USE OF MODEL BASED SYSTEMS ENGINEERING (MBSE) TO IMPROVE PROGRAM METRICS

DR. HOWARD GANS

Systems Engineer

Contact information:

hgans@harris.com

(321) 727-4379

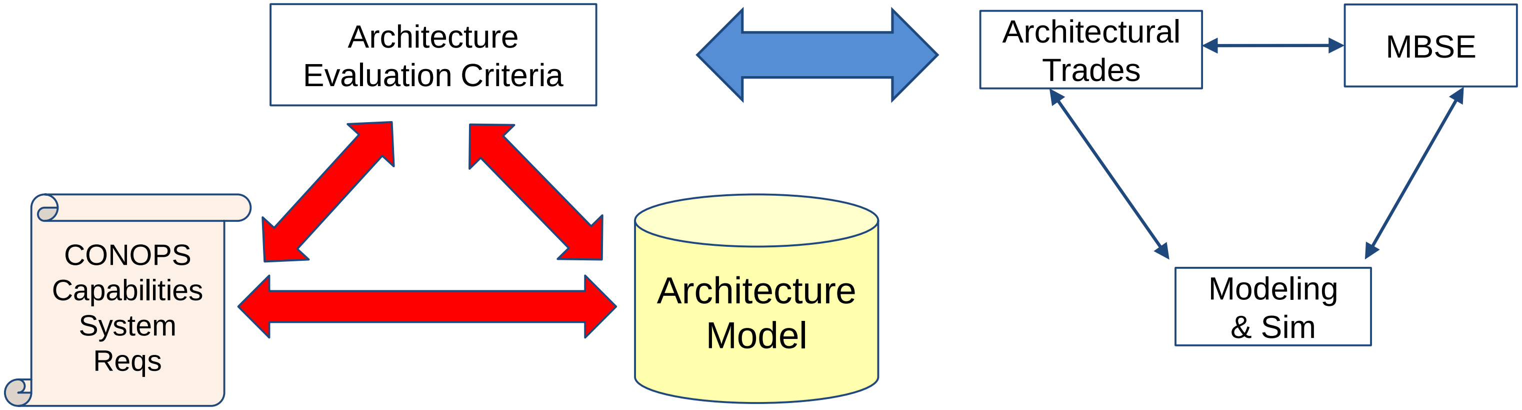
HARRIS® TECHNOLOGY TO CONNECT, INFORM AND PROTECT™

Overview

- **Model-Based Systems Engineering (MBSE) in the context of Systems Engineering**
- **Identification of four major issues**
  - Requirements gaps
  - Development costs
  - Early detection and resolution of defects
  - Requirements volatility
- **MBSE application to a requirements development problem**
- **Completion of the requirements allocation process**
- **Summary of MBSE effects on metrics**

Problem statement: Use of MBSE to improve program metrics

- **Business cases require defining the tangible benefits of MBSE**
- **Look at this issue in the context of solving key problems affecting system development**
  - Have all requirements needed to define system functionality been identified?
    - “Requirements gaps”
  - How can the cost of instantiating a system be controlled?
    - “Development costs”
  - How can defects in the design be found and solved early in the program life cycle?
    - “Early detection and resolution of defects”
  - How can requirements changes late in the program be avoided?
    - “Requirements volatility”



MBSE – Management of Complexity

Why MBSE?

- **Modern day systems are increasingly complex**
- **Enables concurrent engineering and traceability of design artifacts**
- **Enables better communication among stakeholders**
- **Government and Primes are requiring MBSE implementation in programs**
- **Becoming an industry standard development approach**
- **Reduces design defects; especially system engineering induced defects**
- **Universities, and even high schools are teaching MBSE ... new grads are ready for it**

"MBSE is not a new way of doing Systems Engineering; it is a set of tools to do it better."
Philomena Zimmerman, Deputy Director for Engineering Tools and Environments, ODASD/SE

MBSE is considered a required skillset at higher levels of industry competition
Brute Force Systems Engineering is no longer an option

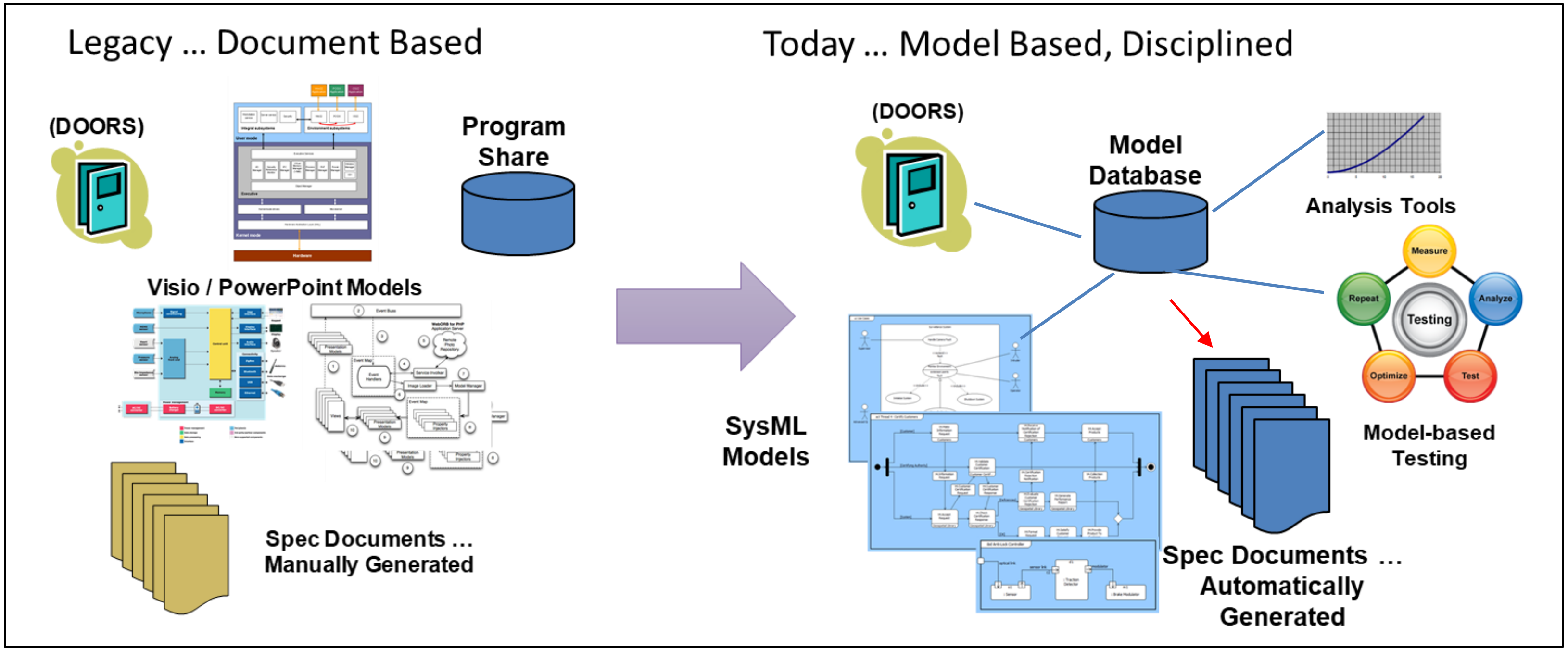
Systems Engineering and MBSE

INCOSE Systems Engineering Handbook, 2015[1]:

- **"Systems engineering is an interdisciplinary approach and means to enable the realization of a successful system."**
- **Iterative process of top-down synthesis and development**
- **Looks at a problem in its entirety with an iterative methodology**

MBSE enhances ability to capture, analyze, share and manage information

Metrics will show the benefit of using MBSE



Issue 1: Requirements Gaps

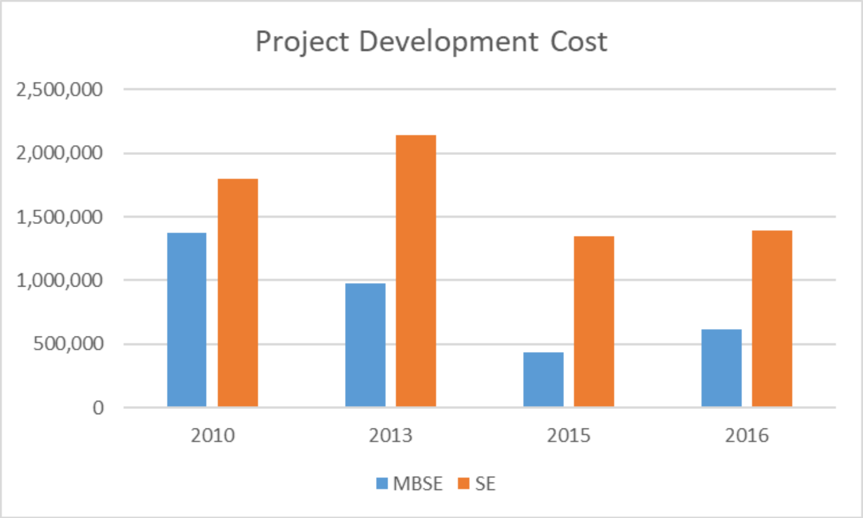
- **Issue: Necessary system functionality is not mapped to a requirement**
- **Well constructed MBSE model identifies gaps**
  - See Mike Russell, "Using MBSE to Enhance System Design Decision Making[2]."
- **If left undiscovered, gaps can lead to missed functionality or interfaces**
  - Failure during V&V
  - Escapes found by the customer
- **Requirements volatility (RV) is not affected until the gap is corrected (lagging indicator)**
- **Question: how does MBSE close the requirements gap?**

Gap	Description	Issues
Incomplete	Specification does not completely define needed requirements	Ill defined project scope
		Lack of traceability to design
		Result: orphan requirements at the lower level
Inconsistent	Requirements in conflict	Implementation uncertainty
		Undetected requirements errors (e.g. cut and paste from existing specs)
		May not satisfy customer expectations
Misunderstood	Poorly written requirements	Lack of mission definition
		SE's do not understand the problem

Derived from Ivy Hooks, "Why Johnny Can't Write Requirements"[3]

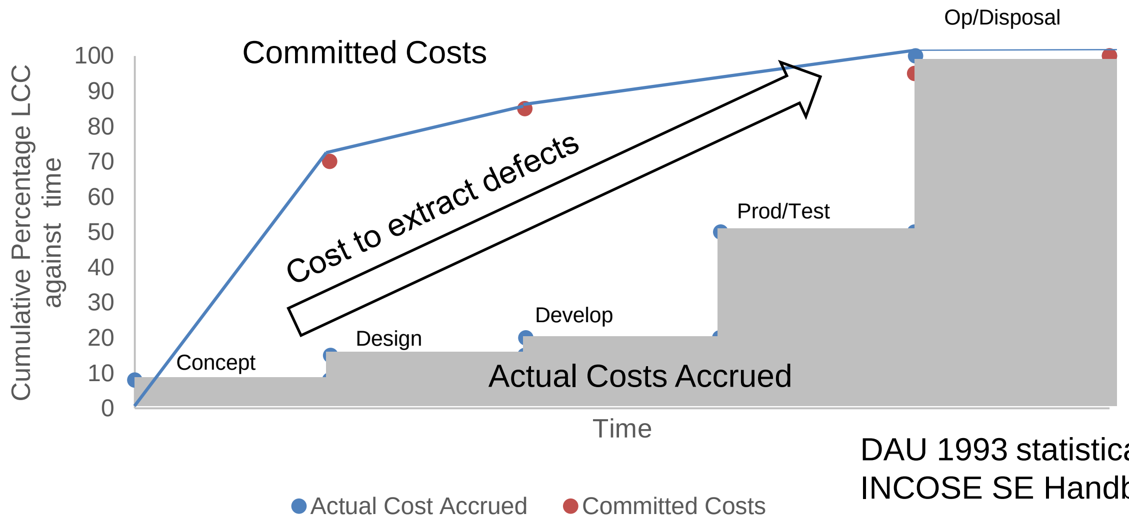
Issue 2: MBSE and Development Cost

- **Previous studies have shown the correlation between system development cost and application of MBSE**
  - "Return on Investment for Complex Projects using Model Based Systems Engineering," Michael Gooden, George Washington University[4]
- **Savings result from the following benefits**
  - Improved communication among the development stakeholders
  - Increased ability to manage system complexity using a system model viewed from multiple perspectives
  - Improved product definition from an enhanced system definition
  - Model facilitates SME knowledge capture and reuse



MBSE Cost Savings vs Traditional Systems Engineer as documented in Michael Gooden study

Issue 3: Cost to Extract Defects



DAU 1993 statistical study, from INCOSE SE Handbook

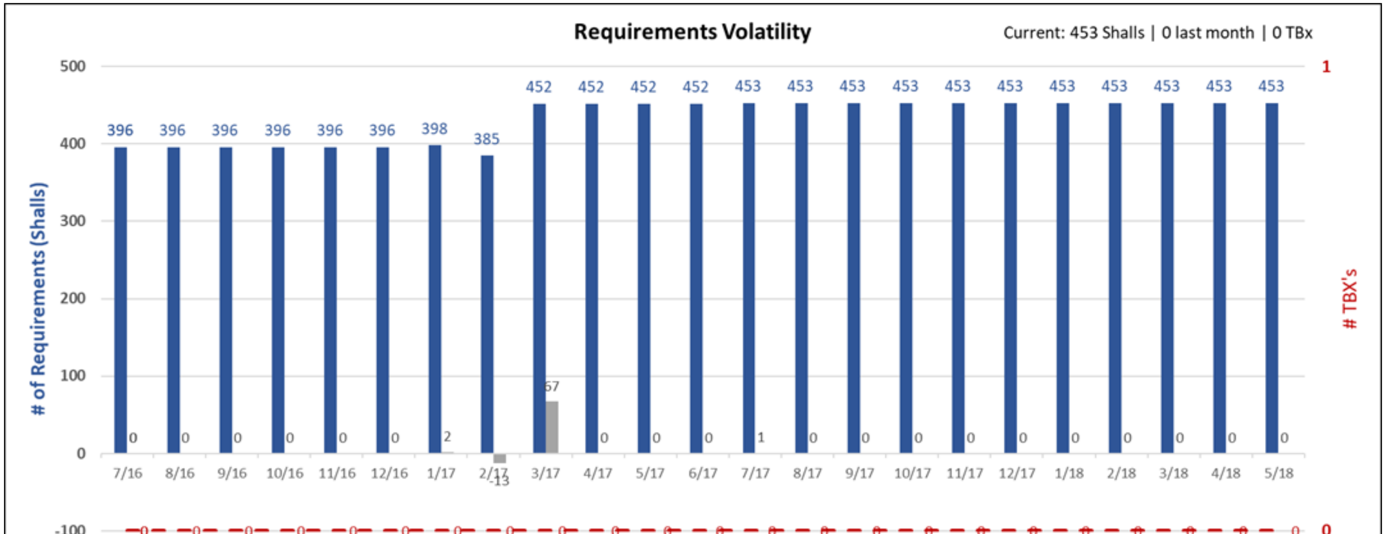
- **From INCOSE SE Handbook, 2015, reducing risk associated with new systems or modification of existing systems is the primary goal of systems engineering.**
  - DAU 1993 statistical study showed life cycle cost (LCC) increases over time with early cost commitment
  - Example, by Development phase, 80% of costs have been committed while 20% of costs have been accrued
- **The cost of finding and fixing a defect increases while the funding available to correct decreases.**
- **Steve Sanders (2011 Raytheon study)[5] proposed that the ability of MBSE to manage complexity can help detect and correct defects early**

Issue 4: Requirements Volatility (RV)

- **Defined by requirements added, deleted and modified (Elwahab, *et al*)**
- **Excess RV results in additional development and costs**
- **Causes**
  - Poor stakeholder communication
  - Insufficient analysis of requirements
  - Incomplete understanding of user CONOPS

Result: Requirements Gaps

- **Impact: May increase cost by over 20% and lengthen schedule**



	7/16	7/16	8/16	9/16	10/16	11/16	12/16	1/17	2/17	3/17	4/17	5/17	6/17	7/17	8/17	9/17	10/17	11/17	12/17	1/18	2/18	3/18	4/18	5/18
Shall Count	396	396	396	396	396	396	396	398	385	452	452	452	452	453	453	453	453	453	453	453	453	453	453	453
Delta Shalls	0	0	0	0	0	0	0	2	-13	67	0	0	0	1	0	0	0	0	0	0	0	0	0	0
TBx	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Shall Count | Delta Shalls | TBx

Methods to Manage RV Using MBSE

General Method	Implementation Approach
Reject RV	Freeze requirements
Eliminate Avoidable Causes	Ensure all functions are modeled
	Use MBSE to define requirements (structured requirements modeling)
	Have a stable MBSE team
RV "Coping"	Interative requirements-architecture development
Manage Effects of RV	Use change management for requirements and model
	Use the model to resolve conflicts between teams
Essential Elements	Management commitment to MBSE
	MBSE developers have access to SMEs
	Leads have authority for model governance
	Link to V&V in the model

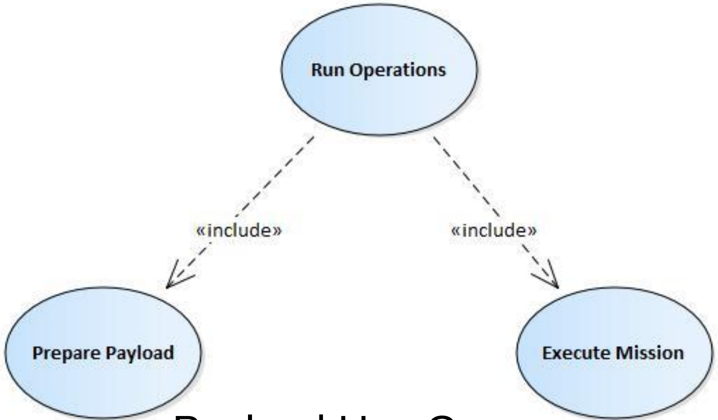
- See Sachidanandam Sakthivel, “Manage Requirements Volatility to Manage Risks in an IS Development Project[6]”
- Focus is on MBSE applications

Application of MBSE and Metrics

- **Start with a simple set of Use Cases (UC’s) and requirements for payload operations**
- **Include a first draft of a state diagram for the payload**
- **Assumption: the payload can zeroize and write to NVM only in maintenance state**

ID	Subsystem 2
PAY-1	**1 Payload**
PAY-3	The Payload **shall** power up on command.
PAY-4	The Payload **shall** execute commands as received from the Bus.
PAY-5	The Payload **shall** execute autonomously housekeeping functions.
PAY-6	The Payload **shall** report H&S.
PAY-7	The Payload **shall** transfer mission data to the bus.

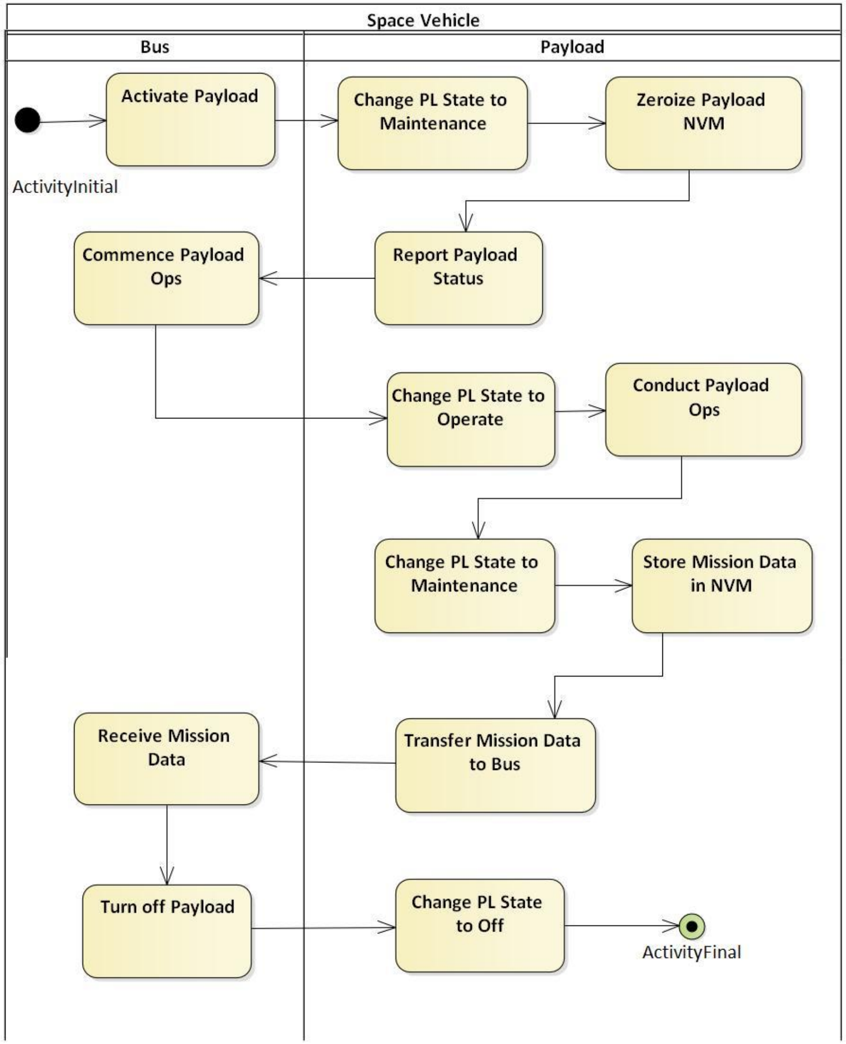
Initial Requirements in DOORS



Payload Use Cases

Initial Payload State Diagram

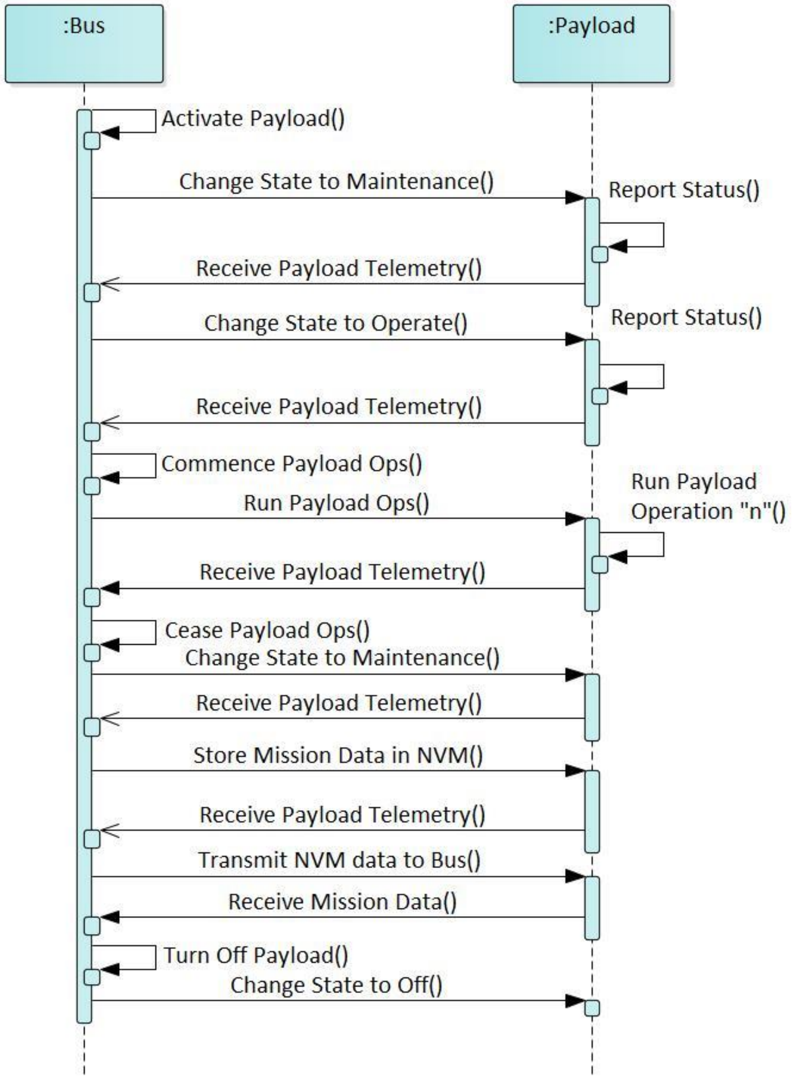
First Draft of Activity Diagram (SV-4)



- **Issues**
  - Payload IPT says "We need to protect the Payload. We are changing the state diagram so that the PL can go to 'off' from any state'.
    - Defect detected and corrected
  - Customer: "The Payload must send acknowledgements to the Bus after every successful (or unsuccessful) command
    - Possible RV issue detected early
    - Would result in major design change if found at later program phase.
  - Design Review Team: "The PL is storing data in NVM. There is no requirement for that."
    - Discovered and corrected requirements gap

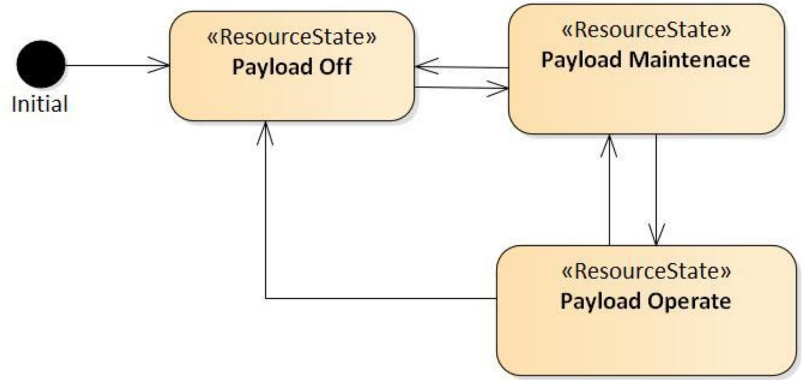
Updated Architecture and Requirements

Updated Sequence Diagram (SV-10c)



ID	Subsystem 2
PAY-1	**1 Payload**
PAY-3	The Payload **shall** power up on command.
PAY-4	The Payload **shall** execute commands as received from the Bus.
PAY-5	The Payload **shall** execute autonomously housekeeping functions.
PAY-6	The Payload **shall** report H&S.
PAY-8	The Payload **shall** store mission data in nonvolatile memory
PAY-7	The Payload **shall** transfer mission data to the bus.

Version 2 of Requirements in DOORS



Updated State Diagram (SV-10b)

Requirements Allocation To Architecture

ID	Subsystem 2
PAY-1	**1 Payload**
PAY-3	The Payload **shall** power up on command.
PAY-4	The Payload **shall** execute commands as received from the Bus.
PAY-5	The Payload **shall** execute autonomously housekeeping functions.
PAY-6	The Payload **shall** continuously report H&S.
PAY-8	The Payload **shall** store mission data in nonvolatile memory.
PAY-7	The Payload **shall** transfer mission data to the bus.
PAY-9	The Payload **shall** change states when commanded by the Bus.
PAY-10	The Payload **shall** be able to turn off from any state

Requirements Version 3

	Change PL State to Maintenance	Change PL State to Off	Change PL State to Operate	Powerup Payload	Report Payload Status	Run Payload Operations	Store Mission Data in NVM	Turn Off Payload	Zeroize Payload NVM
PAY-1				X					
PAY-10		X							
PAY-3								X	
PAY-4						X			
PAY-5									X
PAY-6					X				
PAY-7							X		
PAY-8							X		
PAY-9	X	X	X						

Requirements-Architecture Cross Reference

- **Requirements changes shown in red and were extracted from DOORS**
- **Requirements were brought into the MBSE model and linked to the system functionality as determined by the analysis**
- **Result: System functions are covered by requirements without gaps**

Summary of MBSE Effects on Metrics

- **Metrics issue 1: Requirements Gaps**
  - ➢ Needed requirements were added to close the requirements gaps (Issue 1)
- **Metrics issue 2: Reduction in Development Costs**
  - ➢ Customer and IPT interaction resulted in more stable payload design with improved communication to the Bus
- **Metrics issue 3: Early Defect Detection and Resolution**
  - ➢ Defects in Payload design were detected and fixed early
- **Metrics issue 4: Requirements volatility.**
  - ➢ MBSE model improved communication with stakeholders
  - ➢ Requirements analysis improved by MBSE
- **Next steps**
  - ➢ Monitor program costs over program lifetime to see if MBSE is a positive controlling activity
  - ➢ Continue to check for defects in the model to avoid downstream integration issues and possible escapes
  - ➢ Track RV across programs to see if MBSE is having a positive systemic effect
  - ➢ Expand this analysis to include control of interfaces (external and internal) by MBSE

References

1. “Systems Engineering Handbook, 4th edition,” INCOSE-TP-2003-03-04, Wiley, 2015.

2. Russell, M., “Using MBSE to Enhance System Design Decision Making,” New Challenges in Systems Engineering and Architecting Conference on Systems Engineering Research (CSER), 2012 – St. Louis, MO, retrieved from: https://www.sciencedirect.com/science/article/pii/S1877050912000427

3. Hooks, Ivy, “Why Johnny Can’t Write Requirements,” retrieved from https://www.reqexperts.com/wp-content/uploads/2015/07/why_johnny_cant_write_good_requirements.pdf

4. Gooden, M., “Return on Investment for Complex Projects Utilizing MBSE,” retrieved from: https://ndiastorage.blob.core.usgovcloudapi.net/ndia/2016/systems/18854_MichaelGooden.pdf

5. Sanders, S., “Does a Model Based Systems Engineering Approach Provide Real Program Savings? – Lesson Learnt,” 25 Oct 2011, retrieved from: http://www.omgsysml.org/Does_a_MBSE_Approach_Provide_Savings-Lessons_Learnt-Saunders-200111.pdf

6. Sakthivel, S., “Manage Requirements Volatility to Mange Risks in IS Development Projects,” ISACA Journal, Volume 5, 2010, retrieved from: https://www.isaca.org/Journal/archives/2010/Volume-5/Documents/jpdf1005-manage-require.pdf