


Adoption Practices for Model Driven Engineering (MDE)

presented at:

Lockheed Martin Architects Workshop

August 17, 2011

by

Mark R. Blackburn, Ph.D.

(Mark.Blackburn@stevens.edu)

561.637.3452

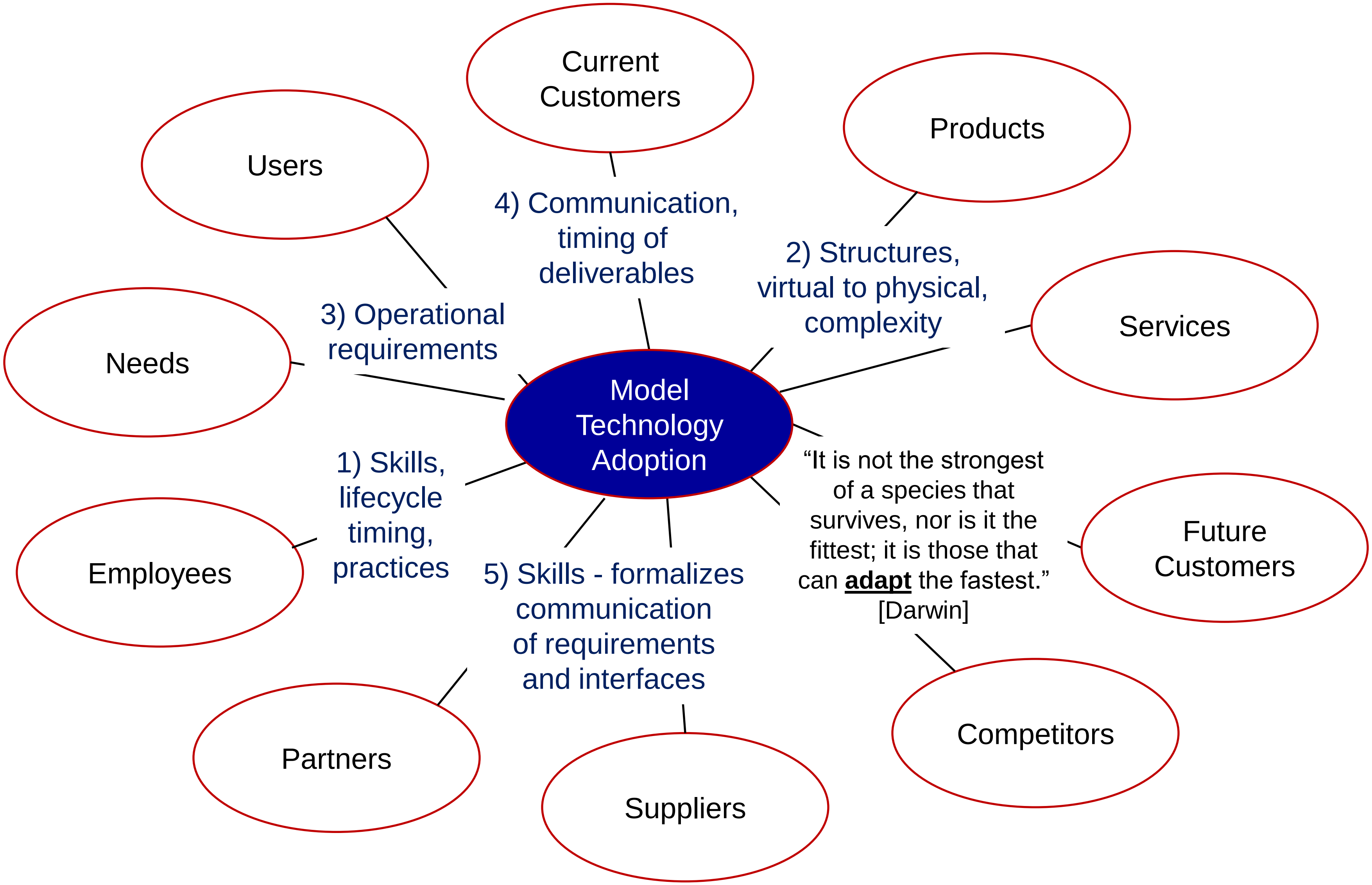
STEVENS INSTITUTE of TECHNOLOGY

THE INNOVATION UNIVERSITY

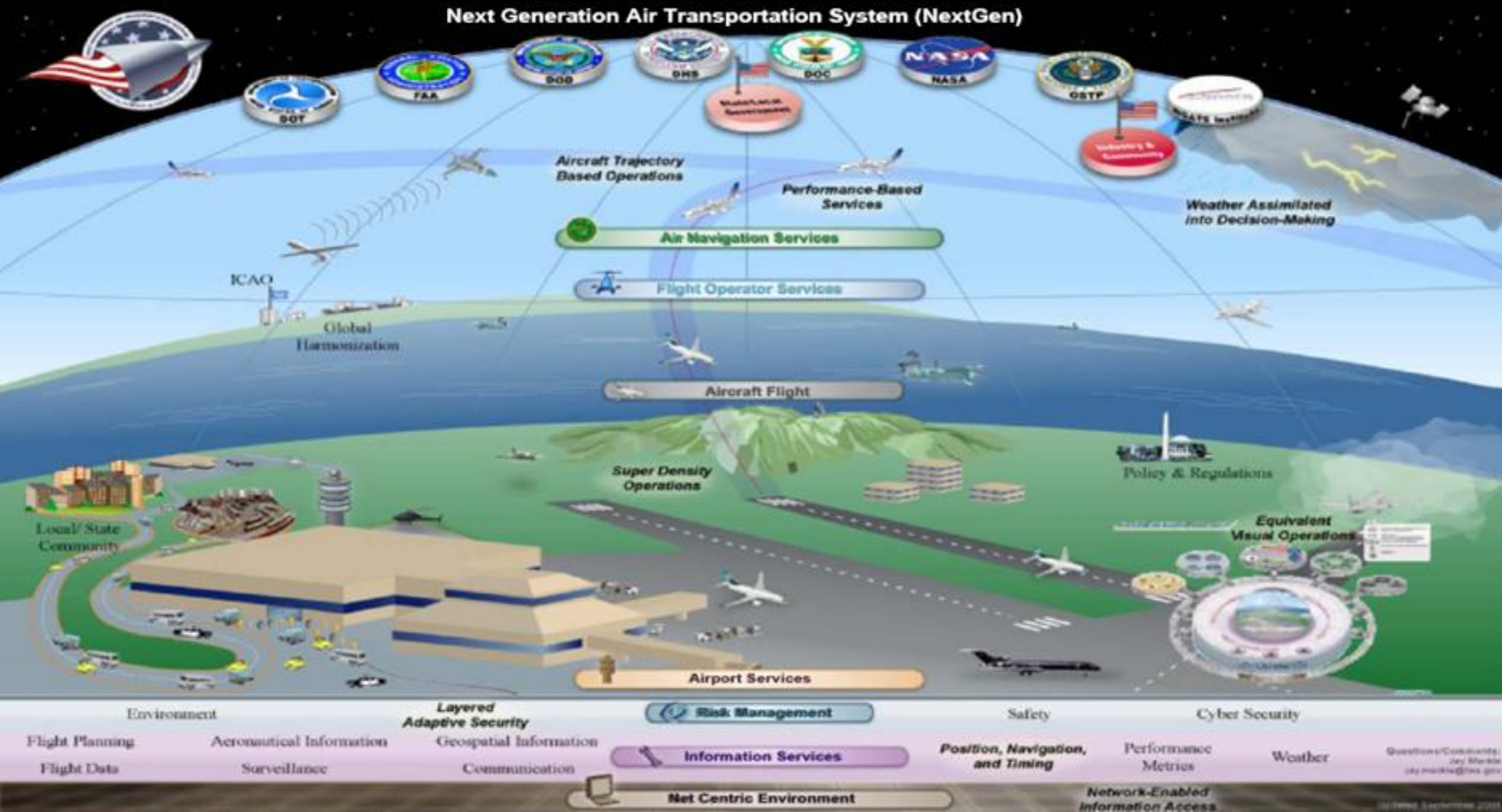
Overview of Presentation

- Context setting
  - Stakeholder perspectives on model adoption
  - Current situation
  - Vision for the future
  - Strategy for getting to the future state
- Part I: challenges and why architecting matters
- Part II: model adoption practices
- Part III: research topics that are emerging as necessary practices for System of Systems (SoS) MDE

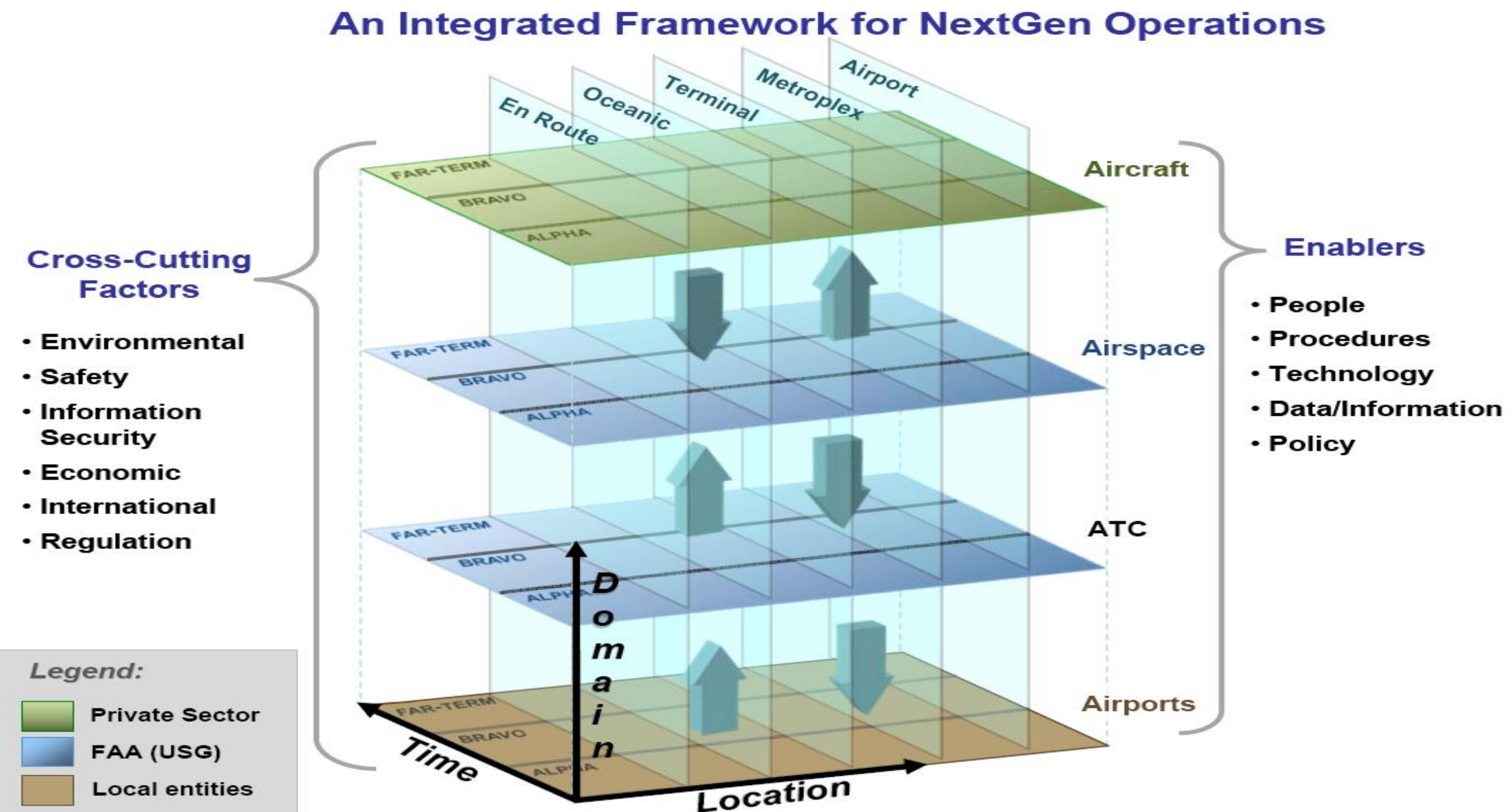
What are the Impacts of Model Adoption on the wide array of Stakeholders?



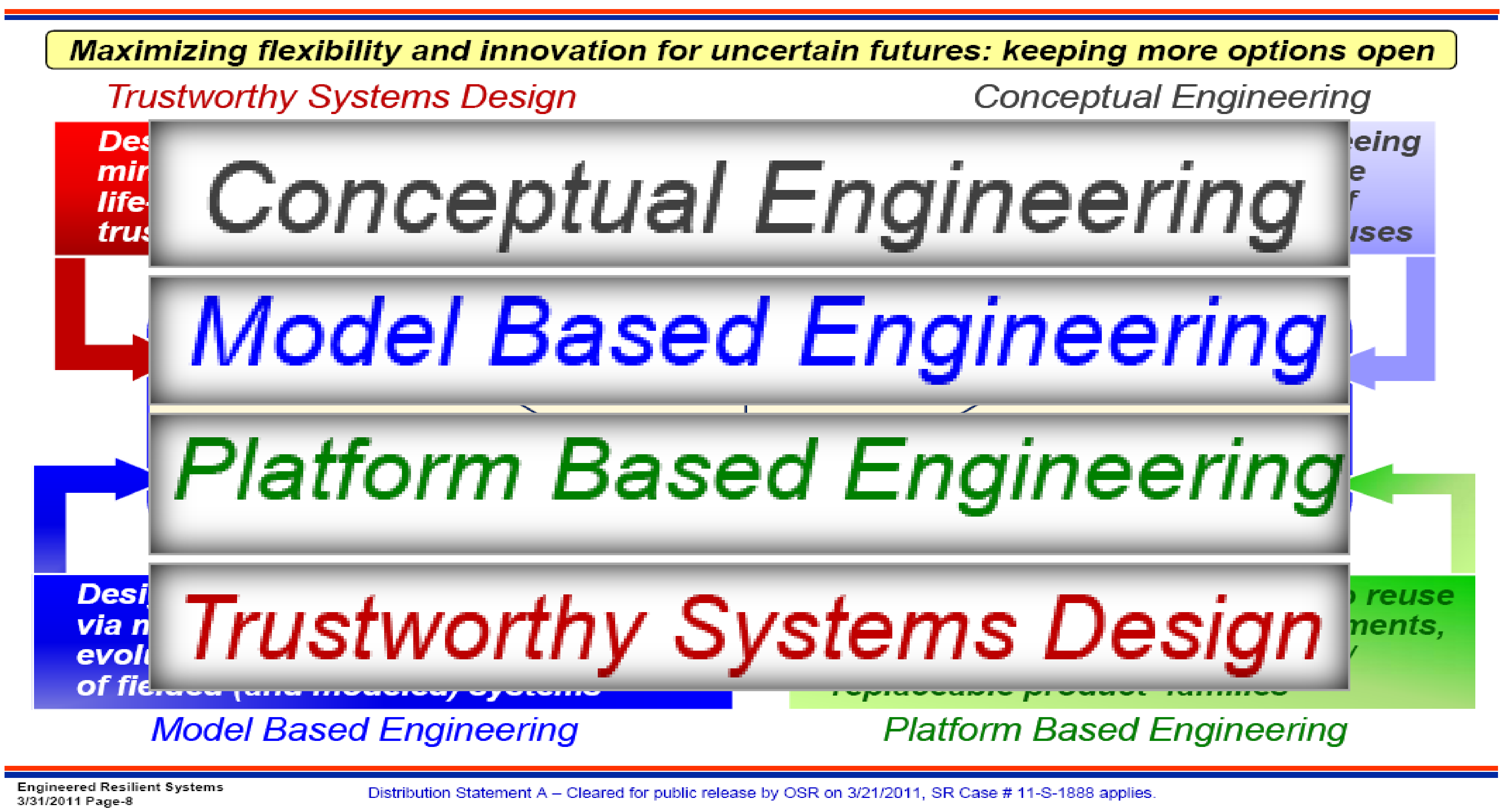
Today: Customers need continuous integration of capabilities spanning wide range of mission and system domains



Future Vision: Managing complexity in all dimensions through Systems of System Engineering

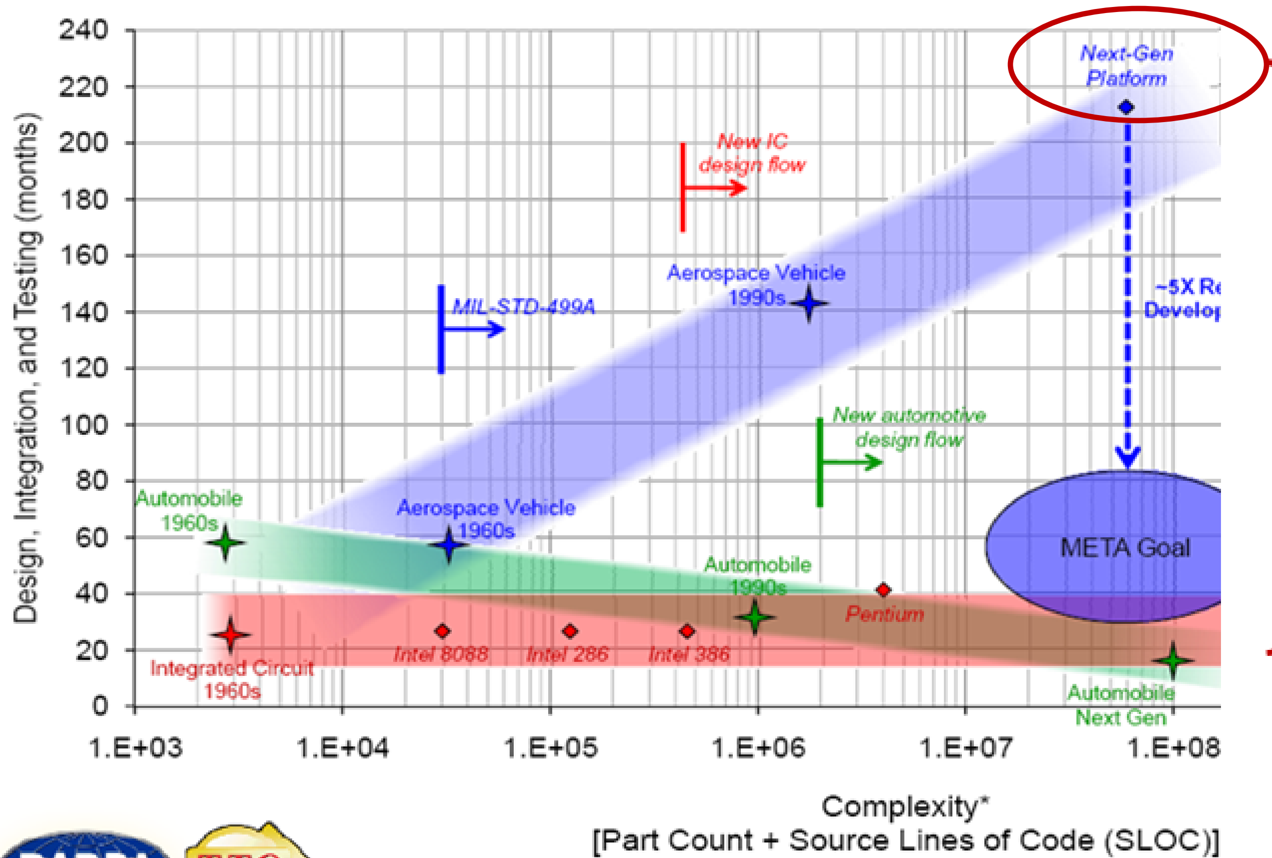


How do we get there (one strategy)? Engineer Resilient Systems: Architecting to rapidly adapt to user needs in uncertain futures



Systems of systems are emerging in many domains enabling unimagined complexity

We must produce systems of the same complexity as hardware with similar costs and schedules

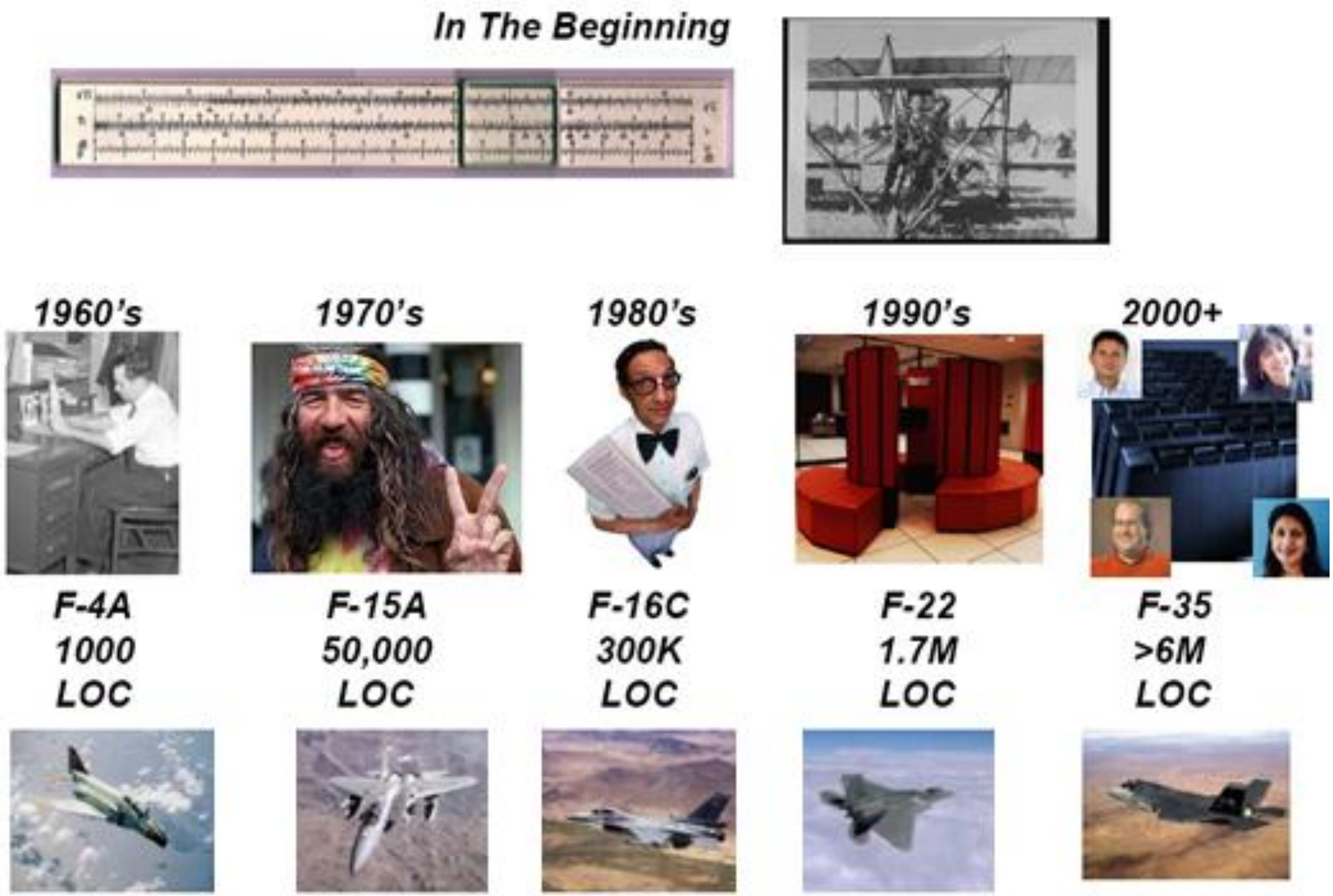


Note (*): Not a great metric. But that's what we have today. META will come up with better metrics.

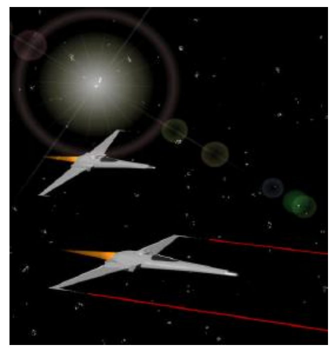
What's Different?

Software behavior often relies on floating point variables with nonlinear relationships and constraints

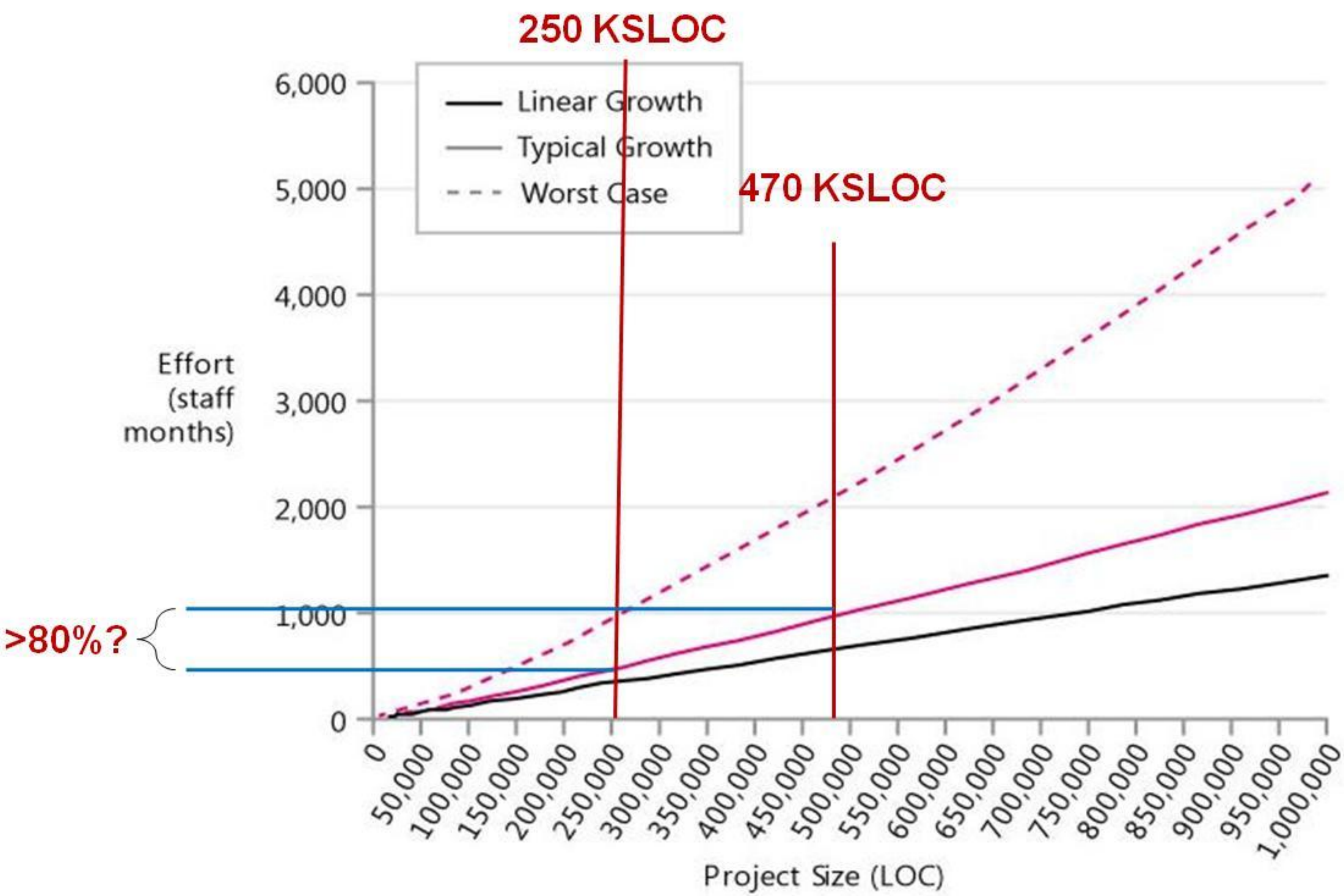
Augustine's Law – Growth of Software: Order of Magnitude Every 10 Years



2080?
F-50
>4.7B
LOC

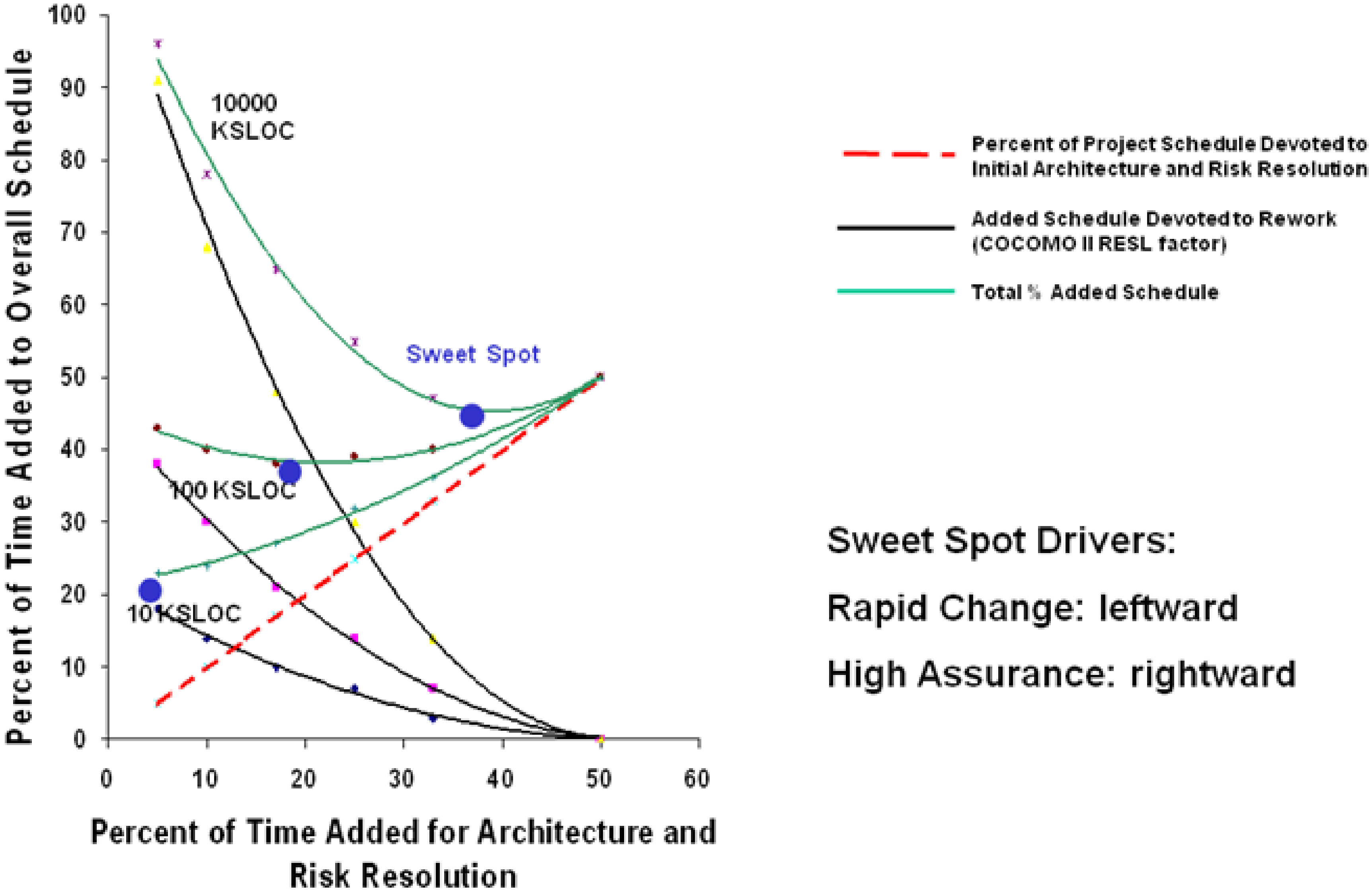


Complexity Results in Diseconomy of Scale In Software often impacting size, scope and cost estimates



Source: Computed using data from the Cocomo II estimation model, assuming nominal and worst-case diseconomies of scale (Boehm, et al 2000).

Optimal Architecting Matters: sequential path of least resistance often leads to delivery of poorly performing systems

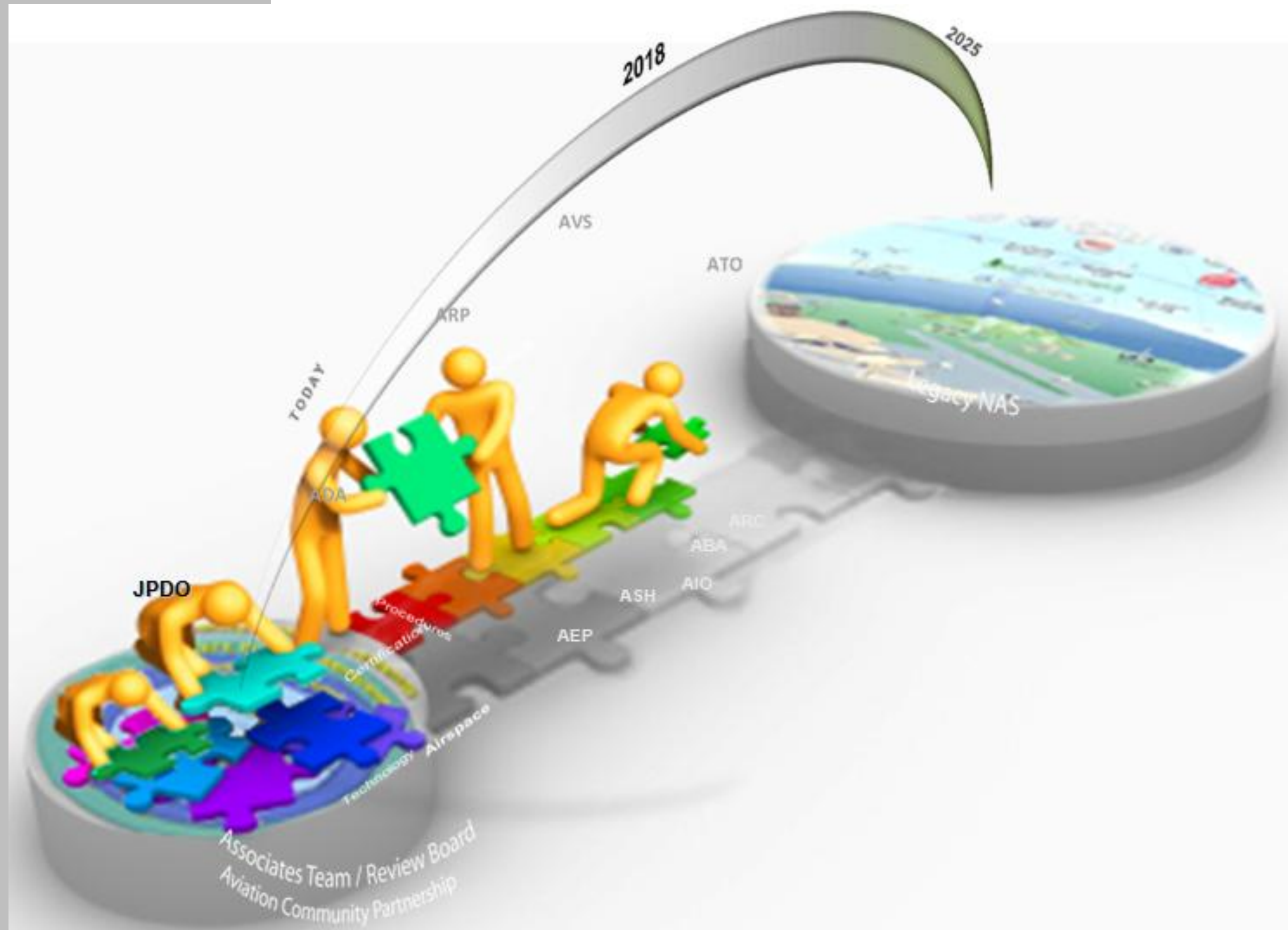


Sweet Spot Drivers:

Rapid Change: leftward

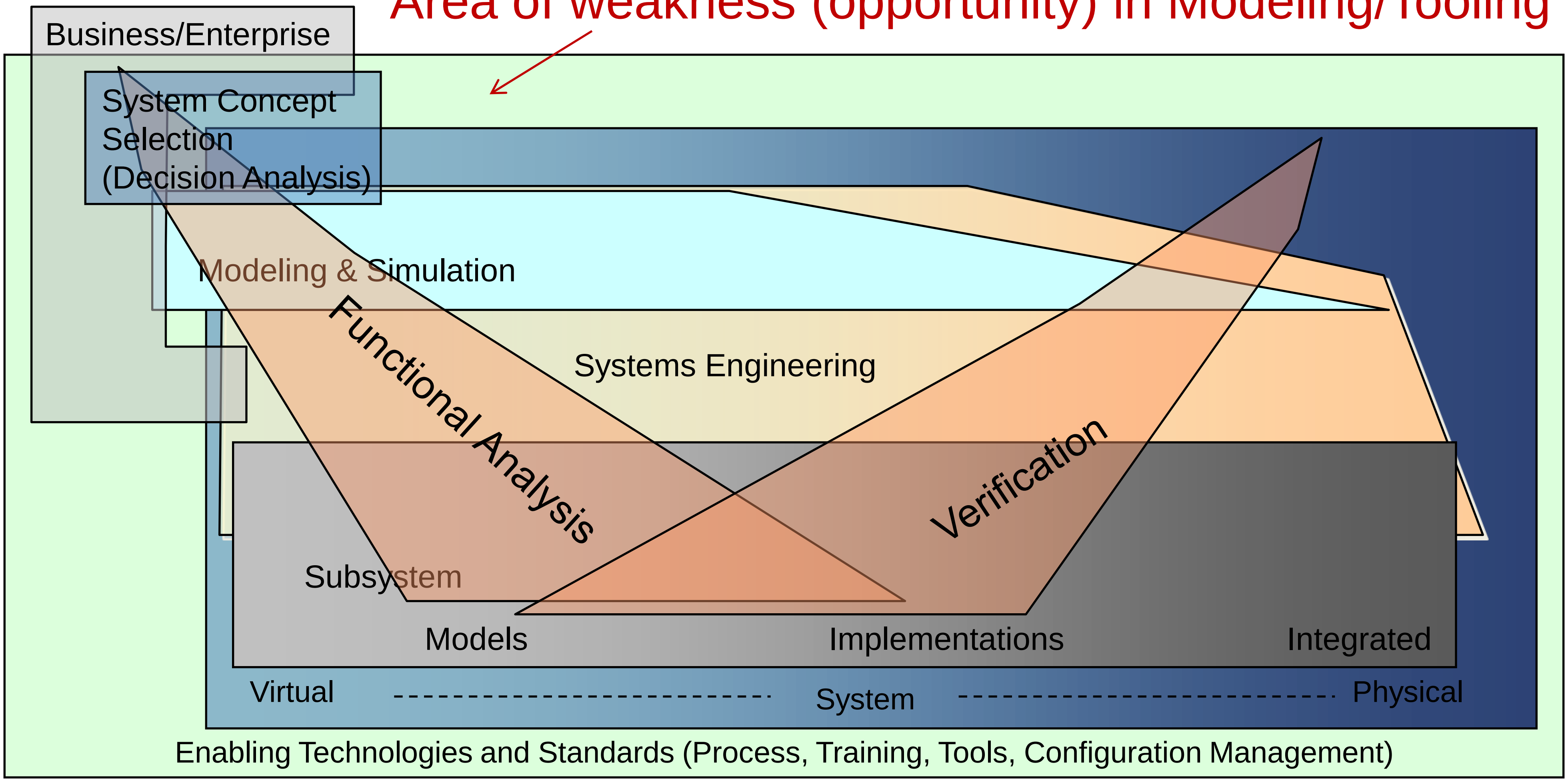
High Assurance: rightward

New capabilities need to be continuously developed, deployed, and evolved

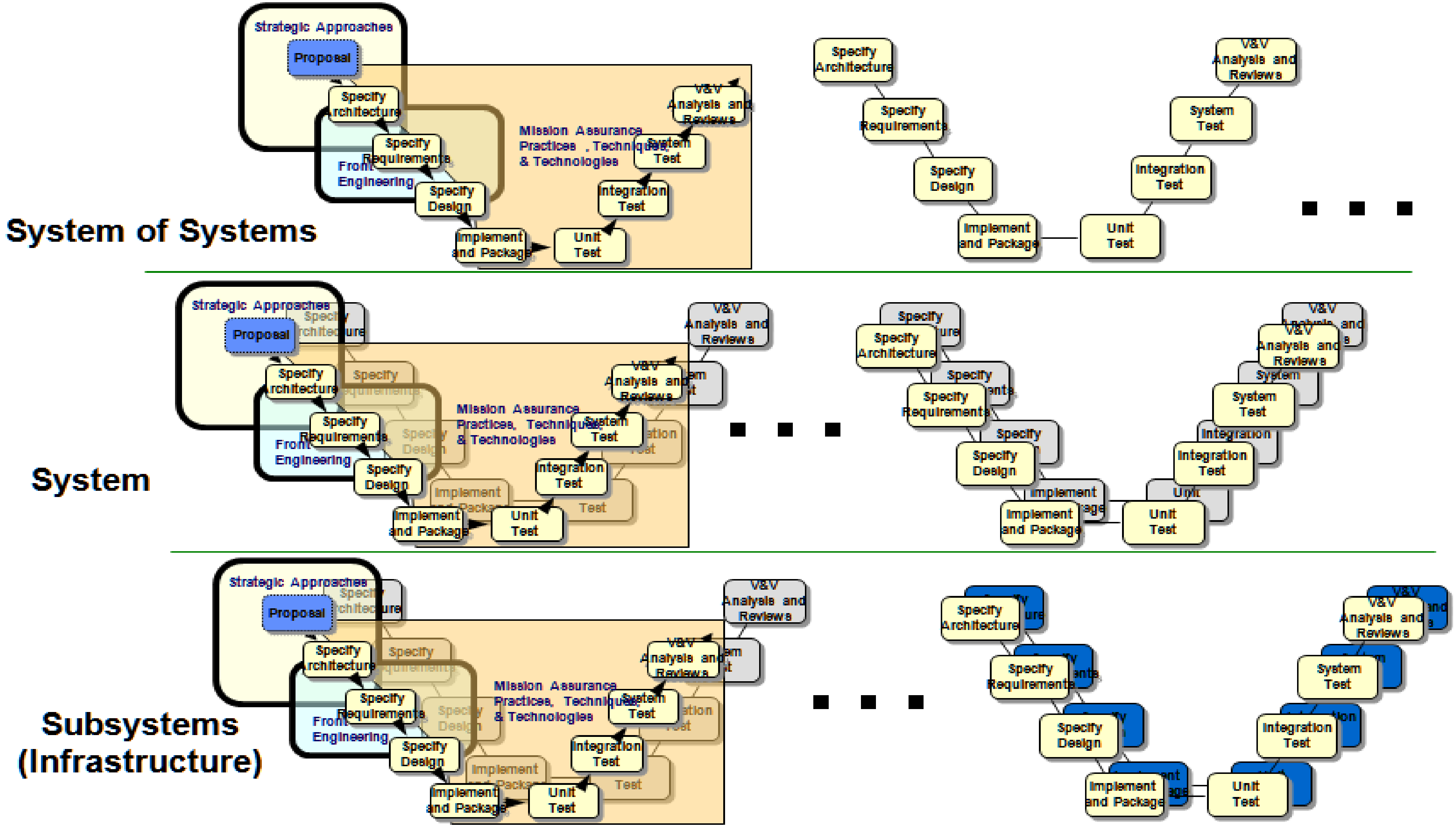


Functional analysis across SoS is required to understand tradeoff of capabilities (CONOPS) and impact analysis

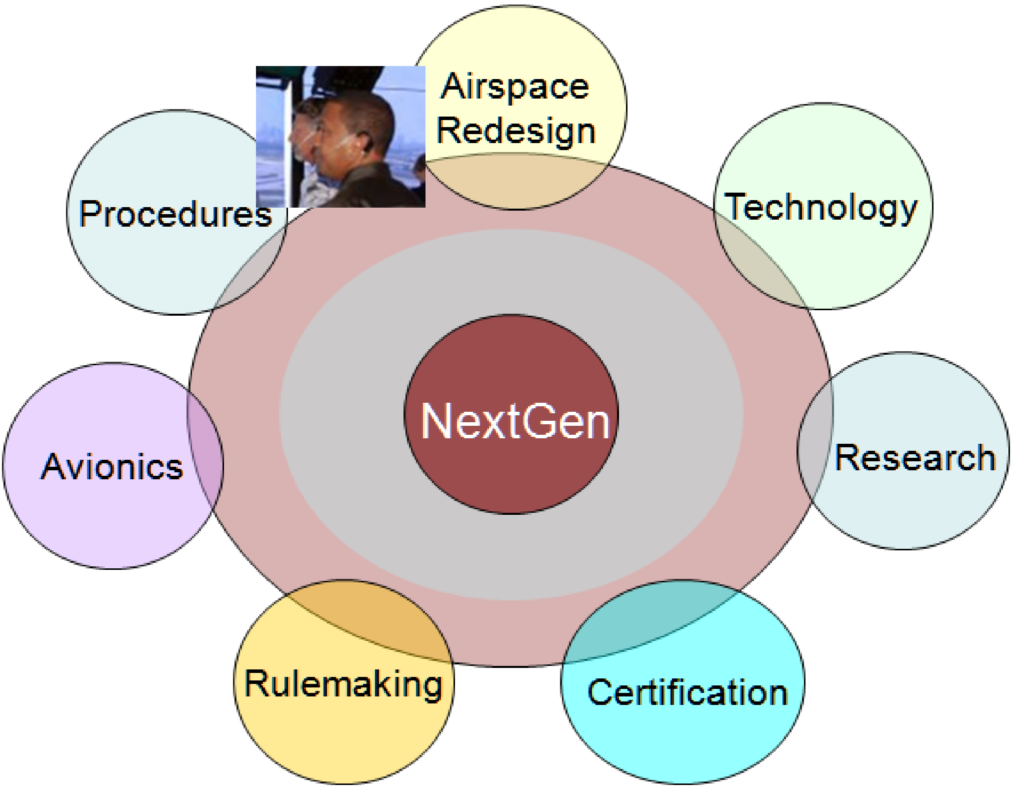
Area of weakness (opportunity) in Modeling/Tooling



Architecting is required for asynchronous integration and test across multiple SoS layers

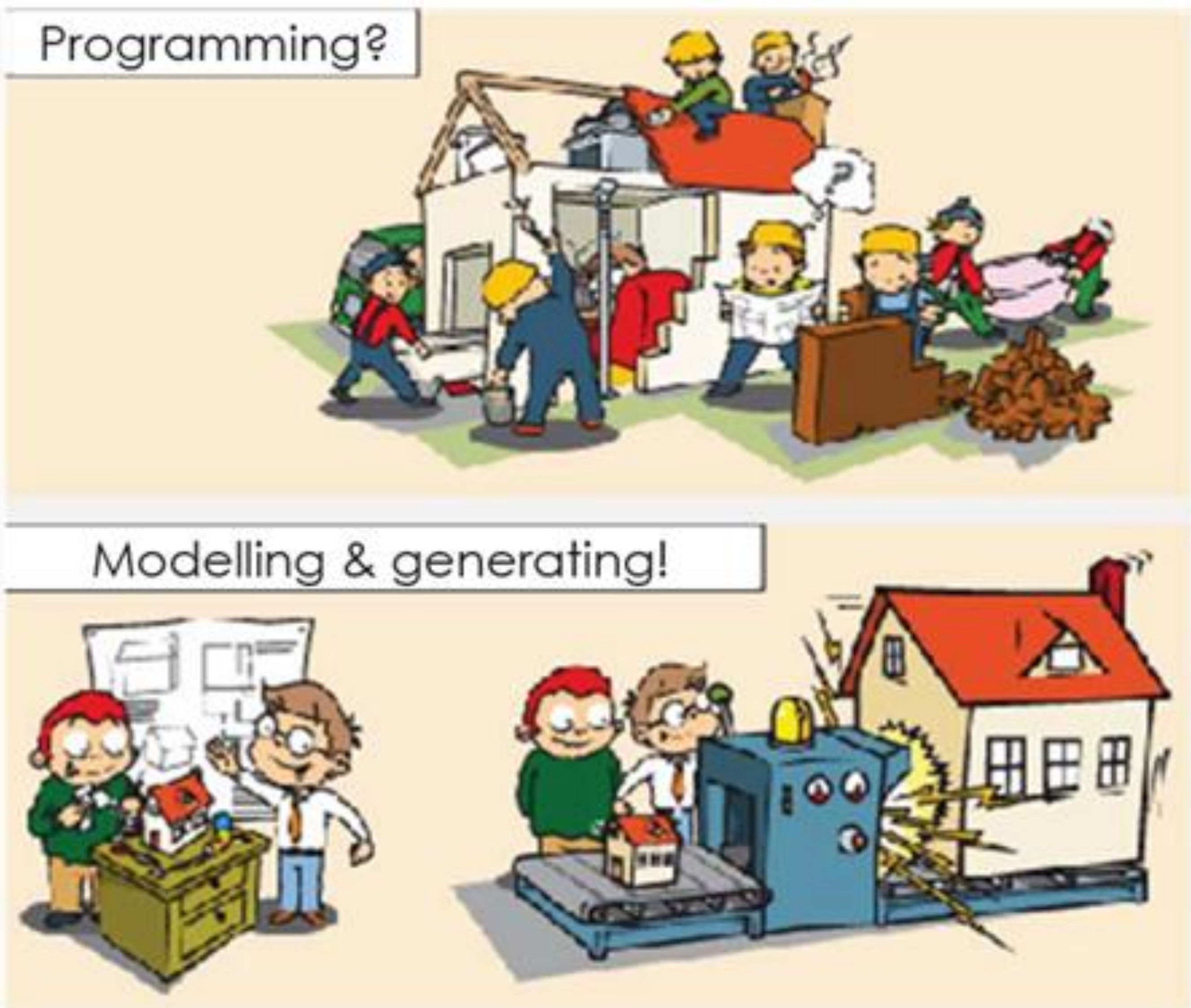


Transitioning into operations must accommodate users with mixed operational capabilities and maintain trusted system properties

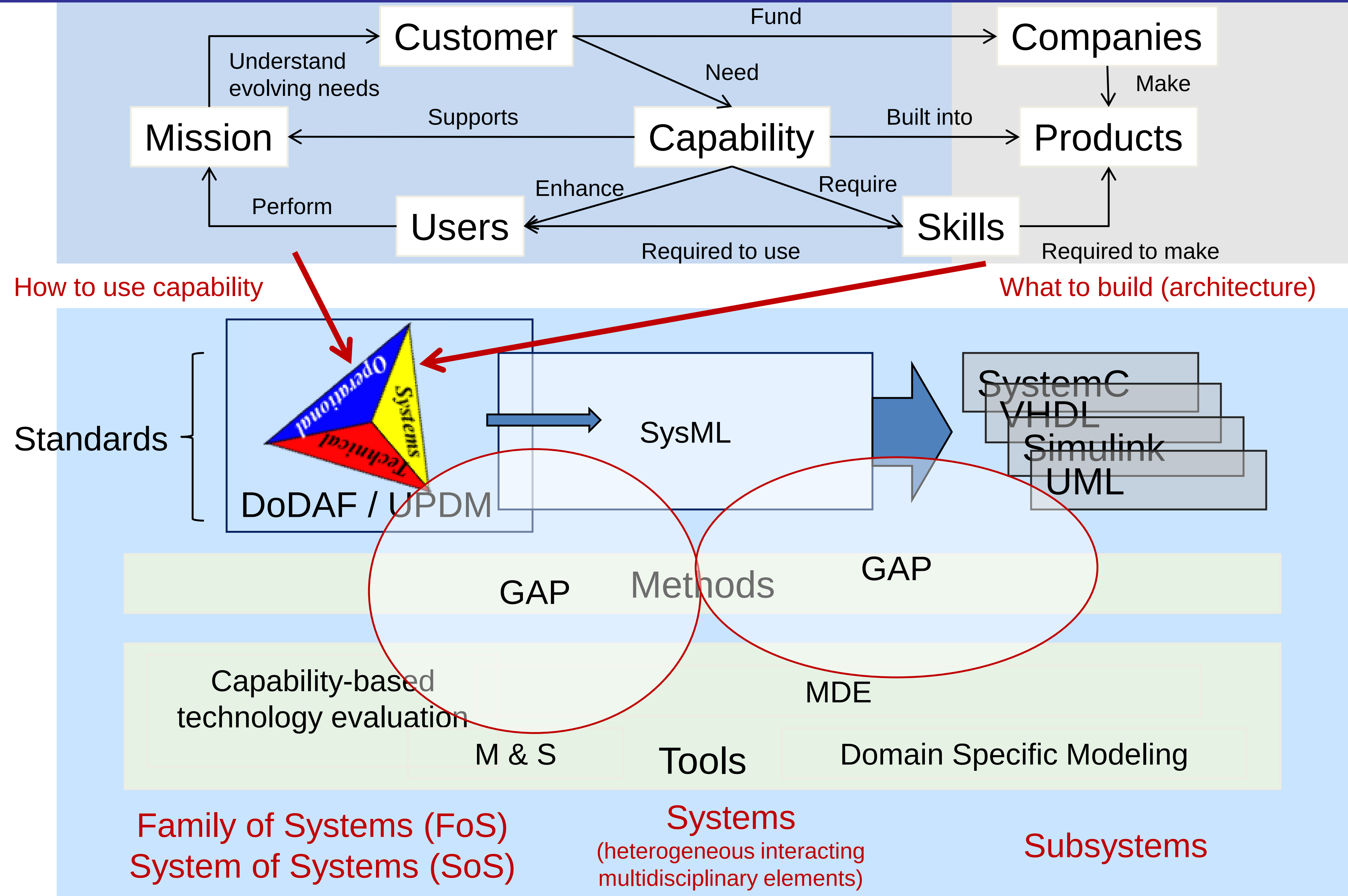


“We model to reason about the problem… And to communicate with others.”

Model Driven Engineering will revolutionize concurrent engineering

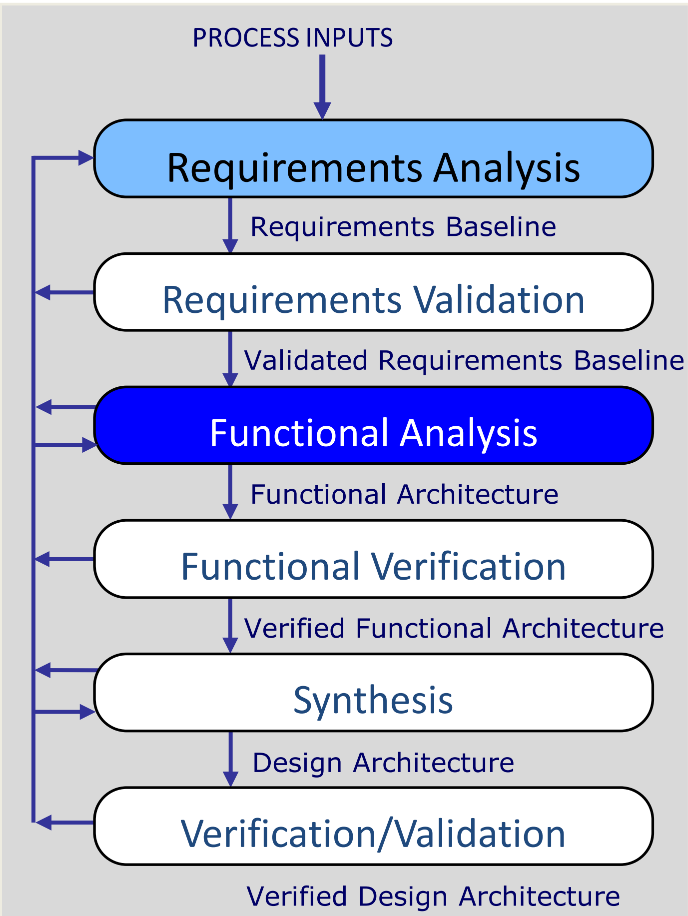


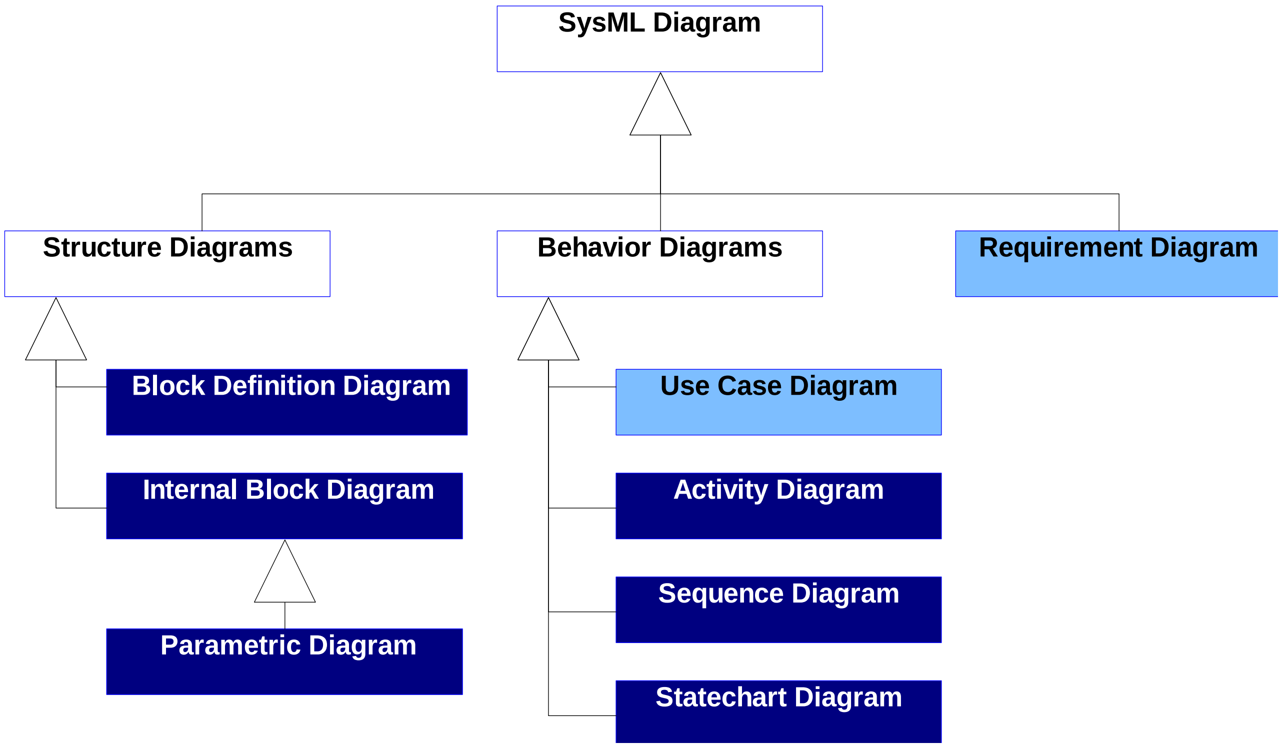
MDE approaches and tools must address gaps



From traditional SE → Model-Based SE – standard, structured, rigorous, & linked

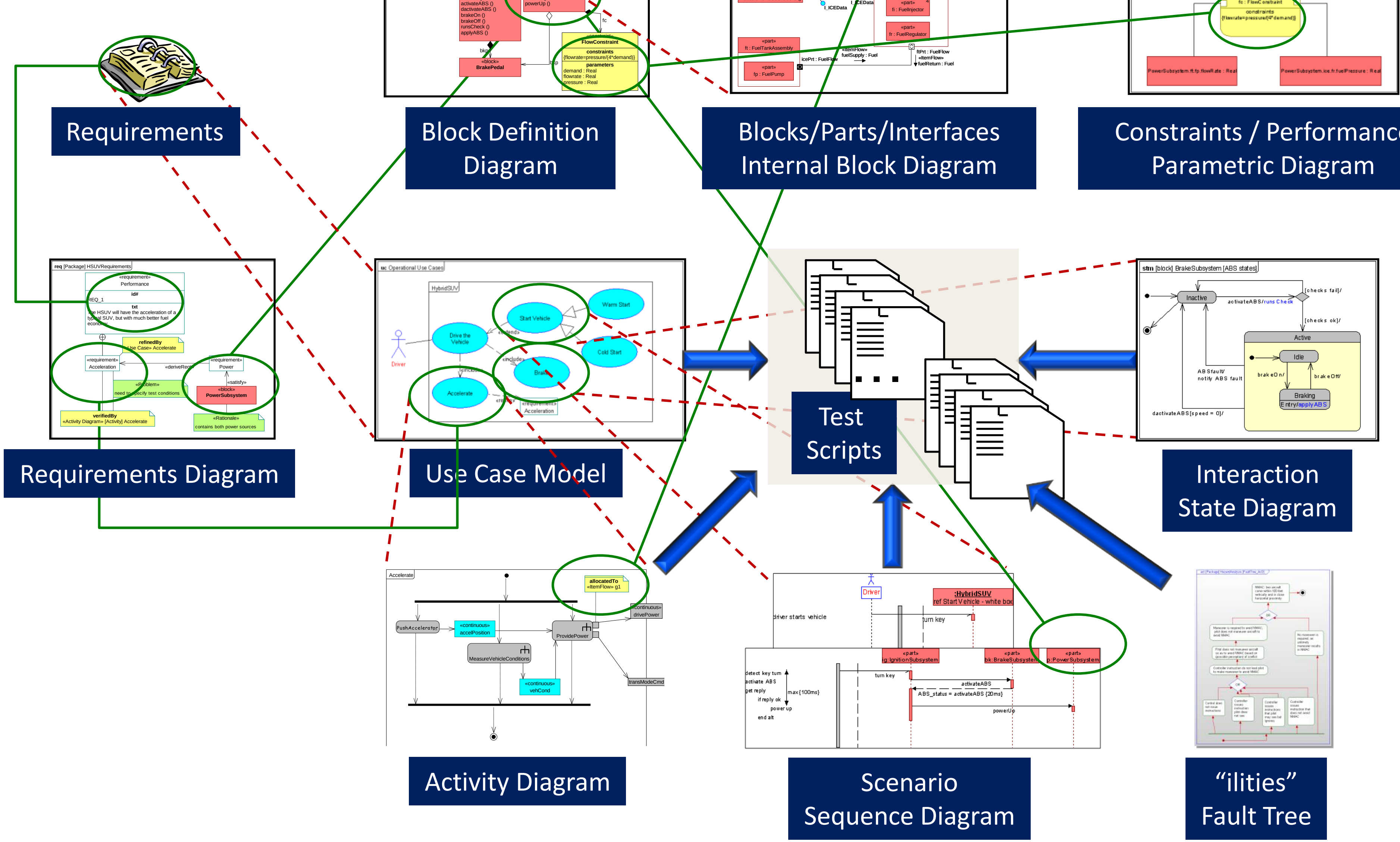
IEEE 1220 SE Process



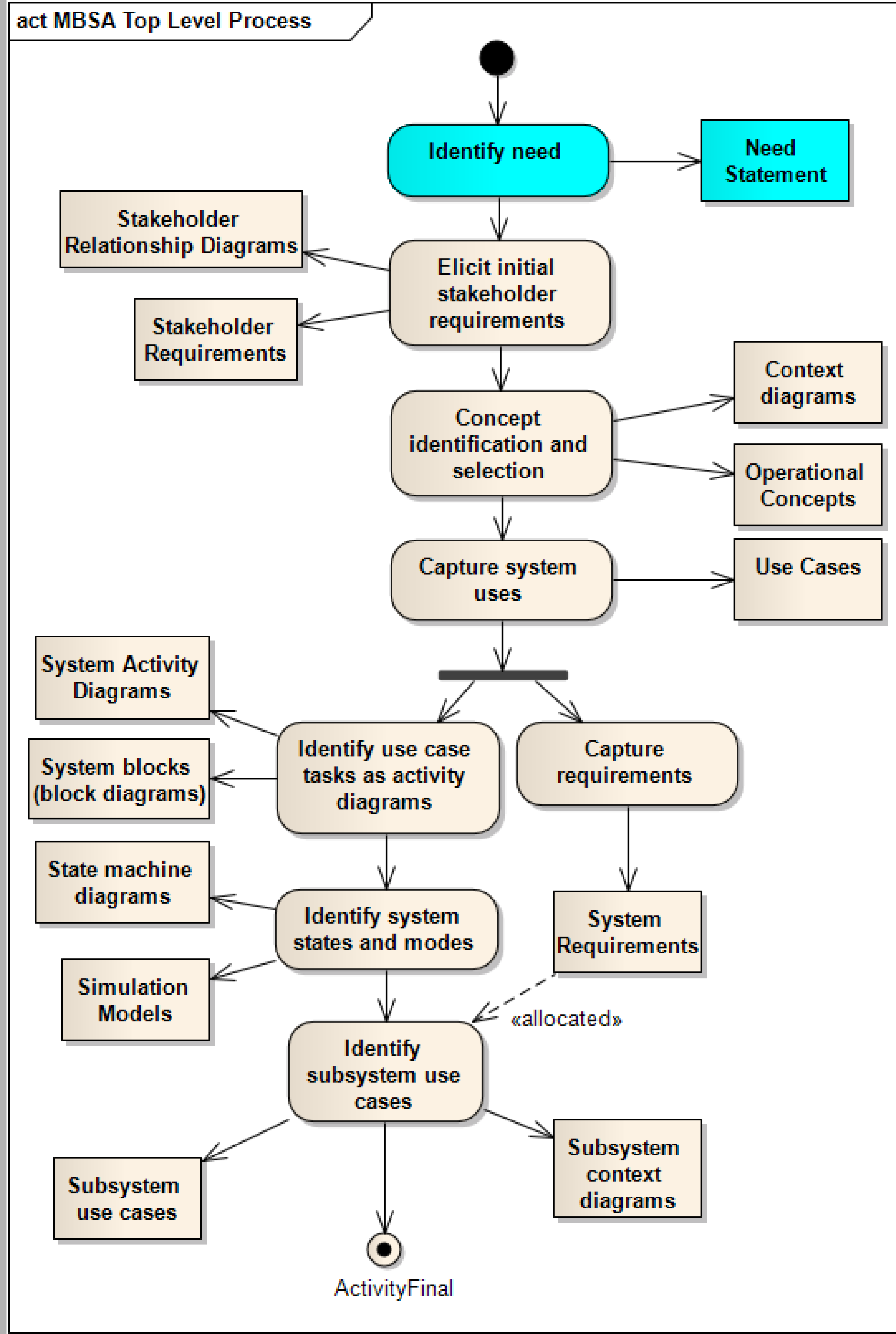


Many tools support a typical SysML usage scenario

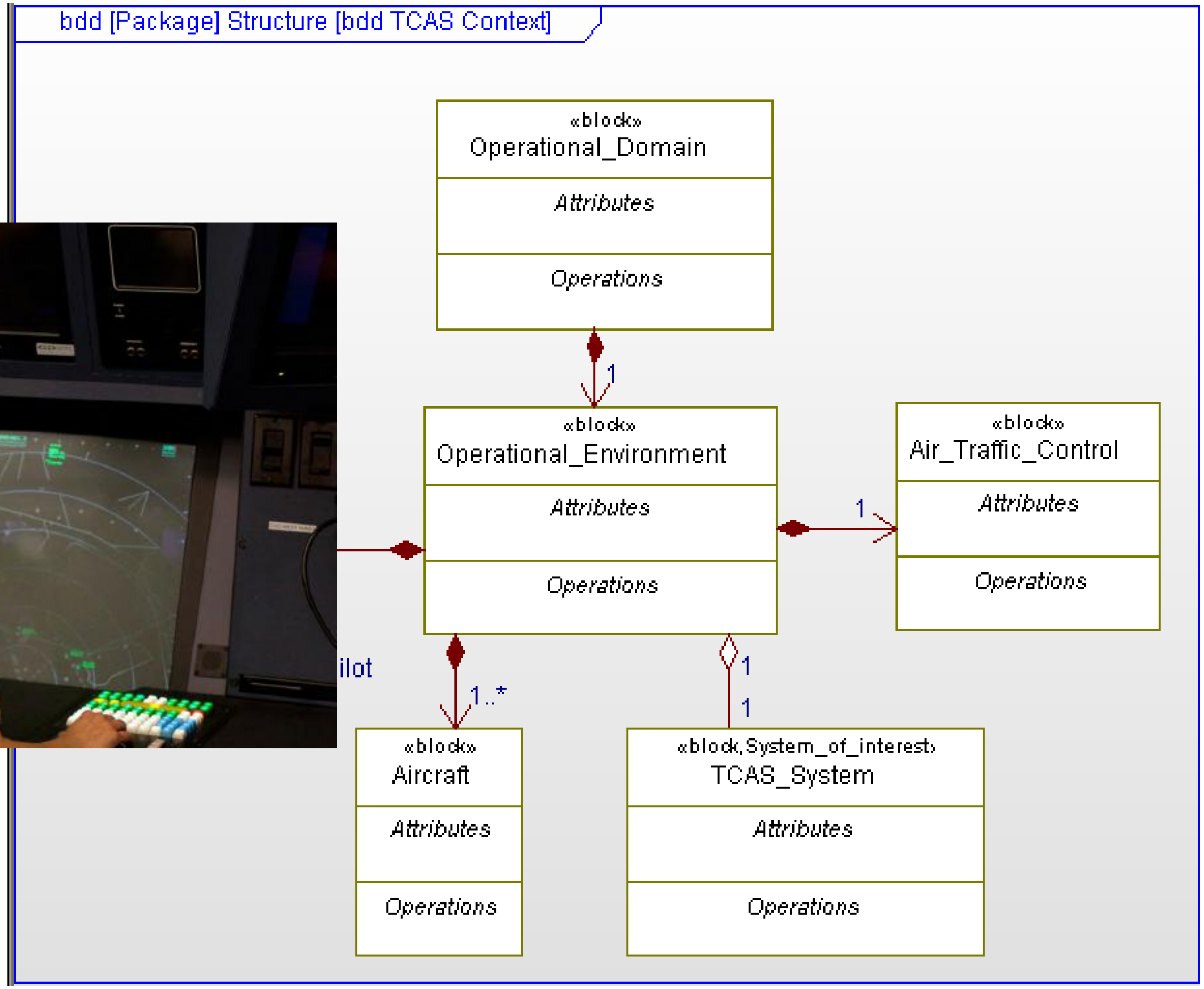
(Build me to see how it works)



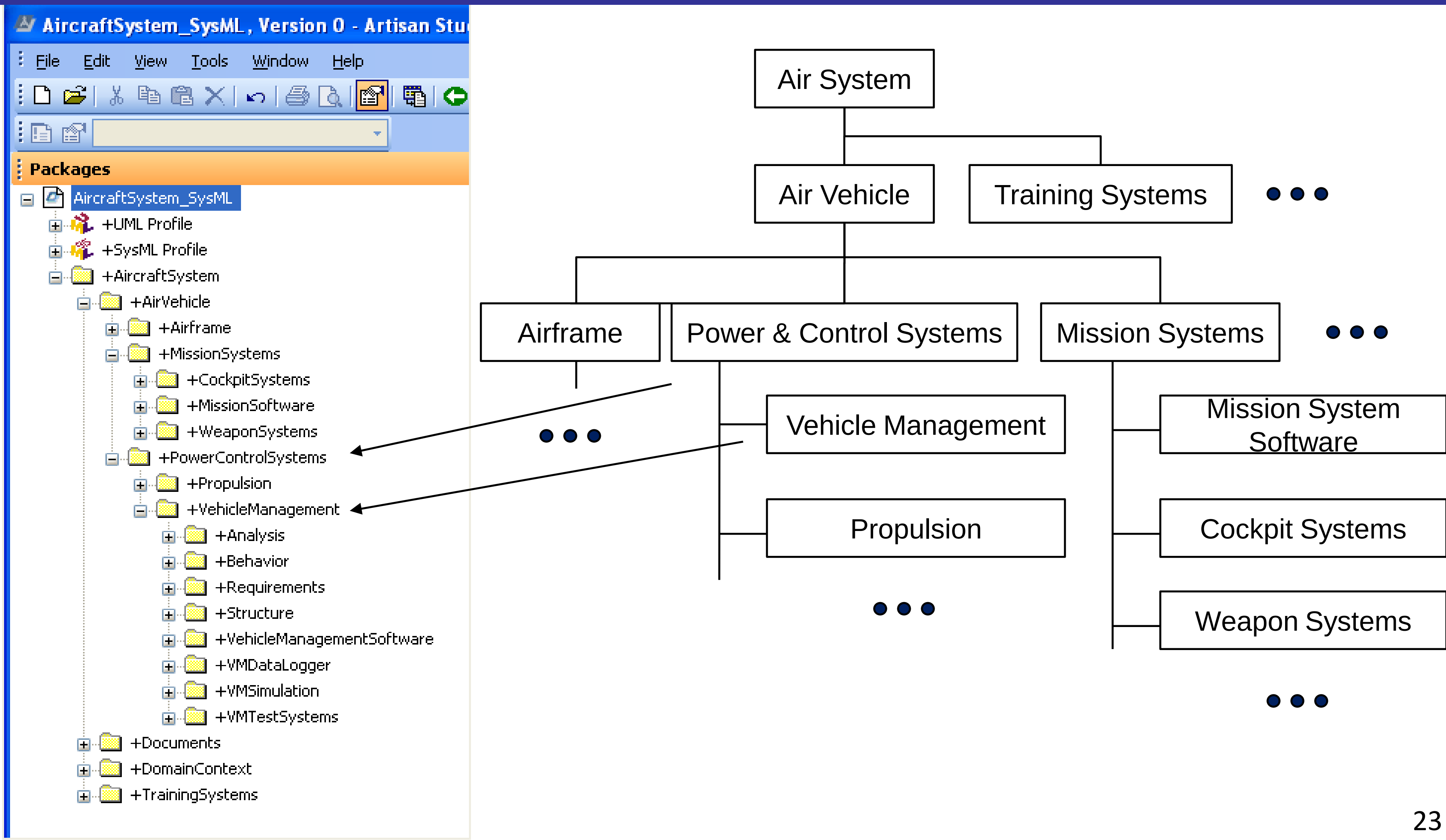
Model Driven Engineering covers concepts, practices, tools, and future ideas – this is a core process for MBSA/MBSE



Structural views should include system domain, context, and interfaces

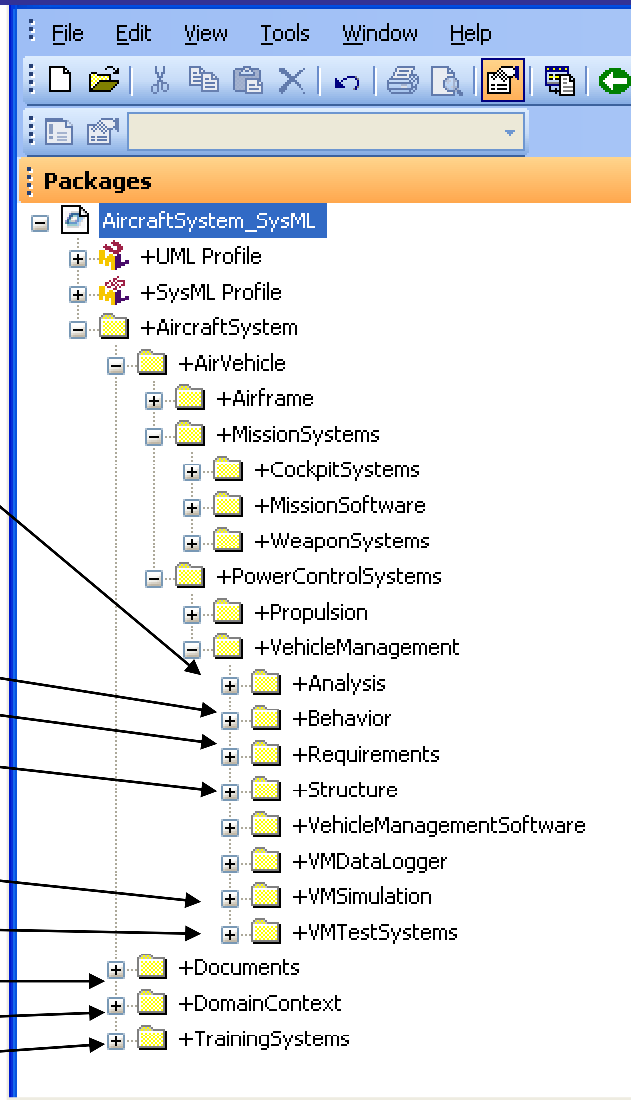


Model topology often mirrors architecture of system

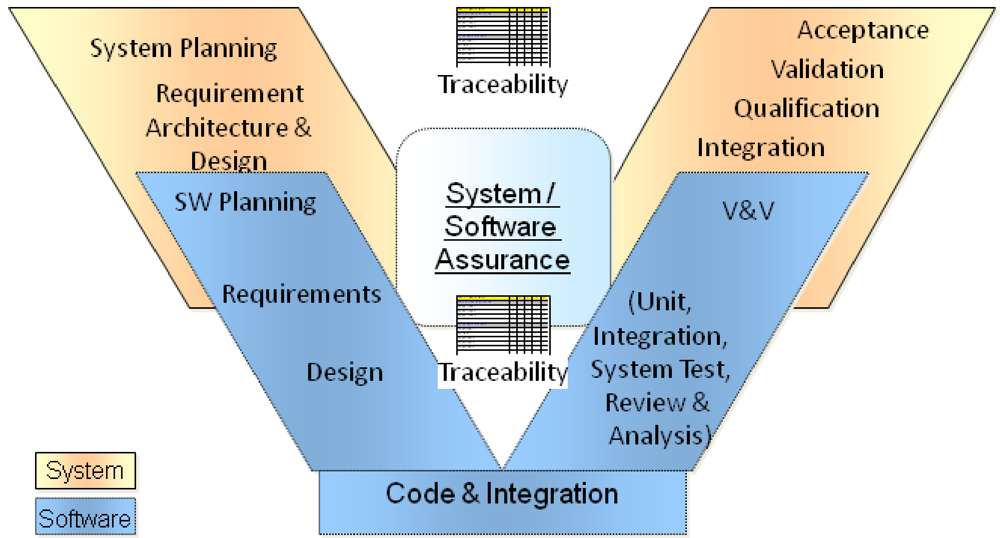


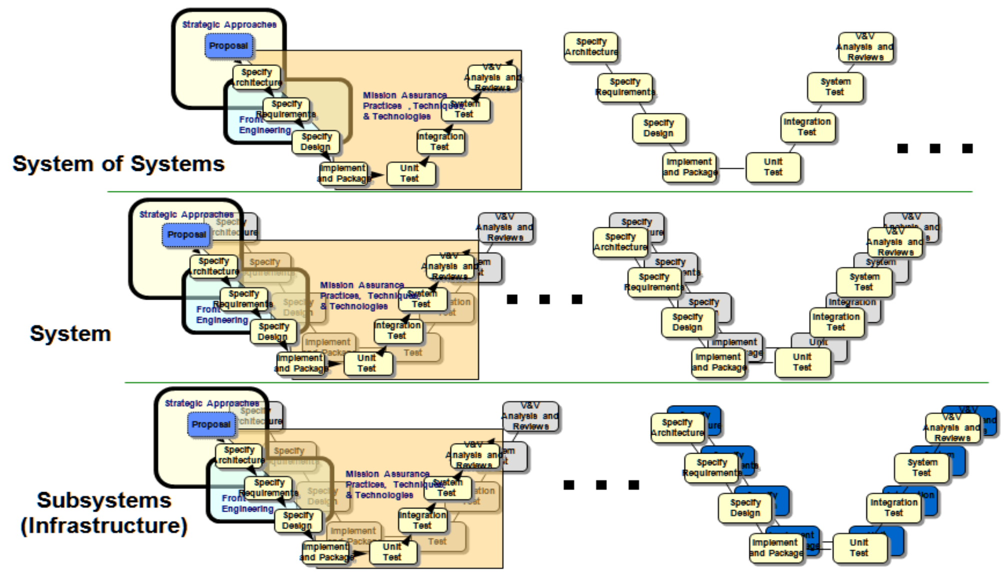
Establish common package elements to organize and structure model

- Change log
- Recursively applied
  - Analysis – could have subpackages
    - Context
    - Mission concept
    - Operational concept
    - Stakeholders
    - Scenarios
    - User cases
  - Behavior
  - Requirements
  - Structure
    - Architecture
- Simulations
- Variants
- Verification and test
- Documents (controlled)
- Domain context
- Environment



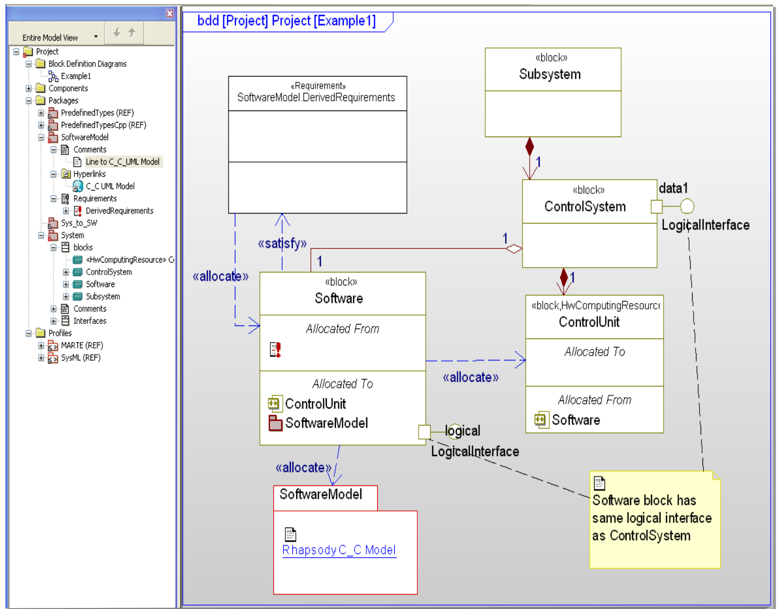
Model artifacts trace requirements through views and map derived requirements to software / hardware subsystems



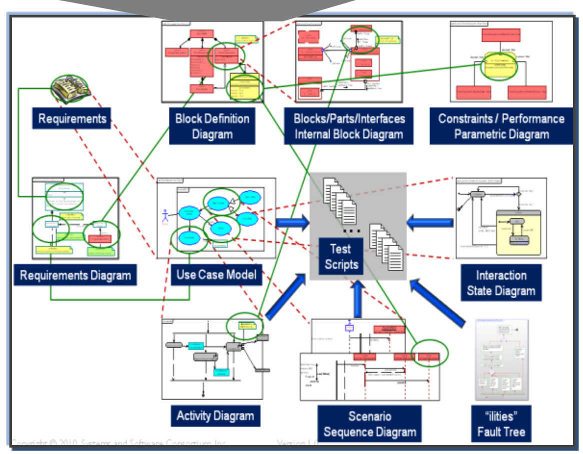


For continuous integration and test, we must be able to understand all of the interfaces and allocated requirements

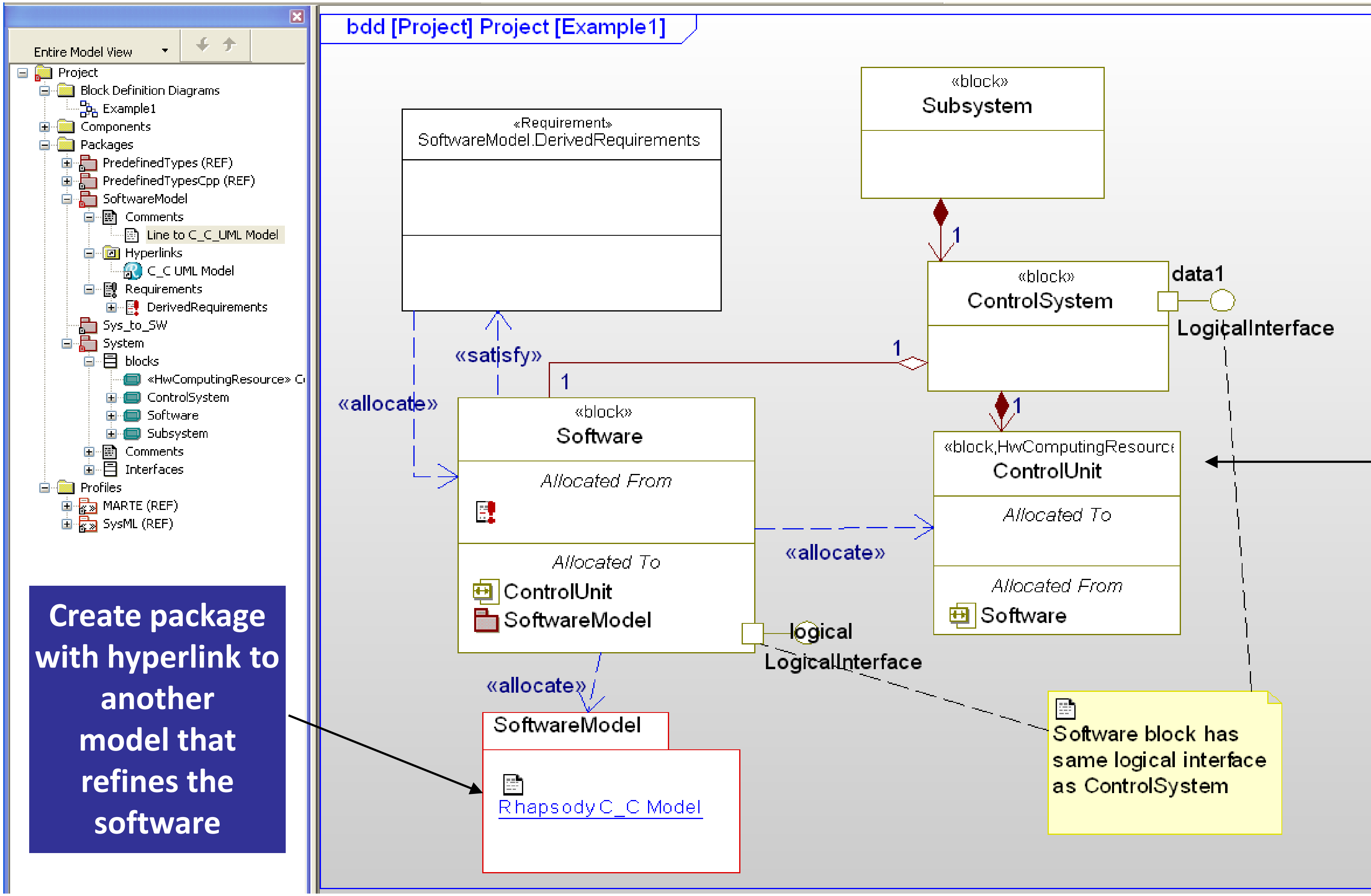
See next slide for example pattern for mapping HW/SW



Context



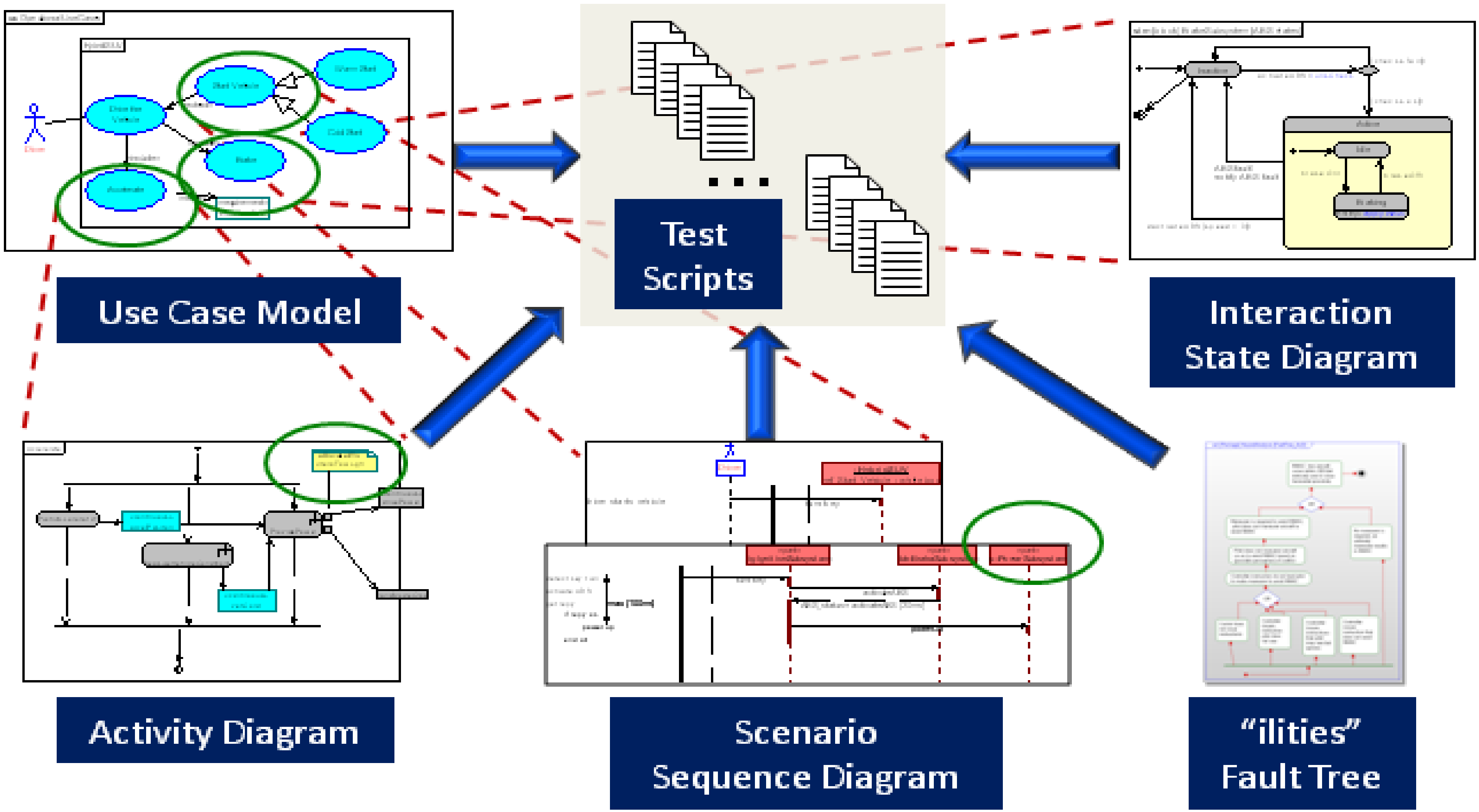
Models depict interfaces and derived requirements allocated to SW & HW



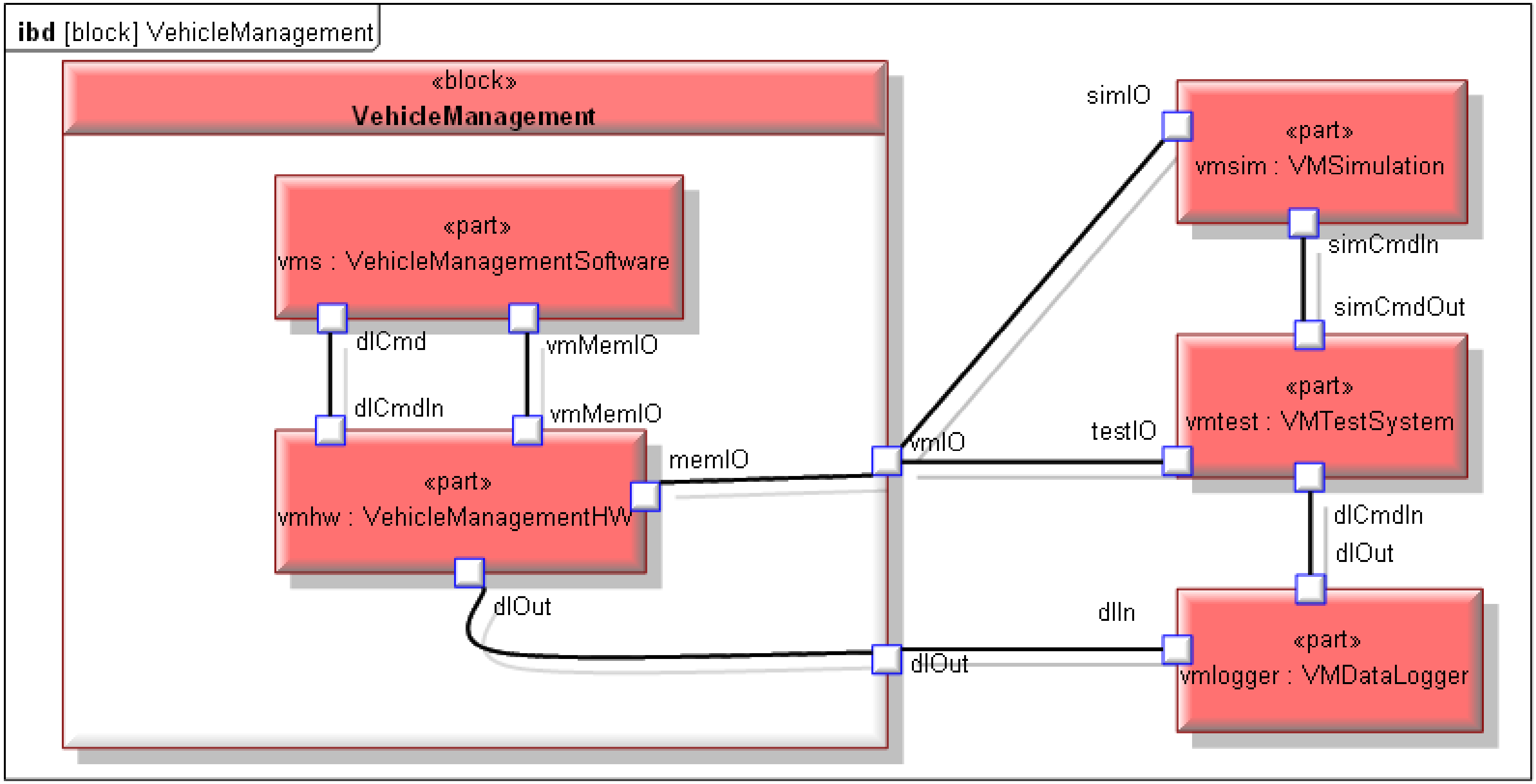
Create package with hyperlink to another model that refines the software

Example use of MARTE Stereotype for HW Computing Resource

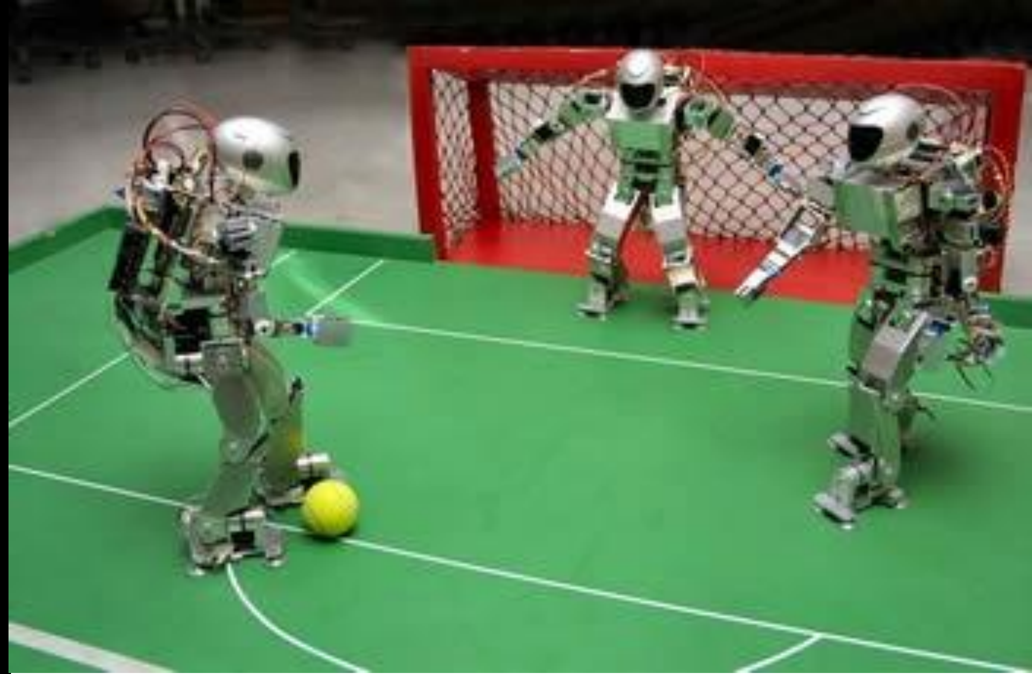
Behavioral views provide inputs for continuous integration and test planning and execution



Architect for testability to support automation and leverage simulation and legacy data



We must engage stakeholders in new ways to adapt faster and to determine what works, what doesn't, and how it should be used



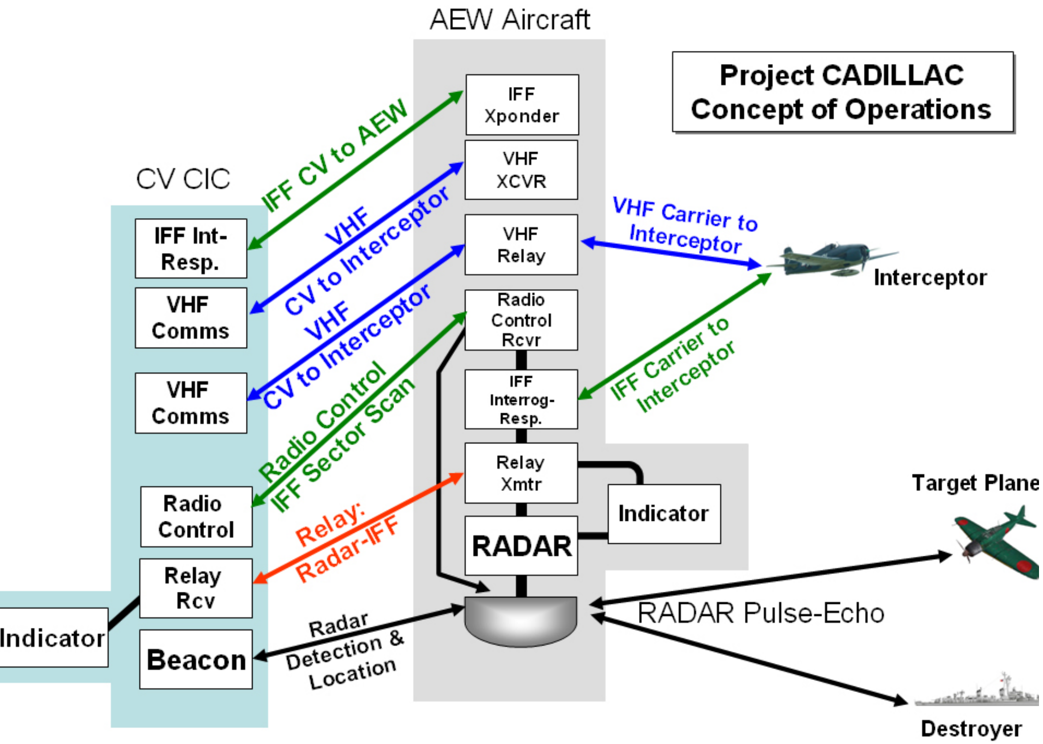
Choose your game

A key focus is on developing the CONOPS for capabilities that need rapid deployment

Operational views are critical as they represent how the system is to be used

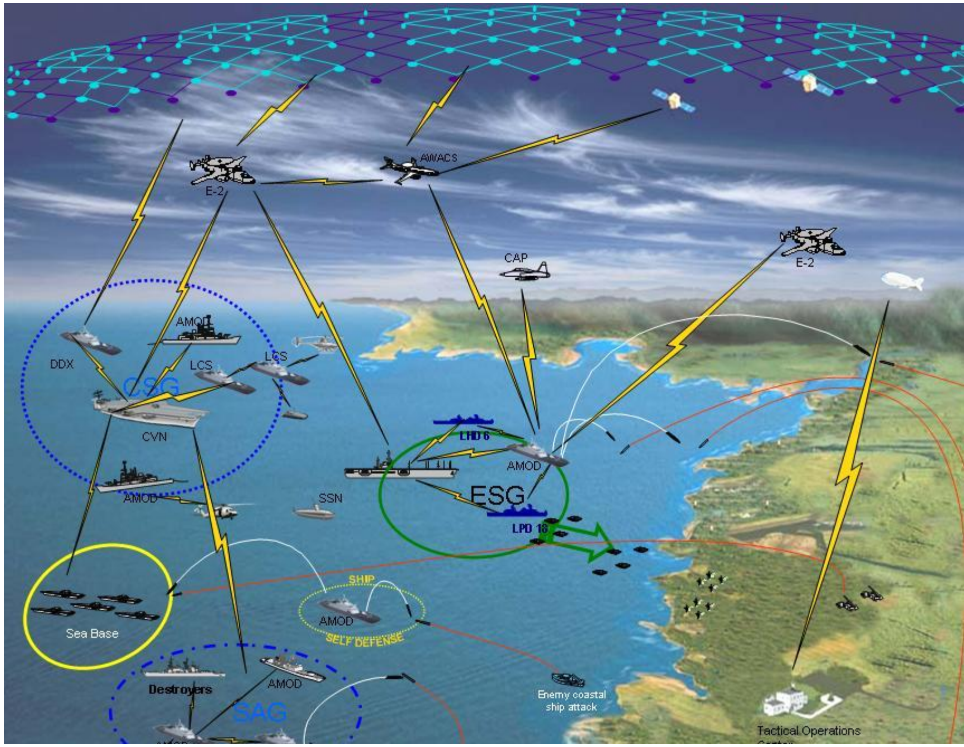
CONOPS: Then and Now
We have not progressed far - no meaning behind graphics, no human roles represented, takes too long, and customer often not involved

First Airborne Early Warning System to defend against aircraft (1945)



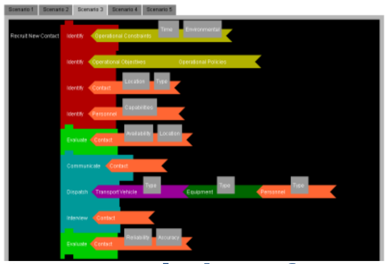
US Naval Institute Blog, http://blog.usni.org/?s=AEW&x=0&y=0

CONOPS from any current Naval program

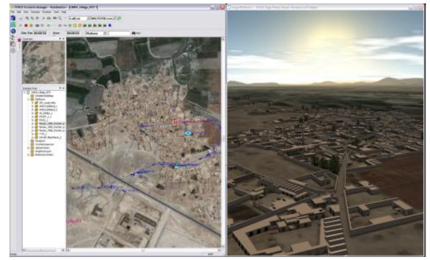


New modalities and engineering capabilities are required to manage exponential complexity

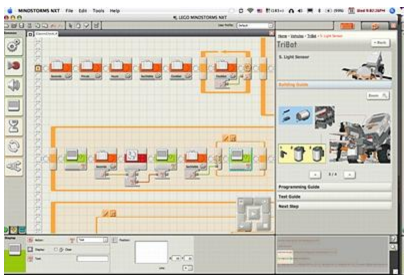
Concept engineering through graphical storytelling builds capability scenarios that are executable to understand dynamics and tradeoffs



Lego-style interfaces



Rapid Virtual Environment generation



Graphical Programming

Virtual Environment to CAD tool translation

"Human-Centered Design"

Gaming Platforms

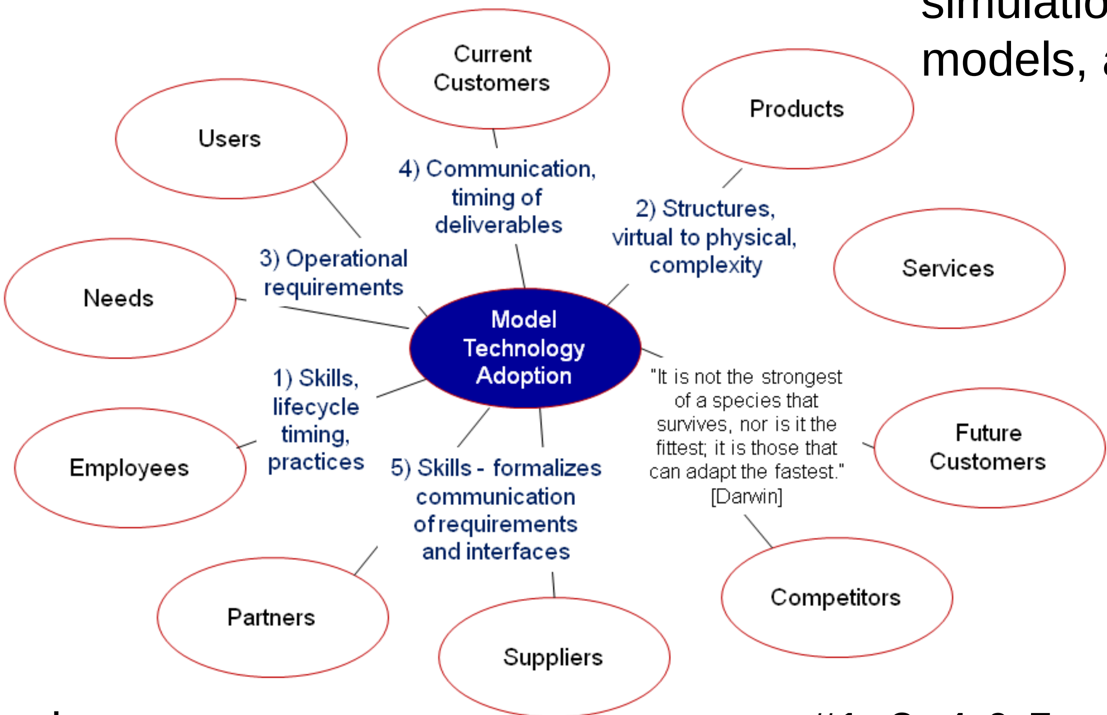
Immersive Virtual Environments

Graphical CONOPS can be leveraged for virtual training addressing challenge of evolving operational capabilities

Successful model adoption often uses pilot projects to reduce risk

#1 – Understand how to relate traditional process activities to modeling practices and modeling artifacts

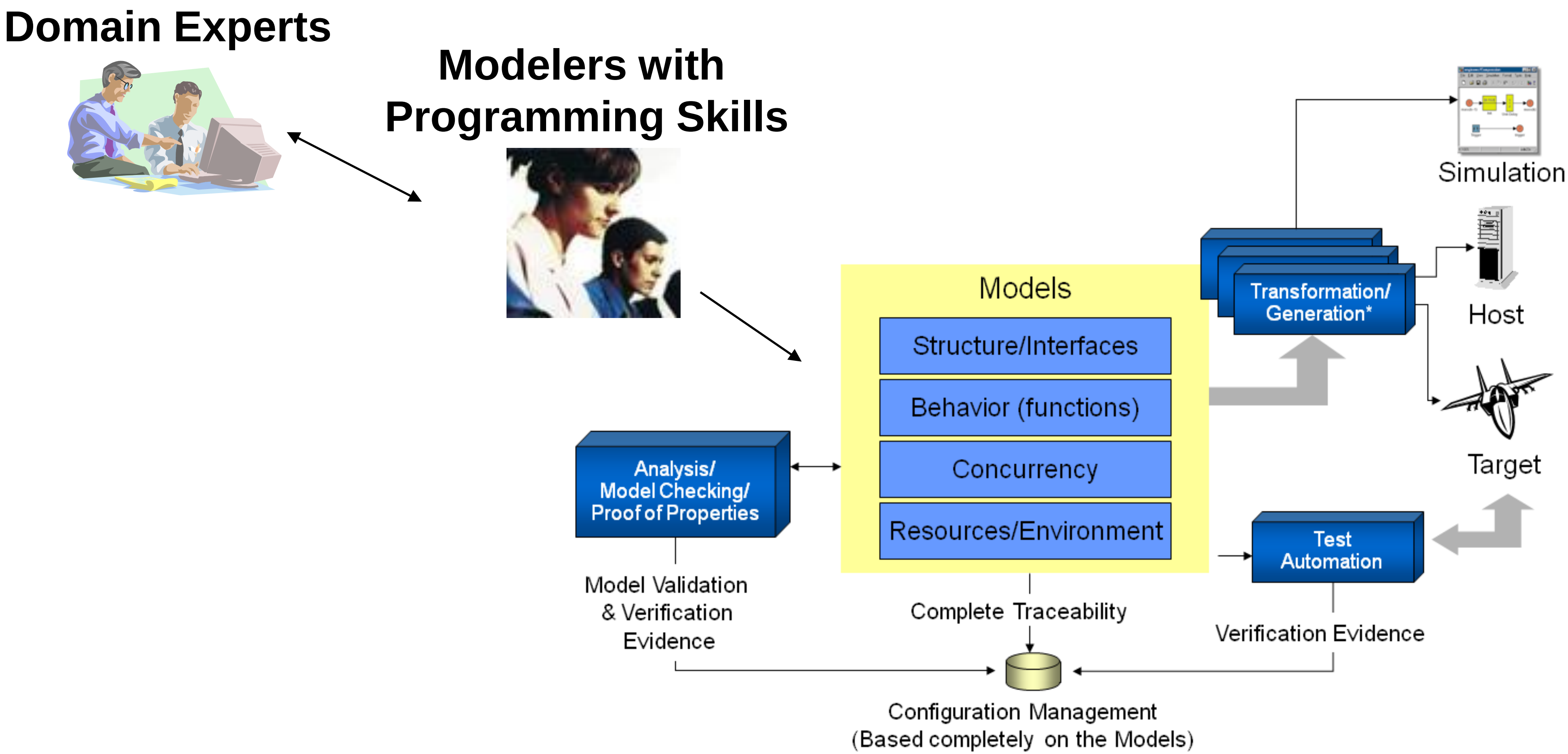
2, 3 – Structure modeling context, domain, actors, target system, interfaces to test, simulation, environmental models, and external systems



#1 & 2 - Need to incorporate modeling methods, structure, practices and standards

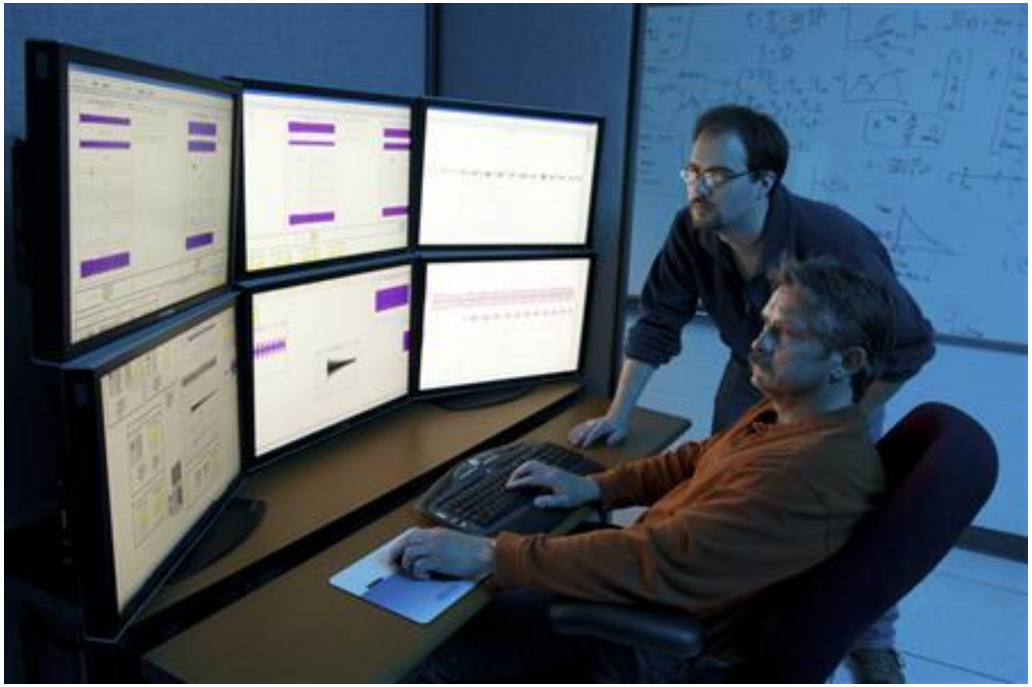
#1, 3, 4 & 5 – Address changes to lifecycle schedule and deliverables that can impact proposals, reviews and stakeholders

Skills – Don't "jump" into projects without knowing how to use MDE tools; have the right balance of modeling and domain expertise



Talk with customer about technology, process, and deliverable changes

Incomplete or inconsistent models are obvious and difficult to review



Model-based artifacts contribute to multiple phases of reviews and downstream needs (e.g., V&V)

Topics discussed today provide coverage over some emerging issues and gaps

Emerging Issues and Gaps	Concept Engineering & Graphical CONOPS	Modeling Architectures for I&T
Architecting for resilience	X	X
Capability mapping bi-directionally	X	X
Capability impact analysis for systems of systems	X	X
Tradeoff analysis	X	X
Continuous asynchronous integration and test	X	X
Transition into operations	X	
Transforming the systems engineering workforce	X	X
Involving disparate stakeholders	X	
Methods and standards		X
MDE adoption practices		X

There are many important ideas that we did not explore

Summary:

- ✔ Optimal architecting will be critical in engineering resilient systems that can rapidly adapt to user needs in uncertain futures
- ✔ MDE can better formalize the architecture to support adaptable, evolvable systems important in complex systems of systems
- ✔ MDE can provide early insights into V&V and better support impact analysis needed for continuous integration of capabilities
- ✔ Adoption practices and method guidance should be considered and refined in pilot projects to manage risk
- ✔ Our research in Graphical Concept Engineering can help address operational needs while formalizing capabilities that span the SoS and can be leveraged for virtual training addressing challenges of continuous operational changes

About Systems Engineering @ Stevens

- Largest Graduate Program in Systems Engineering in the United States
  - Broad engagement with Industry and Government
  - International Outreach
- Relevant and Flexible Curriculum Architecture
  - Developed and continually refined in collaboration with Industry and Government partners and sponsors
  - Individual Courses, Graduate Certificates and Degree Programs
  - Convenient Delivery Formats
- Experienced Faculty
- Leadership within the Systems Engineering community (US and Globally)

Abbreviations

AADL	Architecture Analysis & Design Language
AP233	Application Protocol 233
ATL	ATLAS Transformation Language
BPML	Business Process Modeling Language
CAD	Computer-Aided Design
CASE	Computer-Aided Software Engineering
CATIA	Computer Aided Three-dimensional Interactive Application
CDR	Critical Design Review
CMM	Capability Maturity Model
CMMI	Capability Maturity Model Integration
CWM	Common Warehouse Metamodel
DBMS	Database Management System
DoDAF	Depart of Defense Architectural Framework
DSL	Domain Specific Languages
HW	Hardware
IBM	International Business Machines
ICD	Interface Control Document
IEEE	Institute of Electrical and Electronics Engineers
INCOSE	International Council on Systems Engineering
IO	Input / Output
IPR	Integration Problem Report
ISO	International Organization for Standardization
IT	Information Technology
Linux	An operating system created by Linus Torvalds
MAP	Modeling Adoption Practices
MARTE	Modeling and Analysis of Real Time Embedded systems
MATRIXx	Product family for model-based control system design produced by National Instruments
MBT	Model Based Testing
MBSA	Model Based System Architecture
MBSE	Model Based System Engineering
MDA®	Model Driven Architecture®
MDD™	Model Driven Development
MDE	Model Driven Engineering
MDSD	Model Driven Software Development
MDSE	Model Driven Software Engineering
MIC	Model Integrated Computing
MMM	Modeling Maturity Model
MoDAF	United Kingdom Ministry of Defence Architectural Framework
MOF	Meta Object Facility
MVS	Multiple Virtual Storage
NASA	National Aeronautics and Space Administration
OCL	Object Constraint Language
OMG	Object Management Group
OO	Object oriented
PDR	Preliminary Design Review
PIM	Platform Independent Model
Pro/EPro/ENGINEER	
PSM	Platform Specific Model
RFP	Request for Proposal
ROI	Return On Investment
RTW	Mathworks Real Time Workshop
SSCI	Systems and Software Consortium
SCR	Software Cost Reduction
SDD	Software Design Document
SE	System Engineer
Simulink/Stateflow	Product family for model-based control system produced by The Mathworks
SOAP	A protocol for exchanging XML-based messages – originally stood for Simple Object Access Protocol
Software Factory	Term used by Microsoft
SQL	Structured Query Language
SRS	Software Requirement Specification
SW	Software
SysML	System Modeling Language
SystemC	IEEE Standard 1666
UML	Unified Modeling Language
XMI	XML Metadata Interchange
XML	eXtensible Markup Language
xUML	Executable UML
Unix	An operating system with trademark held by Open Group
VHDLVerilog	Hardware Description Language
VGS	T-VEC Vector Generation System
VxWorks	Operating system owned by WindRiver

Trademarks

- **OMG®, MDA®, UML®, MOF®, XMI®, SysML™, BPML™ are registered trademarks or trademarks of the Object Management Group.**
- **IBM™ is a trademark of the IBM Corporation**
- **Java™ and J2EE™ are trademark of SUN Microsystems**
- **XML™ is a trademark of W3C**
- **BridgePoint is a registered trademark of Mentor Graphics.**
- **Java is trademarked by Sun Microsystems, Inc.**
- **Linux is a registered trademark of The Linux Mark Institute.**
- **MagicDraw is a trademark of No Magic, Inc.**
- **MATRIXx is a registered trademark of National Instruments.**
- **MVS is a trademark of IBM.**
- **Real-time Studio Professional is a registered trademark of ARTiSAN Software Tools, Inc.**
- **Rhapsody is a registered trademark of Telelogic/IBM.**
- **Rose XDE is a registered trademark of IBM.**
- **SCADE is copyrighted to Esterel Technologies.**
- **Simulink is a registered trademark of The MathWorks.**
- **Stateflow is a registered trademark of The MathWorks.**
- **Statemate is a registered trademark of Telelogic/IBM.**
- **TAU/Developer is registered to Telelogic/IBM.**
- **T-VEC is a registered trademark of T-VEC Technologies, Inc.**
- **UNIX is a registered trademark of The Open Group.**
- **VAPS is registered at eNGENUITY Technologies.**
- **VxWorks is a registered trademark of Wind River Systems, Inc.**
- **VectorCAST is a trademark of Vector Software.**
- **Windows is a registered trademark of Microsoft Corporation in the United States and other countries.**
- **All other trademarks belong to their respective organizations.**