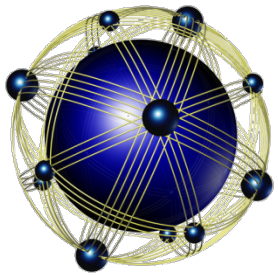
NDIA 10th Annual CMMI Technology Conference

"How MBE can support Requirements Development and Technical Solution."

November 18, 2010

Frank Salvatore
High Performance Technologies, inc.
3159 Schrader Road
Dover NJ, 07801
(973) 442-6436 ext 249
fsalvatore@hpti.com



HPTi

Outline

- Current State of Practice
- Improving the State of Practice
- Vision, Definitions
- How MBSE can support RD and TS
- Planning Consideration
- Conclusion

Current State of Practice

- Requirements are captured in database applications
  - They are often not accurate or complete
  - They take a long time to develop
  - They are published and viewed as documents
  - It is difficult to achieve IPT consensus
- Technical Solution is captured in various forms
  - CAD/CAE, PowerPoint Slides, Simulations, and assembled in paper based documents.
- Often traceability is weak and hard to maintain.
- Our current engineering practices lack the rigor and discipline necessary to be explicit

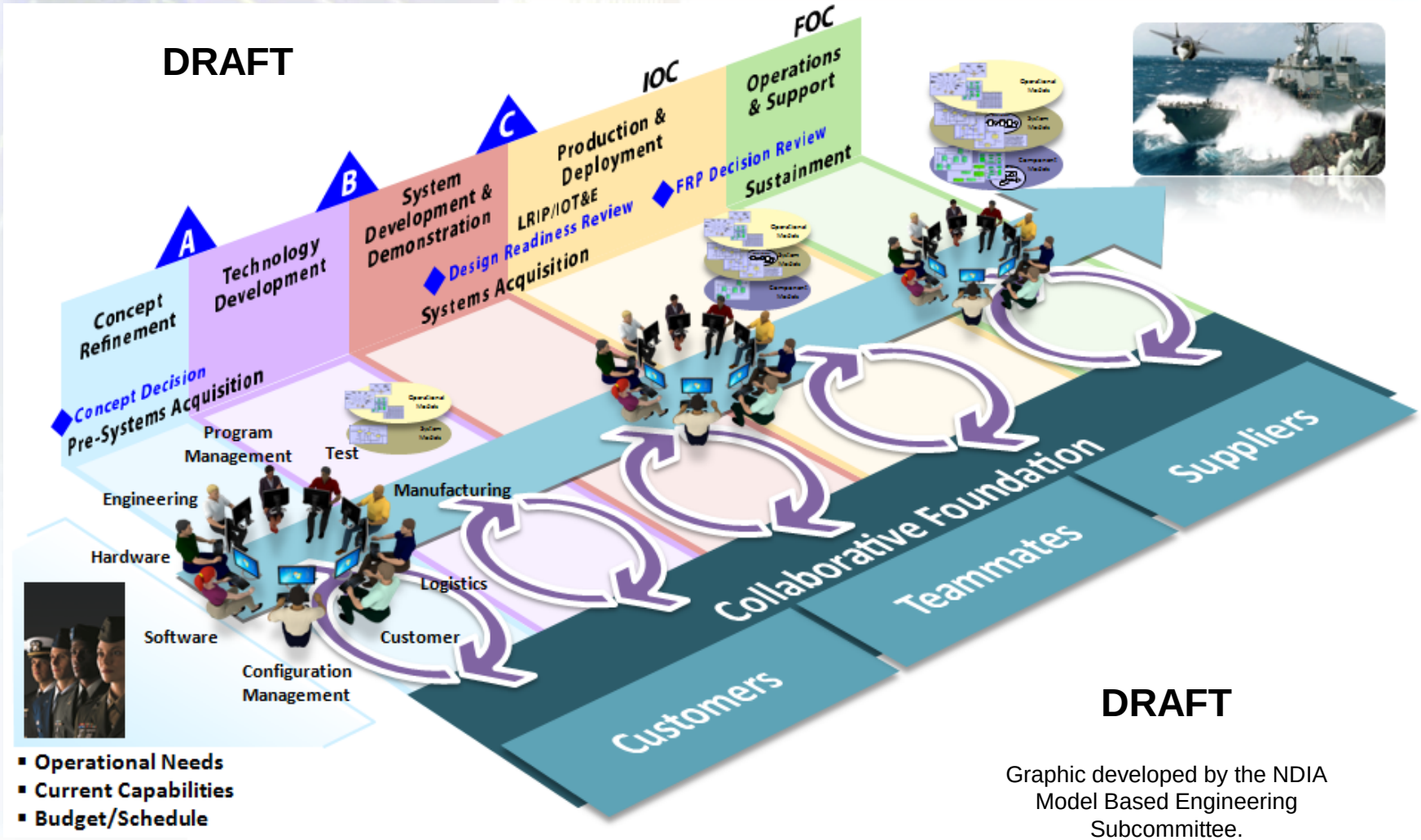
Is There a Better Way?

- What if the Requirements and Technical Solutions were all captured in one location and everything was traceable?
- What if we could validate our requirements at the same time we are developing our technical solution?
- What if the IPT could reach agreement to what the requirements are and what they mean.

- Can a Model Based Engineering Approach Help?

"You cannot engineer something if you cannot see it."
Scott Workinger

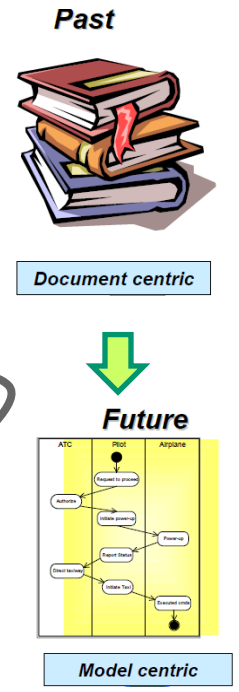
Vision for Model Based Engineering (MBE)



Graphic developed by the NDIA Model Based Engineering Subcommittee.

MBSE - General Definition

- It is about System Modeling
  - System Model is a ***cohesive, unambiguous representation*** of what the System ***is*** and ***does.***
- It provides a description of
  - Requirements and
  - Technical Solution and
    - Operational Scenarios
    - System Behavior (including I/O)
    - Physical Architecture (Structure, interfaces)
    - Dynamic Simulation (requires “executable” models)
  - Verification Procedures
- MBSE is used to produce SE products
- It requires a Modeling Language (SysML) that is computer interpretable



Minimum Required to Define System

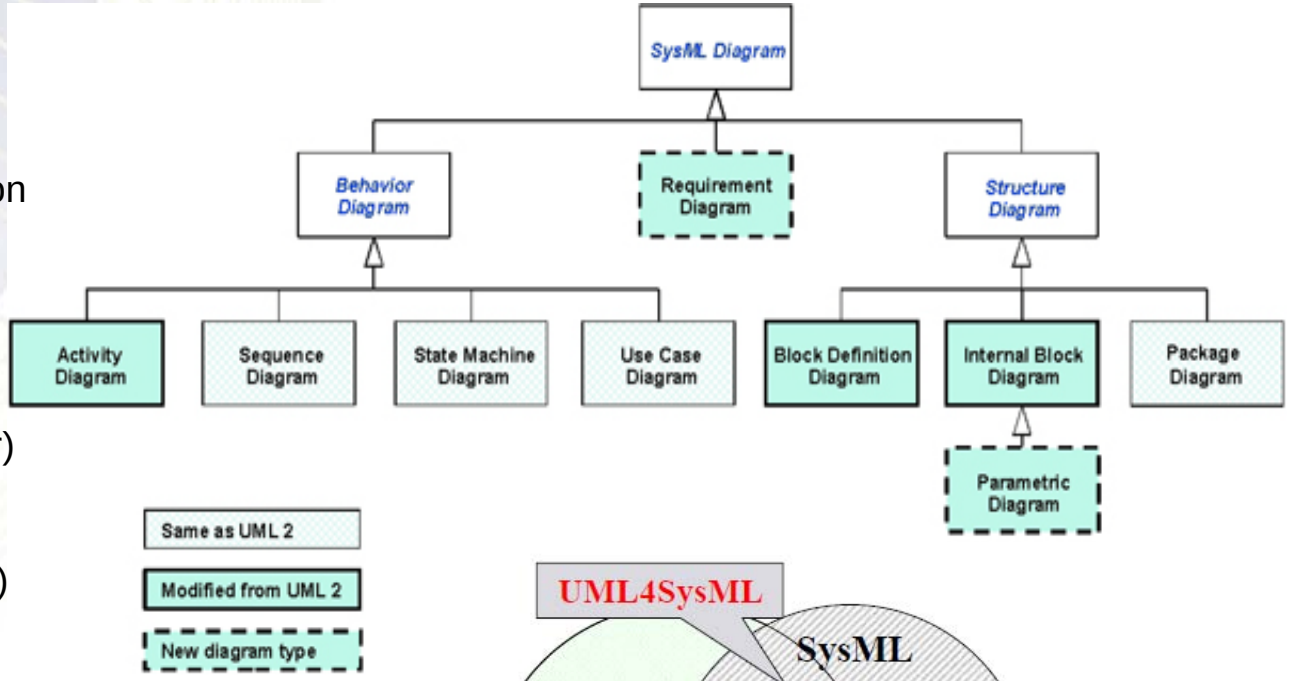
SysML Overview

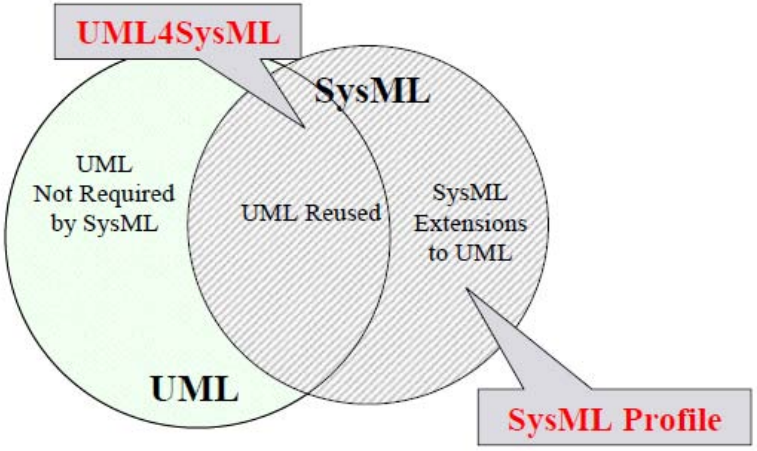
Descriptive Modeling

- General Purpose Graphical Modeling
  - Structure
  - Behavior
  - Requirements
  - Parametric
- Supports: specification, analysis, design, verification and validation
- Supports model and data interchange via XMI and the evolving AP233 standard (in-process)

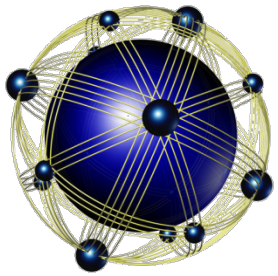
SysML is Derived from UML

- **Name Change**
  - Class – Block Definition
  - Composite – Internal Block
- **Removed Diagrams**
  - Deployment (Behavior)
  - Object (Behavior)
  - Component (Behavior)
  - Interaction(Structure)
  - Communication (Structure)
  - Timing (Structure)

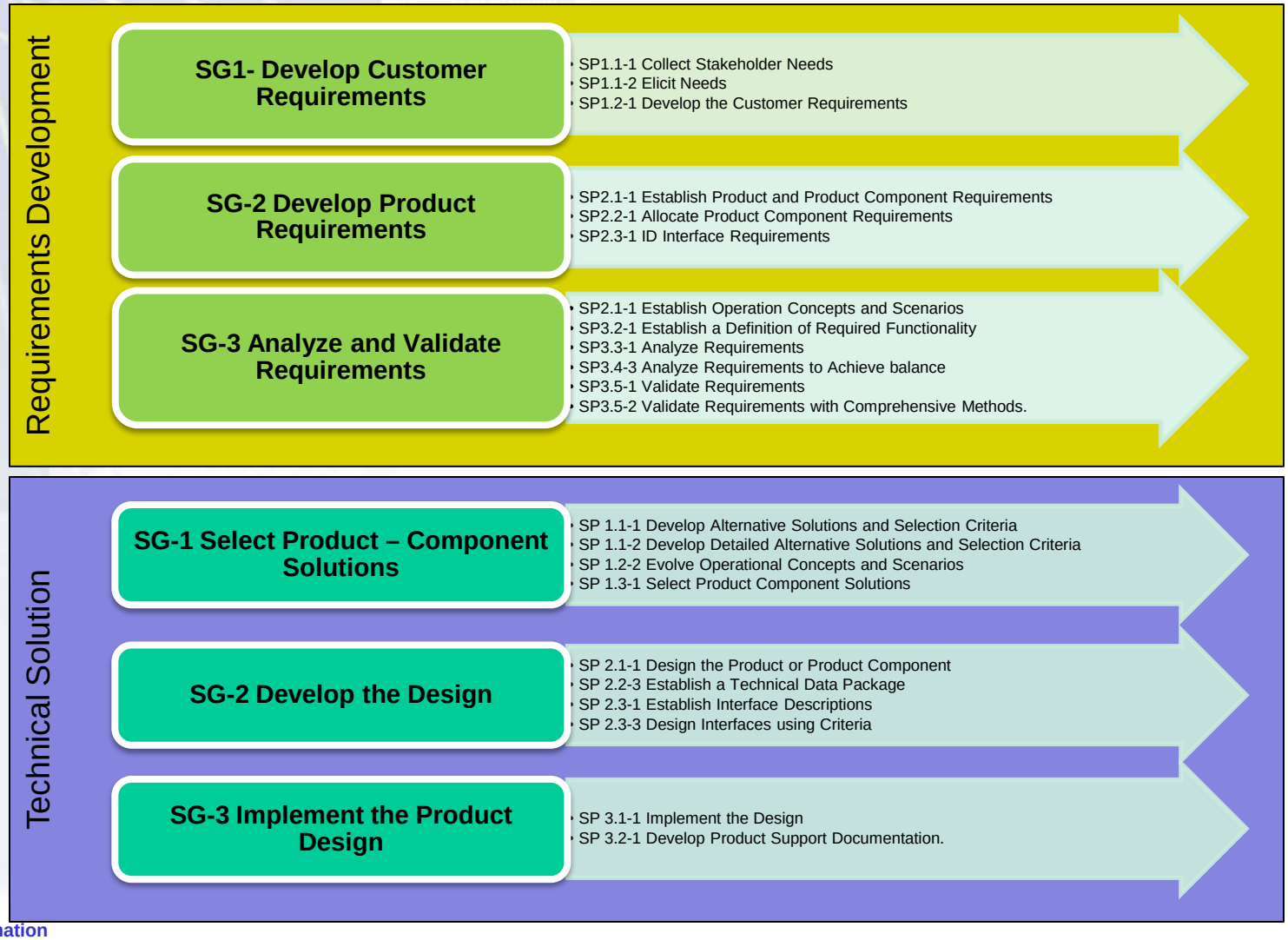




How can MBSE support RD and TS



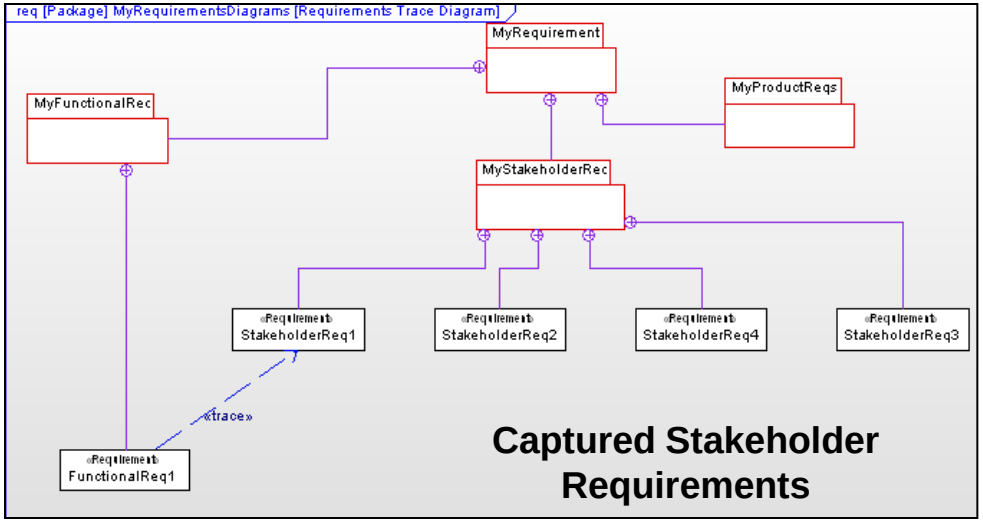
CMMI V1.1 Continuous Representation



Requirements Development:
SG-1 Develop Customer Requirements

- Collect Stakeholder needs (SP 1.1-1) using the Requirements model element
- Or capture them in a requirements database and synch them with the model
- SysML can be used to capture and manage relationships between requirements
- Auto Generate Requirements Trace Matrices and Specifications

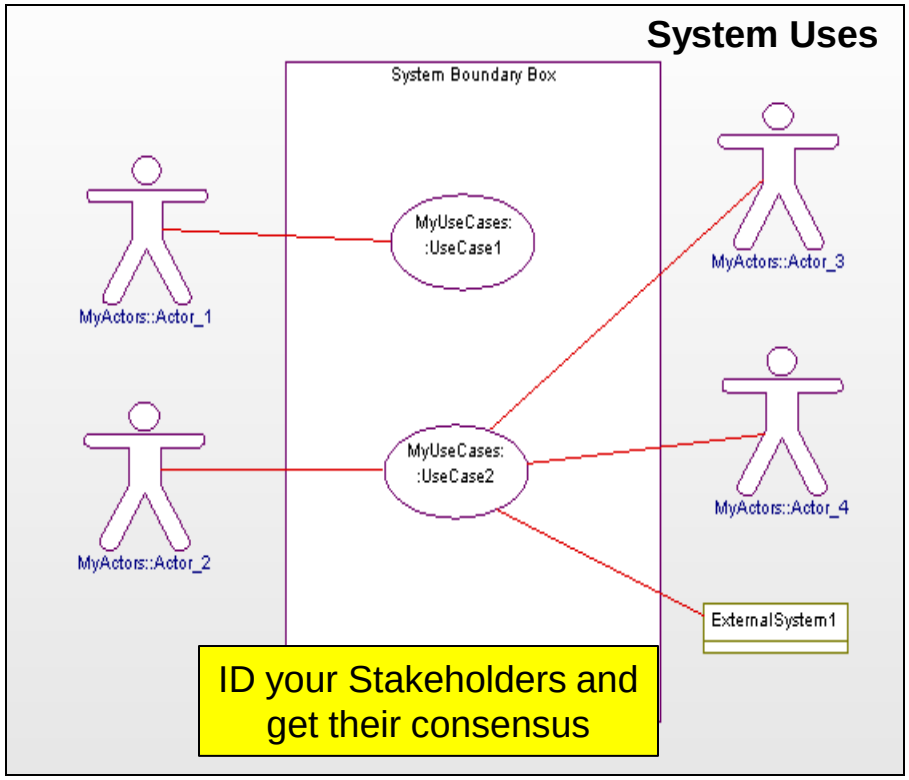
ID	Name	Specification
SR0001	StakeholderReq1	Req Text for SR1
SR0002	StakeholderReq2	Req Text for SR2
SR0003	StakeholderReq3	Req Text for SR3



Captured Stakeholder Requirements

Requirements Development:
SG-1 Develop Customer Requirements

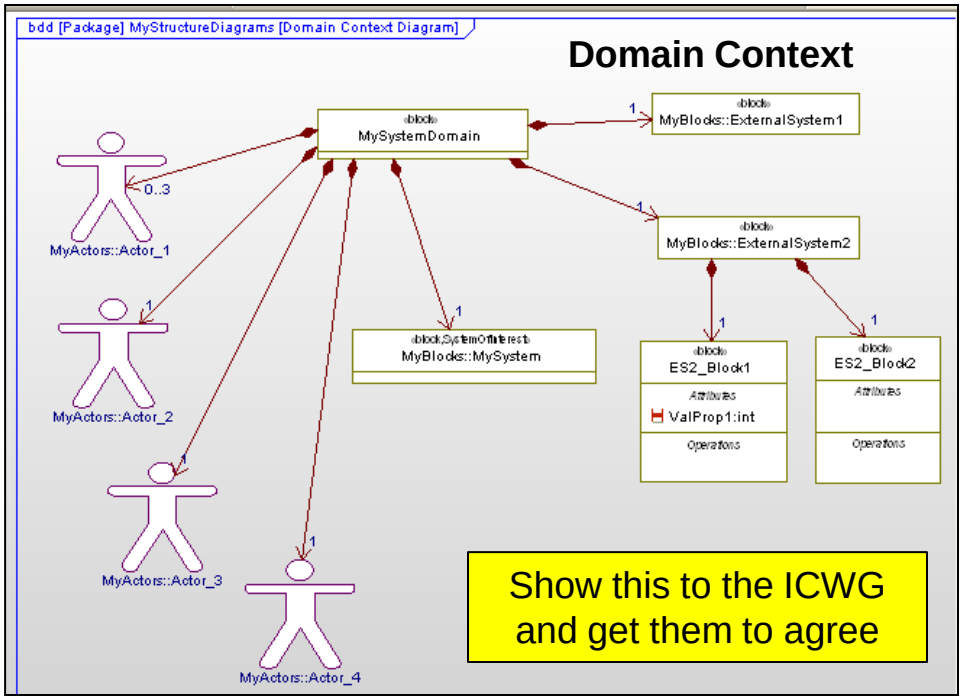
- Use “Use Case Analysis” to capture capabilities and elicit requirements from stakeholders (SP1.1-1, SP 1.1-2)
- Trace Requirements to Use Cases
- Seeing requirements in a diagram tends to draw out requirements



Diagramming and visually presenting what has been captured (SP 1.1-1)

Requirements Development:
SG-1 Develop Customer Requirements

- Use Block Definition Diagrams (BDD) to capture where your system of interest fits, defining the Physical System Boundary (TS-SP 1.1-2, 1.2-2)
- Perform Domain Analysis to understand context and further drive out a clearer understanding of the problem that needs to be solved.
- Drive out external interfaces

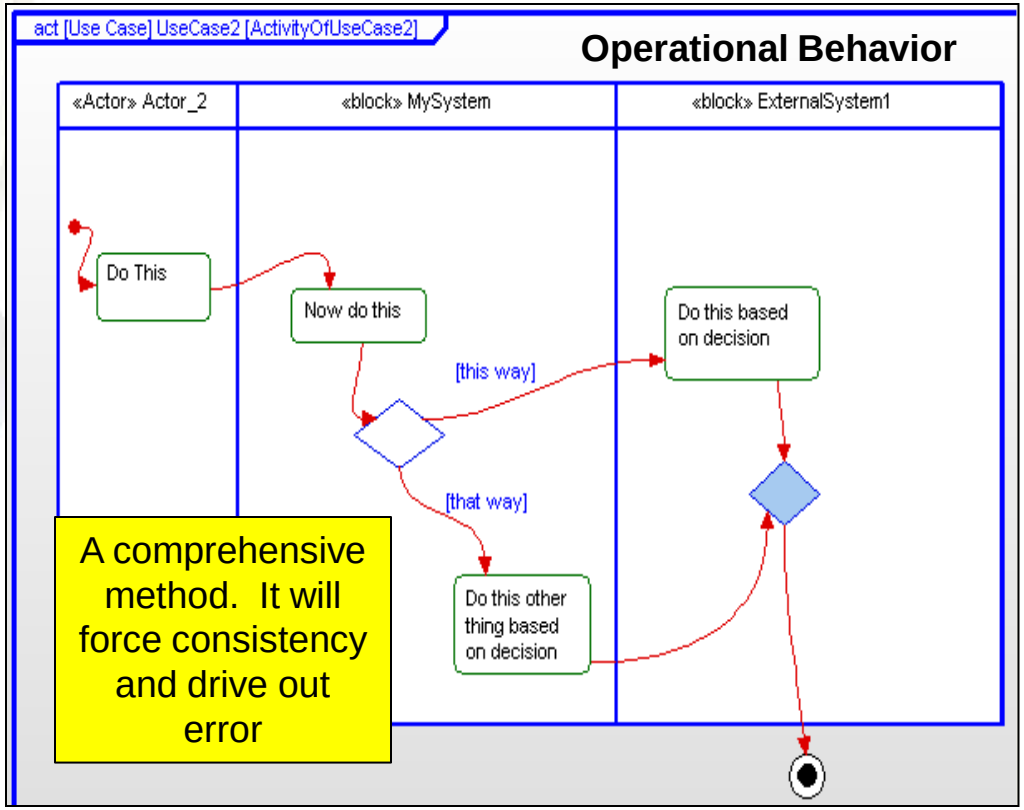


Diagramming and visually presenting what has been captured (SP 1.1-1)

Update use cases and Requirements as you learn more

Requirements Development:
SG-3 Analyze and Validate Requirements

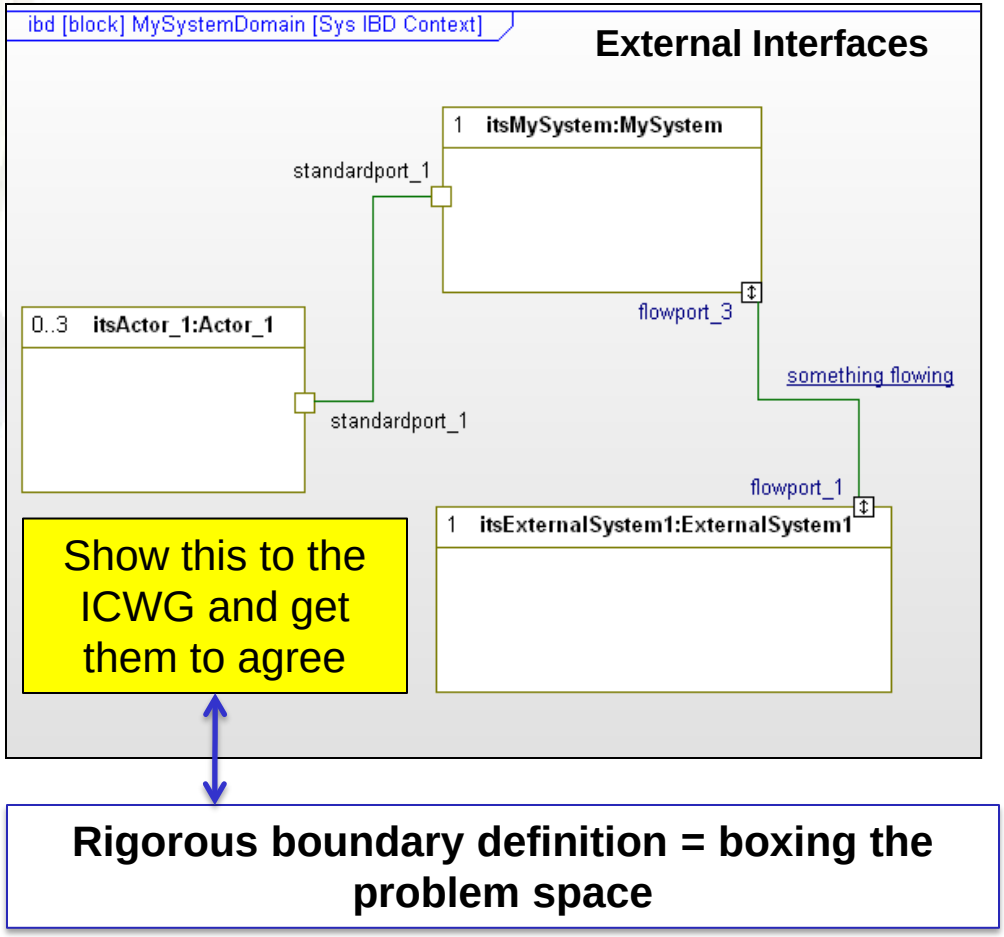
- Model Domain Activities with Activity Diagram (ACT) to Elicit needs and to capture behavior (SP 1.1-2)
- It starts to capture what the operational concepts and scenarios are (SP 3.1-1)
- Provides a precise definition of required functionality (SP3.2-1)
- It serves the purpose of Analyzing and validating the requirements (SP3.3-1, SP3.5-1, SP3.5-2)
- Establishes Functional System Boundary



Update everything as you learn more.

Requirements Development:
SG-3 Analyze and Validate Requirements

- Use an Internal Block Diagram (IBD) to elicit requirements and identify interface requirements (SP 1.1-2, 2.3-1).
- Rigorous and explicit capture and documentation of external interfaces.
- Serves the purpose of Analyzing and validating the interface requirements (SG 3).
- Merge of physical and functional system boundary definition

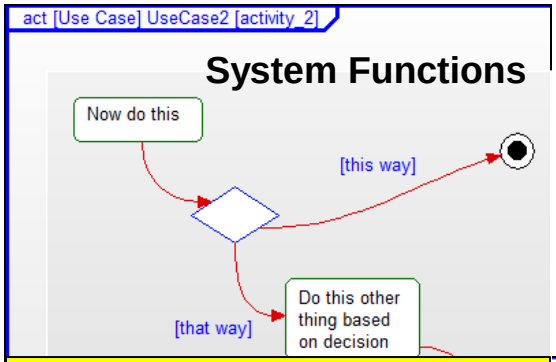


Update everything as you learn more.

Requirements Development:
SG-2 Develop Product Requirements

- Identify behaviors and derive functional requirements of your system and system components (SP 3.2-1)
- Use IBD to capture the arrangement and association between a selected system solution alternative.
- Identify Interface Requirements (SP 2.3-1)

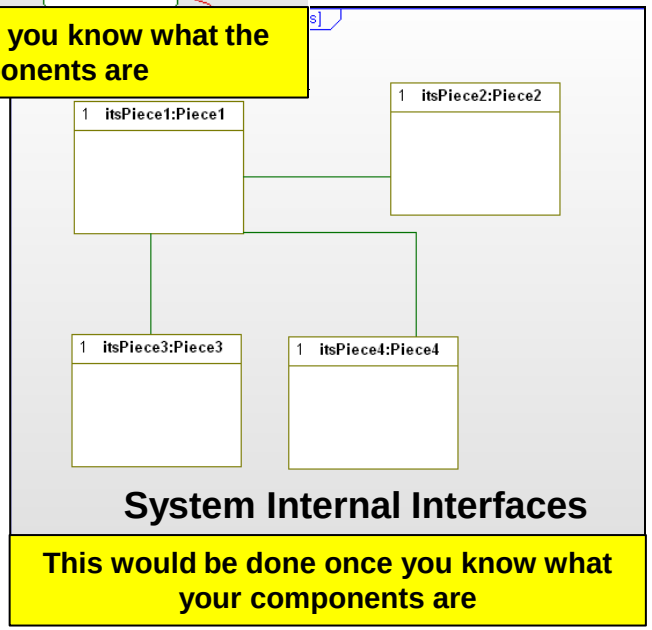
Input from Technical Solution (SP1.1-2, SP2.1-1)



Identify System Functions

Do this before you know what the components are

Identify Component Interactions



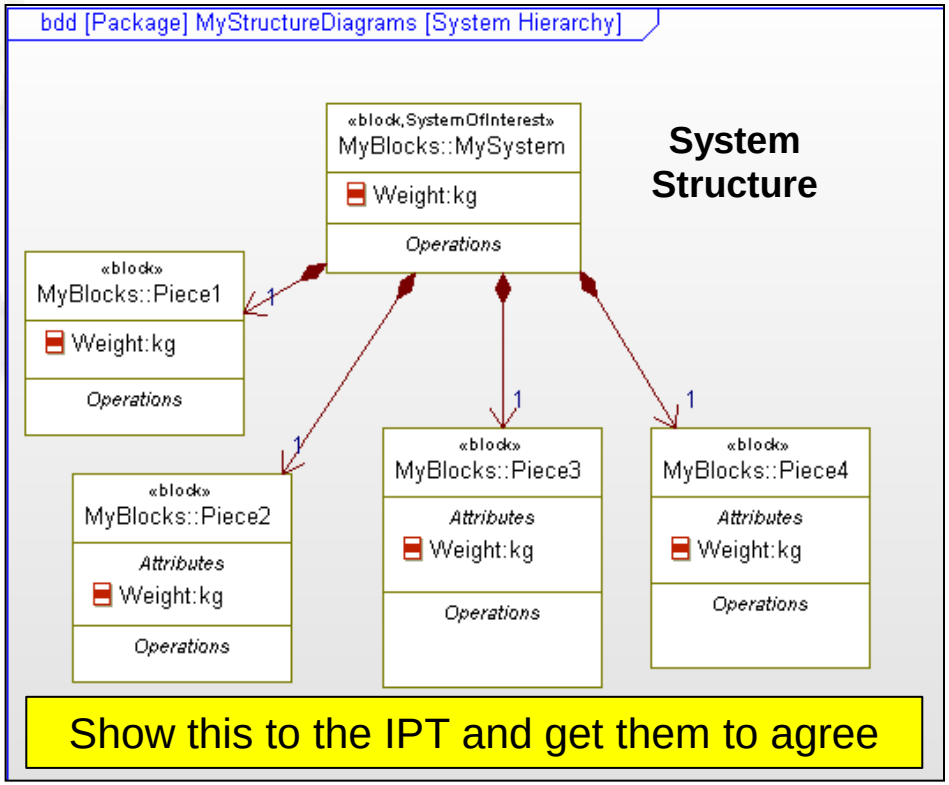
This would be done once you know what your components are

Update everything as you learn more.

Requirements Development:
SG-2 Develop Product Requirements

- Capture the Systems Physical Structure or Hierarchy with a Block Definition Diagram (BDD).
- If you know what these are begin Developing Product Requirements (SP2.1-1).
- Else perform Technical Solution Practices.
- Identify associations and quantities.
- Allocate requirements to each system component/block (SP2.2-1)

Input from Technical Solution (SP1.1-1,SP 2.1-1)



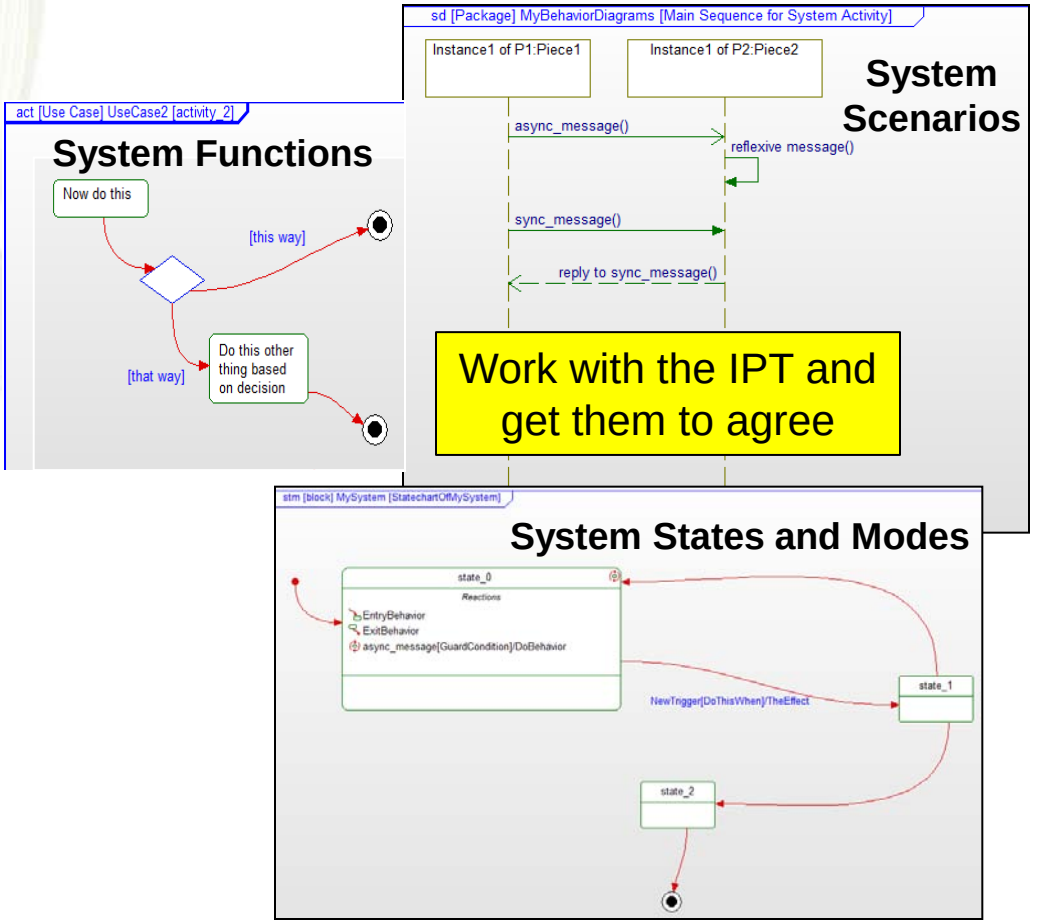
Identify System components

Update everything as you learn more.

Requirements Development:
SG-2 Develop Product Requirements

- Use Sequence Diagrams, Activity Diagrams and State Machine Diagrams to Identify behaviors and derive functional requirements of your system and system components (SP 3.2-1)
- This should serve as the authoritative source for describing design details
- Requirements can be directly traced to design elements in the model

Input from Technical Solution (SP1.1-2, SP2.1-1)

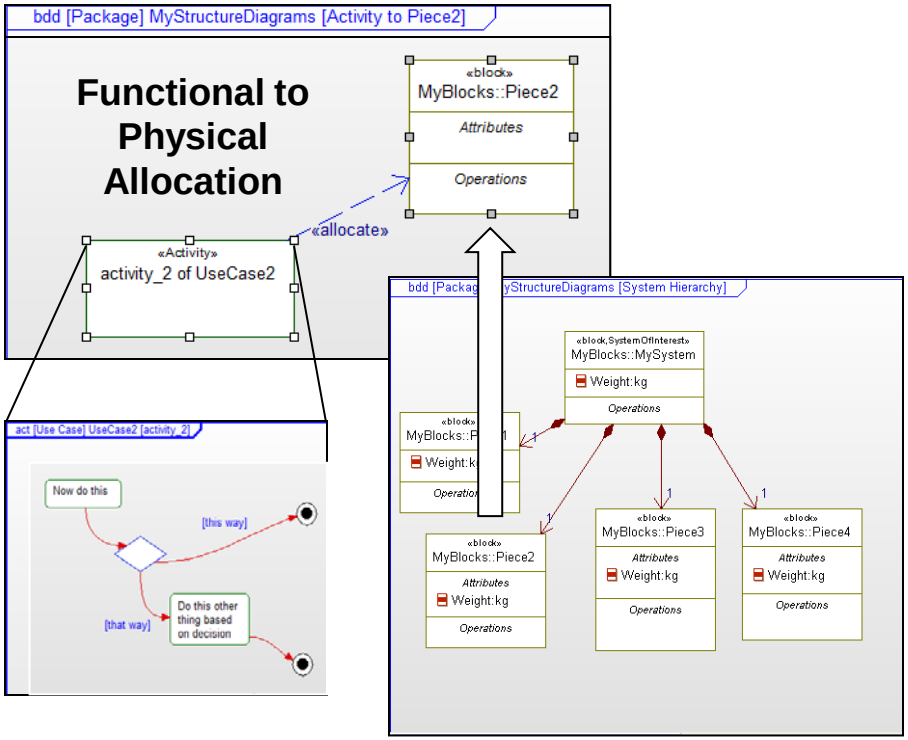


Results in a Functional Architecture

Requirements Development:
SG-2 Develop Product Requirements

- Allocate Product Component Requirements to System Elements (SP2.2-1).
- This requires execution of the Technical Solution Process Area
- If modeled properly a change to the model elements will invoke a change to all diagrams and uses of that model element

Input from Technical Solution (SP1.1-2, SP2.1-1)



To: block **Scope:** MyBlocks

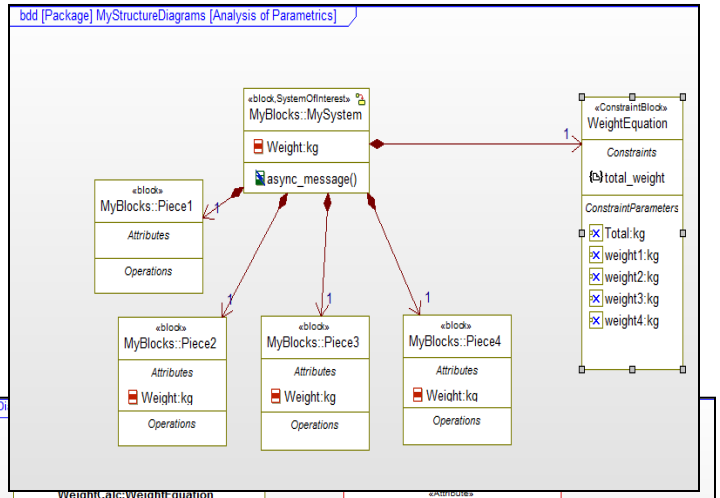
	MySystem	ExternalSystem1	ExternalSystem2	ES2_Block1	ES2_Block2	MySystemDomain	Piece1	Piece2	Piece3	Piec
activity_0										
activity_2								Piece2		

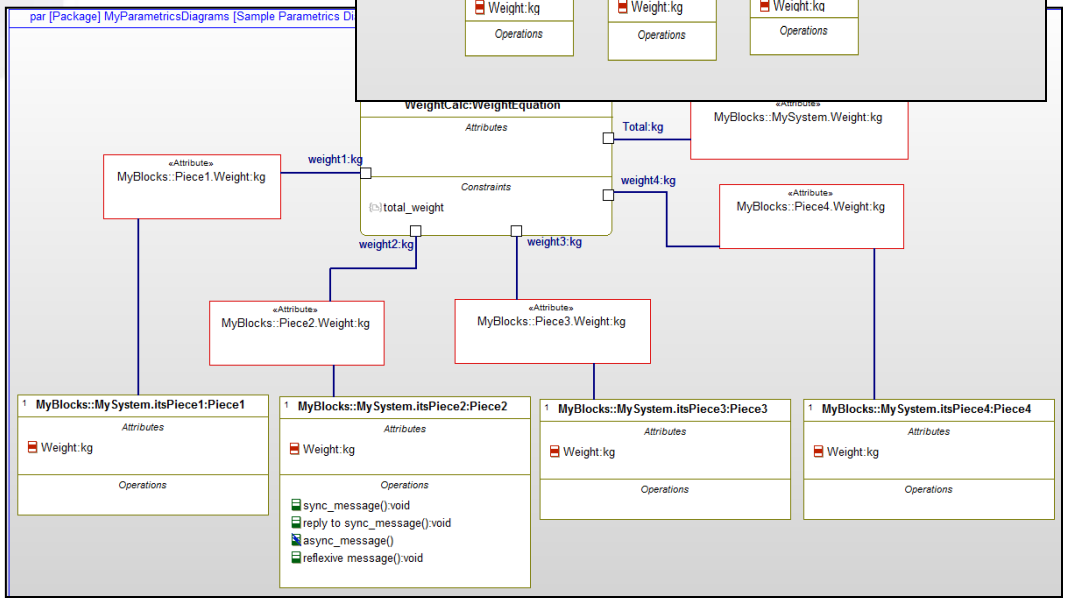
Results in a Physical Architecture

Requirements Development:
SG-2 Develop Product Requirements

- Use a BDD and Parametric Diagrams to show allocation of product component requirements and to capture Design Constraints (SP 1.1-1, SP-2.2-1).
- Used in to define design trade offs (SP 2.4-3) and to balance the requirements (SP3.4-3)
- Parametric Diagrams capture how requirements can be analyzed (SP 3.3-1) for validation (SP3.5-1) and evaluate performance measures (PA EV).

Input from Technical Solution (SP1.1-2)

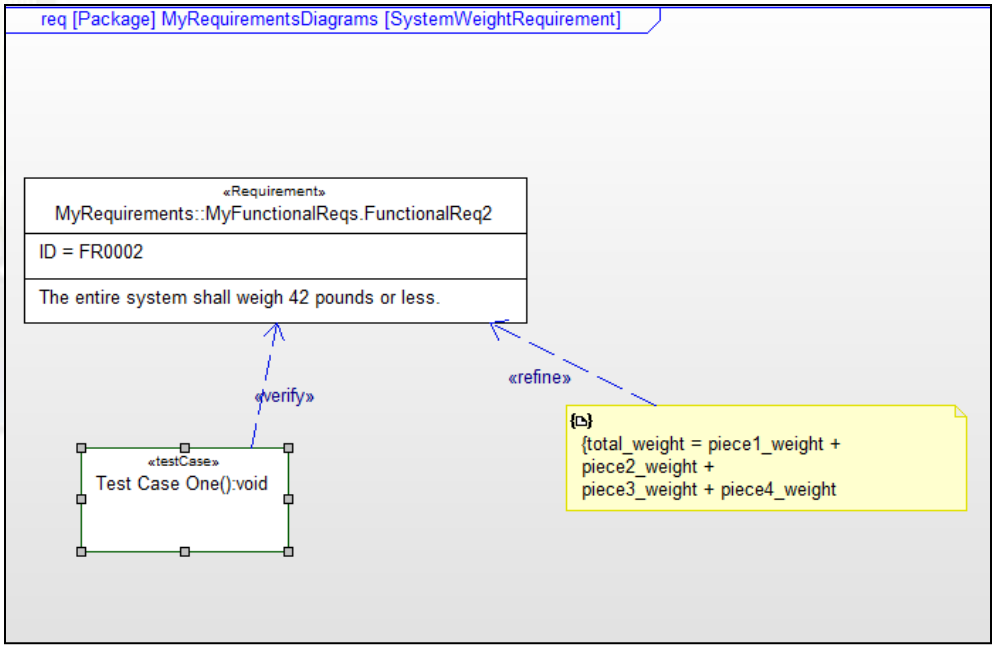




Work with the IPT and get them to agree

Requirements Development:
SG-3 Analyze and Validate Requirements

- Use Requirements Model Elements to capture Test Case Descriptions
- Use Requirements Diagrams to trace between Test Cases and Requirements.
- A diagram of this may help the team relate to how many tests are necessary to verify a requirements and vice versa.
- Auto generation of Verification Matrix



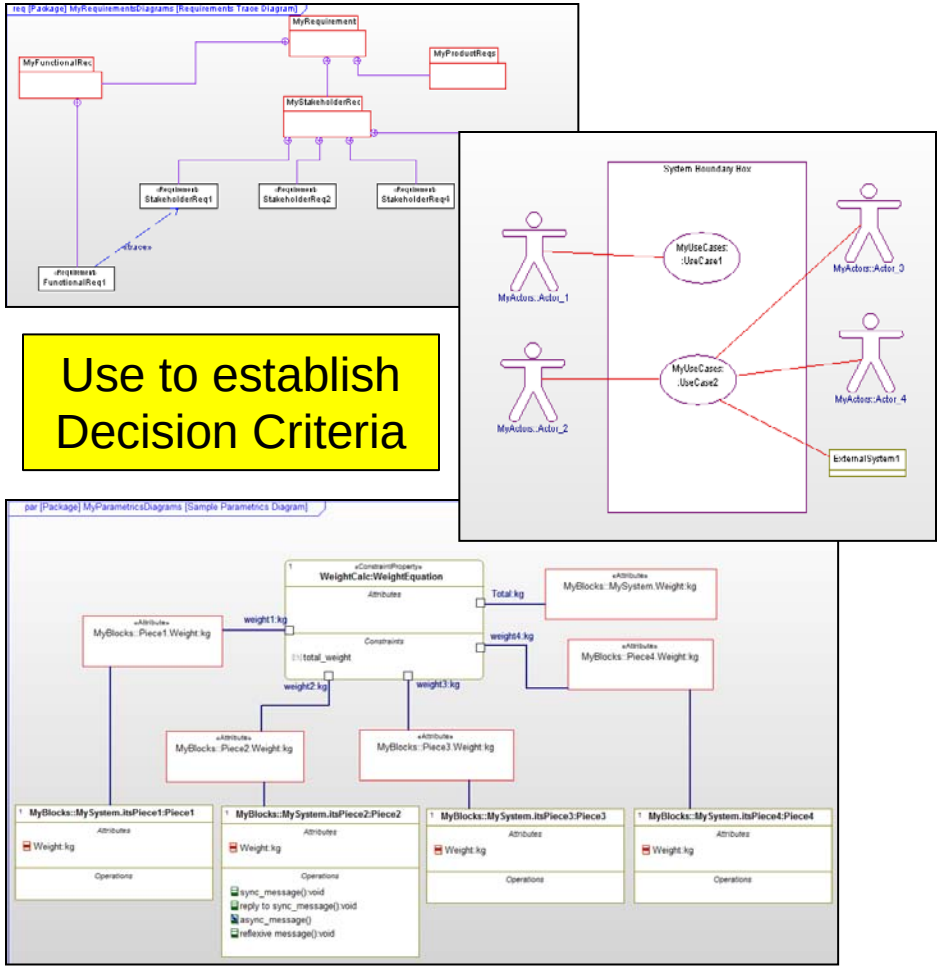
	FunctionalReq1	FunctionalReq2	StakeholderReq1
Test Case One		FunctionalReq2	

All Requirements should have Associated Verification Methods

Technical Solution:

SG-1 Select Product – Component Solutions

- You will need to construct behavior, cost, architecture, physical mock ups, prototypes of alternative solutions.
- Using SysML to capture a description to each alternative can help to communicate with the IPT
- The Decisions can be captured as notes in SysML
- Evaluation Criteria will likely emerge as requirements so capture them as such.

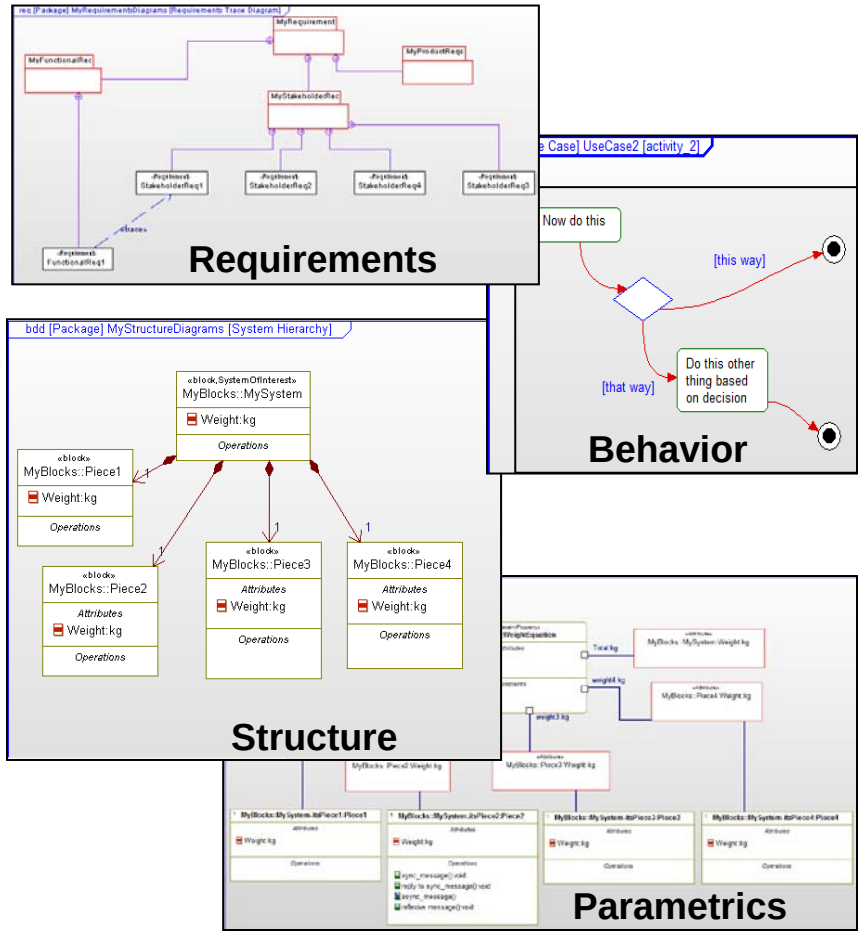


Use to establish Decision Criteria

Decision Criteria should be consistent with the model

Technical Solution:
SG-2 Develop the Design

- Use a BDD to allocate requirements from the logical design to each design Alternative (SP 1.1-2)
- Use BDD and IBD to capture an abstraction of the Physical Design. This is the Product Architecture and the authoritative description of the design.
- The whole process is recursive till you are at the lowest level in the system hierarchy
- The System model will organize your product description data



Include the model with your technical data package

Technical Solution:
SG-3 Implement the Product Design

- For SW products SysML and UML can be used to auto generate Code.
- For Electrical and Mechanical Systems CAD product models can be used to directly support manufacturing, assembly, inspection, and test.
- A rigorously modeled design will support resolving problems as they come up.

Model Based Engineering Can Help

Planning Considerations

- Use people who are open to this approach and are willing to go the extra mile. Pair them with someone who is experienced. It will accelerate the learning process and keep them from stalling out.
- Conduct a Gap Analysis
- You will need to pick a tool
- You will need a Methodology that describes how, why, when, and what to model.
- You will need to train even the most experienced Systems Engineers (Language, Tool , Methodology)
- You will want to pilot it on several projects to learn your own lessons.
- Keep it simple at first and expand to more complex modeling problems.
- Model what you know best and see if everyone really understands it the same way. (You will need to work towards this.)

Conclusion

- Established the need to change
- Discussed how Modeling supports many of the RD and TS CMMI practices.
- It will take time and requires training and practice.
- We can’t keep operating the same way.