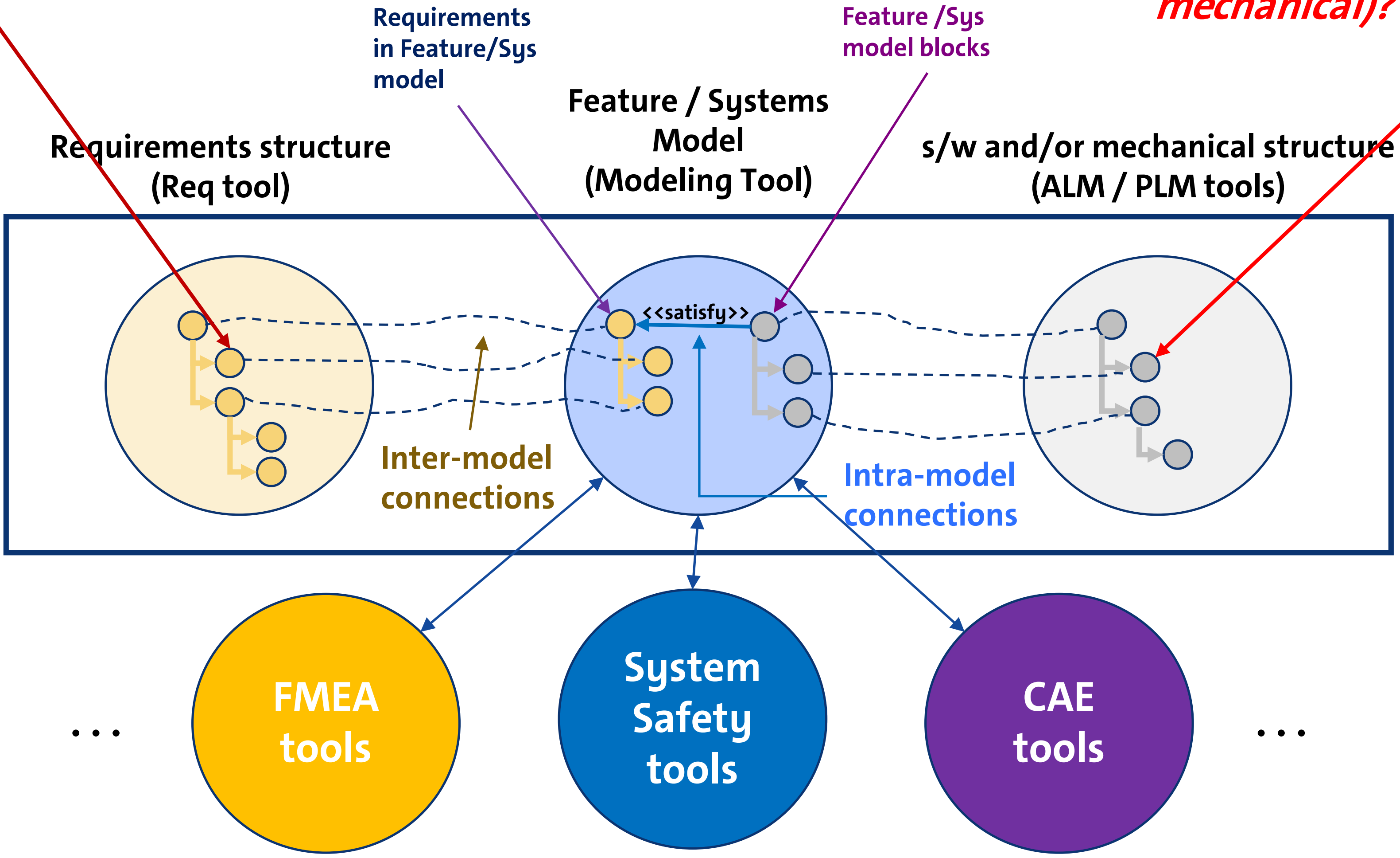


FUTURE-STATE

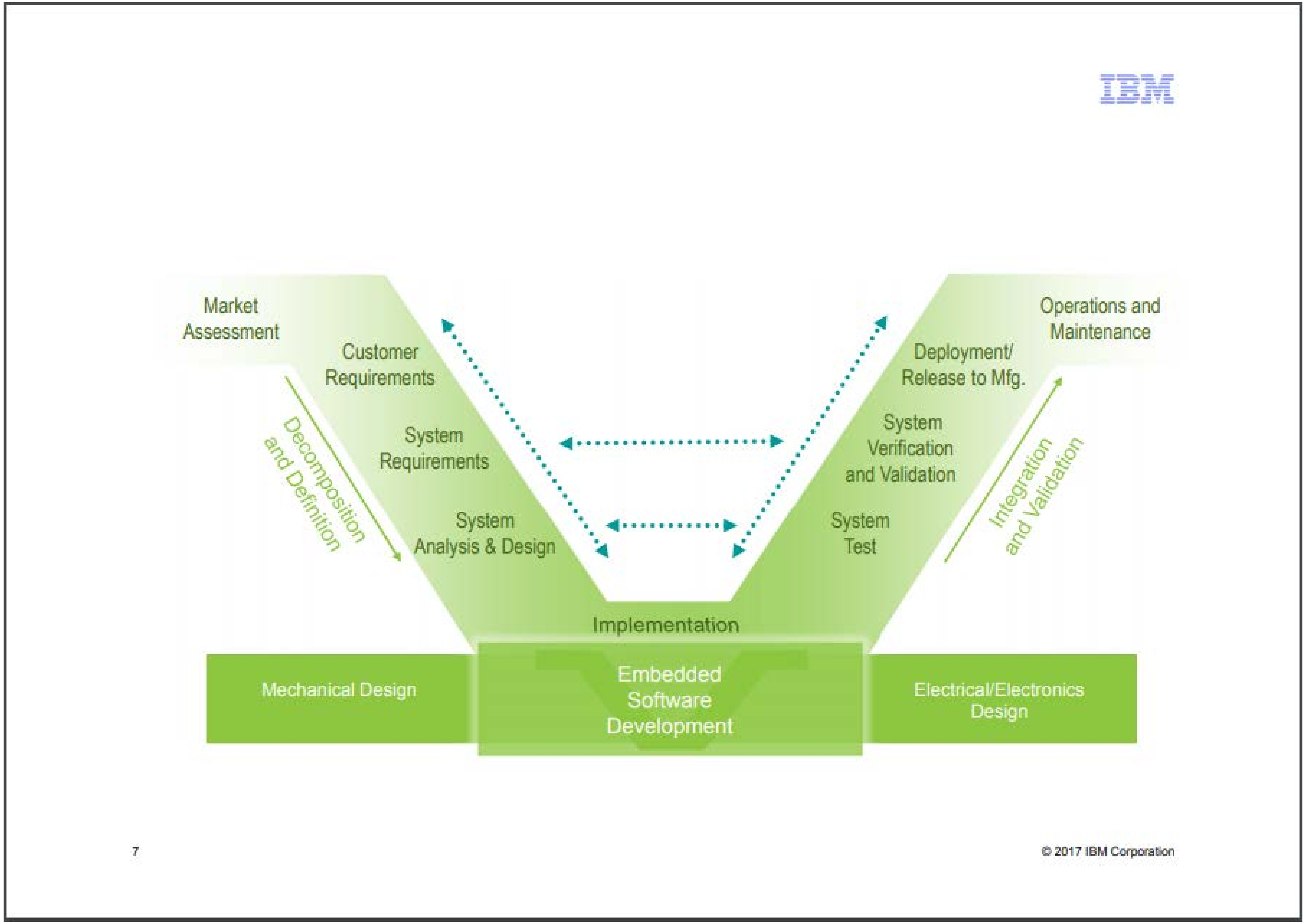
MBSE USES A CONNECTED TOOL CHAIN

If I change this requirement, what is the downstream impact, e.g. to CAD models in my PLM tool chain, or software in development?

What requirements does this part correspond to, and what is the upstream impact of changes of failures of this part (s/w or mechanical)?

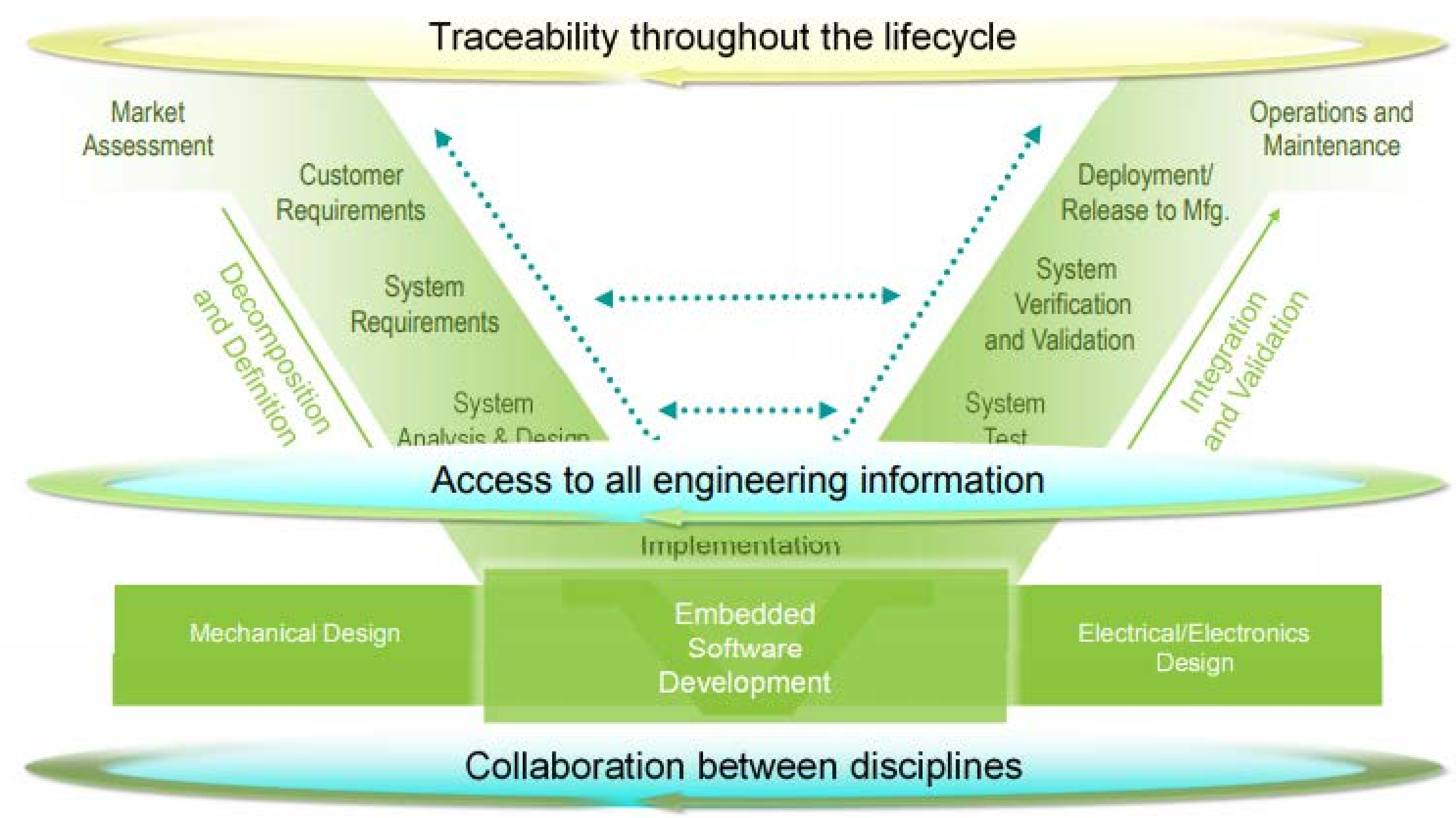


Same data "gold source" used by specialized tools



IBM
Market Assessment
Customer Requirements
System Requirements
System Analysis & Design
Decomposition and Definition
Implementation
Operations and Maintenance
Deployment/ Release to Mfg.
System Verification and Validation
System Test
Integration and Validation
Mechanical Design
Embedded Software Development
Electrical/Electronics Design
7
© 2017 IBM Corporation

An open, integrated systems approach delivers:



8

OPPORTUNITIES FOR EXPANDED INTEGRATION

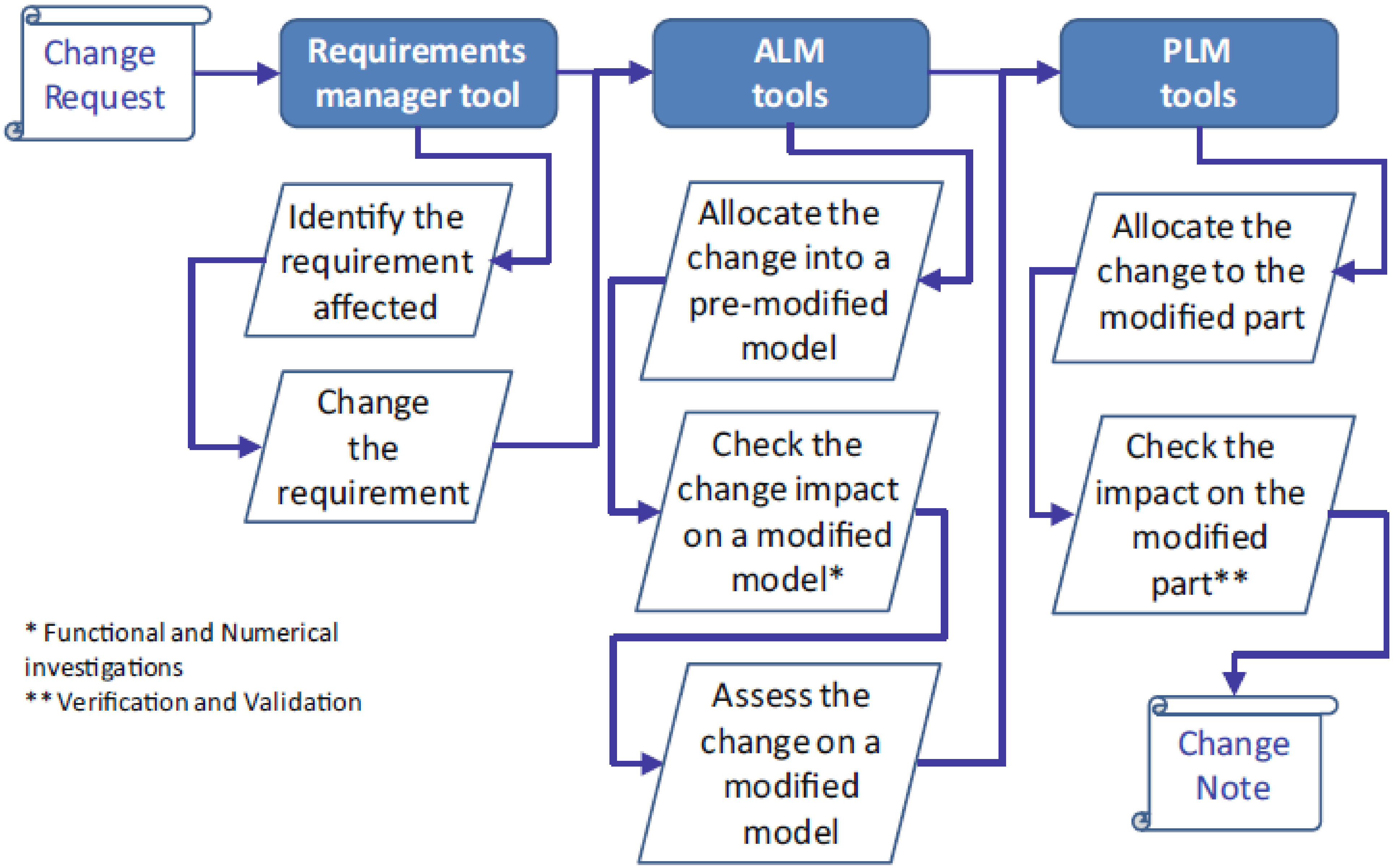
The System model provides an **abstracted** authoring environment that identifies the content of a system and the relationships between those elements

Many engineering operations and tools provide the authoring environment for defining the **details** of system content and relationships

GM, through the use of our extensions, has started to allow the abstracted model definition to assist in preforming the detailed operations, but there is significant opportunity for additional integrations

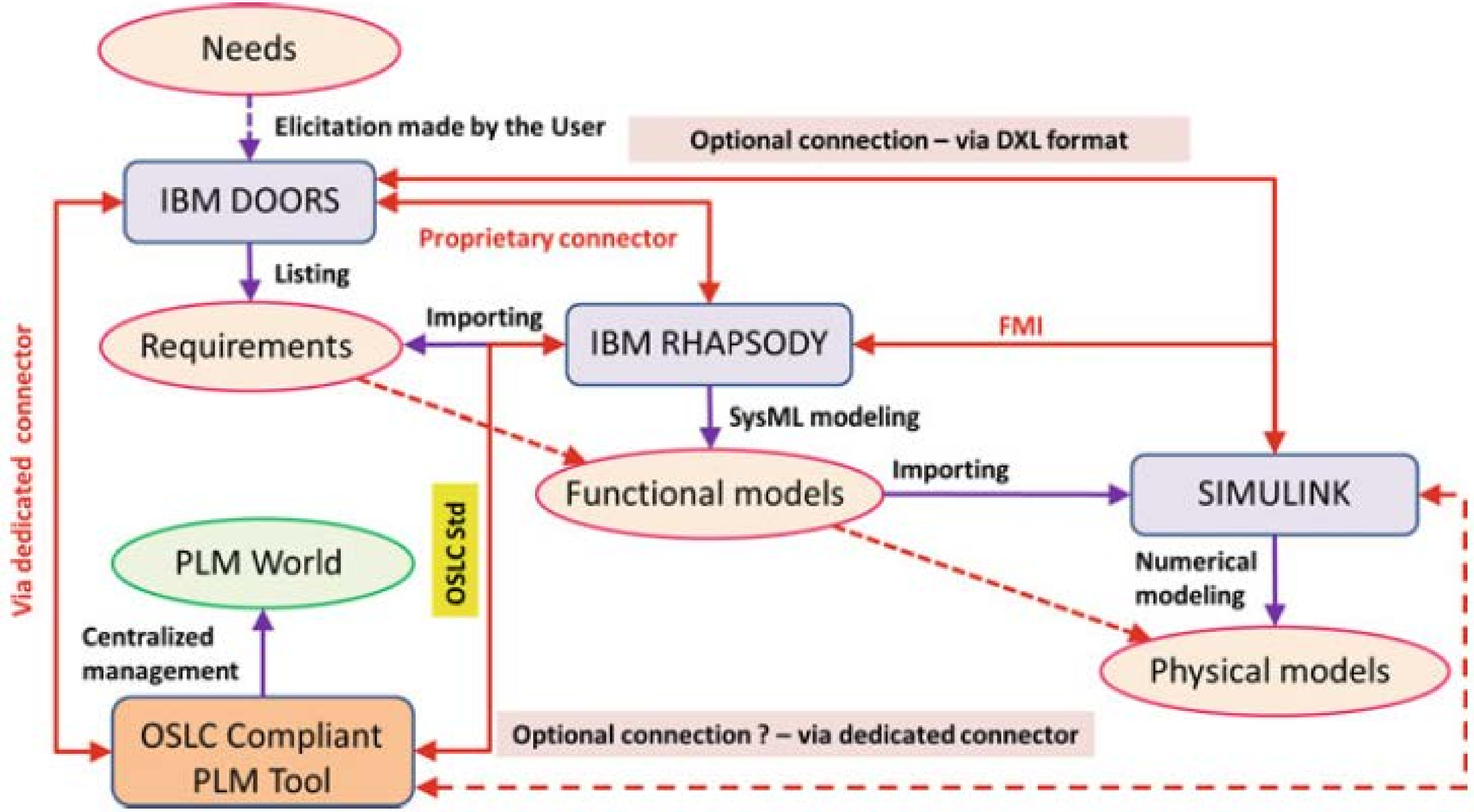
Two examples follow:

Example of change control implementation



* Functional and Numerical investigations
** Verification and Validation

Example of integration with the PLM process of tools used for the industrial test case



WHY SYSTEMS ENGINEERING?

... the power of the Systems Engineering is that it demonstrated to be suitably applicable to several industrial domains with high level of confidence and few differences in some tailored tasks, basically following a common generalized, intrinsically systematic and effective approach.

...

Main benefits of this approach are the reusability in different projects, the traceability of each requirement, being allocated to functions and system components as well as the automatic documentation of the whole product development activity.

Systems Engineering and Its Application to Industrial Product Development, Studies in Systems, Decision and Control, Volume 134, Section 1.2

CONCLUSION

- In a Model Based System Lifecycle Engineering methodology, the model is not something created in addition to traditional engineering tasks
    - The Models are created in support of / as alternative to traditional engineering tasks
- A key to success is freeing the model data to be used in support of all lifecycle tasks
- This is possible today, but is too hard
- Improved capabilities are needed to leverage the infrastructure that is already there
    - Focus needs to be on extensible frameworks, not fixed integrations and capabilities


THANK YOU

POLARION

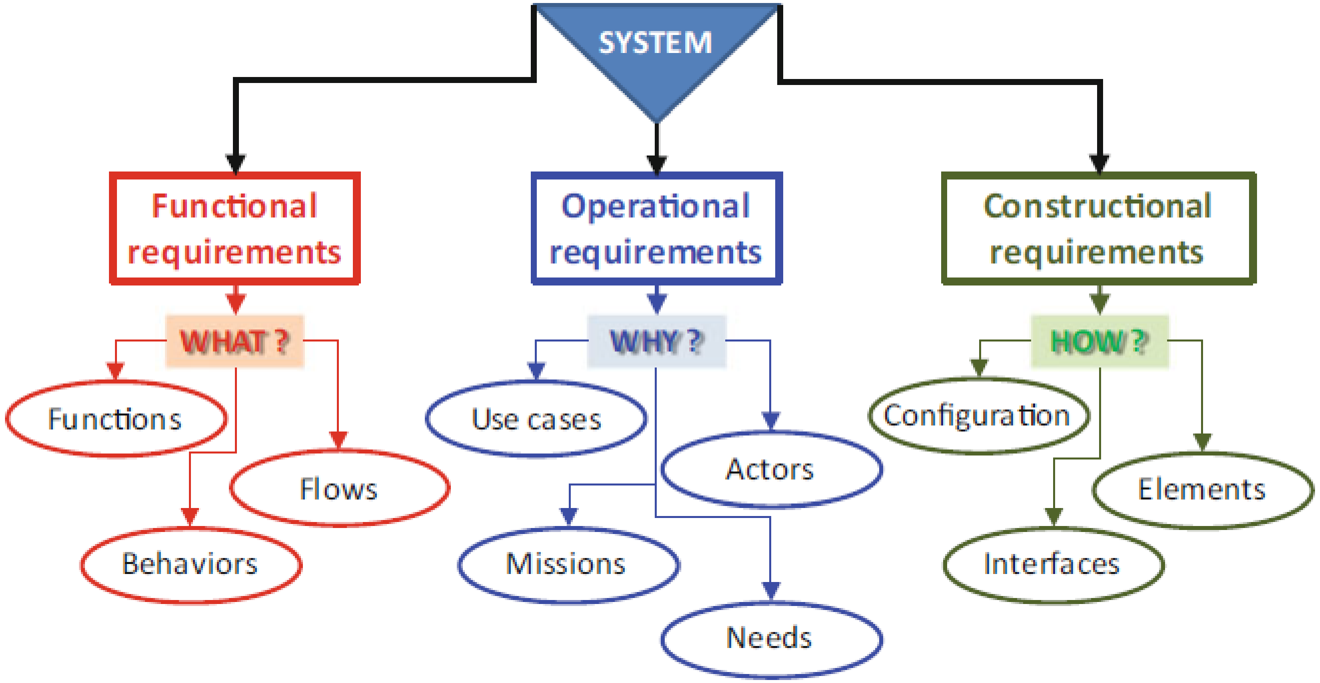
The convergence of PLM and ALM - something we call “product and application lifecycle management” PALM represents the future generation of management control for Systems Engineering.

https://polarion.plm.automation.siemens.com/polarion-alm-plm-integration

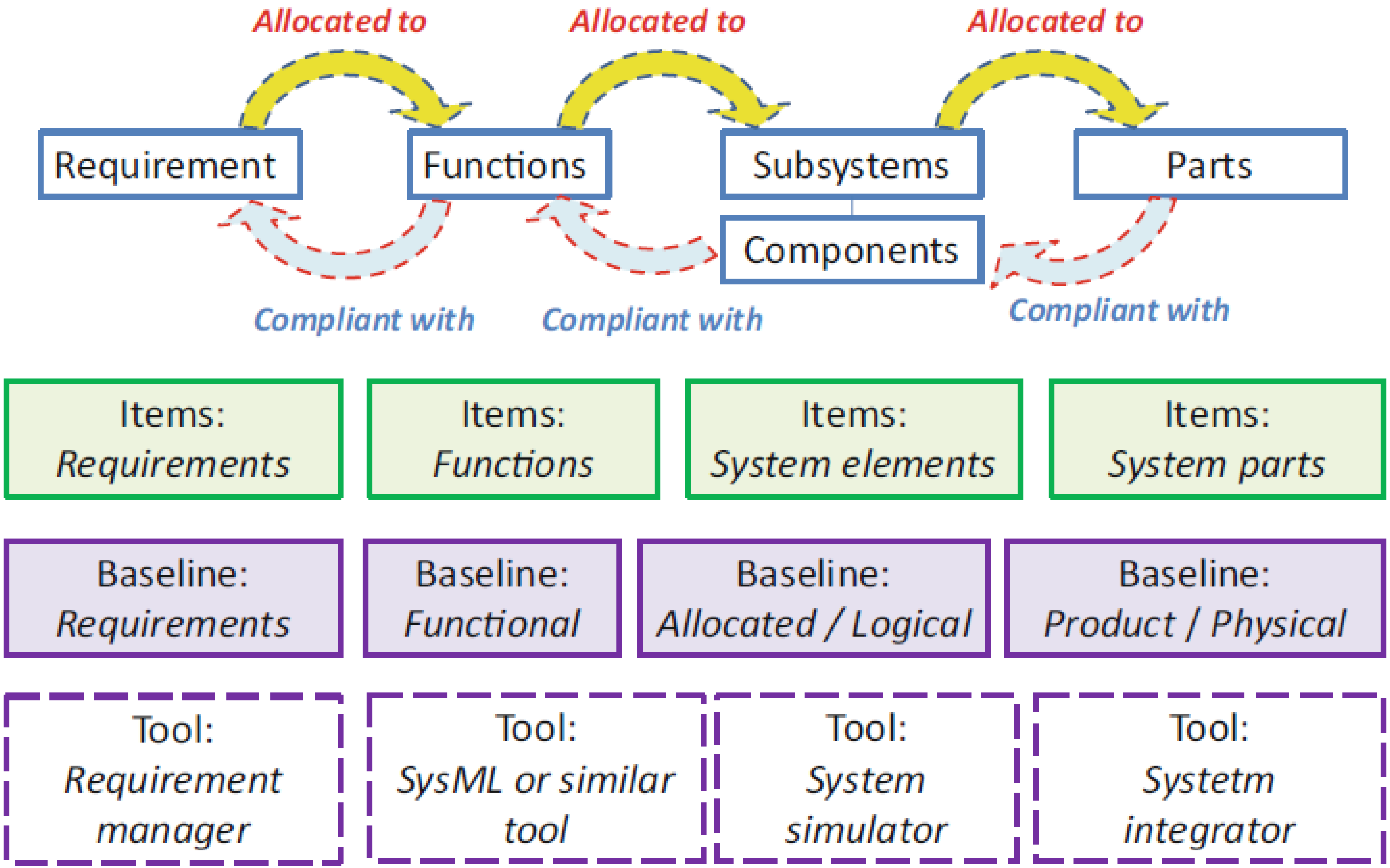
PLE, PLM, AND ALM CONVERGENCE

The relative strengths of PLM, ALM, and PLE converge in near-perfect jigsaw puzzle fashion. PLM's limited support for software can be overcome by ALM tools. ALM's limited support for product line variation in space can be overcome by PLE tools. The result of this technology convergence is a powerful solution for managing the full engineering lifecycle of software-intensive mechanical product lines.

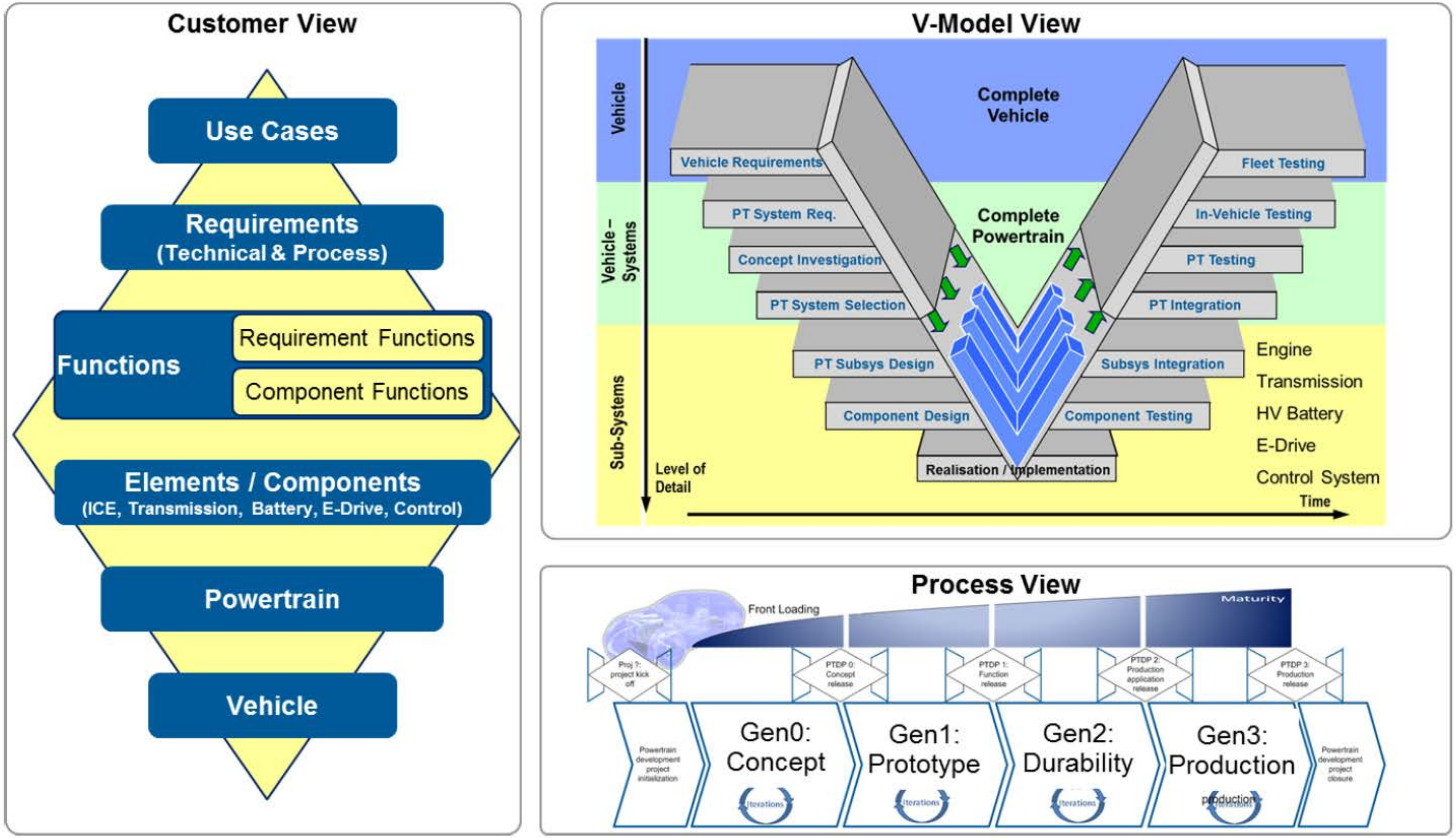
http://www.biglever.com/technotes/plm_alm_spl.html



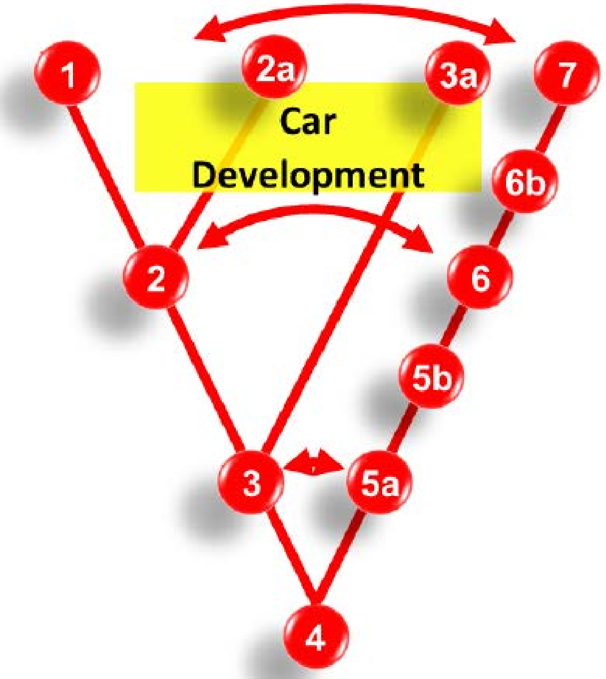
Main classification of requirements and related content

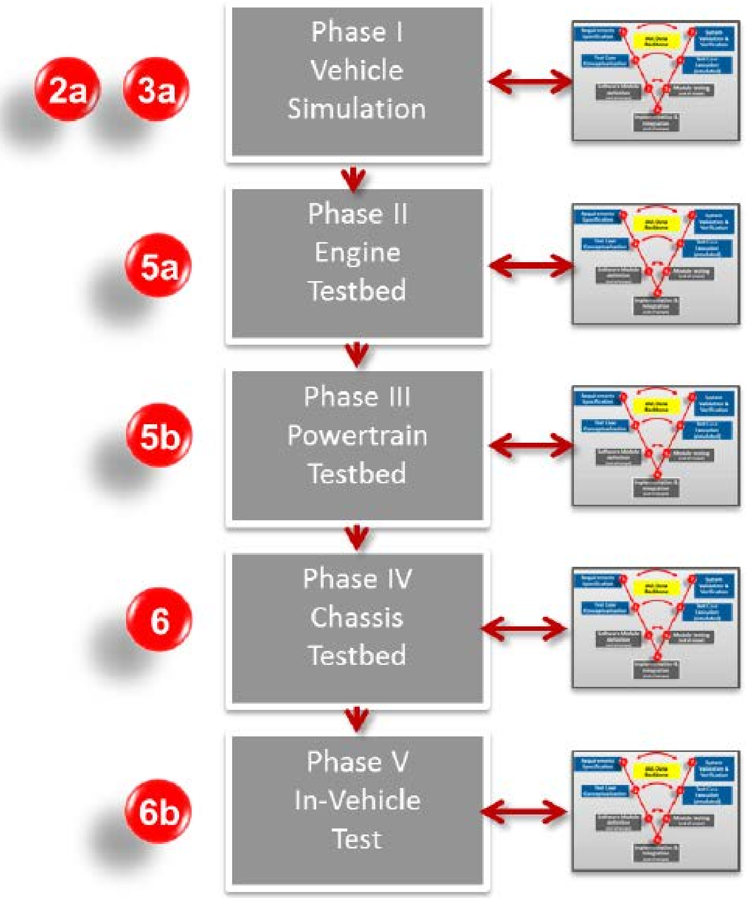


Overview of the traceability of requirements and description of the Configuration Control Management and of its items



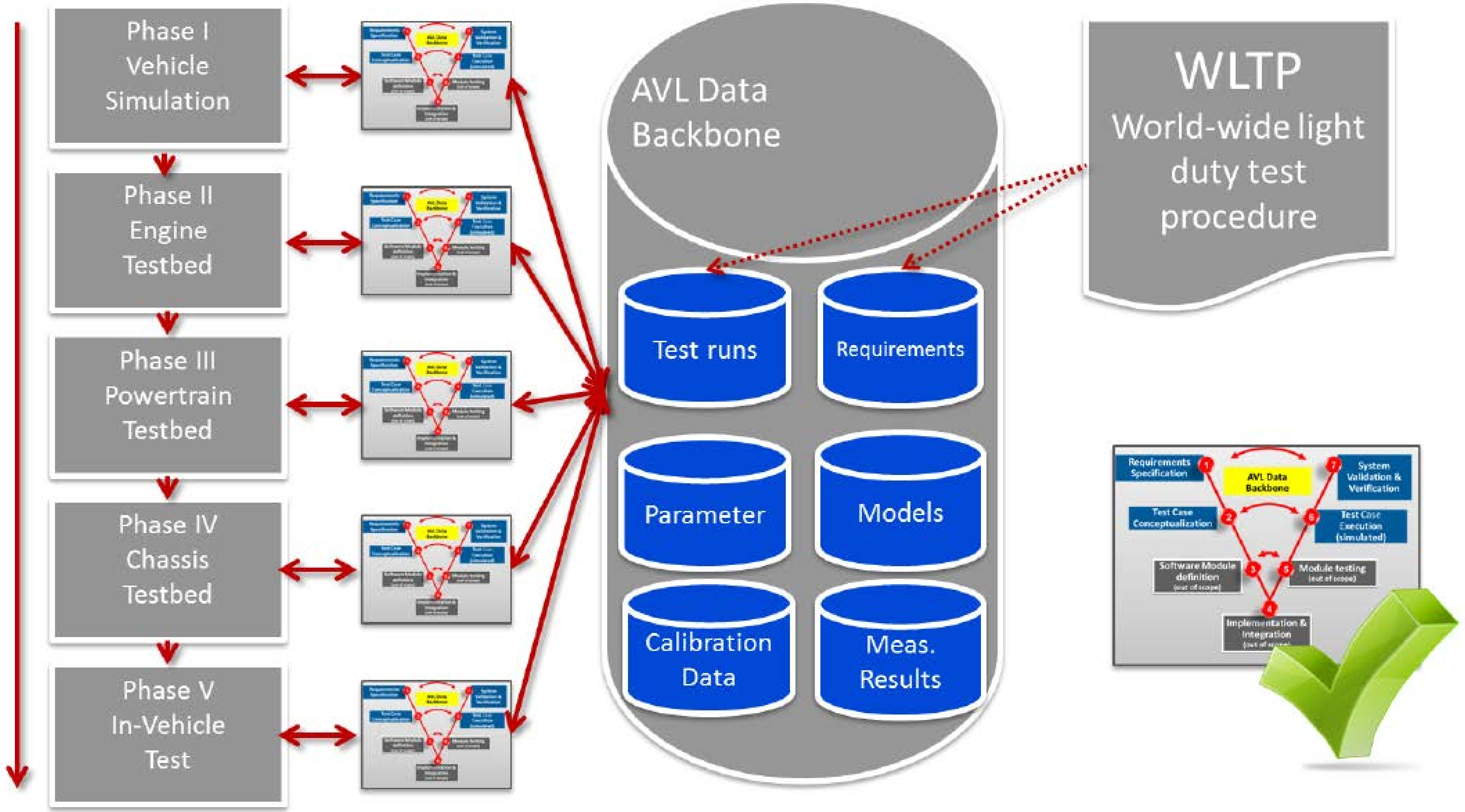
From V3 Automotive Use Cases, CRYSTAL





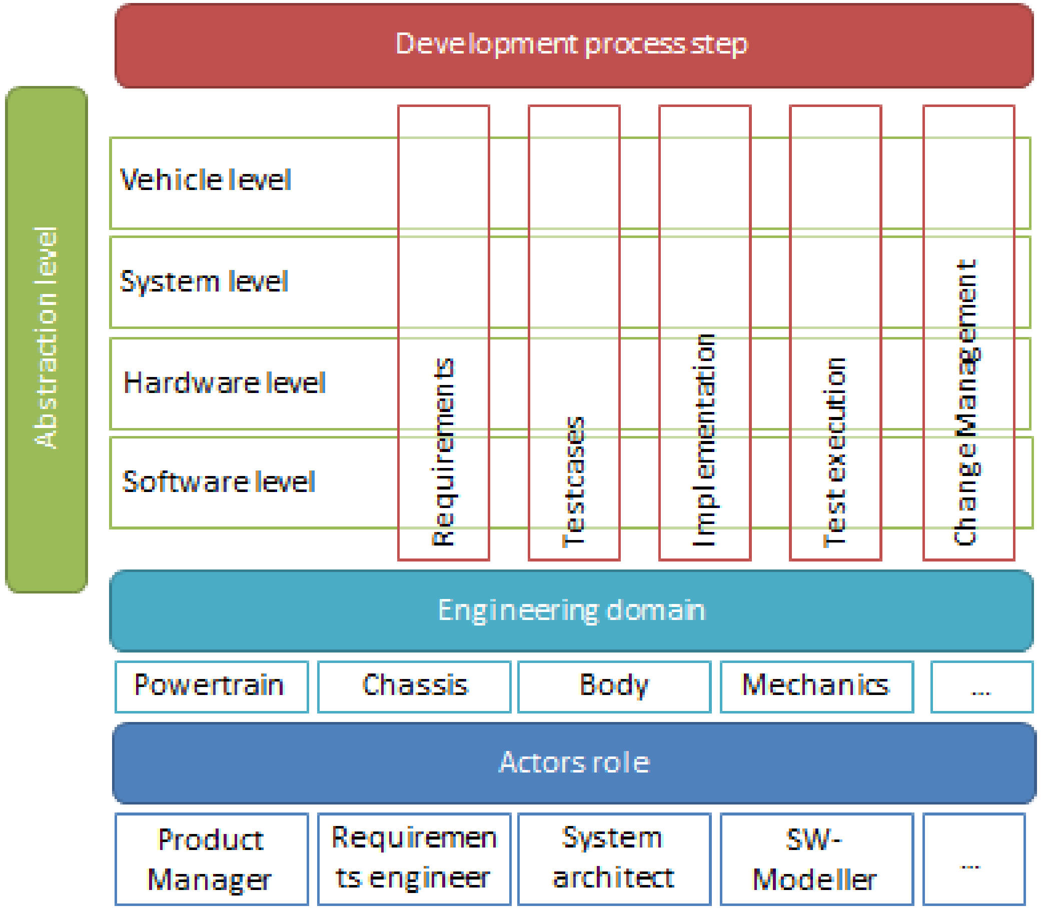
From V3 Automotive Use Cases, CRYSTAL

Mapping the W-model to Vehicle Test Phases

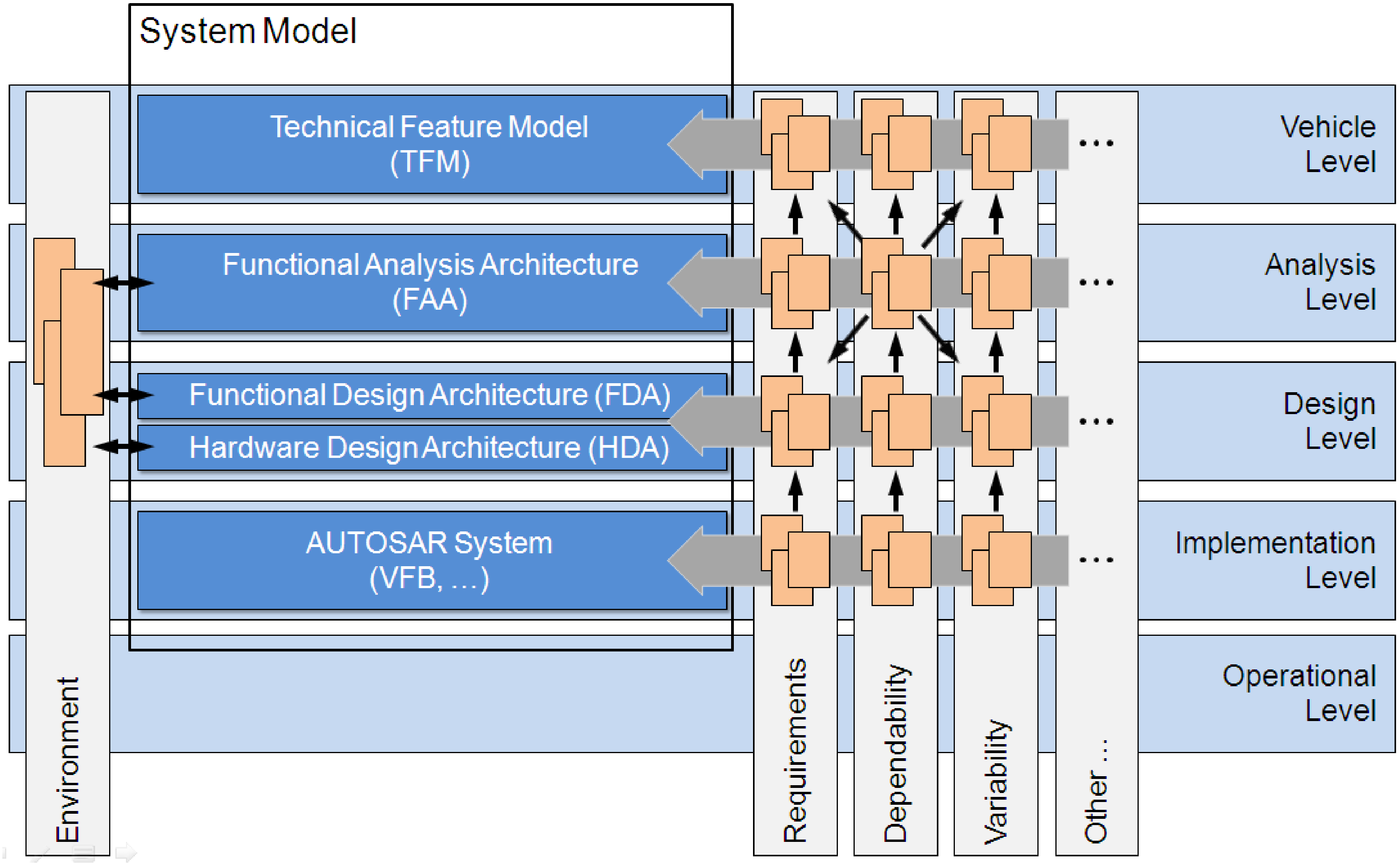


From V3 Automotive Use Cases, CRYSTAL

Data backbone concept for testing phases



Overview of Development Dimensions
From V3 Automotive Use Cases, CRYSTAL



EAST-ADL overview (www.east-adl.info/Specification.html)

HEADLINE HERE

Abbreviated as ALM, Application Lifecycle Management refers to the capability to integrate, coordinate and manage the different phases of the software delivery process. From development to deployment, ALM is a set of defined process and tools that include definition, design, development, testing, deployment and management. Throughout the ALM process, each of these steps are closely monitored and controlled.

https://www.webopedia.com/TERM/A/Application_Lifecycle_Management.html

"Systems engineering", ..., refers to the distinctive set of concepts, methodologies, organizational structures (and so on) that have been developed to meet the challenges of engineering effective functional systems of unprecedented size and complexity within time, budget, and other constraints.

https://en.wikipedia.org/wiki/Systems_engineering

INCOSE Webinar

Approach and Limitations

Key insight
Systems and software engineering depend on knowledge in other domains. Understanding how knowledge domains come together in systems is central to building conceptual foundations for systems engineering.

- Inquire into the relationship between knowledge, systems and engineering
- Frame the standard understanding of systems engineering in a form that reflects systems ideas, and in a form amenable to mathematical expression
  - Stay as close to traditional understanding and concepts as possible
  - Most of the ideas presented here are common knowledge. Focus is not on novelty, but on explication and synthesis
- Limitations of current work
  - Only a strawman version with seed ideas, needs considerable refinement
  - Limited to mechanistic systems: does not grapple with purposefulness
- Presentation objectives
  - Share current state of the research, obtain feedback
  - Invite participation in INCOSE SSWG SE Conceptual Model project

EMBEDDED SOFTWARE

Embedded software is specialized programming in a chip or on firmware in an embedded device to controls its functions.

Hardware makers use embedded software to control the functions of various hardware devices and systems.

http://internetofthingsagenda.techtarget.com/definition/embedded-software

Unlike application software, embedded software has fixed hardware requirements and capabilities, and addition of third-party hardware or software is strictly controlled.

Embedded software needs to include all needed device drivers at manufacturing time, and the device drivers are written for the specific hardware. The software is highly dependent on the CPU and specific chips chosen.

https://en.wikipedia.org/wiki/Embedded_software

EMBEDDED SYSTEMS

An **embedded** system is a computer system with a dedicated function within a larger mechanical or electrical system, often with real-time computing constraints. It is **embedded** as part of a complete device often including hardware and mechanical parts. **Embedded** systems control many devices in common use today.

Since the embedded system is dedicated to specific tasks, design engineers can optimize it to reduce the size and cost of the product and increase the reliability and performance. Some embedded systems are mass-produced, benefiting from economies of scale.

Automobiles, electric vehicles, and hybrid vehicles increasingly use embedded systems to maximize efficiency and reduce pollution. Other automotive safety systems include anti-lock braking system (ABS), Electronic Stability Control (ESC/ESP), traction control (TCS) and automatic four-wheel drive.

Embedded systems are designed to do some specific task, rather than be a general-purpose computer for multiple tasks. Some also have real-time performance constraints that must be met, for reasons such as safety and usability; others may have low or no performance requirements, allowing the system hardware to be simplified to reduce costs.

Embedded systems are not always standalone devices. Many embedded systems consist of small parts within a larger device that serves a more general purpose. For example, the Gibson Robot Guitar features an embedded system for tuning the strings, but the overall purpose of the Robot Guitar is, of course, to play music.[10] Similarly, an embedded system in an automobile provides a specific function as a subsystem of the car itself.

Embedded systems range from no user interface at all, in systems dedicated only to one task, to complex graphical user interfaces that resemble modern computer desktop operating systems.

Embedded systems often reside in machines that are expected to run continuously for years without errors, and in some cases recover by themselves if an error occurs. Therefore, the software is usually developed and tested more carefully than that for personal computers, and unreliable mechanical moving parts such as disk drives, switches or buttons are avoided.

https://en.wikipedia.org/wiki/Embedded_system

INCOSE Webinar

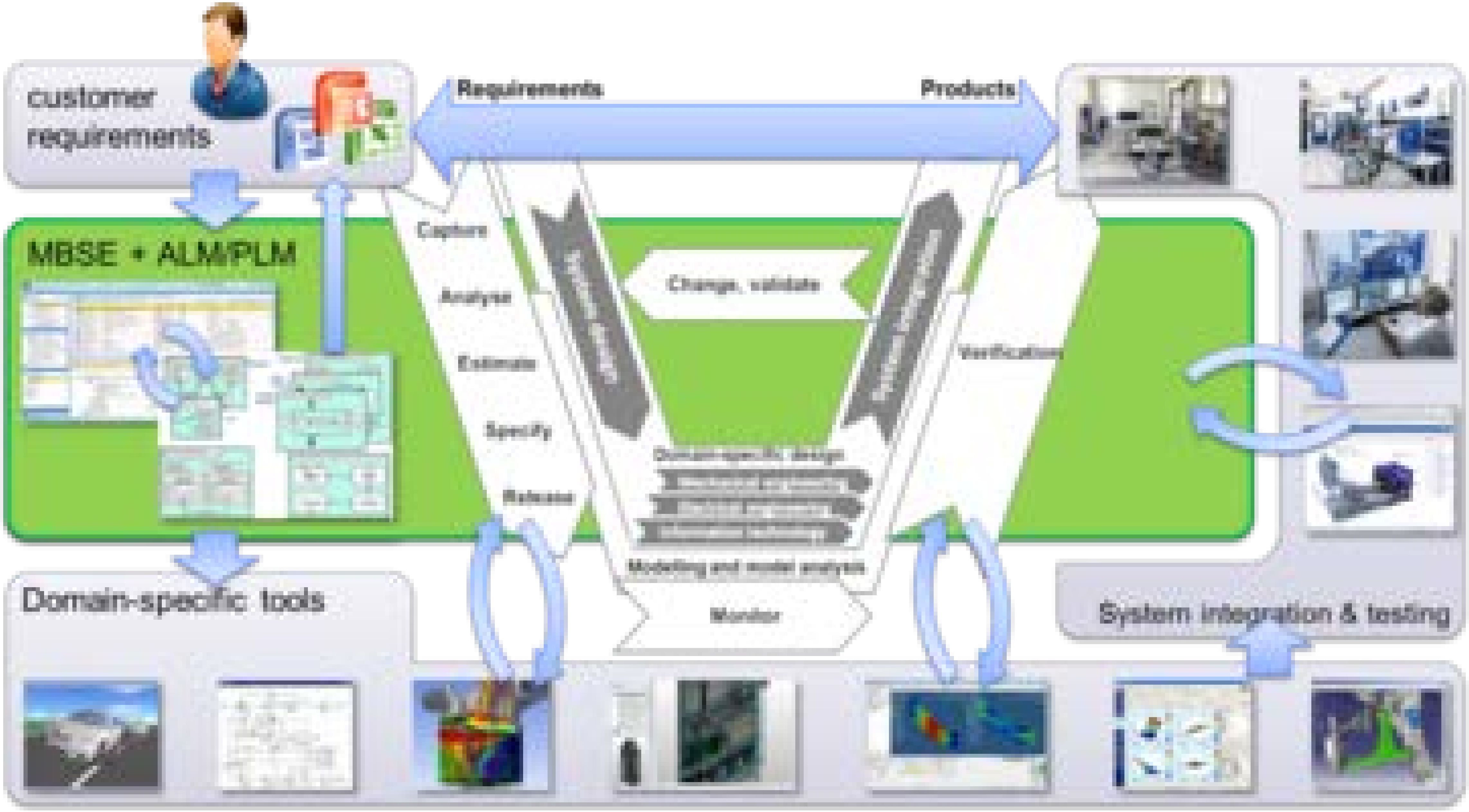
Systems Engineering: Use of Systems Knowledge

INCOSE

Systems engineering transforms real-world situations
to achieve desired outcomes in the real world
using resources available in the real world
based on knowledge

- The problem of engineering is to achieve desired behaviours (i.e. to transform situational results)
  - Constraint: use only resources available or producible in the environment
  - To qualify as "engineering", the transformation design must be based on application of knowledge (includes innovation: creation of knowledge)
    - Decisions and actions must be consistent with applicable knowledge
    - May include informal & tacit knowledge

Challenge: The knowledge belongs to many different domains, with different *vocabularies* used to express knowledge.
This leads to the problem of integration across knowledge domains



http://www.crystal-artemis.eu/typo3temp/pics/387401a247.png