GO BEYOND

INTRODUCTION TO SYSML

WITH CAMEO SYSTEMS MODELER

Northeast Chapter INCOSE MBSE Workshop , Oct 17,18
Instructor: Randy Skelding - Pratt & Whitney Aircraft , Advanced Concepts & Technology
email: randy.skelding@pw.utc.com

A HANDS-ON INTRODUCTION TO SYSML FOR MBSE

GO BEYOND

GETTING STARTED

A UNITED TECHNOLOGIES COMPANY

GETTING STARTED

This section includes:

- Course Overview
- Course Objectives
- Course Approach
- Installing Cameo Systems Modeler on your laptop
- Starting Cameo Systems Modeler and connecting to licenses.

COURSE OVERVIEW

This course will address the following topics:

- High level overview of Systems Engineering, Model Based Systems Engineering, Descriptive Modeling

COURSE OVERVIEW

This course will address the following topics:

- High level overview of Systems Engineering, Model Based Systems Engineering, Descriptive Modeling
- Introduction to SysML – Modeling Structure, Behavior, Requirements and Parametric Relationships

COURSE OVERVIEW

This course will address the following topics:

- High level overview of Systems Engineering, Model Based Systems Engineering, Descriptive Modeling
- Introduction to SysML – Modeling Structure, Behavior, Requirements and Parametric Relationships
- Introduction to MBSE methods – OOSEM and similar approaches

COURSE OVERVIEW

This course will address the following topics:

- High level overview of Systems Engineering, Model Based Systems Engineering, Descriptive Modeling
- Introduction to SysML – Modeling Structure, Behavior, Requirements and Parametric Relationships
- Introduction to MBSE methods – OOSEM and similar approaches
- Introduction to the SysML tool - Cameo Systems Modeler

COURSE OVERVIEW

This course will address the following topics:

- High level overview of Systems Engineering, Model Based Systems Engineering, Descriptive Modeling
- Introduction to SysML – Modeling Structure, Behavior, Requirements and Parametric Relationships
- Introduction to MBSE methods – OOSEM and similar approaches
- Introduction to the SysML tool - Cameo Systems Modeler
- Hands-on modeling with Cameo Systems Modeler on sample problems

COURSE OVERVIEW

This course is intended for:

- Anyone who is interested in Model Based Systems Engineering (MBSE) and the use of SysML in MBSE but has not used SysML before.
- Engineers who specify and derive requirements safety critical software and hardware but who are not familiar with SysML.

Assumptions:

- Systems Engineering background – and / or
- Software / Computer Systems Analysis background
- No prior exposure to SysML or UML required or assumed.

COURSE OVERVIEW

Resources:

- “A Practical Guide to SysML: The Systems Modeling Language”; Friedenthal, Moore, and Steiner; 2009, Elsevier, Inc.
- “SysML Distilled A Brief Guide to the Systems Modeling Language”; Delligatti; 2014, Addison-Wesley
- “Modeling with SysML”; Joe Wolfrom (John Hopkins APL); 2010, Tutorial presentation at INCOSE 2010 conference.

The structure of this course follows similar content in INCOSE tutorials, including the “Modeling with SysML” course by Joe Wolfrom and “An Introduction to MBSE with SysML” by Sandy Friedenthal.

Software:

- NoMagic “Cameo Systems Modeler” V19.0 SP 2

ONLINE RESOURCES

- John Hopkins APL on-line tutorial by Joe Wolfrom https://www.jhuapl.edu/About/OTTSysML
  This requires that you register on the site, but the download is free.

- INCOSE tutorial http://www.omgsysml.org/INCOSE-OMGSysML-Tutorial-Final-090901.pdf

- Short instructional Videos… from Dr. Saulius Pavalkis at NoMagic.
- Flashing light example – model execution https://www.youtube.com/watch?v=Il3V3KOFvTM
- Thermostat model with model execution https://www.youtube.com/watch?v=sdwJ93-wVFY
- Sequence Diagrams with model execution https://www.youtube.com/watch?v=C8lk5gRnXyo
- Information Exchange Between Blocks Simulation https://www.youtube.com/watch?v=Cwz_tuKX_xs
- Send Signal Between Subsystems / Blocks https://www.youtube.com/watch?v=vcj5xIfrORk
- Pass Values Through Ports https://www.youtube.com/watch?v=1PDKgfX-uhM
- Parametric diagram example https://www.youtube.com/watch?v=xk74YxFbCLA
- Trade Study example https://www.nomagic.com/events/webinars/item/perfroming-trade-off-studies?category_id=6
- Logical Interfaces modeling https://www.youtube.com/watch?v=VO_VjyNaj50

COURSE OBJECTIVE

Primary course objective:

- Develop understanding of the elements of the SysML language
- Become comfortable with the use the Cameo Systems Modeler tool.
- Gain an appreciation for where SysML should be used and what it does and does not do.

After completing this course, you should be able to create basic SysML models with Cameo Systems Modeler for both software and hardware systems, and validate the correctness of behavioral content by model execution.

COURSE APPROACH

- The course has several sections, each taking from ½ hour to 1 hour to complete. Each section has a PowerPoint lecture and class discussion based on the presentation.
- Where appropriate, labs will follow the section slides, and should take ½ hour or so to complete the steps in the lab.
- Labs are “check pointed” so that if you cannot complete a lab in time for the start of the next section, we provide a worked out example as a starting point for the next lab.

Regardless of the modeling tool that is used, significant practice is needed to establish modeling ability. Working on the labs is essential for developing proficiency.

Modeling ability for any particular purpose gets better as you make and use more models.

COURSE SETUP

- You will need to install Cameo Systems Modeler in your laptop or desktop machine if you have not done so already.

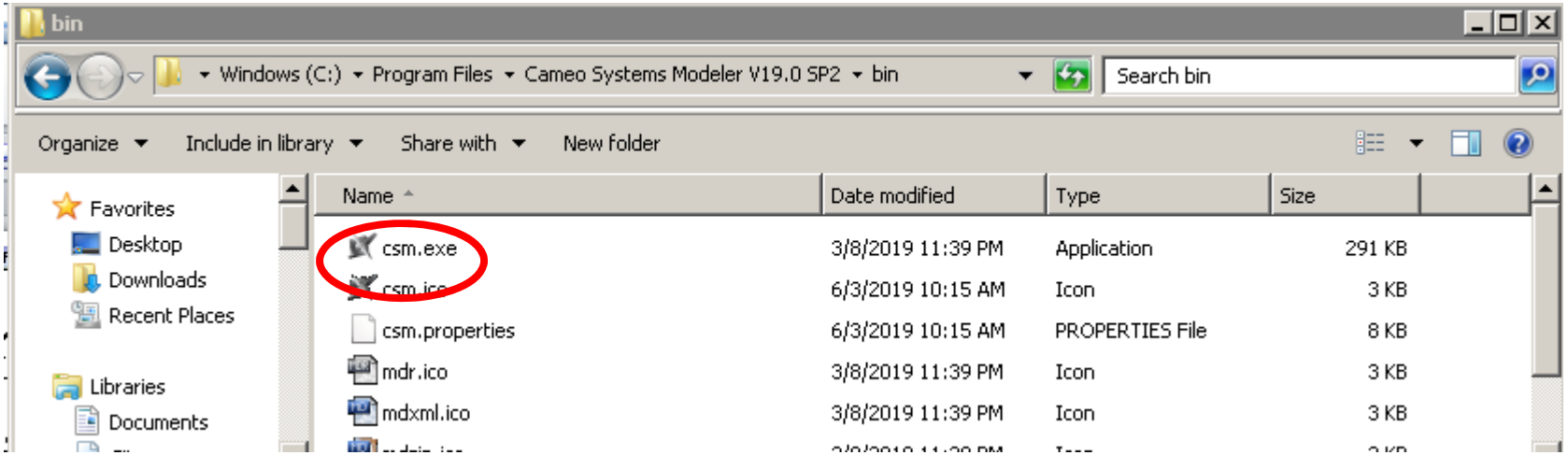
- An installation executable will be provided on a usb-stick that you can use.

STARTING CAMEO SYSTEMS MODELER

You may have the Cameo Start Icon on your desktop or toolbar. Just click it.

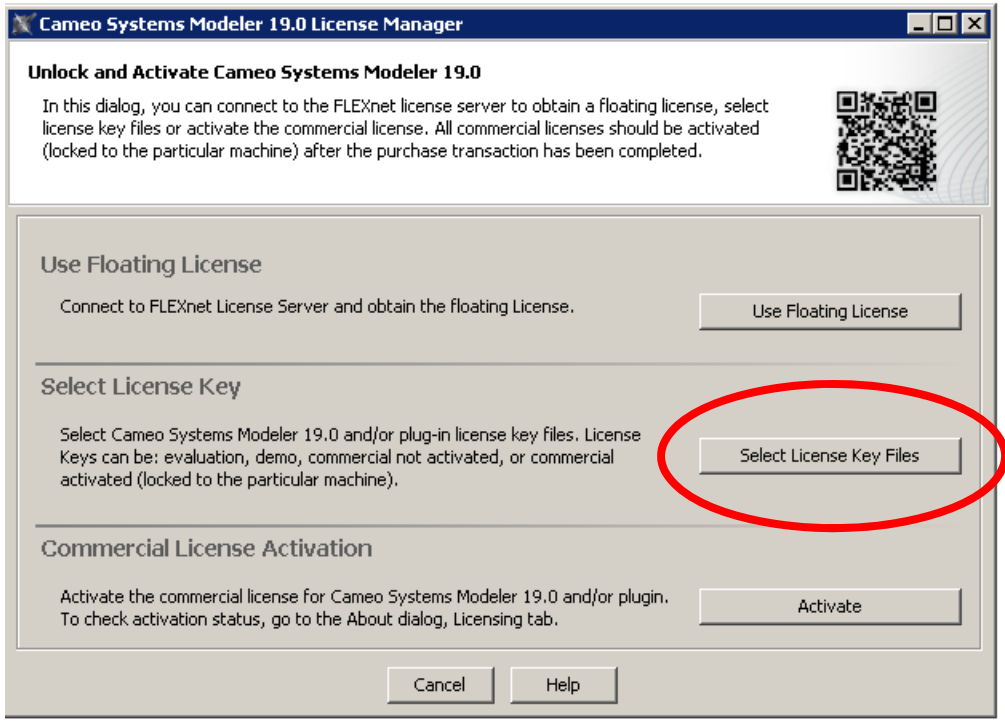
If you do not have this shortcut, find the csm.exe program in the bin area where Cameo is installed.

Double click to run csm.exe.

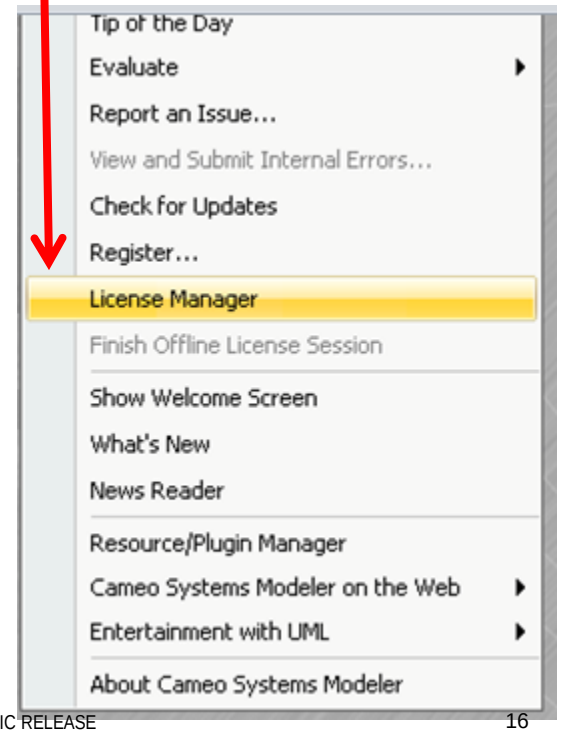


STARTING CAMEO SYSTEMS MODELER

Point Cameo at the evaluation license.

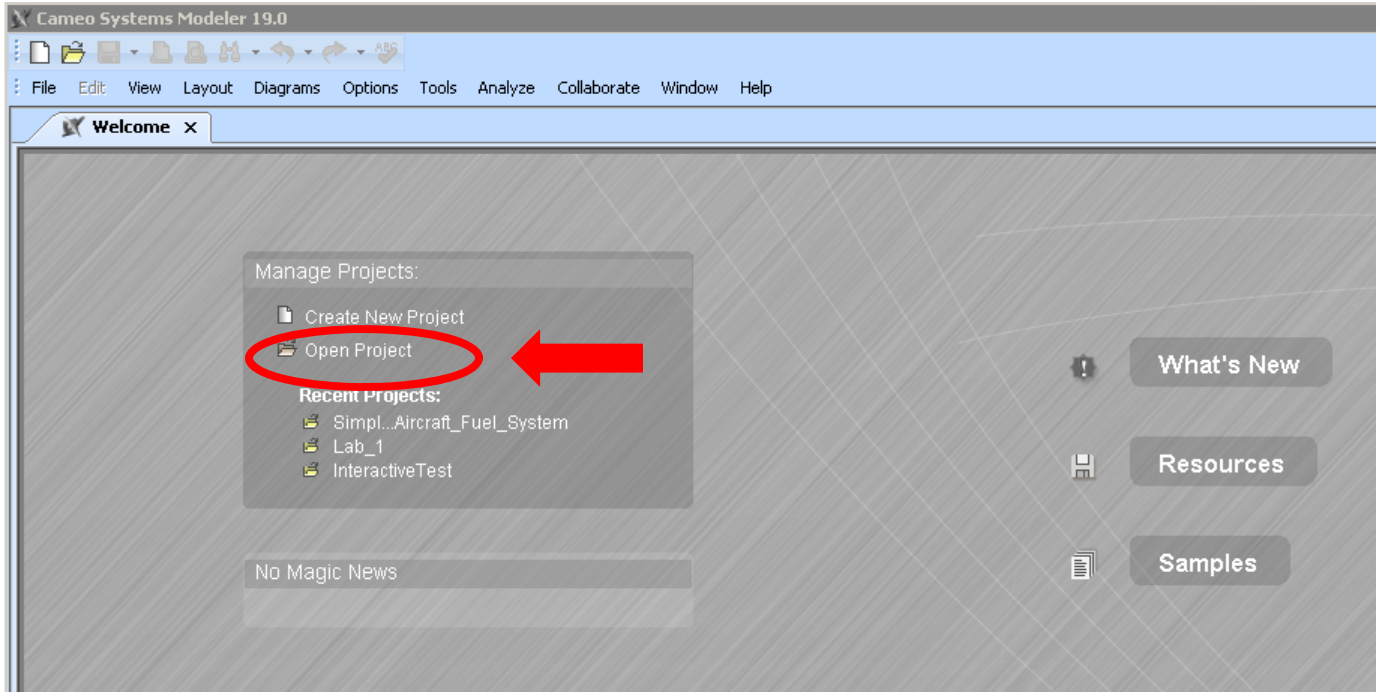


To get here … find "Help" and click it to get the pop up with the License Manager option.



GETTING STARTED

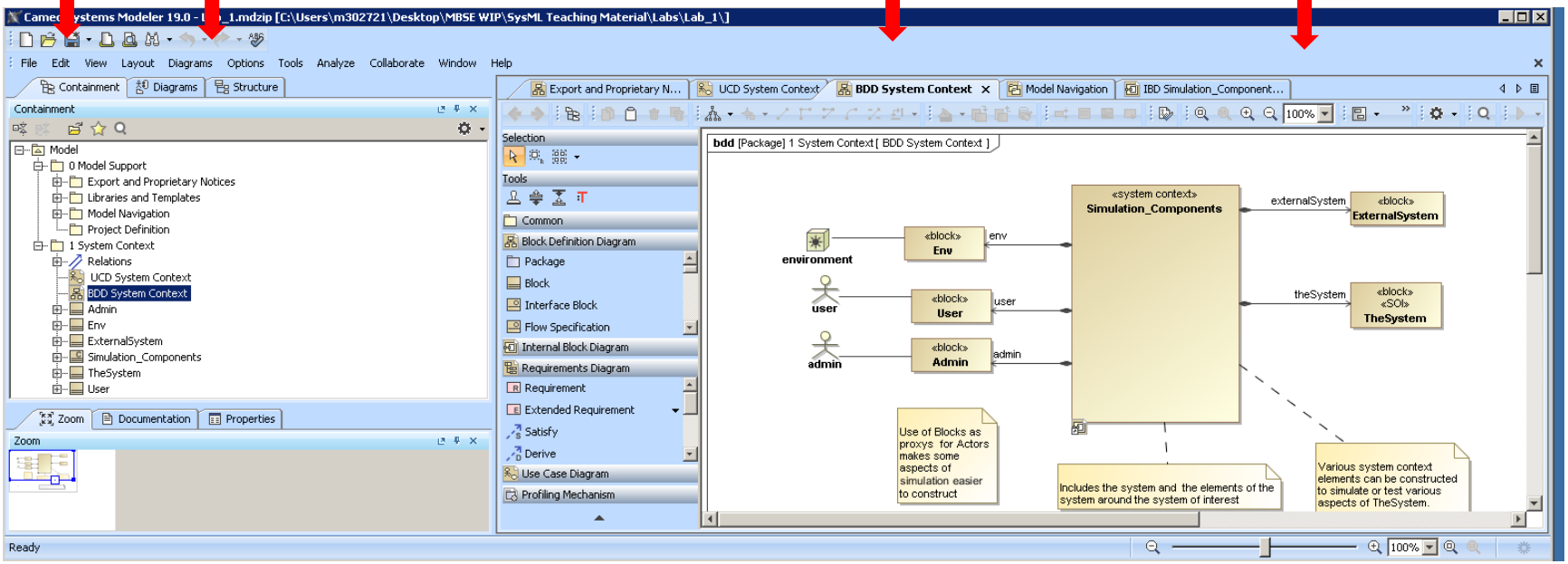
Cameo Start Screen – New Project / Open Project



GETTING STARTED

Cameo Systems Modeler Screen and Toolbar Layout

Save Undo / Redo Tab to select diagram Zoom



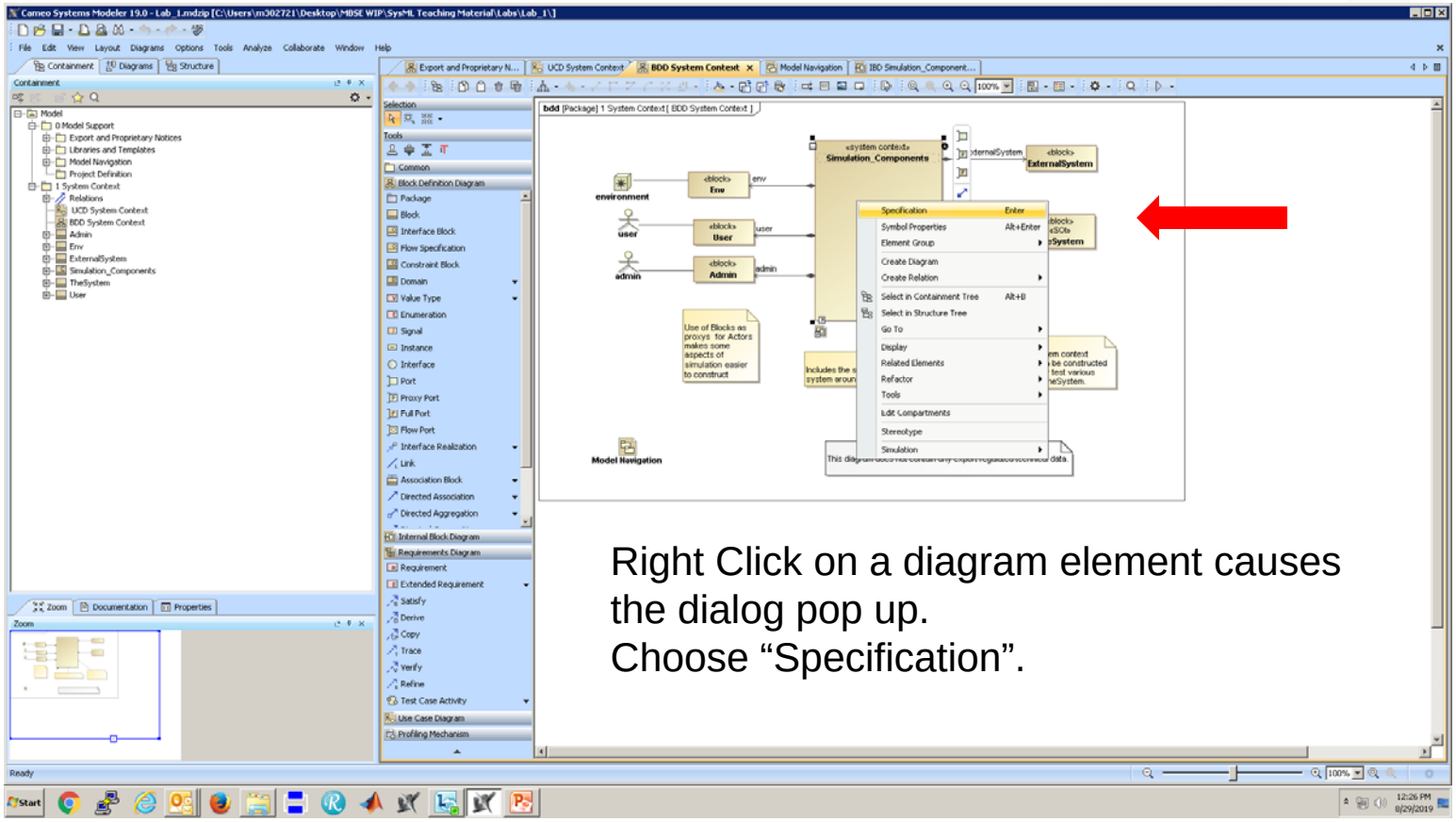
Model Browser

Pallet

Diagram Area

GETTING STARTED

Specification pop up – view or modify a model element

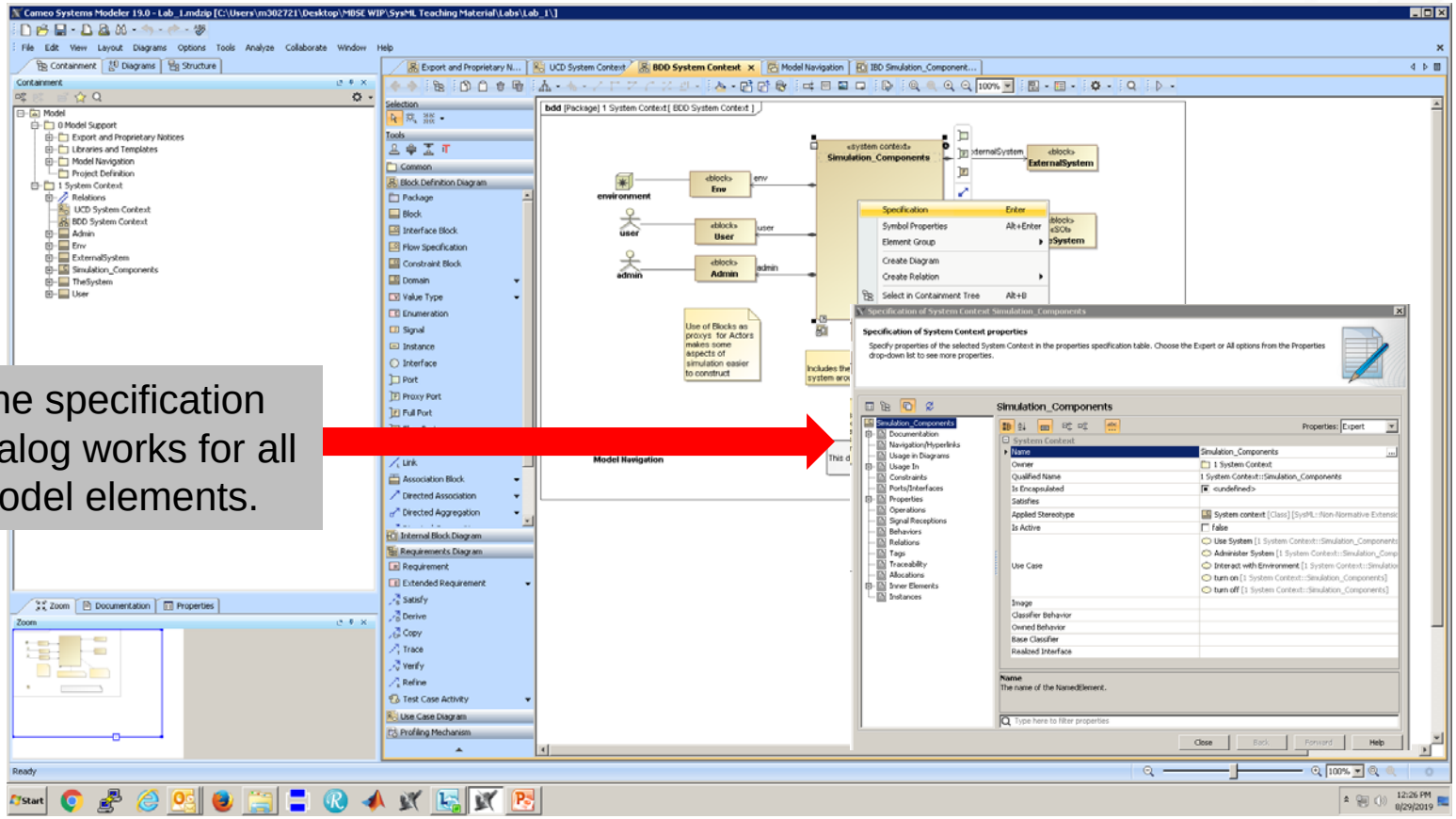


Right Click on a diagram element causes the dialog pop up.
Choose “Specification”.

GETTING STARTED

Specification pop up – view or modify a model element

The specification dialog works for all model elements.



IMPORTANT TRICK – HOW TO DELETE

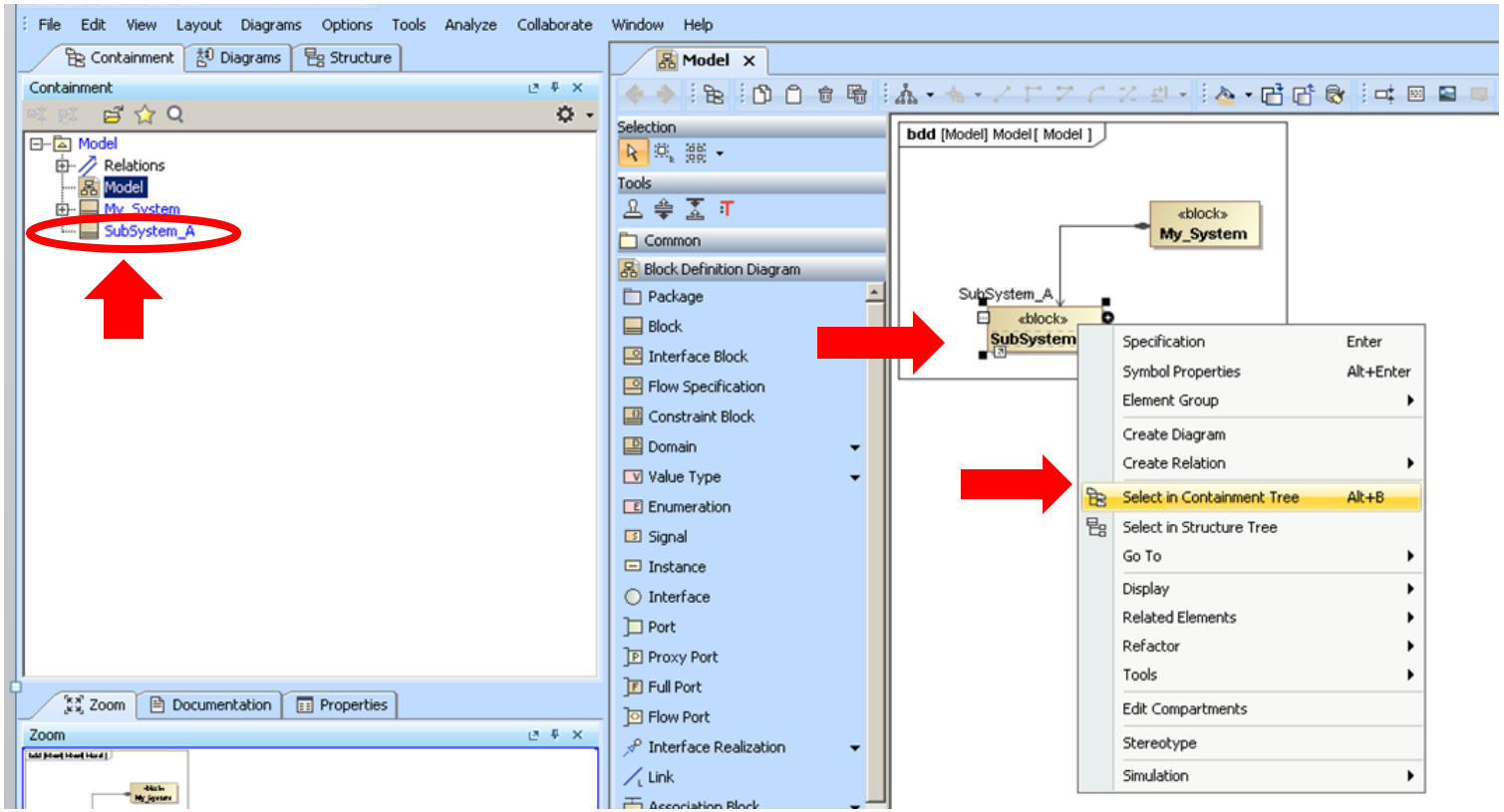
Deleting something from a diagram does NOT delete it from the model unless you take some extra steps!

- This is so you can show only what you want to show on a diagram, so it does not get cluttered. You can show a particular element on multiple diagrams, and you can produce different views of the system, all from a single consistent model.
- To really, really, delete from the model, select the item and hit the Ctrl key + the ‘D’ key. ^D.

Or…

GETTING STARTED … DELETE FROM MODEL

Select the item and right click. Choose “Select in Containment Tree”. You will see the item highlighted in the Containment tree list view. Hit the “Delete” key.



LAB 1 – GETTING STARTED

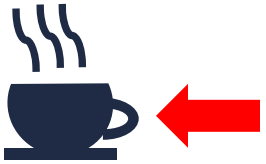
1. Copy in the SysML_Intro folder into your Documents folder to hold the content of this course.

- Set your default directory to C:\Users\<yourid>\Documents\SysML_Intro\Labs

2. Start Cameo Systems Modeler
3. Create a new SysML project named “Lab_1” in \Labs.
4. Create a new Block Definition diagram. It will be named “Model”, since this is the name of the model until we change it. (We will explain blocks later). Drag a block onto the diagram and name it “MySystem”.
5. Select the “MySystem” block by single click. Hit the delete key. See the model – the block is still there, but it is gone from the diagram.
6. Hit UNDO (on the top toolbar) to get “MySystem” back on the diagram.

LAB 1 – GETTING STARTED

8. Select the “MySystem” block again and hit ^D.
9. The block is gone from the model.
10. Hit UNDO to bring it back.
11. Select “MySystem” block, and right click for the pop up window.
12. Choose “Select in Containment Tree”, and hit the “Delete” key.
13. The block is gone from the model.
14. Hit UNDO to bring it back.
15. Save the model.
16. Coffee Break!



Engineering is fueled by caffeinated beverages

GO BEYOND

KEY QUESTIONS TO BE ANSWERED

QUESTIONS

- What is SysML?
- What is it used for?
- Who should use it?
- Why should I use it?
- When should I use it?
- What do I need to know to use it?

This course attempts to address the following questions.

GO BEYOND

SYSML, SE, AND MBSE

A UNITED TECHNOLOGIES COMPANY

SYSML - SE AND MBSE

SysML is a modeling language for representing systems.

A system is a construct or collection of different elements that together produce results not obtainable by the elements alone.

SYSML - SE AND MBSE

SysML is a modeling language for representing systems.

A system is a construct or collection of different elements that together produce results not obtainable by the elements alone.

These system elements may include people, procedures, environments, physical things, hardware, software, data, interfaces, interconnections, flows, and transforms. The system elements may include other systems and “sub systems”.

A system accomplishes a set of specific purposes by the interactions between the component elements.

SYSML – SE AND MBSE

"Systems Engineering" (SE) is the discipline that considers *all* the connected elements of a system and focuses on understanding the interactions between these elements."

Model Based Systems Engineering (MBSE) is modern, computer aided, tool assisted, systems engineering.

SYSML – SE AND MBSE

"Systems Engineering" (SE) is the discipline that considers *all* the connected elements of a system and focuses on understanding the interactions between these elements."

Model Based Systems Engineering (MBSE) is modern, computer aided, tool assisted, systems engineering.

"MBSE is the *formalized application of modeling* to support system requirements, design, analysis, verification and validation activities beginning in the conceptual design phase and continuing throughout development and later life cycle phases." INCOSE SE Vision 2020, Sept 2007

SysML is *designed* to enable and assist MBSE tasks.

SYSML – SE AND MBSE

"Systems Engineering" (SE) is the discipline that considers *all* the connected elements of a system and focuses on understanding the interactions between these elements."

Systems Engineering *requires clear communication* between personnel with different engineering backgrounds.

SYSML – SE AND MBSE

"Systems Engineering" (SE) is the discipline that considers *all* the connected elements of a system and focuses on understanding the interactions between these elements."

Systems Engineering *requires clear communication* between personnel with different engineering backgrounds.

This makes it a difficult specialty on its own. It requires the ability to work with and understand the work of specialists, and recognize the important interactions between specialties.

SYSTEMS ENGINEERING (SE)

- Systems Engineering Enables
  - Lean Operation
  - Agile Operation

SYSTEMS ENGINEERING (SE)

- Systems Engineering Enables
  - Lean Operation
  - Agile Operation

Leads to

Cost Savings, Increased Competitiveness, Innovative Culture

SYSTEMS ENGINEERING (SE)

- Systems Engineering Enables
  - Lean Operation
  - Agile Operation

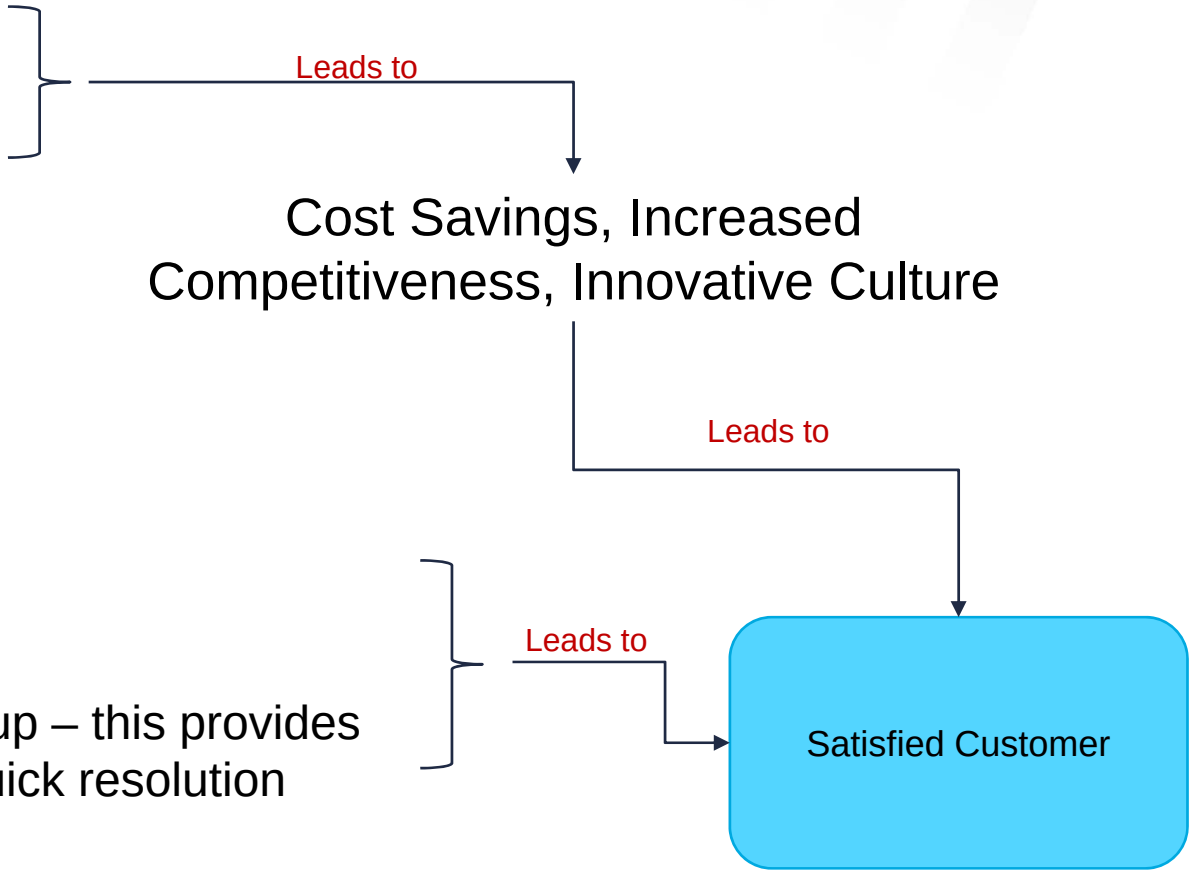
Leads to

Cost Savings, Increased Competitiveness, Innovative Culture

- Do it right, once
  - If a problem pops up – this provides a mechanism to quick resolution

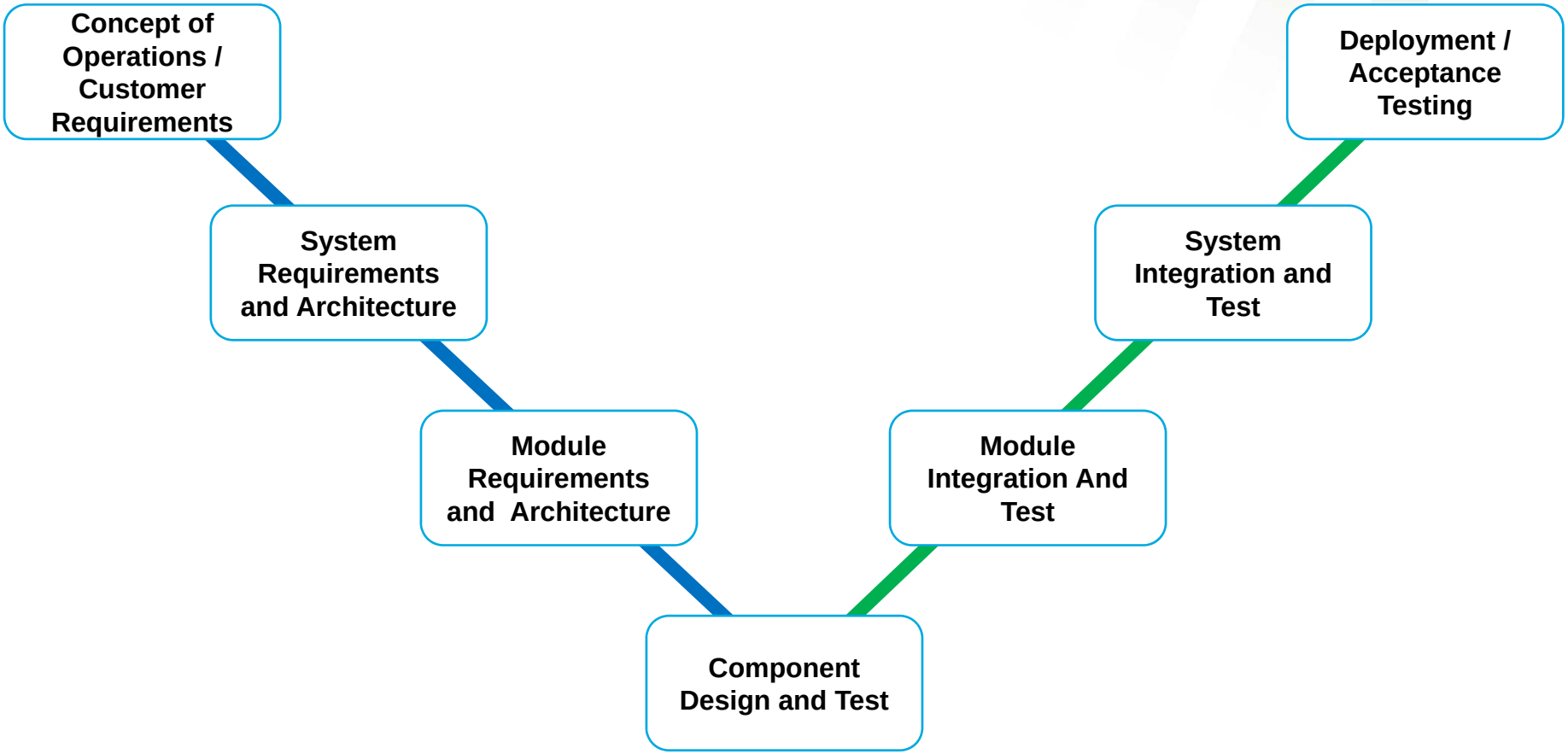
SYSTEMS ENGINEERING (SE)

- Systems Engineering Enables
  - Lean Operation
  - Agile Operation

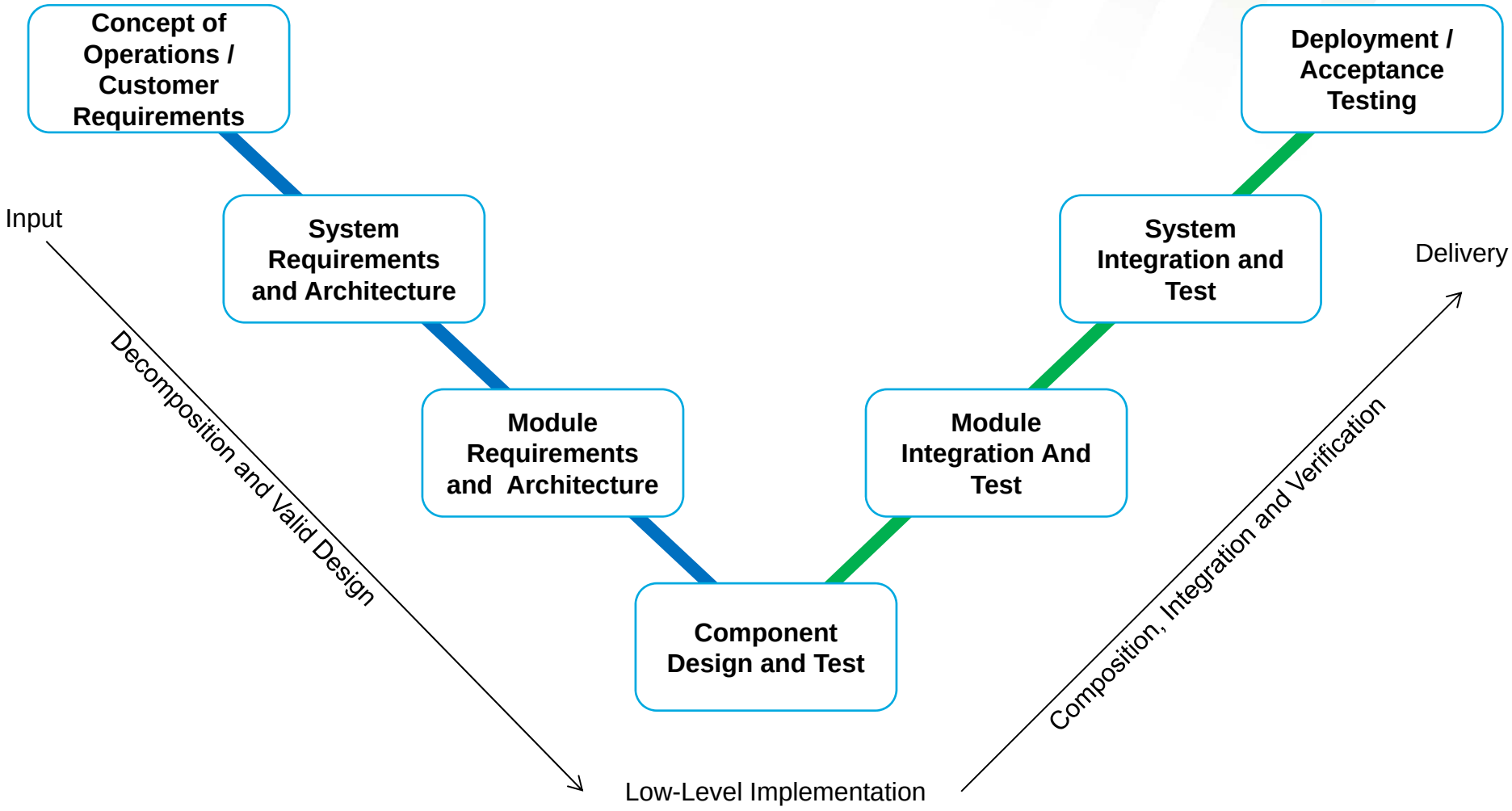


- Do it right, once
  - If a problem pops up – this provides a mechanism to quick resolution

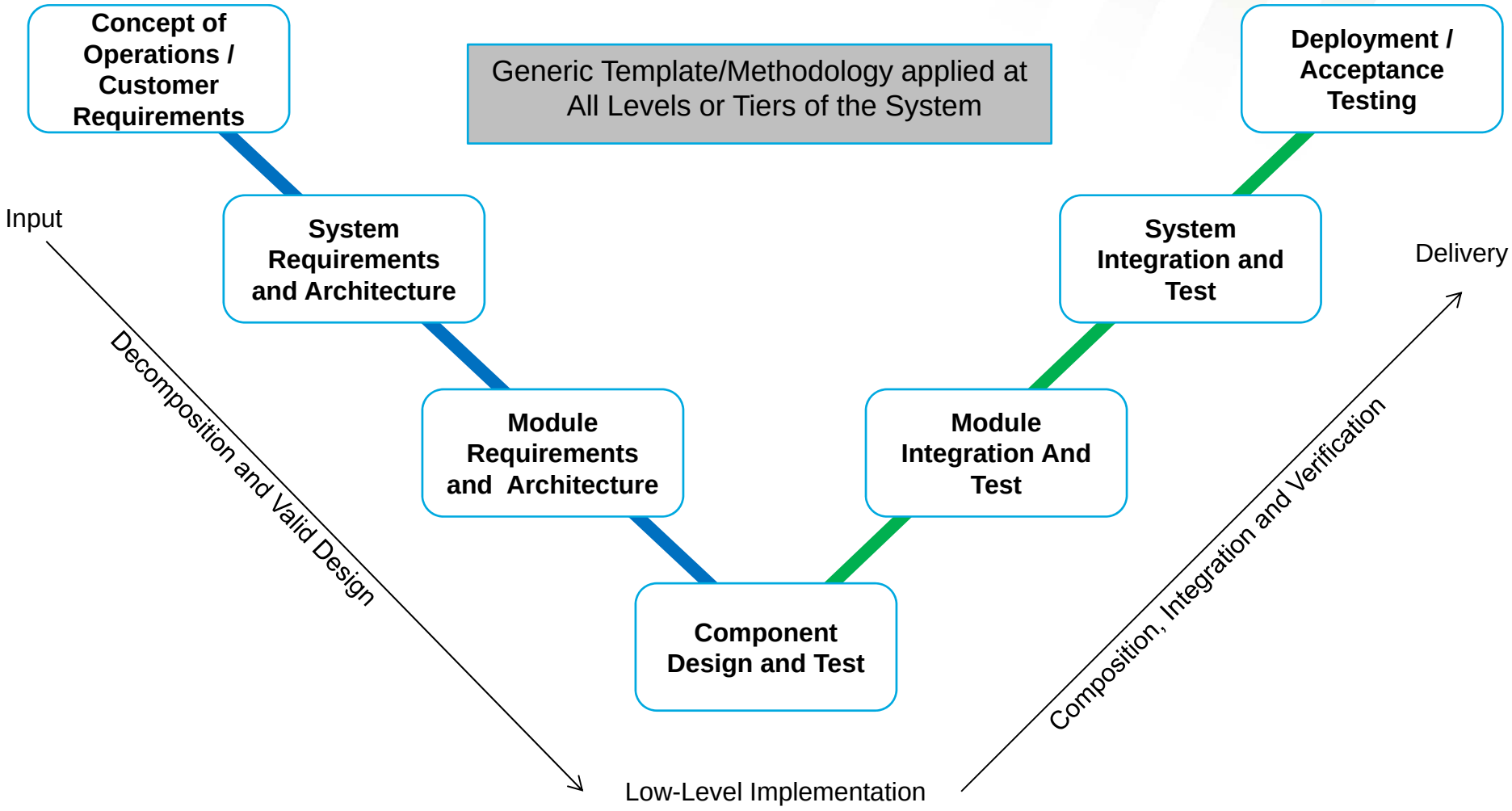
SYSTEMS ENGINEERING “V” DIAGRAM



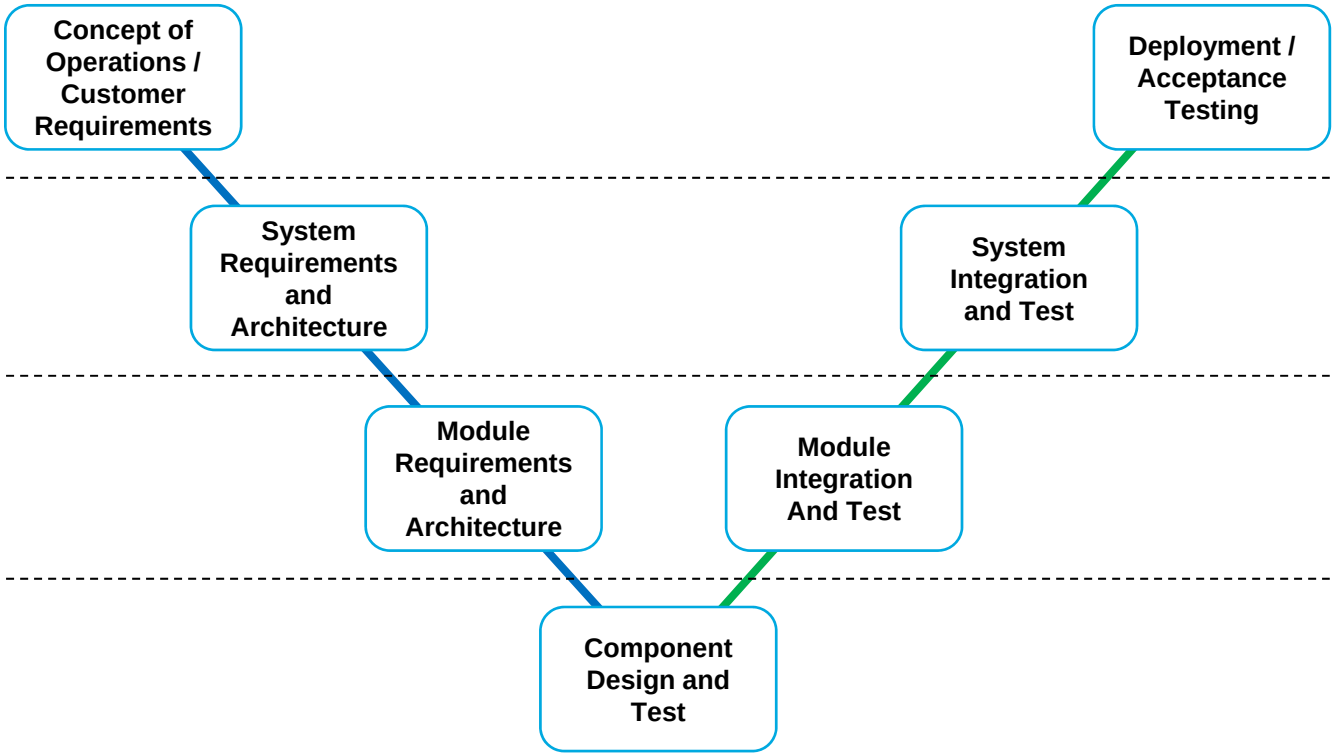
SYSTEMS ENGINEERING "V" DIAGRAM



SYSTEMS ENGINEERING "V" DIAGRAM



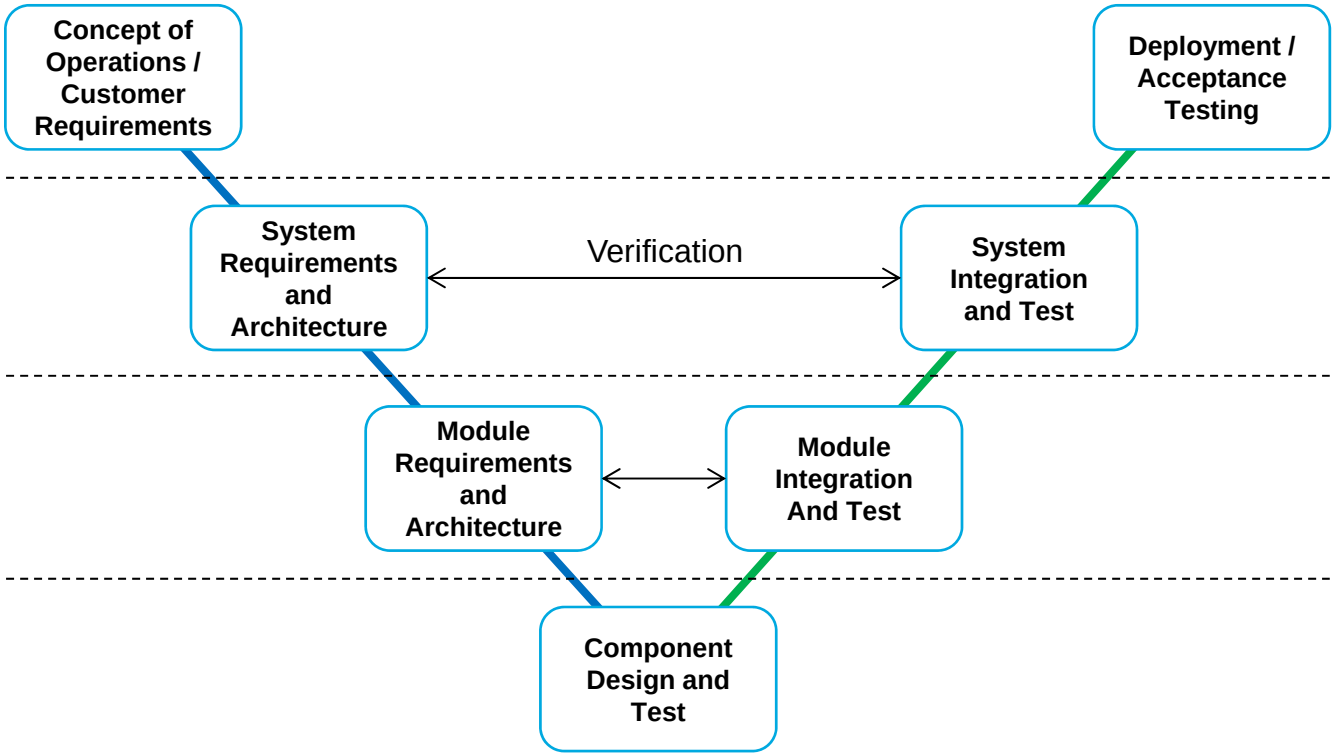
SYSTEMS ENGINEERING "V" DIAGRAM



Tiers and Boundaries

- Provide:
  - Clear boundaries
  - Clear ownership

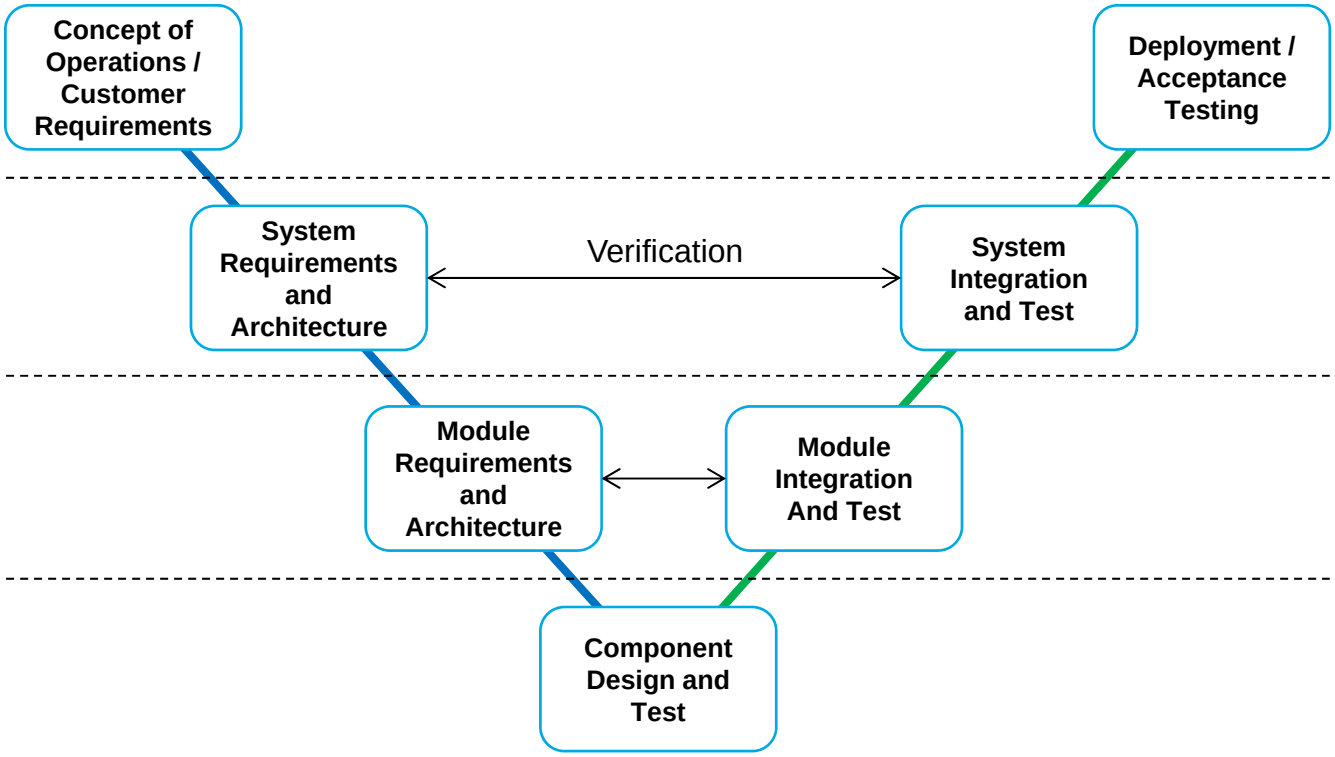
SYSTEMS ENGINEERING "V" DIAGRAM



Tiers and Boundaries

- Provide:
  - Clear boundaries
  - Clear ownership
- Verification Level Definition

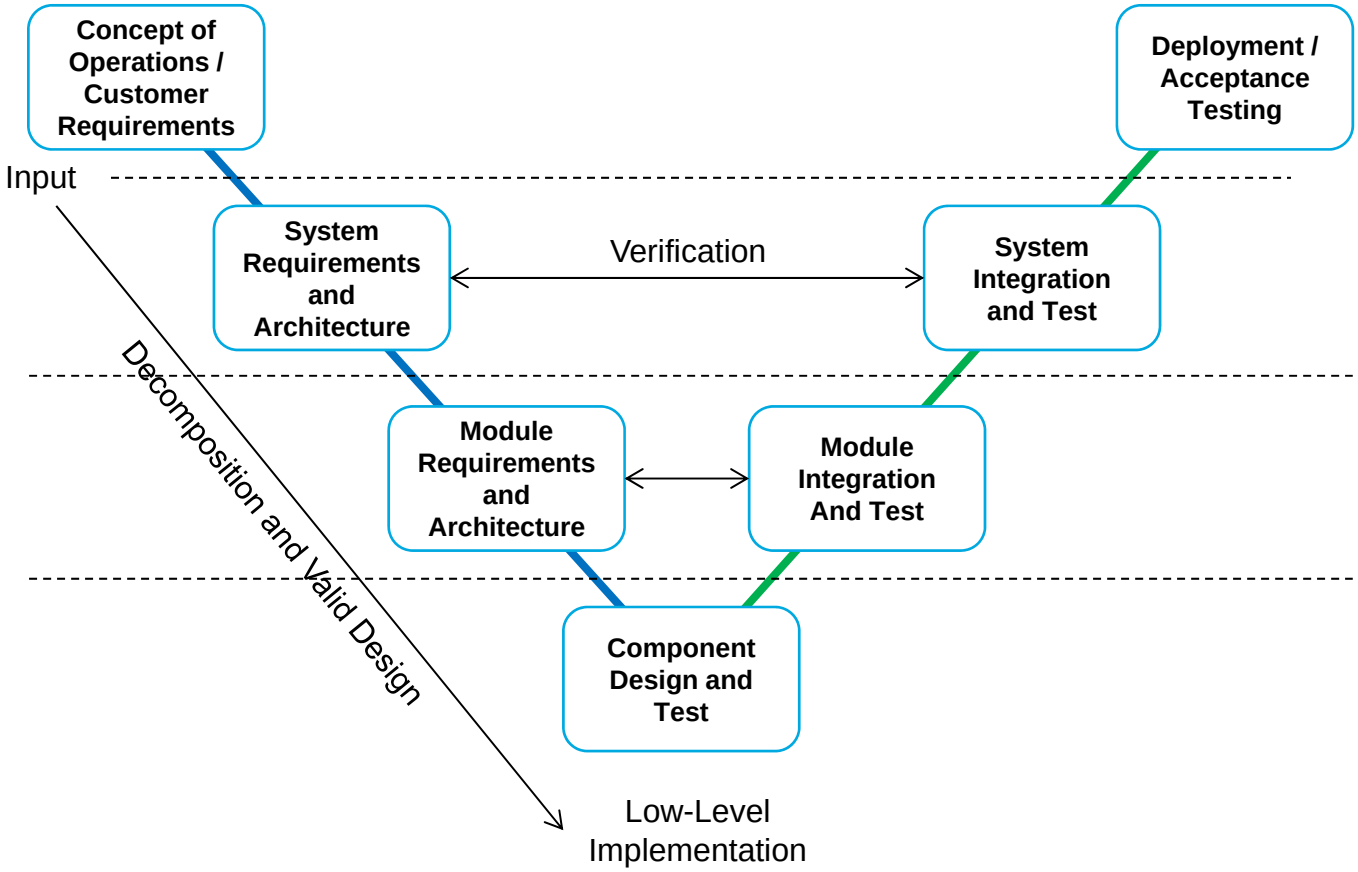
SYSTEMS ENGINEERING "V" DIAGRAM



Tiers and Boundaries

- Provide:
  - Clear boundaries
  - Clear ownership
- Verification Level Definition
- Tiers provide complete scope definition
  - Told what needs to be done
  - Provide what needs to be done to next tier down

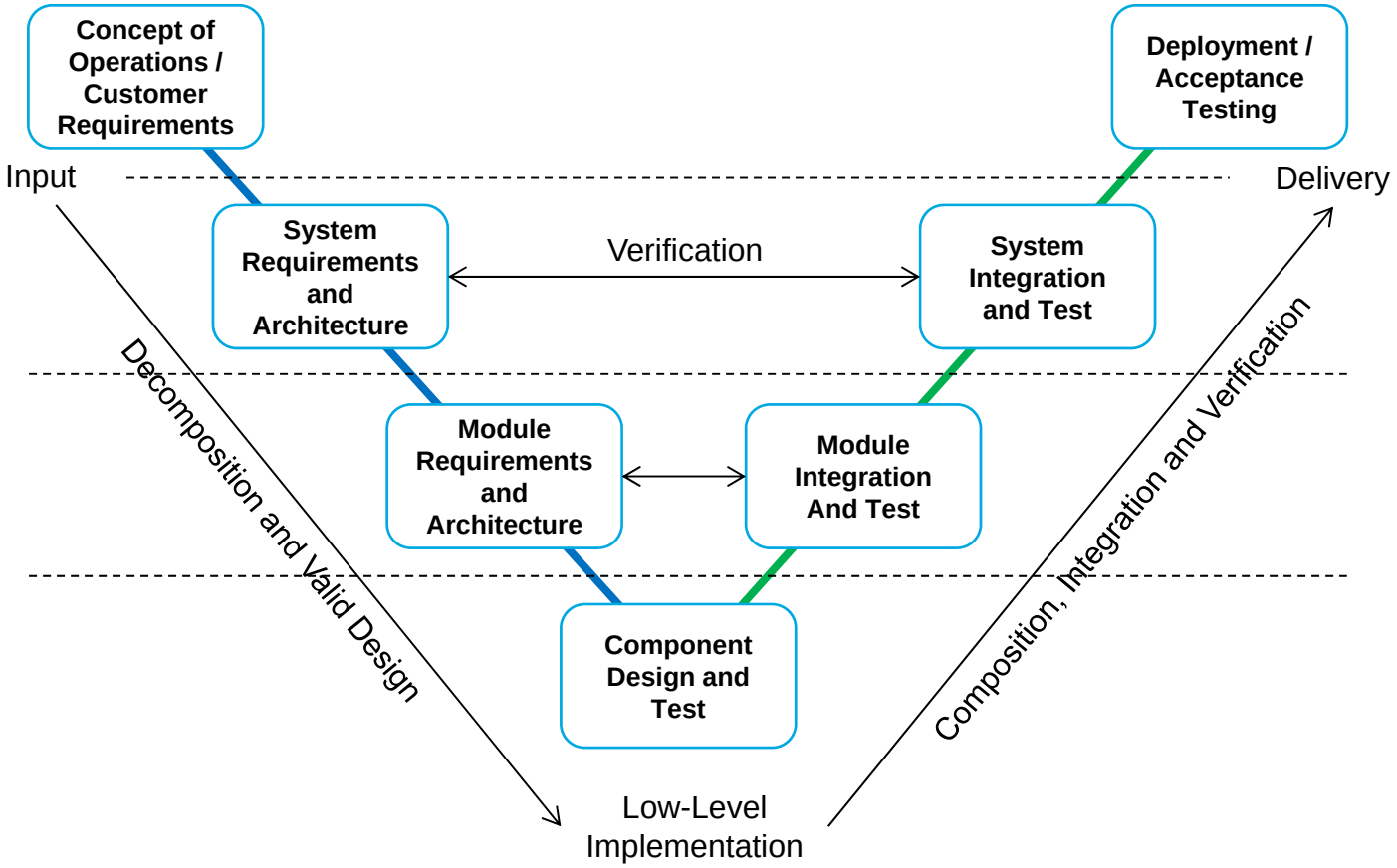
SYSTEMS ENGINEERING "V" DIAGRAM



Tiers and Boundaries

- Provide:
  - Clear boundaries
  - Clear ownership
- Verification Level Definition
- Tiers provide complete scope definition
  - Told what needs to be done
  - Provide what needs to be done to next tier down
- Tiers are scoped to drive **valid** design – design occurs at each tier
  - Decide how to implement at tier level

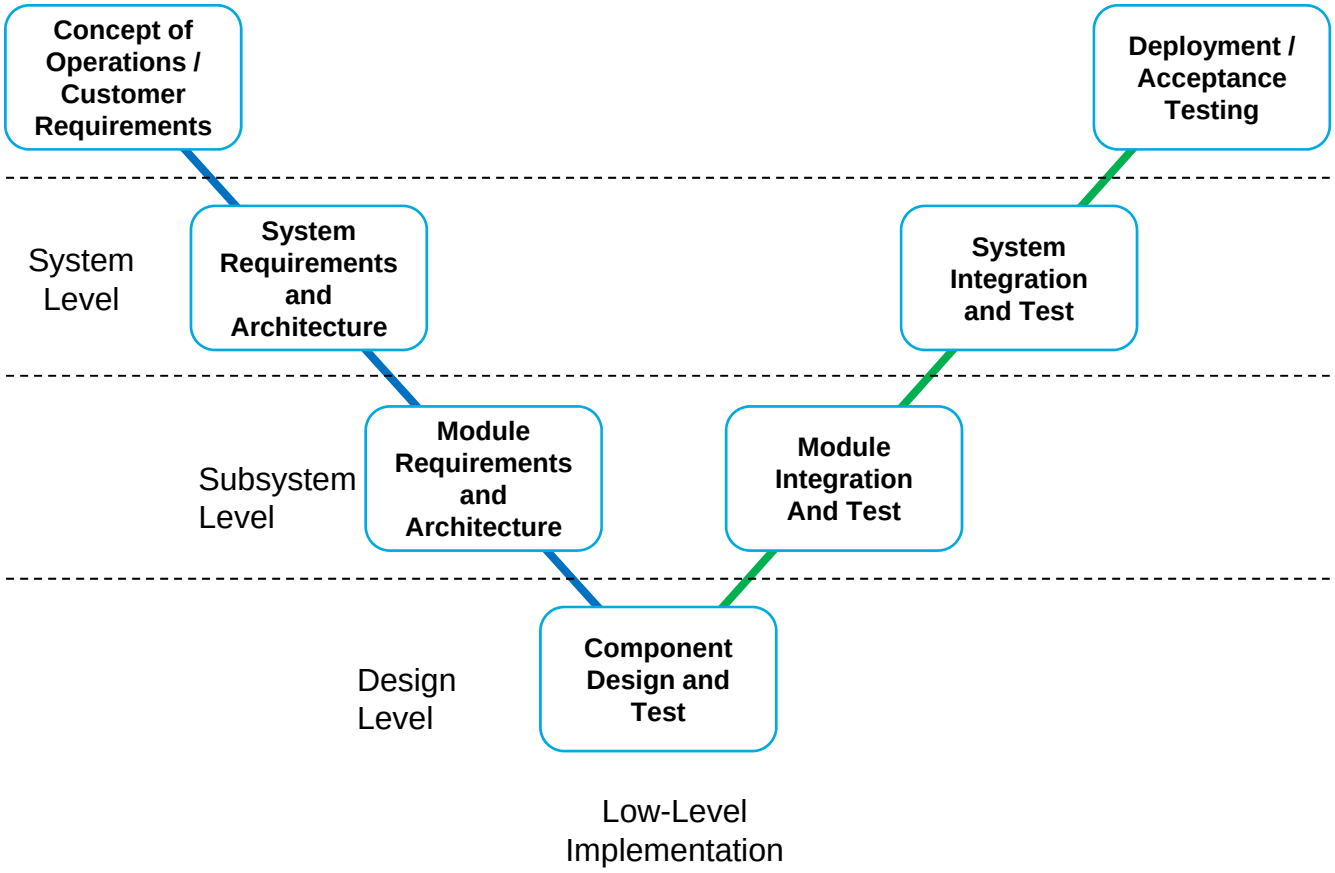
SYSTEMS ENGINEERING "V" DIAGRAM



Tiers and Boundaries

- Provide:
  - Clear boundaries
  - Clear ownership
- Verification Level Definition
- Tiers provide complete scope definition
  - Told what needs to be done
  - Provide what needs to be done to next tier down
- Tiers are scoped to drive **valid** design – design occurs at each tier
  - Decide how to implement at tier level
- Verification of valid design – is it built as intended?

SYSTEMS ENGINEERING "V" DIAGRAM

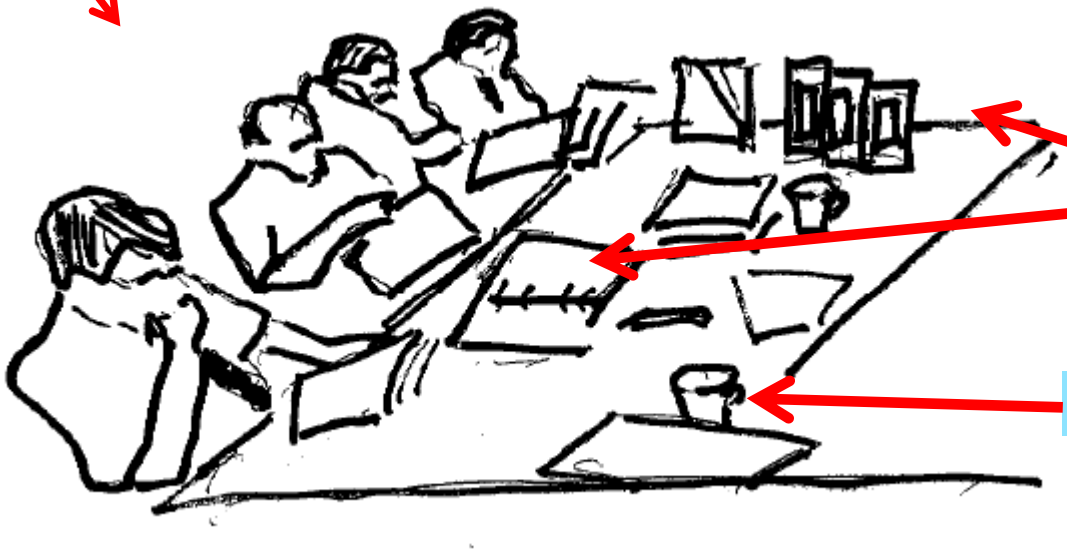


Tiers and Boundaries

- Provide:
  - Clear boundaries
  - Clear ownership
- Verification Level Definition
- Tiers provide complete scope definition
  - Told what needs to be done
  - Provide what needs to be done to next tier down
- Tiers are scoped to drive **valid** design – design occurs at each tier
  - Decide how to implement at tier level
- Verification of valid design – is it built as intended?
- Tiers aligned to organizational structure

OLD WAY OF REVIEWING SYSTEMS

Everyone in the same room – that's good.

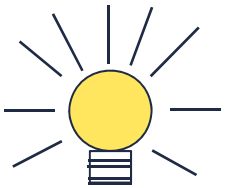


Documents – lots of them. Not so good.

Coffee – necessary

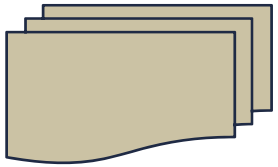
"On page 2500, the valve shall be open when Starting. On page 300, the valve shall be closed when xxx limit is exceeded . These conditions are not mutually exclusive. "

CURRENT WAY OF REVIEWING SYSTEMS



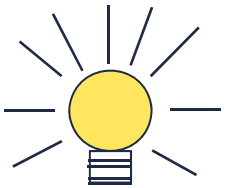
Ideas and Designs

Requirements



Stashed in a RM system – essentially a big spreadsheet

CURRENT WAY OF REVIEWING SYSTEMS



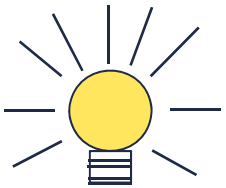
Ideas and Designs

Copied into PowerPoint Presentations

Requirements

Stashed in a RM system – essentially a big spreadsheet

CURRENT WAY OF REVIEWING SYSTEMS

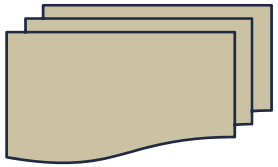


Ideas and Designs

Copied into PowerPoint Presentations

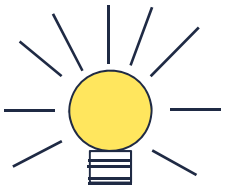
Stashed in folder somewhere – hopefully on a shared drive

Requirements



Stashed in a RM system – essentially a big spreadsheet

CURRENT WAY OF REVIEWING SYSTEMS



Ideas and Designs

Copied into PowerPoint Presentations

Stashed in folder somewhere – hopefully on a shared drive

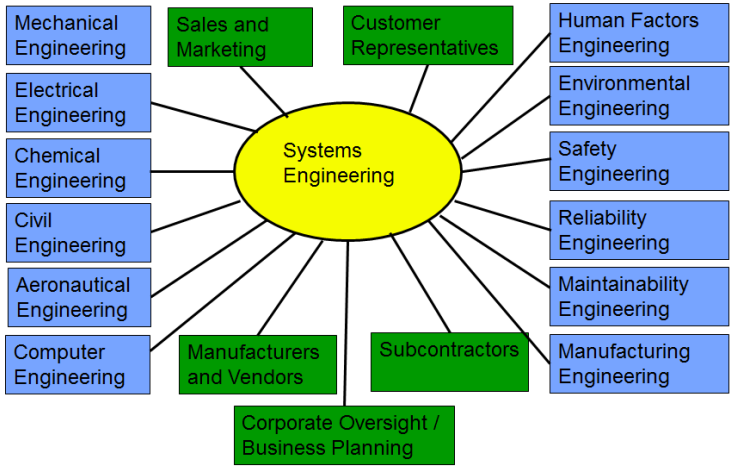
Presented in long meetings and conference calls

Requirements

Stashed in a RM system – essentially a big spreadsheet

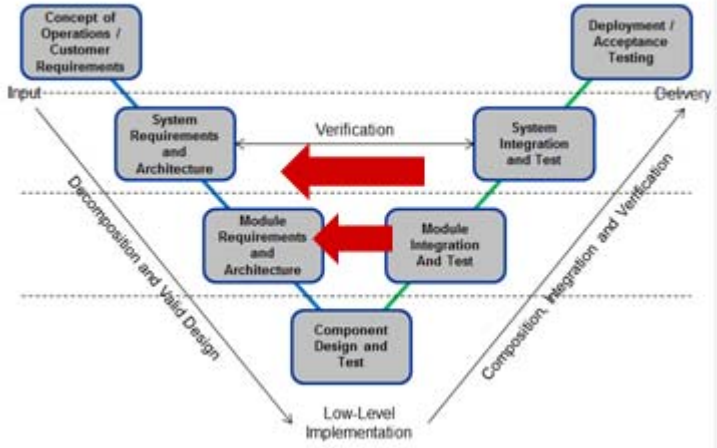
Is there a better way?

SYSML - SE AND MBSE



SE is multi-disciplinary

With SysML and MBSE – different discipline specific views of the system are maintained in the same model.

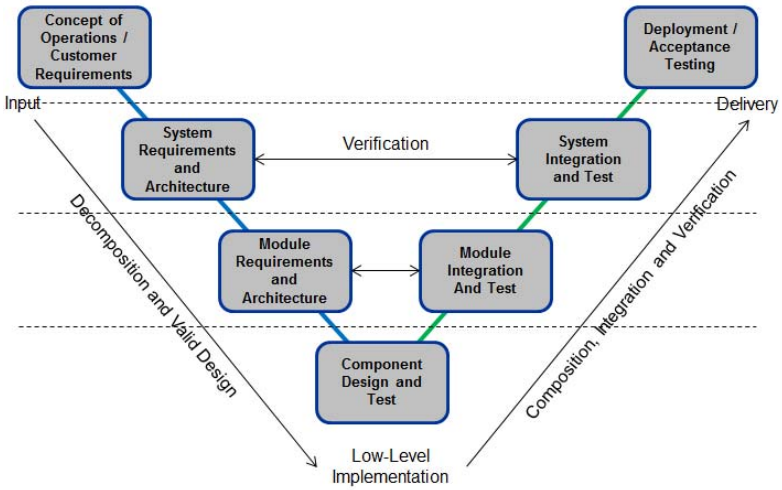


SE is applied throughout the project life cycle

With SysML and MBSE, designs, rationales, data, are all correlated and maintained in the same model.

SYSML - SE AND MBSE

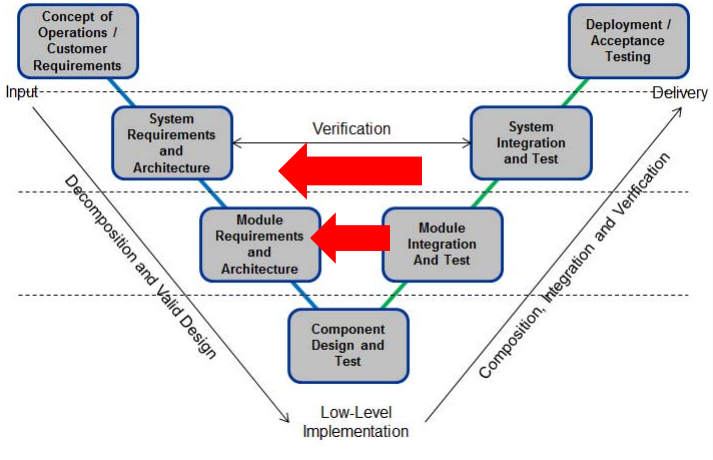
Traditional Life Cycle “V”



Time ➡

MBSE approach

System and Subsystem Validation are done by model as early as possible



Move essential analysis and learning “to the left”!

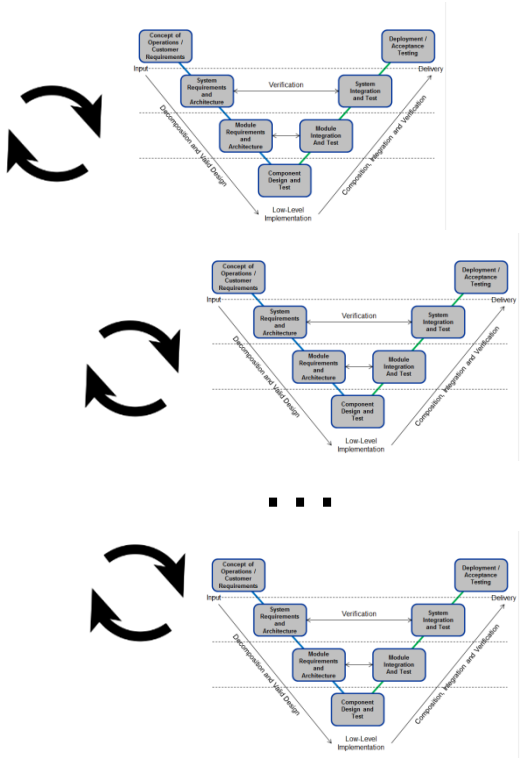
SYSML – SE, MBSE, AND AGILE METHODS

MBSE is compatible with and enables Agile development of hardware systems

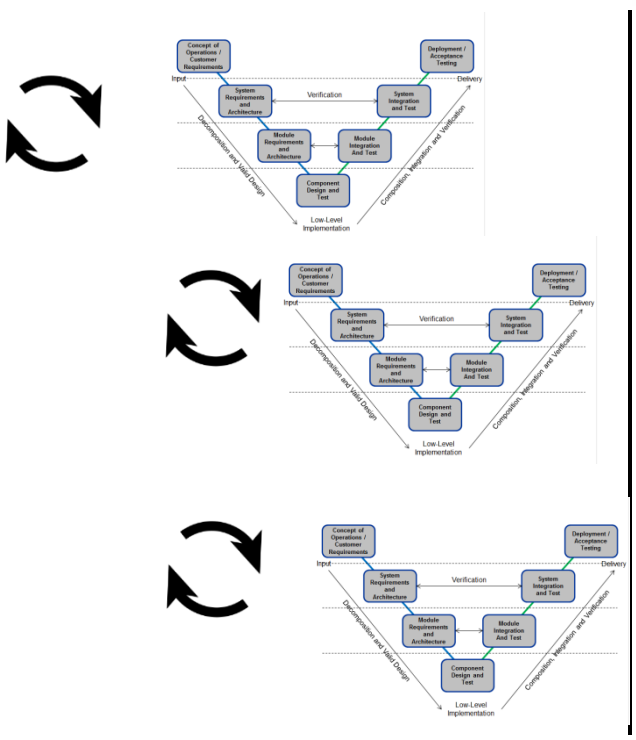
System

Subsystems

Continuous Integration of System and Subsystem Models



PI



PI

The “V”’s are broken up into smaller segments, and simulated early and often.

. . .

SYSML AND SOFTWARE ENGINEERING

Software Systems *are* systems!

- All of the elements we will discuss for working on a general systems design have immediate application for software architecture and design.

SYSML AND SOFTWARE ENGINEERING

Software Systems *are* systems!

- All of the elements we will discuss for working on a general systems design have immediate application for software architecture and design.
- SysML has elements that are particularly useful for designing software that interacts with hardware systems.
- SysML has language elements that are specifically designed for creating, tracing, and managing requirements for highly regulated safety critical systems.

SYSML AND SOFTWARE ENGINEERING

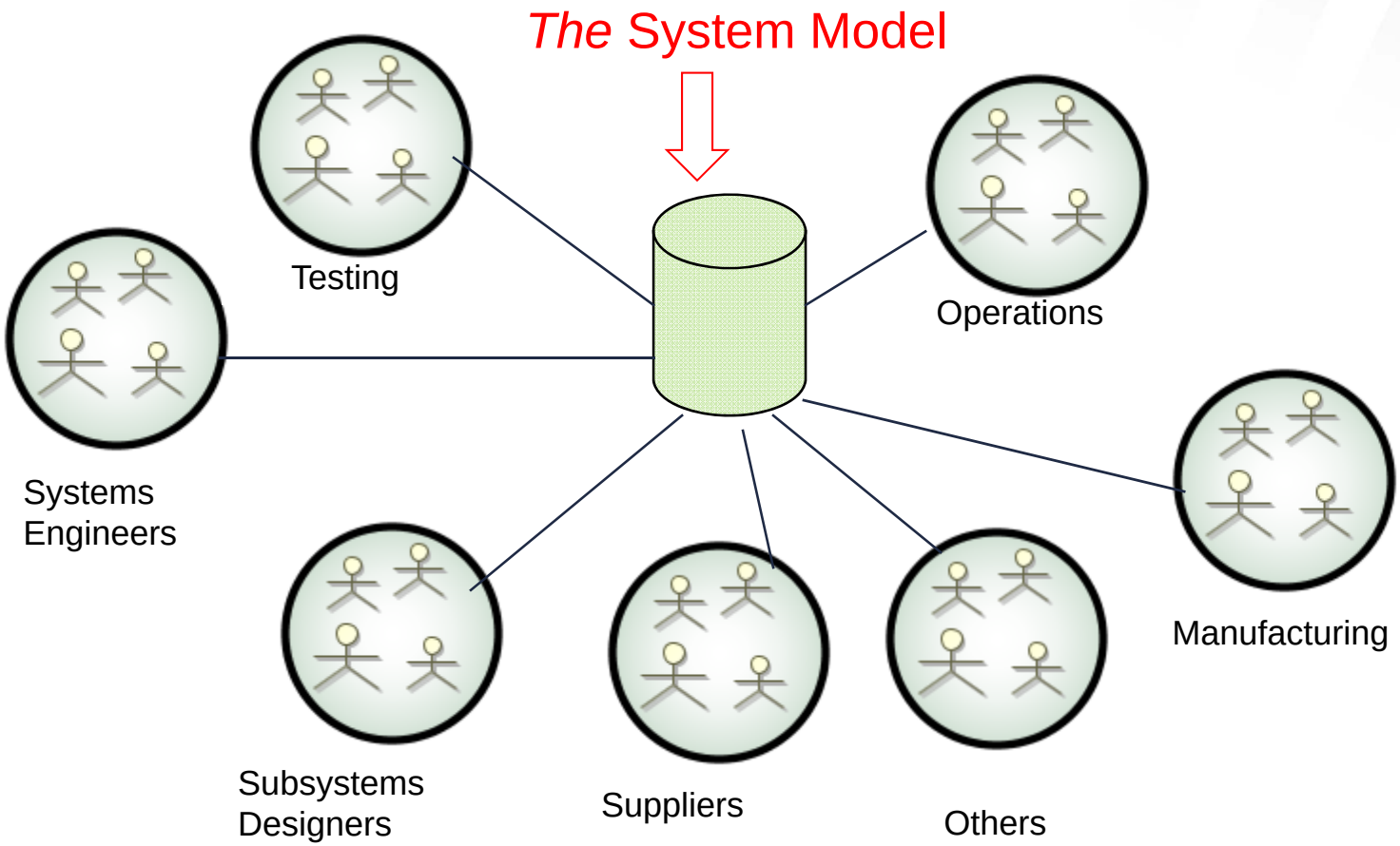
Software Systems *are* systems!

- All of the elements we will discuss for working on a general systems design have immediate application for software architecture and design.
- SysML has elements that are particularly useful for designing software that interacts with hardware systems.
- SysML has language elements that are specifically designed for creating, tracing, and managing requirements for highly regulated safety critical systems.

There are other software modeling languages that have specific purposes, just as there are other hardware modeling languages that have specific purposes. These other modeling languages do not conflict with SysML models, and in fact can be tightly integrated with SysML in various ways.

SYSML – SE AND MBSE – QUESTIONS?

Everyone works with …



GO BEYOND

WHAT IS SYSML?

SO…WHAT IS SYSML?

SysML is a standards based, graphical, object oriented, modeling language for representing the components of a system and their inter-relationships.

- It results in a “descriptive system model”, with multiple views showing various aspects of a system.

SO…WHAT IS SYSML?

SysML is a standards based, graphical, object oriented, modeling language for representing the components of a system and their inter-relationships.

- It results in a “descriptive system model”, with multiple views showing various aspects of a system.
- There are SysML diagrams (views) for representing
  - Structure
  - Behavior
  - Requirements
  - Parametrics

You will be able to recognize and use the various SysML diagrams after this course

WHAT IS SYSML?

- **Standards Based** - defined by the Object Management Group (OMG)
- **Graphical / Visual** - diagrams are used to specify contents of a SysML model
- **Object Oriented** -
- **Modeling** -
- **Language** -

WHAT IS SYSML?

- **Standards Based** - defined by the Object Management Group (OMG)
- **Graphical / Visual** - diagrams are used to specify contents of a SysML model
- **Object Oriented** - leads to reusable design elements and a decomposition into multiple abstraction layers.
- **Modeling** - SysML is best used for “descriptive modeling” of systems
- **Language** - Formal syntax removes ambiguity, adds clarity.

WHAT IS SYSML?

- **Standards Based** - defined by the Object Management Group (OMG)
- **Graphical / Visual** - diagrams are used to specify contents of a SysML model
- **Object Oriented** - leads to reusable design elements and a decomposition into multiple abstraction layers.
- **Modeling** - SysML is best used for “descriptive modeling” of systems
- **Language** - Formal syntax removes ambiguity, adds clarity.

The system involved may be a software system, a physical system, a combination of software and hardware, or a system of systems.

Note that the standard syntax enables interoperability between tools. Standards compliance by tool vendors is converging over time, with Cameo Systems Modeler leading in this aspect.

SYSML IS STANDARDS BASED

SYSML extends UML 2.0

UML 2.0 – a visual language used primarily by software programmers working on object oriented system designs

- Software terms do not resonate with many engineering communities

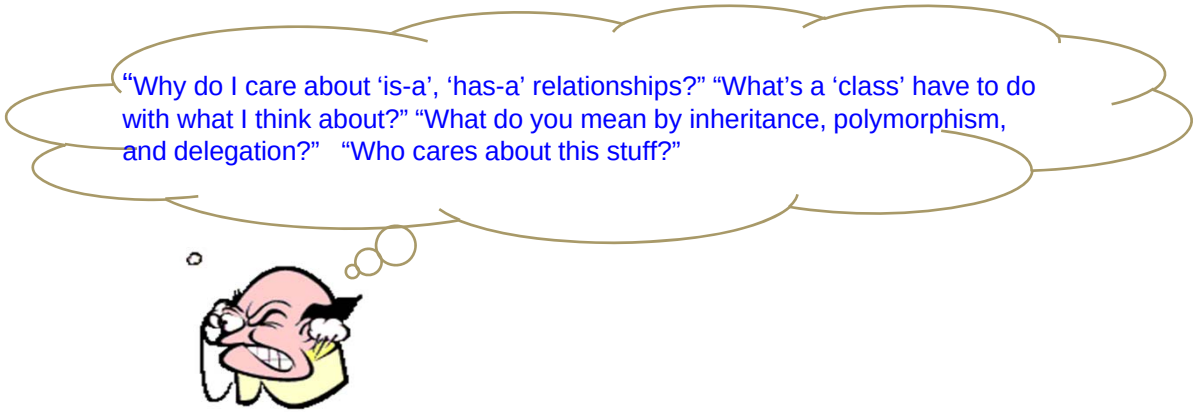


Image from Microsoft PowerPoint 2010Animation library

.

SYSML IS STANDARDS BASED

SYSML extends UML 2.0

UML 2.0 – a visual language used primarily by software programmers working on object oriented system designs

- Software terms do not resonate with many engineering communities

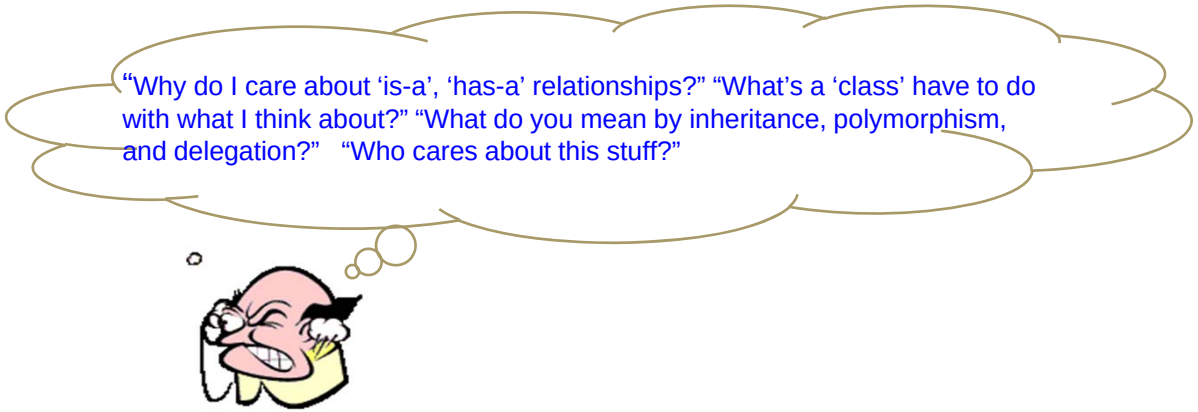


Image from Microsoft PowerPoint 2010Animation library

Yet UML provides infrastructure that is easy to extend for other purposes.

SYSML IS STANDARDS BASED

- UML for Systems Engineering RFP – 2003 by OMG "Object Management Group" Established the requirements for a system modeling language
  - SYSML is an industry / consortium response to the RFP
- SysML RFP - March 2003
  - Solicited submissions that specify a customization of UML for Systems Engineering (SysML)

- American Systems Corporation
- ARTISAN Software Tools
- BAE Systems
- The Boeing Company
- Deere & Company
- EADS Astrium GmbH
- EmbeddedPlus Engineering
- Eurostep Group AB
- Gentleware AG
- Georgia Institute of Technology
- I-Logix
- International Business Machines
- Lockheed Martin Corporation
- Mentor Graphics
- Motorola Inc.
- National Institute of Standards and Technology
- Northrup Grumman Corporation
- Oose.de Dienstleistungen fur innovative Informatik GmbH
- PivotPoint Technology Corporation
- Raytheon
- TelelogicAB
- THALES
- Vitech

SYSML IS STANDARDS BASED

A Customization of UML for Systems Engineering (SysML)

Initial RFP Specification Goals

- Support modeling a broad range of systems
- Hardware, software, data, personnel, procedures, and facilities
- Capture system information precisely and efficiently
- Allow for the analysis and evaluation of the modeled system
- Provide clear communication of systems information among stakeholders

SYSML IS STANDARDS BASED

UML reused for SysML

Actions

Activities

Classes

General Behavior

Information Flows

Interactions

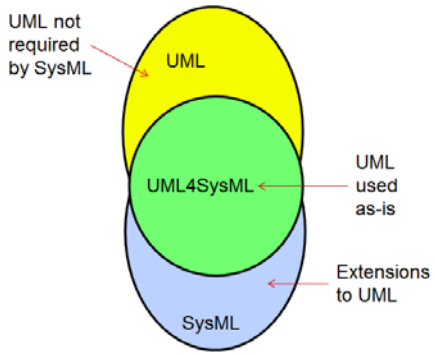
Models

Profiles

State Machines

Structures

Use Cases



Extensions to UML

SysML::Model Elements refactors and extends Kernel

SysML:: Blocks reuses Composite structures & Model Elements

SysML::Constraint Blocks extends Blocks

SysML::Ports & Flows extends UML Ports

SysML::Activities extends UML Activities

SysML::Allocations extends UML dependencies

SysML::Requirements extends Classes and dependencies

WHAT IS SYSML? UML + "SYSTEM STUFF"

SysML is an extension of UML, a graphical language originally designed for modeling object oriented software systems.

- Some UML concepts were renamed to make the language more accessible to systems engineers without a software background.
- "Class" = "Block", "Object" = "Part"

WHAT IS SYSML? UML + "SYSTEM STUFF"

SysML is an extension of UML, a graphical language originally designed for modeling object oriented software systems.

- Some UML concepts were renamed to make the language more accessible to systems engineers without a software background.
- "Class" = "Block", "Object" = "Part"
- Physical units added – "kg", "m", "sec", "volt", etc.
- Requirement objects added with tracing relationships between requirements and model elements supported.
- Parametric descriptions added – to encapsulate constraints and equations.

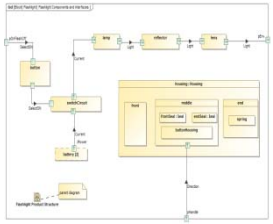
WHAT IS SYSML? UML + "SYSTEM STUFF"

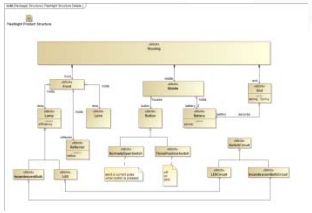
SysML is an extension of UML, a graphical language originally designed for modeling object oriented software systems.

- Some UML concepts were renamed to make the language more accessible to systems engineers without a software background.
- "Class" = "Block", "Object" = "Part"
- Physical units added – "kg", "m", "sec", "volt", etc.
- Requirement objects added with tracing relationships between requirements and model elements supported.
- Parametric descriptions added – to encapsulate constraints and equations.
- Because of its ability to trace requirements, and anchor impact analysis studies, SysML may be preferable to UML for flight or safety critical software specifications.

SYSML IS A VISUAL / GRAPHICAL LANGUAGE

SysML is a language based on graphical representations of key system elements.



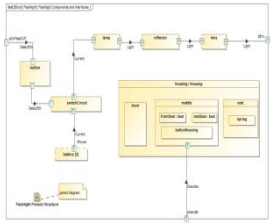


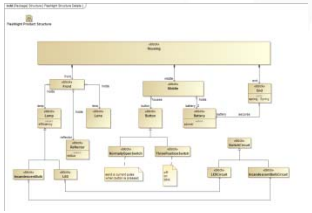
"A picture is worth 1000 words"

This is measurably true, for many types of design detail.

SYSML IS A VISUAL / GRAPHICAL LANGUAGE

SysML is a language based on graphical representations of key system elements.





"A picture is worth 1000 words"

This is measurably true, for many types of design detail.

Note that "graph" is a play on words here - The language is graphical, and the resulting model contains a formal representation of a mathematical "graph" – with nodes and edges.

SysML has several diagram types which provide different views of the elements of the system.

WHAT IS SYSML? – DIAGRAM TYPES

SysML Diagram Types --- Views of:

Behavior
- Use Case Diagram
- Activity Diagram
- Sequence Diagram
- State Machine Diagram

Structure
- Block Definition Diagram
- Internal Block Diagram
- Package Diagram

Requirements
- Requirements Diagram
- Requirements Table

Parametrics
- Parametric Diagram

WHAT IS SYSML? – DIAGRAM TYPES

SysML Diagram Types --- Views of:

Behavior
- Use Case Diagram
- Activity Diagram
- Sequence Diagram
- State Machine Diagram

Structure
- Block Definition Diagram
- Internal Block Diagram
- Package Diagram

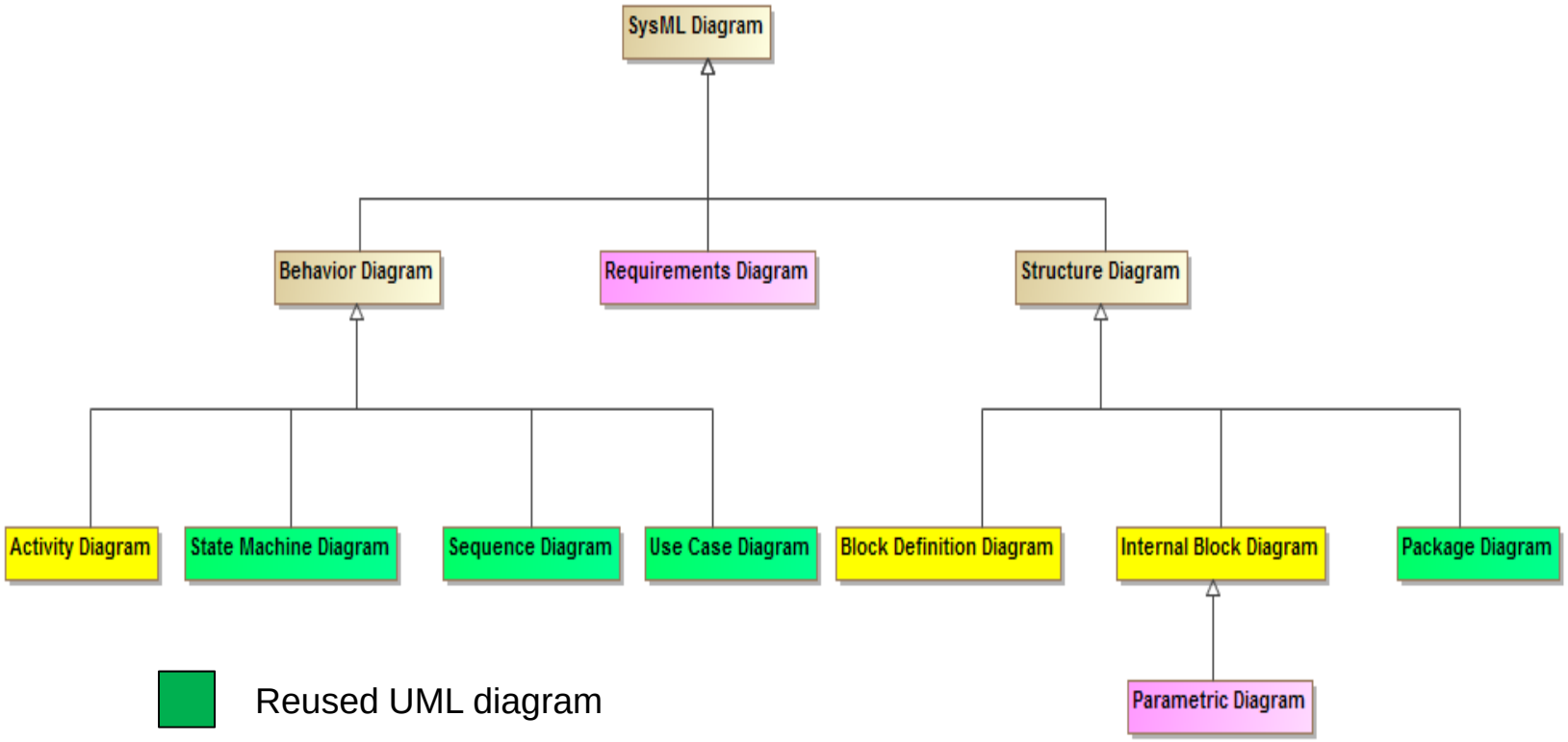
Requirements
- Requirements Diagram
- Requirements Table

Parametrics
- Parametric Diagram

Its NOT just the diagrams or we could use Visio or PowerPoint for modeling!

The diagrams combine to create an interlinked set of elements – a model stored in a repository with consistent naming and strong typing and language syntax enforced. The value of this for elucidating cross cutting relationships across the system cannot be emphasized enough! This is perhaps the key contribution of SysML to SE.

SYSML DIAGRAM TYPES



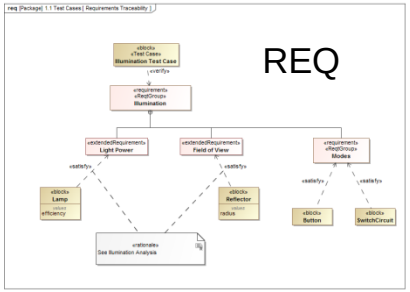
Reused UML diagram

New SysML diagram

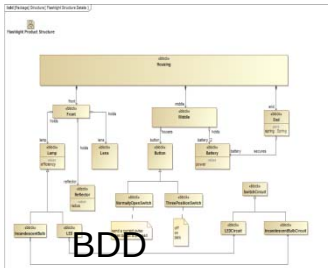
Modified UML diagram

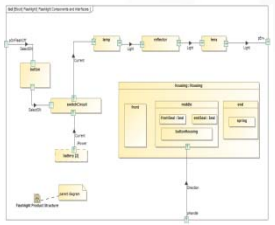
SYSML DIAGRAM TYPES

Requirements

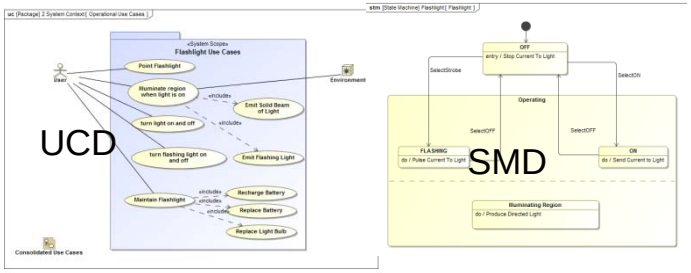


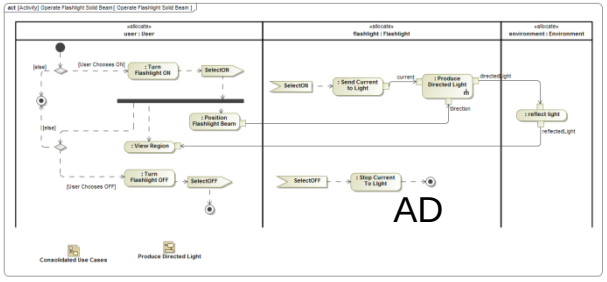
Structure



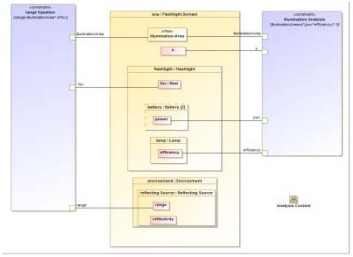


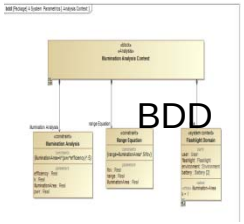
Behavior





Parametrics





We will show examples of diagram types and work with them in following sections.

SYSML DIAGRAM TYPES - ELEMENTS

Requirement Diagram

- Requirement
- Contains / refine / derive / satisfy

Use Case Diagram

- Use Case “Bubble”
- System Boundary or Scope
- Associations
- Actors

Activity Diagram

- Activity
- Fork / Join
- Decision / Guard/ Merge
- Start Activity / End Activity
- Control Flow
- Data Flow
- In / out/ in-out pin

State Machine Diagram

- State
- Entry Action / Do Action / Exit Action
- Orthogonal Region / Sub Machine
- Transition / Guard

Sequence Diagram

- Covered in another course

Block Definition Diagram

- Block
- Is-a, has-a, directed associations

Internal Block Diagram

- Parts
- Ports, types, interfaces, connections, flows

Parametric Diagram

- Covered in another course

Not all diagrams are fully covered in the course labs due to limitations on time.

WHAT IS SYSML? ITS DESCRIPTIVE MODELING

SysML is best used for Descriptive Modeling.

- Descriptive Modeling is different than physics based modeling.

WHAT IS SYSML? ITS DESCRIPTIVE MODELING

SysML is best used for Descriptive Modeling.

- Descriptive Modeling is different than physics based modeling.
- Physical models involve calculations to produce curves of important physical system parameters under various constraints.

WHAT IS SYSML? ITS DESCRIPTIVE MODELING

SysML is best used for Descriptive Modeling.

- Descriptive Modeling is different than physics based modeling.
- Physical models involve calculations to produce curves of important physical system parameters under various constraints.
- Descriptive modeling, on the other hand, *describes* these calculations along with explicitly showing the flows of energy, momentum, fluids, electrical signals, and data, between the components of a large system. It highlights the connections between system components and what flows between the components.

WHAT IS SYSML? ITS DESCRIPTIVE MODELING

SysML is best used for Descriptive Modeling.

- Descriptive Modeling is different than physics based modeling.
- Physical models involve calculations to produce curves of important physical system parameters under various constraints.
- Descriptive modeling, on the other hand, *describes* these calculations along with explicitly showing the flows of energy, momentum, fluids, electrical signals, and data, between the components of a large system. It highlights the connections between system components and what flows between the components.

SysML shows “Nodes and Edges” in graphs (diagrams) that represents various views of the system.

WHAT IS SYSML? SUMMARY

SYSML is a formalized, visual language for specifying a *descriptive model* of a system based on an extension of UML 2.0.

It is NOT a methodology or workflow by itself.

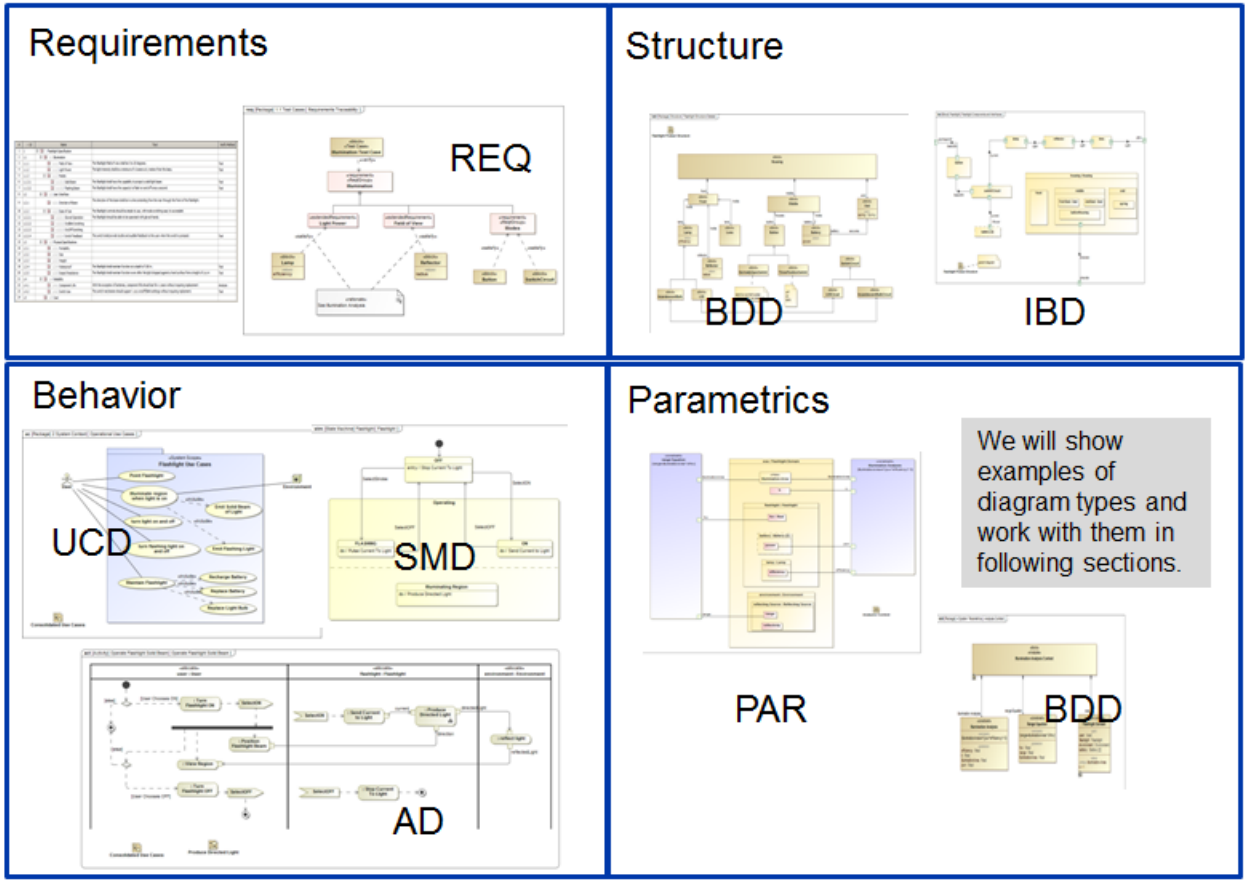
It is a key enabler for Model Based System Engineering

There are many ways to use SYSML.

WHAT IS SYSML? SUMMARY

- The visual elements of SYSML are translatable to internal structures that are amenable to formal methods and static analysis techniques.
- With tool support, SYSML drawings can produce executable computer programs.
- The behavior of these programs can be used to verify model correctness, completeness, and to generate test sequence information for use in system verification and validation.

WHAT IS SYSML? QUESTIONS?



GO BEYOND

METHODOLOGY

A UNITED TECHNOLOGIES COMPANY

SIMPLIFIED SYSML METHODOLOGY

- There are several structured methodologies for using SysML on a systems engineering project. Some have tool support with wizards and templates for the method.
  - "Object Oriented Systems Engineering Method (OOSEM) - INCOSE
  - "IBM Harmony Process – IBM Rational Rhapsody",
  - "MagicGrid - Dassault / NoMagic – Cameo Systems Modeler"
  - "CORE" – Vitech - Genesys

SIMPLIFIED SYSML METHODOLOGY

- There are several structured methodologies for using SysML on a systems engineering project. Some have tool support with wizards and templates for the method.
  - "Object Oriented Systems Engineering Method (OOSEM) - INCOSE
  - "IBM Harmony Process – IBM Rational Rhapsody",
  - "MagicGrid - Dassault / NoMagic – Cameo Systems Modeler"
  - "CORE" – Vitech - Genesys
- These methodologies align with Systems Engineering processes.
- There are common tasks and steps in all these methods that follow the natural development of your thoughts about a system as you start to comprehend it.

SIMPLIFIED SYSML METHODOLOGY

- There are several structured methodologies for using SysML on a systems engineering project. Some have tool support with wizards and templates for the method.
  - "Object Oriented Systems Engineering Method (OOSEM) - INCOSE
  - "IBM Harmony Process – IBM Rational Rhapsody",
  - "MagicGrid - Dassault / NoMagic – Cameo Systems Modeler"
  - "CORE" – Vitech - Genesys
- These methodologies align with Systems Engineering processes.
- There are common tasks and steps in all these methods that follow the natural development of your thoughts about a system as you start to comprehend it.

This leads to a simplified set of tasks that I recommend for any SysML modeling effort and that I will present in this course.

SIMPLIFIED SYSML METHODOLOGY

1. The system is decomposed into distinct abstraction layers, starting with the systems *around* the system of interest, and the system itself.

SIMPLIFIED SYSML METHODOLOGY

1. The system is decomposed into distinct abstraction layers, starting with the systems *around* the system of interest, and the system itself.
2. The system is then decomposed into an assembly of subsystems. This may be done in multiple ways for analysis of design tradeoffs.

SIMPLIFIED SYSML METHODOLOGY

1. The system is decomposed into distinct abstraction layers, starting with the systems *around* the system of interest, and the system itself.
2. The system is then decomposed into an assembly of subsystems. This may be done in multiple ways for analysis of design tradeoffs.
3. Interfaces and flows (data, energy, signals, fluids, etc.) between the subsystems are then elucidated.

SIMPLIFIED SYSML METHODOLOGY

1. The system is decomposed into distinct abstraction layers, starting with the systems *around* the system of interest, and the system itself.
2. The system is then decomposed into an assembly of subsystems. This may be done in multiple ways for analysis of design tradeoffs.
3. Interfaces and flows (data, energy, signals, fluids, etc.) between the subsystems are then elucidated.
4. Behavior of the system and subsystems are specified and then simulated. This may occur in the SysML tool, or may involve other modeling environments connected or traced by the SysML model.

SIMPLIFIED SYSML METHODOLOGY

1. The system is decomposed into distinct abstraction layers, starting with the systems *around* the system of interest, and the system itself.
2. The system is then decomposed into an assembly of subsystems. This may be done in multiple ways for analysis of design tradeoffs.
3. Interfaces and flows (data, energy, signals, fluids, etc.) between the subsystems are then elucidated.
4. Behavior of the system and subsystems are specified and then simulated. This may occur in the SysML tool, or may involve other modeling environments connected or traced by the SysML model.
5. Each subsystem is then treated as its own system of interest and the process is repeated for the subsystems.

SIMPLIFIED SYSML METHODOLOGY

1. The system is decomposed into distinct abstraction layers, starting with the systems *around* the system of interest, and the system itself.
2. The system is then decomposed into an assembly of subsystems. This may be done in multiple ways for analysis of design tradeoffs.
3. Interfaces and flows (data, energy, signals, fluids, etc.) between the subsystems are then elucidated.
4. Behavior of the system and subsystems are specified and then simulated. This may occur in the SysML tool, or may involve other modeling environments connected or traced by the SysML model.
5. Each subsystem is then treated as its own system of interest and the process is repeated for the subsystems.
6. Requirements are written for each behavior of the system or subsystem and traced to the elements in the model that implement the behavior.

SIMPLIFIED SYSML METHODOLOGY

Starting with the system, and customer requirements, do the following …

Make 3 lists

- Outside actors – Stakeholders, Users and External systems
- Functions – what does the system need to do?
- Things – essential physical and logical structures in the system.

SIMPLIFIED SYSML METHODOLOGY

Starting with the system, and customer requirements, do the following …

Make 3 lists
- Outside actors – Stakeholders, Users and External systems
- Functions – what does the system need to do?
- Things – essential physical and logical structures in the system.

➡ Make Use Case Diagrams

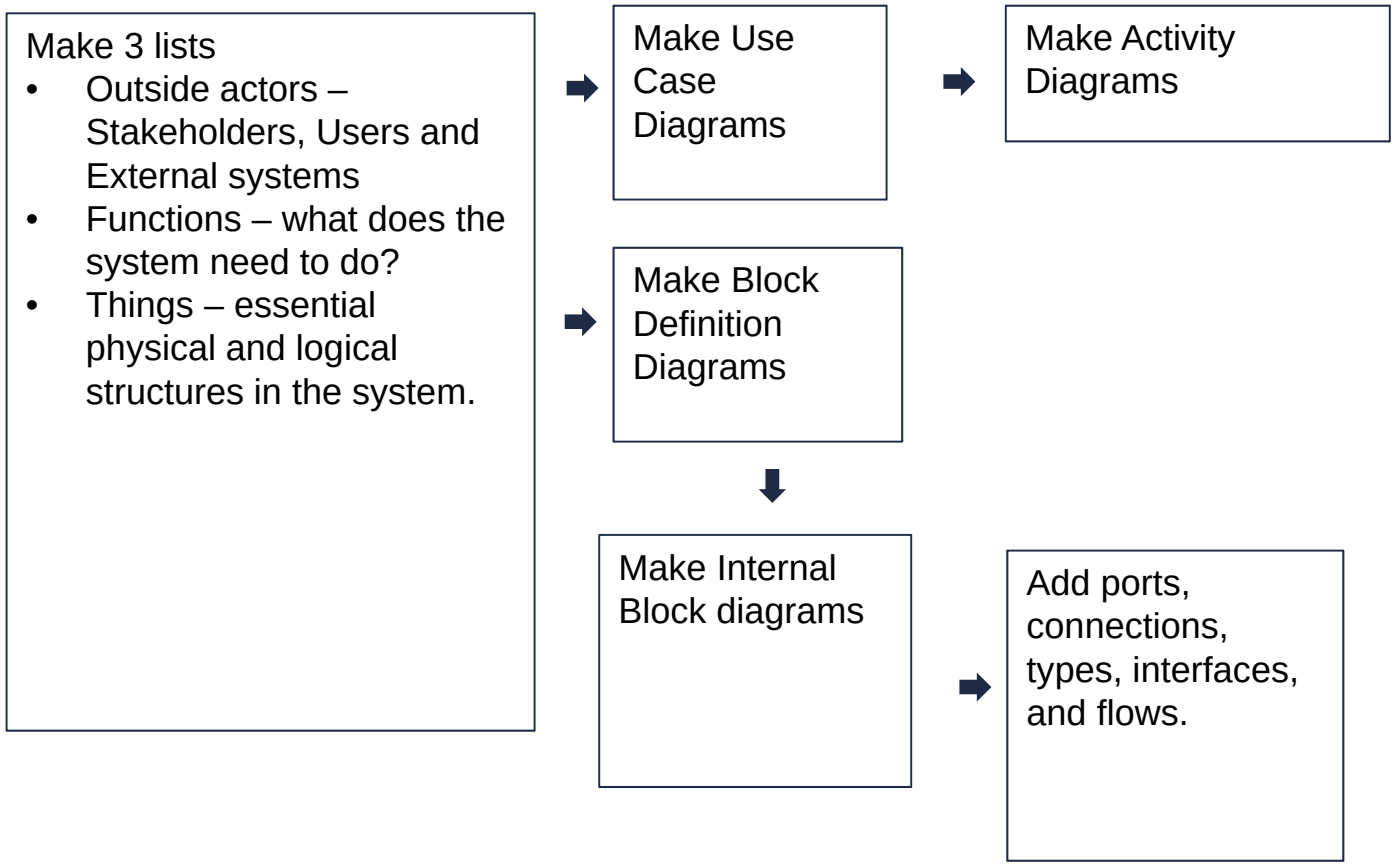
➡ Make Block Definition Diagrams

⬇

Make Internal Block diagrams

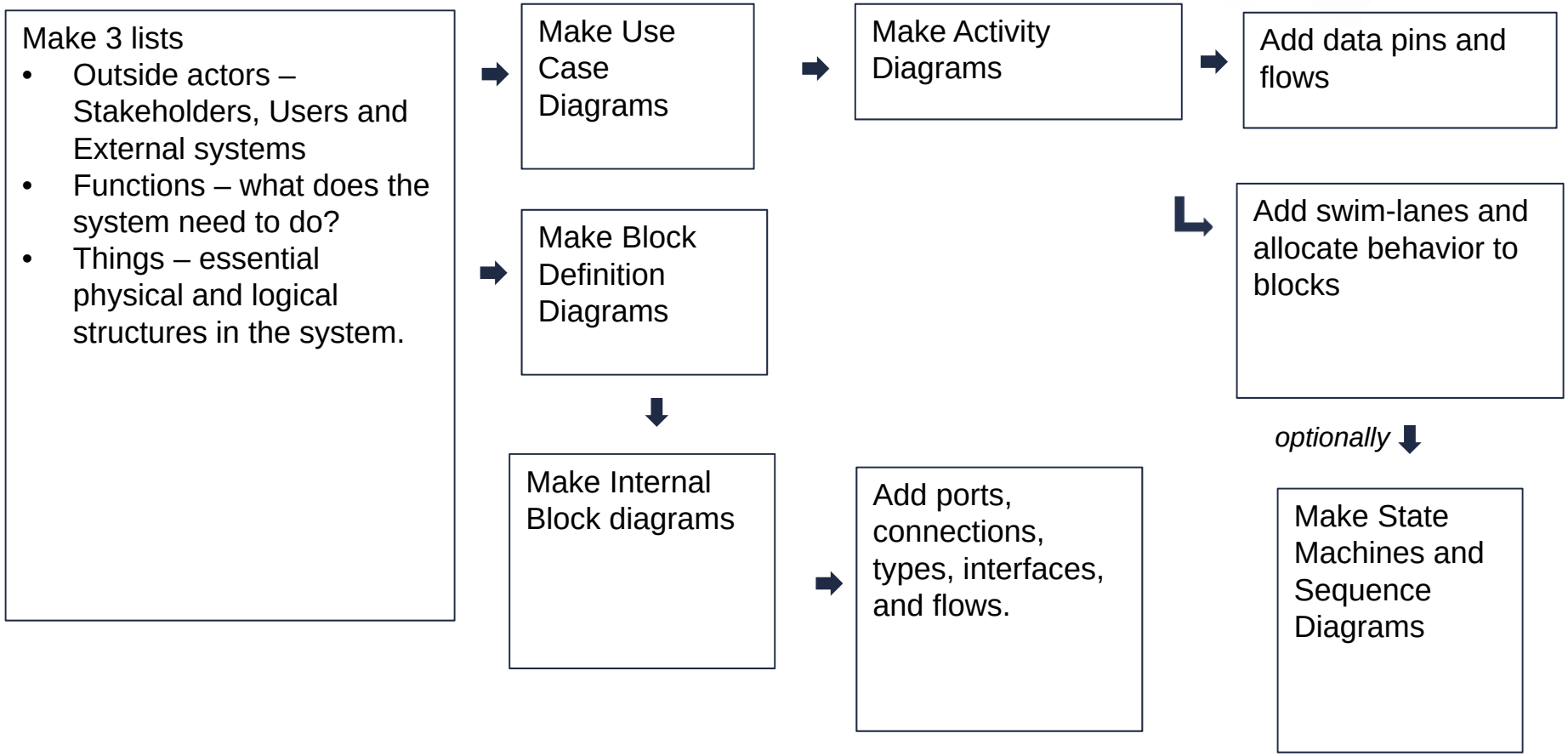
SIMPLIFIED SYSML METHODOLOGY

Starting with the system, and customer requirements, do the following …



SIMPLIFIED SYSML METHODOLOGY

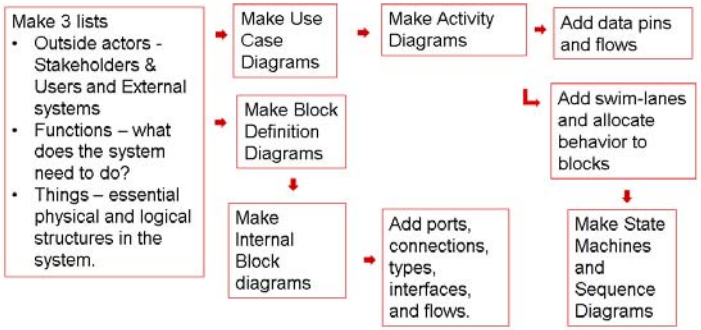
Starting with the system, and customer requirements, do the following …



SIMPLIFIED SYSML METHODOLOGY

For each block or part shown in the IBDs, repeat the same steps that were done for the system. For all elements, trace to requirements and associated tests.

System



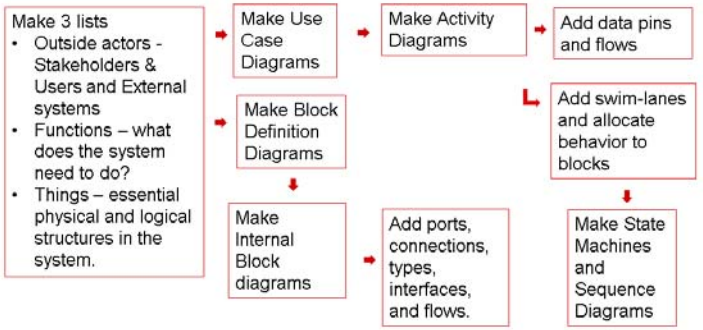
Trace requirements to system elements

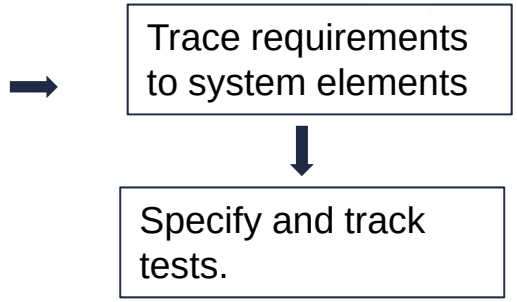
Specify and track tests.

SIMPLIFIED SYSML METHODOLOGY

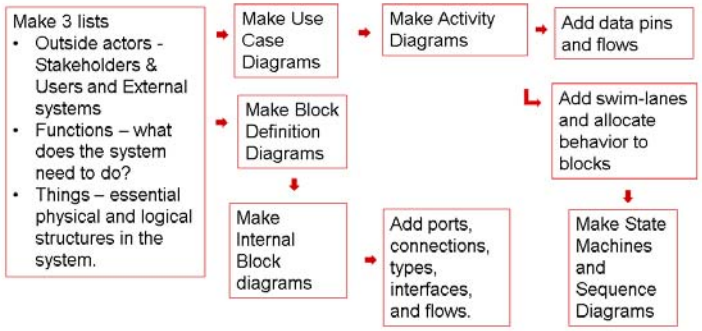
For each block or part shown in the IBDs, repeat the same steps that were done for the system. For all elements, trace to requirements and associated tests.

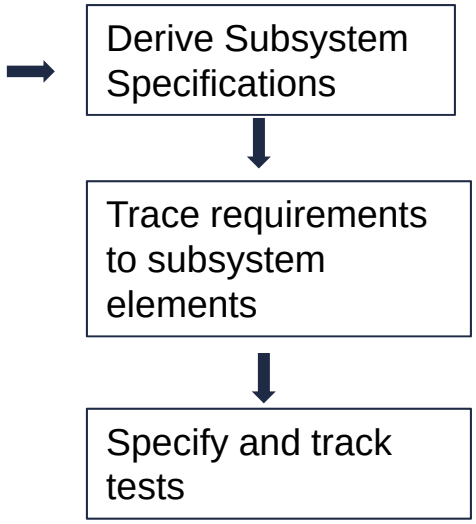
System



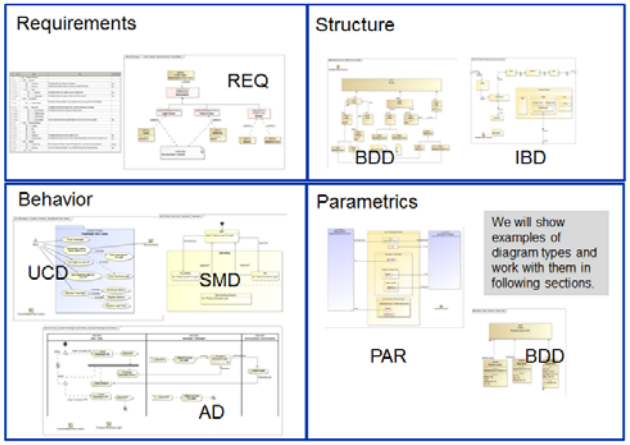


Subsystems

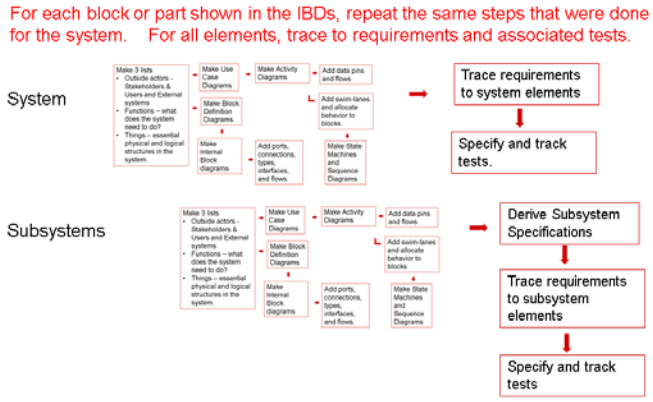




SIMPLIFIED SYSML METHODOLOGY – QUESTIONS?



- Depending upon how the models will be used, there are other steps to connect model elements with engineering analysis artifacts, summarize test results, connect constraints and calculations and set up special views of the system.



- The steps to get the main diagrams drawn and the order that work occurs will be the essentially same in all the methodologies that use SysML and common to all the SysML tools that are available.

GO BEYOND

EXAMPLE SYSML MODEL

A UNITED TECHNOLOGIES COMPANY

LAB – DIVER'S FLASHLIGHT MODEL - PURPOSE

All models should have a purpose. You should be able to state this clearly before you start to model or you are likely to wander during the creation of the model.

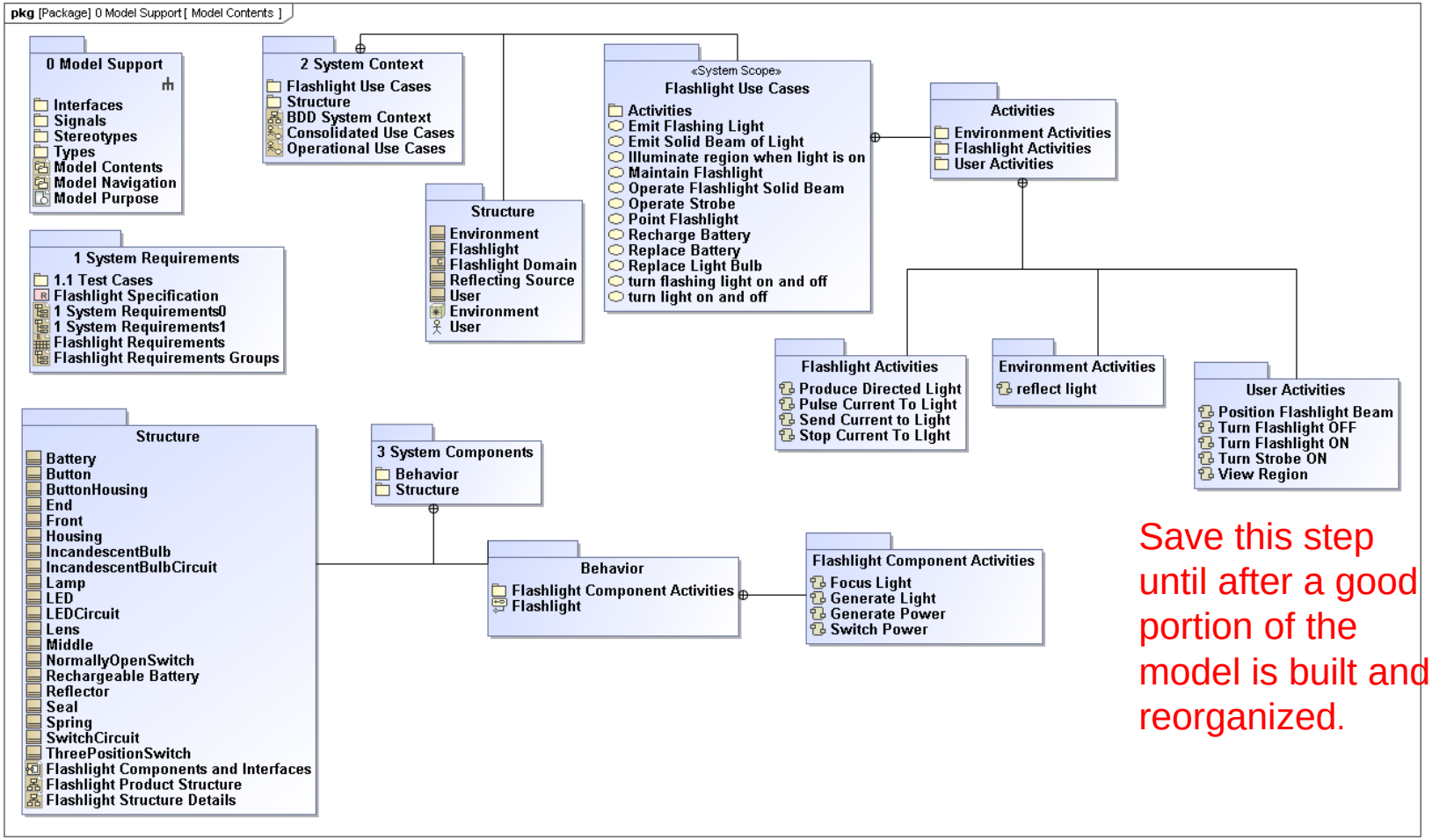
The Free Form Diagram in Cameo Systems Modeler can serve as a good place to note the purpose of the model.

We will create this model in labs, step by step.

This model is an extension of a model that Sandy Friedenthal presented at the Dassault/NoMagic MBSE Cyber Experience 2019 conference and is used with his permission.

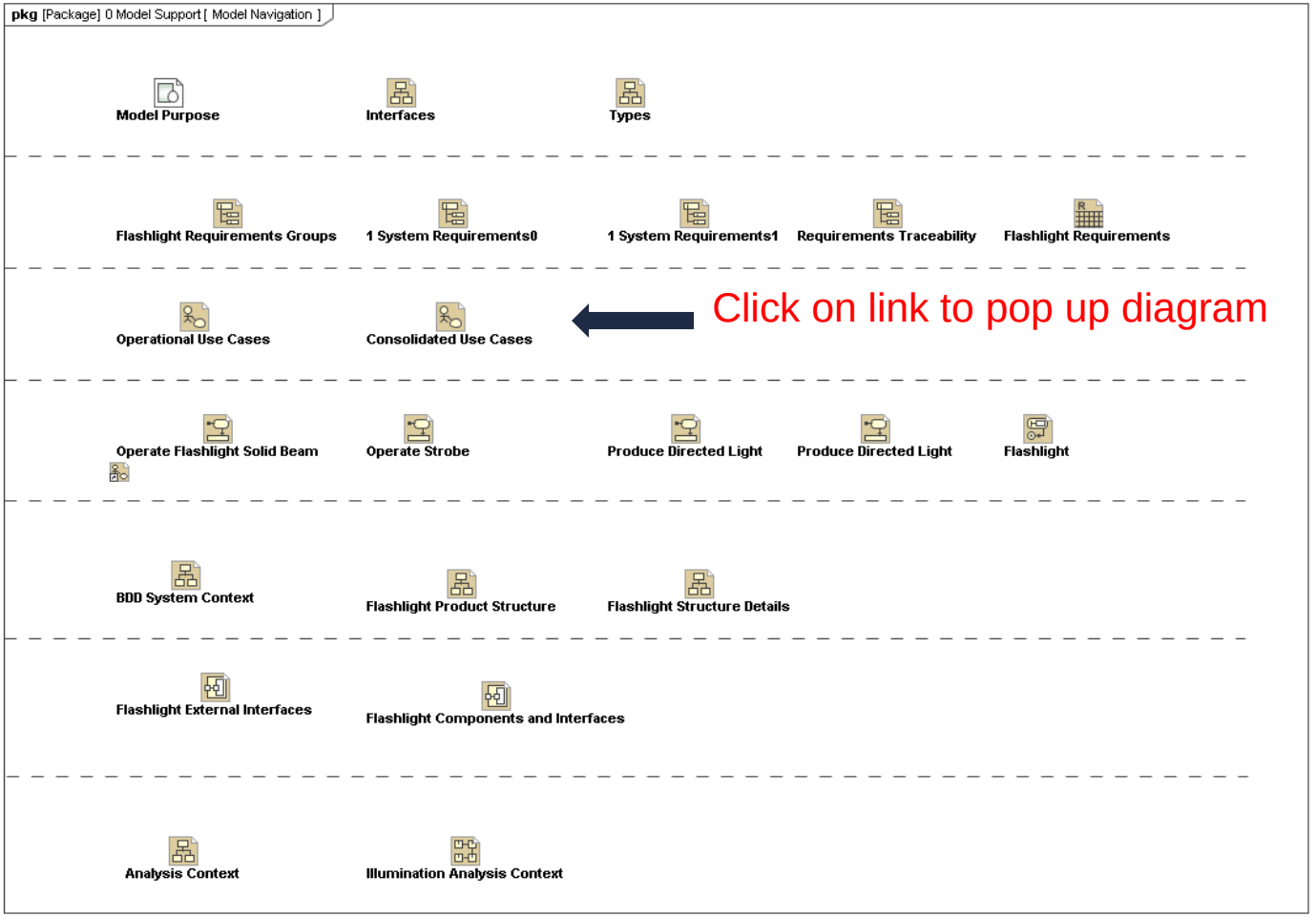
DIVER'S FLASHLIGHT MODEL - CONTENTS

A package diagram can be useful for explaining where things are located in the model.



DIVER'S FLASHLIGHT MODEL - NAVIGATION

A package diagram containing diagram links can assist in moving around the model.



DIVER'S FLASHLIGHT MODEL – REQUIREMENTS TABLE

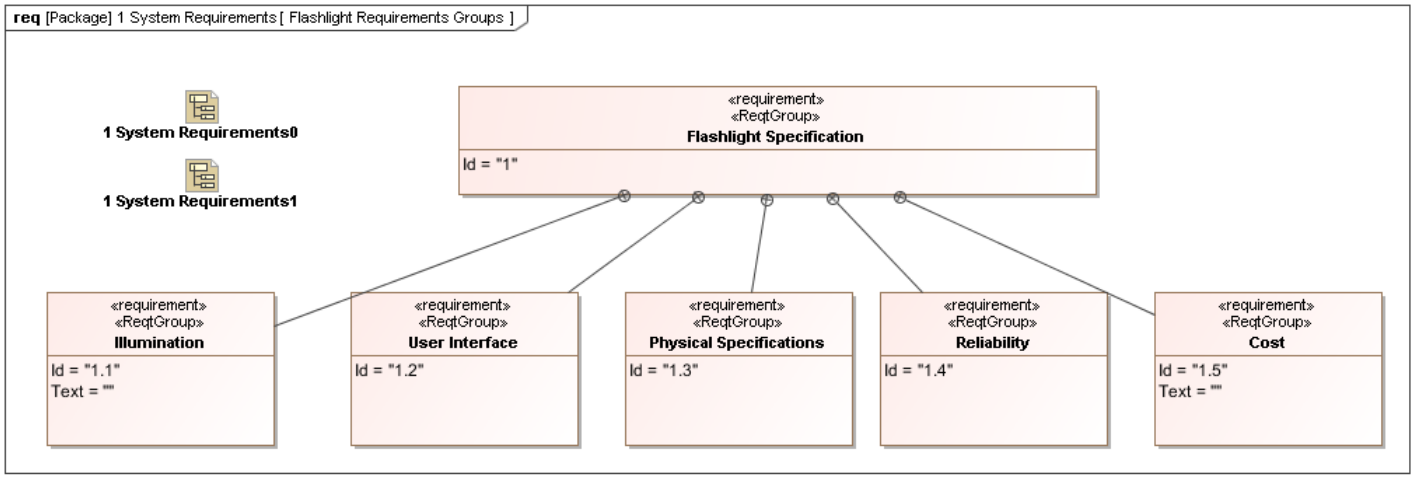
If there are customer requirements at the start, import them into the model. These came from an Excel spreadsheet. Using the DataHub plugin, requirements can be imported from any RecIF compliant RM system.

#	△ Id	Name	Text	Verify Method
1	1	1 Flashlight Specification		
2	1.1	1.1 Illumination		
3	1.1.1	1.1.1 Field of View	The flashlight field of view shall be 0 to 20 degrees.	Test
4	1.1.2	1.1.2 Light Power	The light intensity shall be a minimum of x lumens at y meters from the lamp.	Test
5	1.1.3	1.1.3 Modes		
6	1.1.3.1	1.1.3.1 Solid Beam	The flashlight shall have the capability to project a solid light beam.	Test
7	1.1.3.2	1.1.3.2 Flashing Beam	The flashlight shall have the capacity to flash on and off once a se	
8	1.2	1.2 User Interface		
9	1.2.1	1.2.1 Direction of Beam	The direction of the beam shall be in a line extending from the rea	
10	1.2.2	1.2.2 Ease of Use	The flashlight controls should be simple to use, with mode switchin	
11	1.2.2.1	1.2.2.1 Gloved Operation	The flashlight should be able to be operated with gloved hands.	
12	1.2.2.2	1.2.2.2 On/Blink Switching		
13	1.2.2.3	1.2.2.3 On/Off Switching		
14	1.2.2.4	1.2.2.4 Switch Feedback	The switch shall provide tactile and audible feedback to the user w	
15	1.3	1.3 Physical Specifications		
16	1.3.1	1.3.1 Portability		
17	1.3.2	1.3.2 Size		
18	1.3.3	1.3.3 Weight		
19	1.3.4	1.3.4 Waterproof	The flashlight shall maintain function at a depth of 100 m.	Test
20	1.3.5	1.3.5 Impact Resistance	The flashlight shall maintain function even after the light dropped against a hard surface from a height of yyy m	Test
21	1.4	1.4 Reliability		
22	1.4.1	1.4.1 Component Life	With the exception of batteries, component life should last for x years without requiring replacement	Analysis
23	1.4.2	1.4.2 Switch Use	The switch mechanism should support yyyy on/off/blink settings without requiring replcement.	Test
24	1.5	1.5 Cost		

If a text "Name" is not imported then create one as it helps with visualization and organization

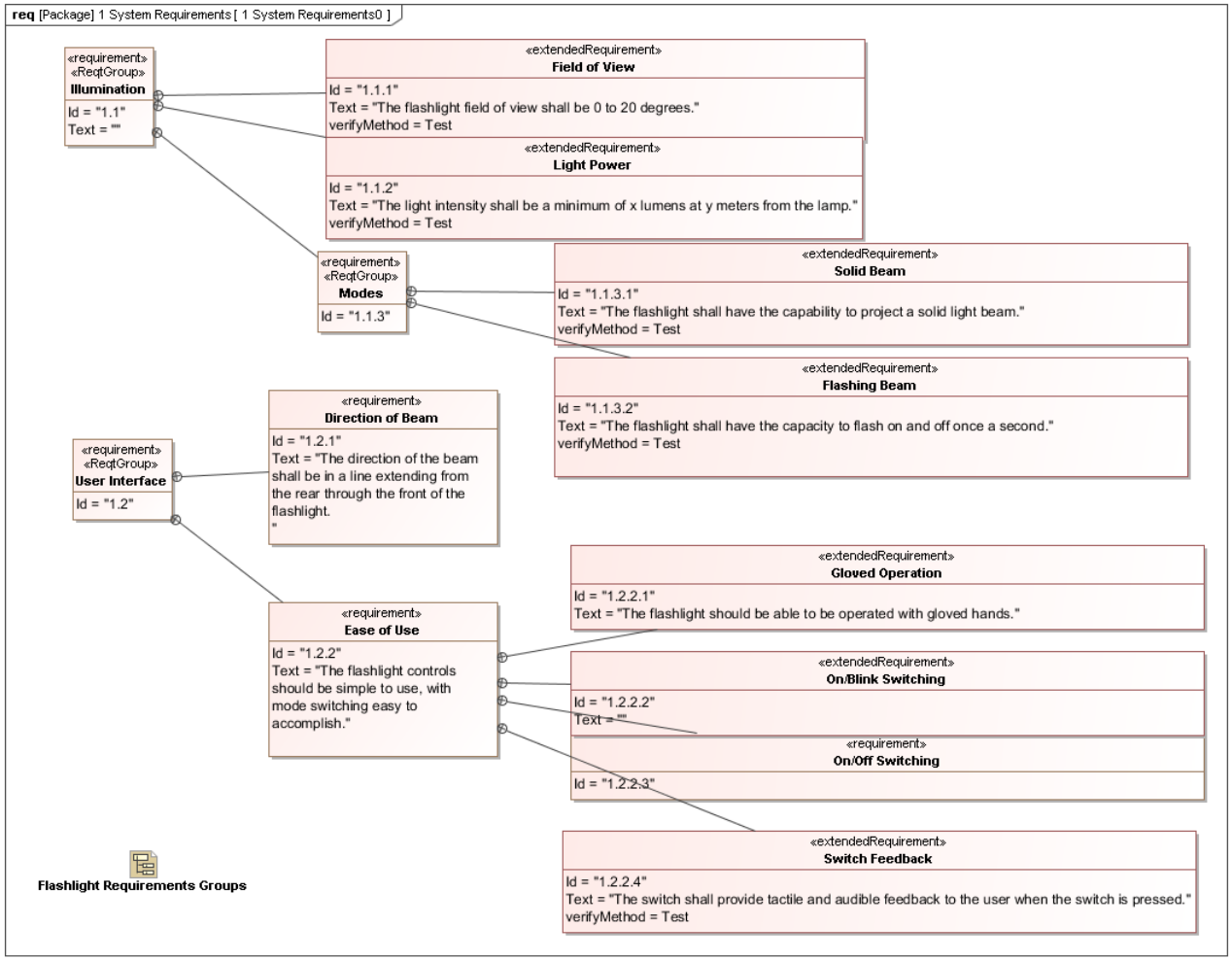
DIVER'S FLASHLIGHT MODEL – REQUIREMENTS DIAGRAMS

The structure of the requirements can be viewed in a requirements diagram. Create by drag and drop, and then use "Display All Paths" to quickly establish relationships. Refactor as necessary to impose a different hierarchy than imported to aid impact analysis.



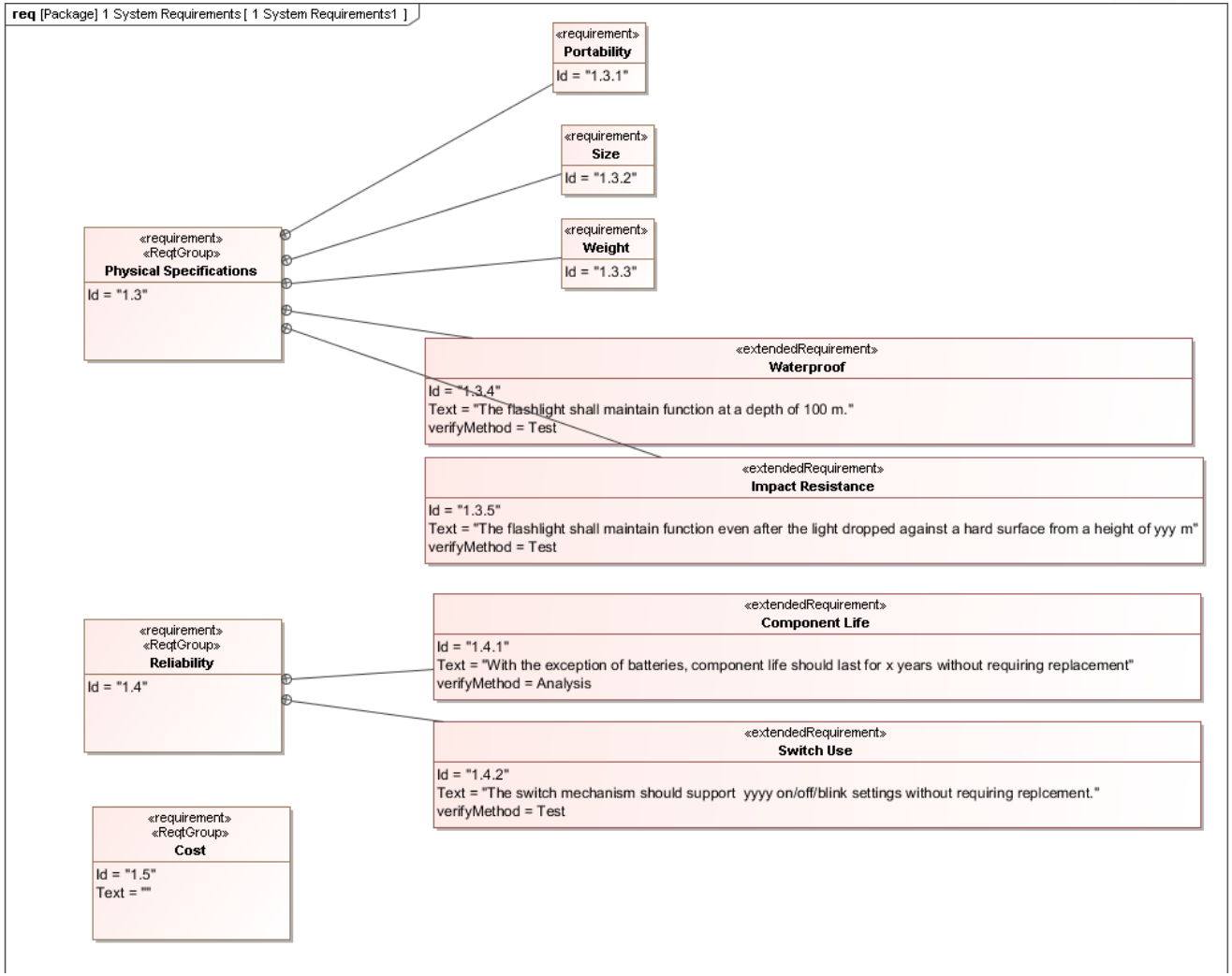
Group Requirements to organize them <<ReqtGroup>> stereotype. Text that describes each group should be added.

DIVER'S FLASHLIGHT MODEL – REQUIREMENTS DIAGRAMS



The structure of requirements can be shown in a graphical manner which is sometimes useful

DIVER'S FLASHLIGHT MODEL – REQUIREMENTS DIAGRAMS

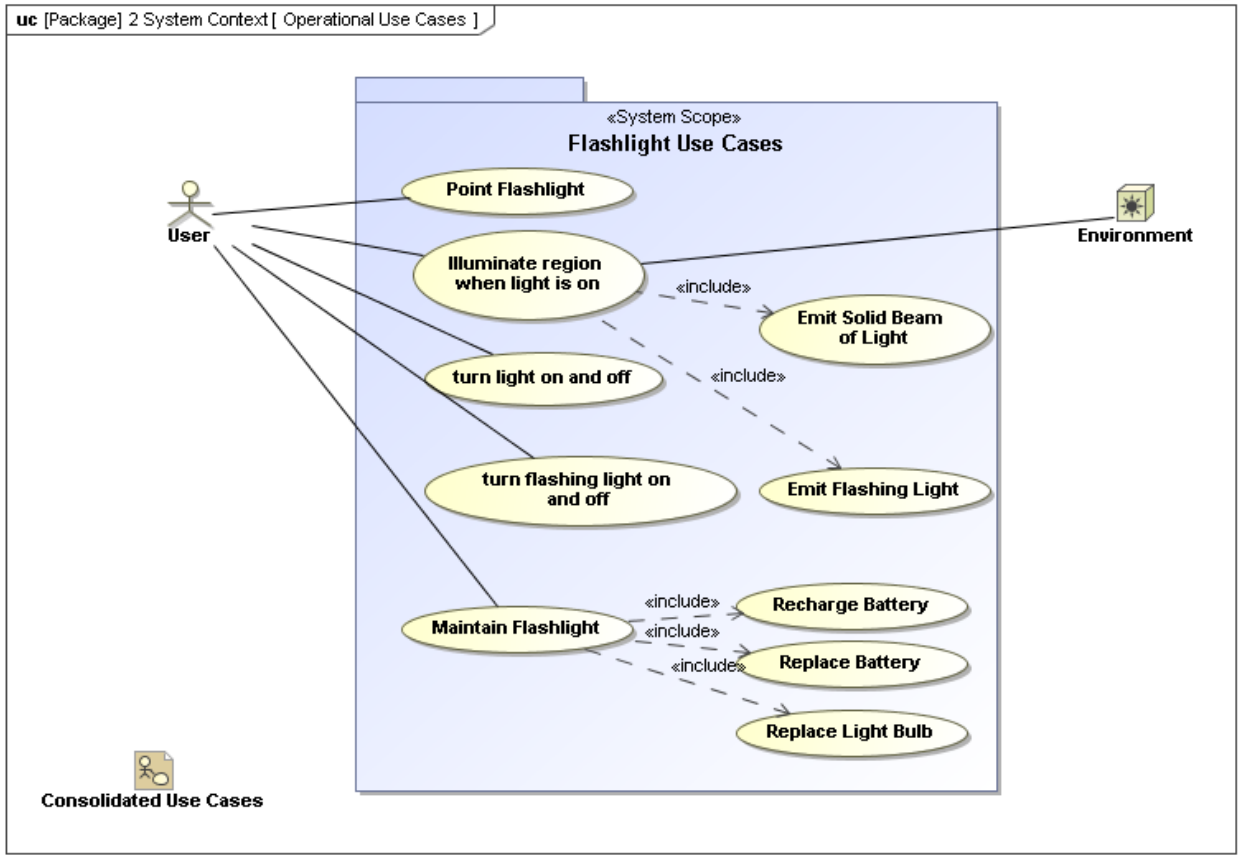


GO BEYOND

LAB 2 IMPORT REQUIREMENTS AND ADD STRUCTURE

DIVER'S FLASHLIGHT MODEL – USE CASES

Use Cases are the "Chapter Headings" in the description of the system.

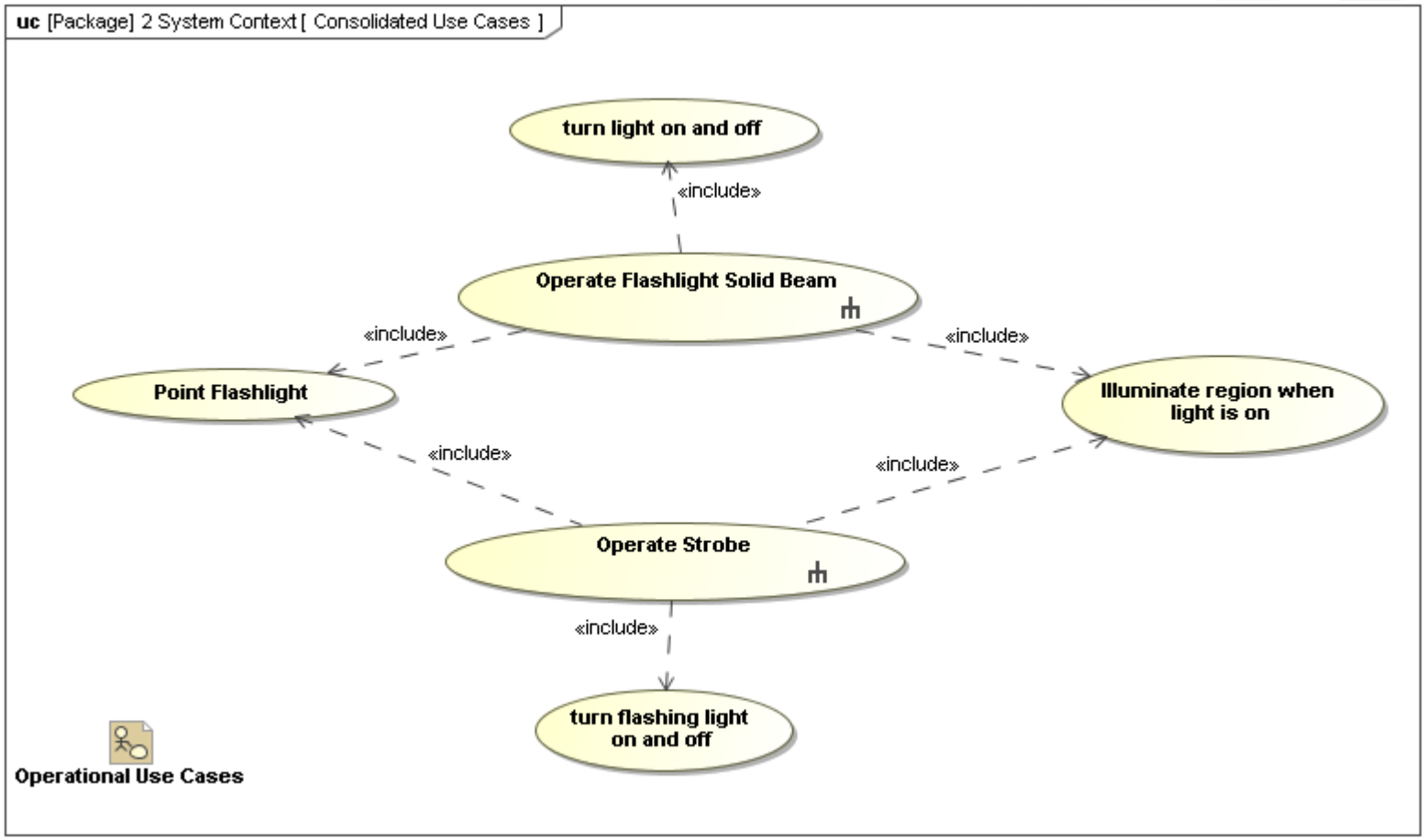


I <u>always</u> produce a use case diagram very early in analysis based on customer requirements.

It may be adjusted or augmented as analysis proceeds.

LAB – DIVER'S FLASHLIGHT MODEL – USE CASES

I find that it is often useful to reorganize and group use cases together for simulation purposes.

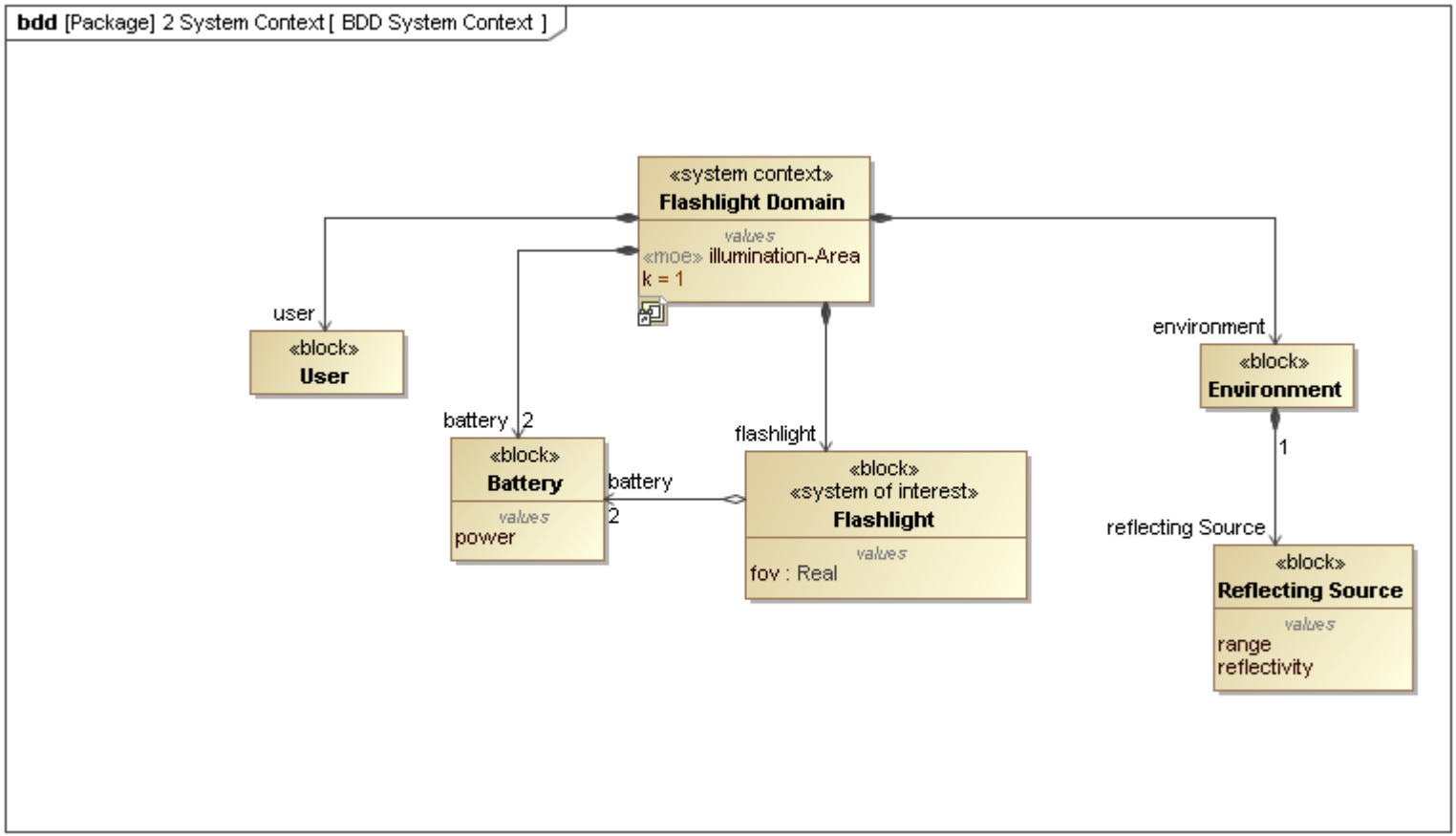


GO BEYOND

LAB 3 USE CASE DIAGRAMS

DIVER'S FLASHLIGHT MODEL – SYSTEM CONTEXT

Next, the structure of the systems "around" the system needs to be established. This is done with a Block Definition Diagram, and an Internal Block Diagram.

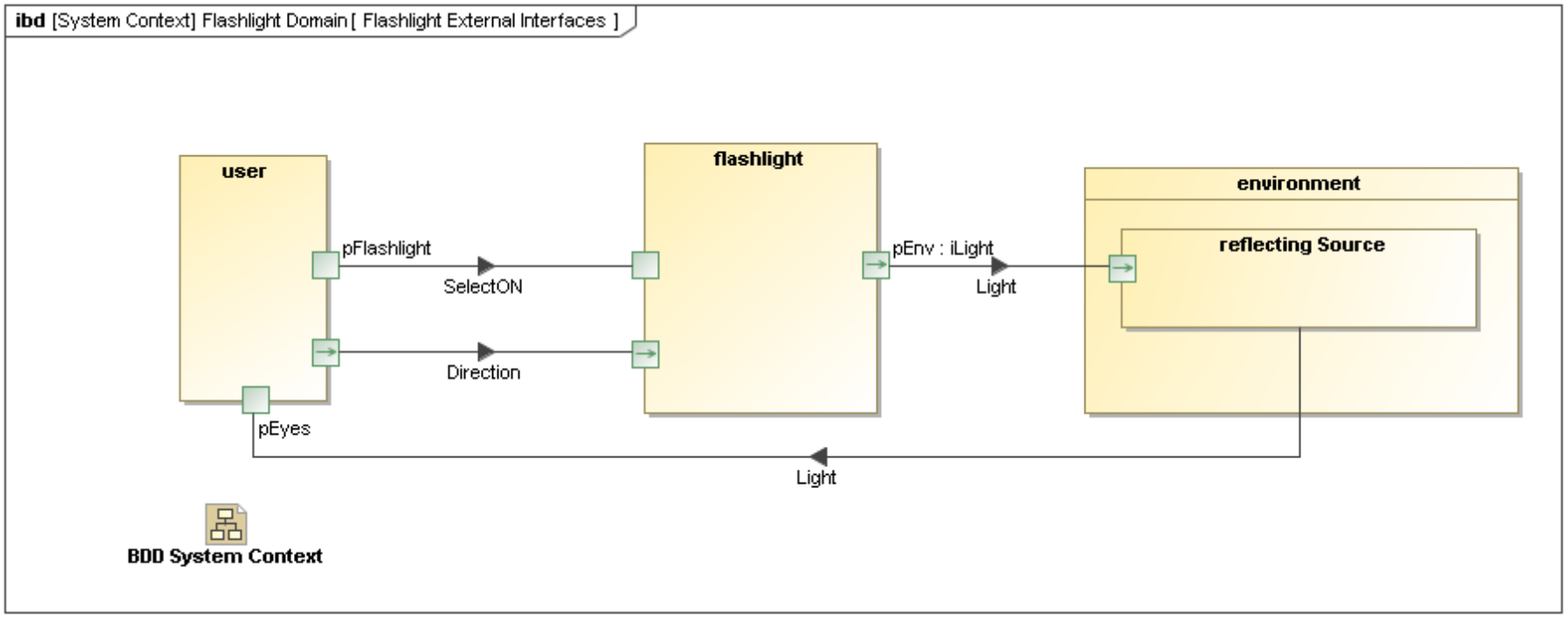


GO BEYOND

LAB 4 BLOCK DEFINITION DIAGRAMS

DIVER'S FLASHLIGHT MODEL – SYSTEM CONTEXT - IBD

The Internal Block Diagram (IBD) is a button click away from the BDD.
Attach ports and connect them. Then fill in the details.



GO BEYOND

LAB 5 INTERNAL BLOCK DIAGRAMS

REVIEW OF LABS SO FAR

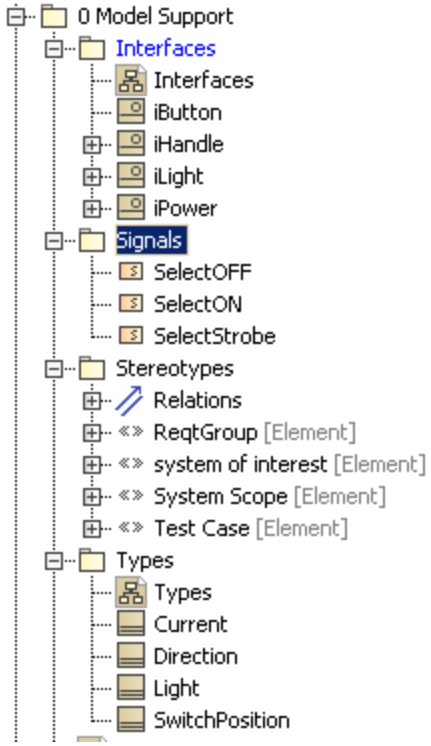
The labs followed the steps for building the structure of the system context.

1. Import requirements (if any)
2. Structure the requirements
3. Derive use cases
4. Refine and group the use cases.
5. Add system context structure
6. Add system context internal structure, interfaces, and flows.

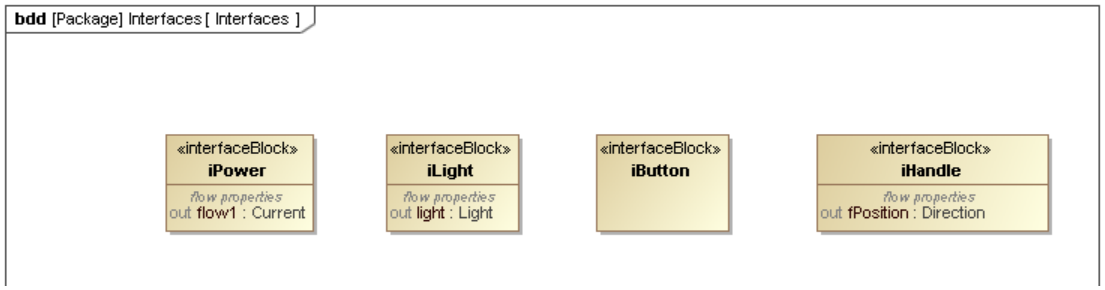
We now need to specify essential system behavior.

DIVER'S FLASHLIGHT MODEL – TYPES AND INTERFACES

In order to add detail to describe what flows between blocks, ports need to be added and typed, and ports need to be connected. I find it useful to bookkeep types in their own package, along with signals, interfaces, and stereotypes. I usually create a BDD for interfaces and another for types used in the model.



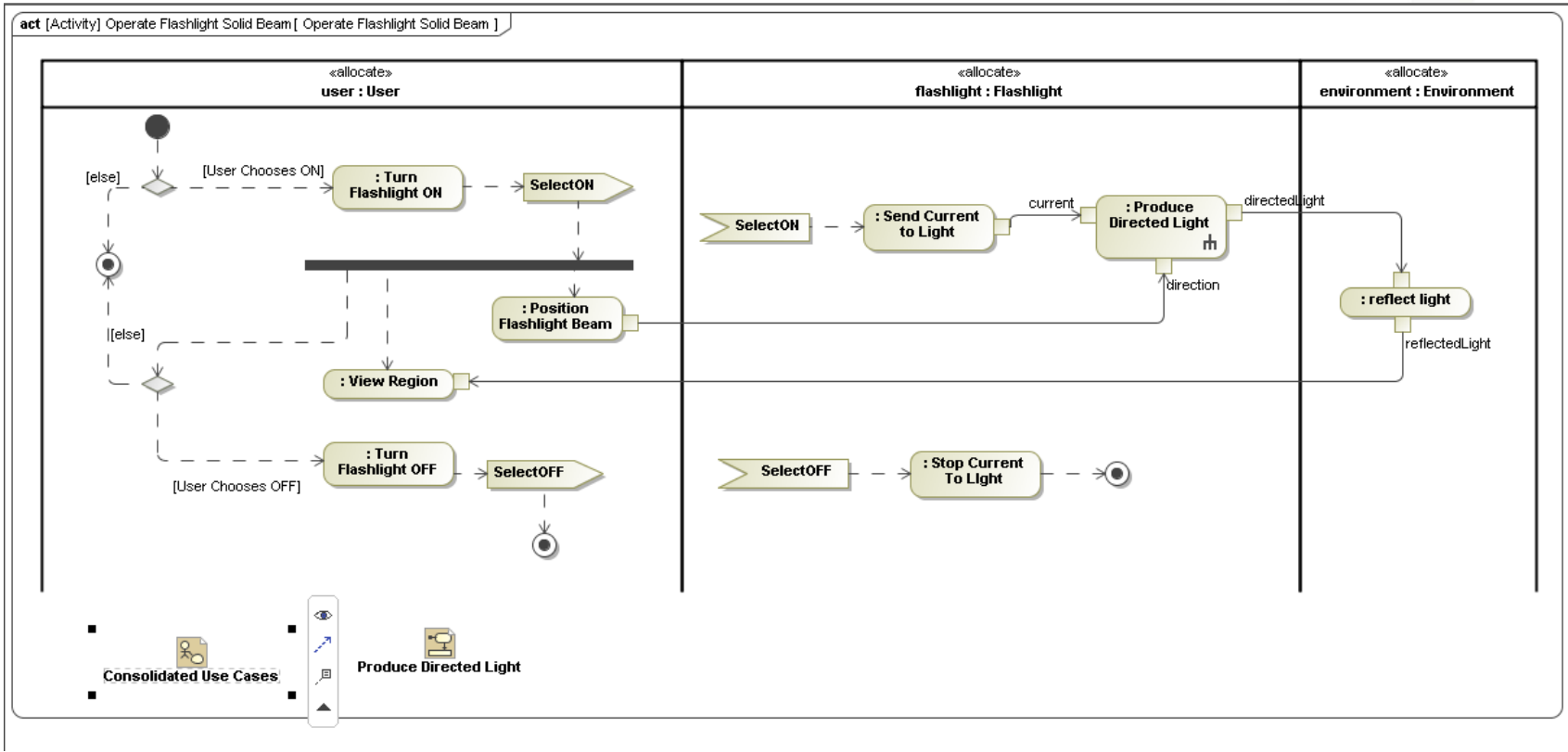
Interfaces are special kinds of blocks indicating what flows between block and in what direction the flow occurs.



Types are blocks that represent a "datatype" or physical thing, and may have units or other properties.

DIVER'S FLASHLIGHT MODEL – SYSTEM BEHAVIOR

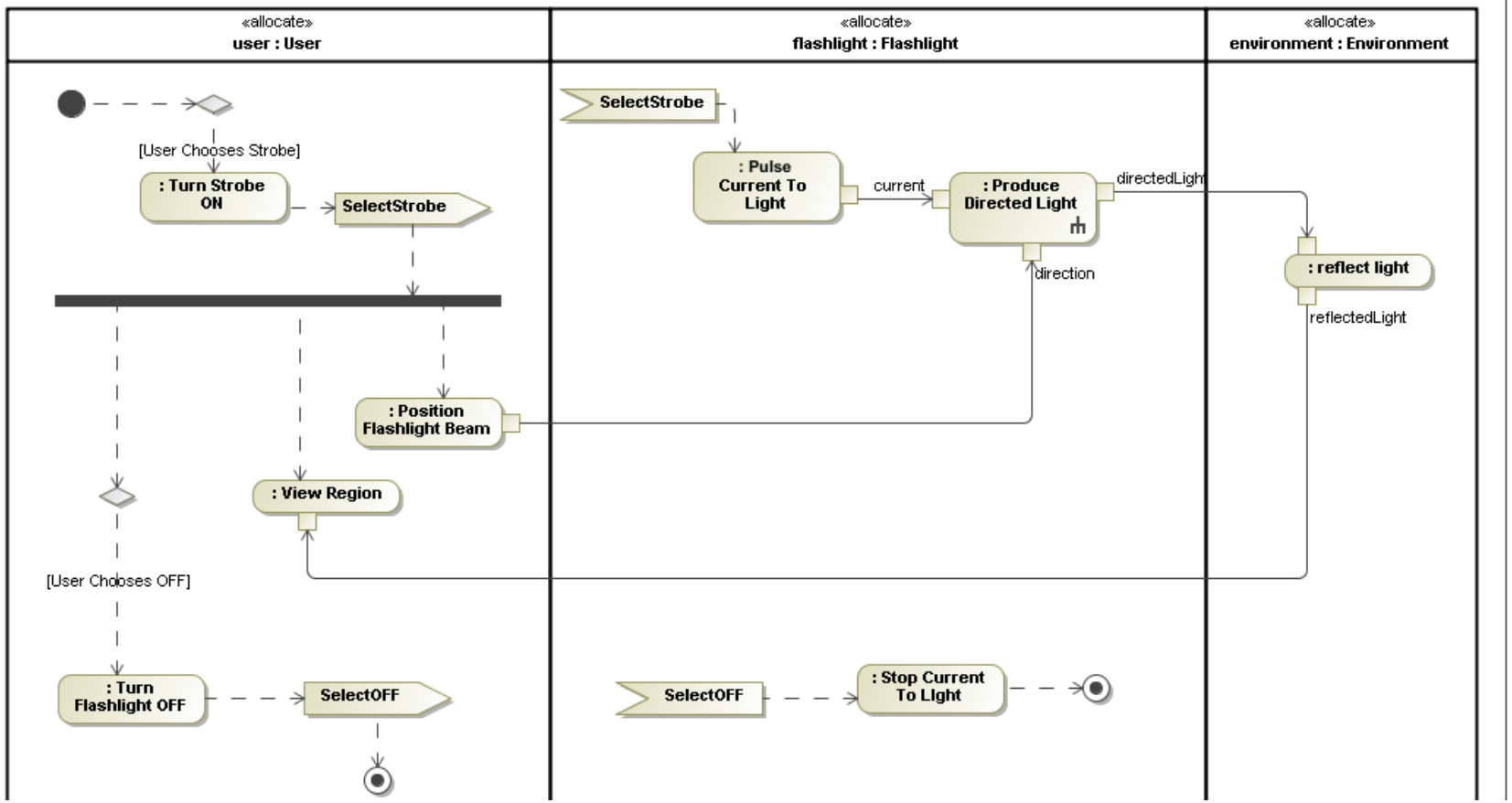
Next, the Use Cases need to be broken down into specific behaviors and capabilities assigned to essential components of the system context.



Simulate AD's as soon as they are drawn!

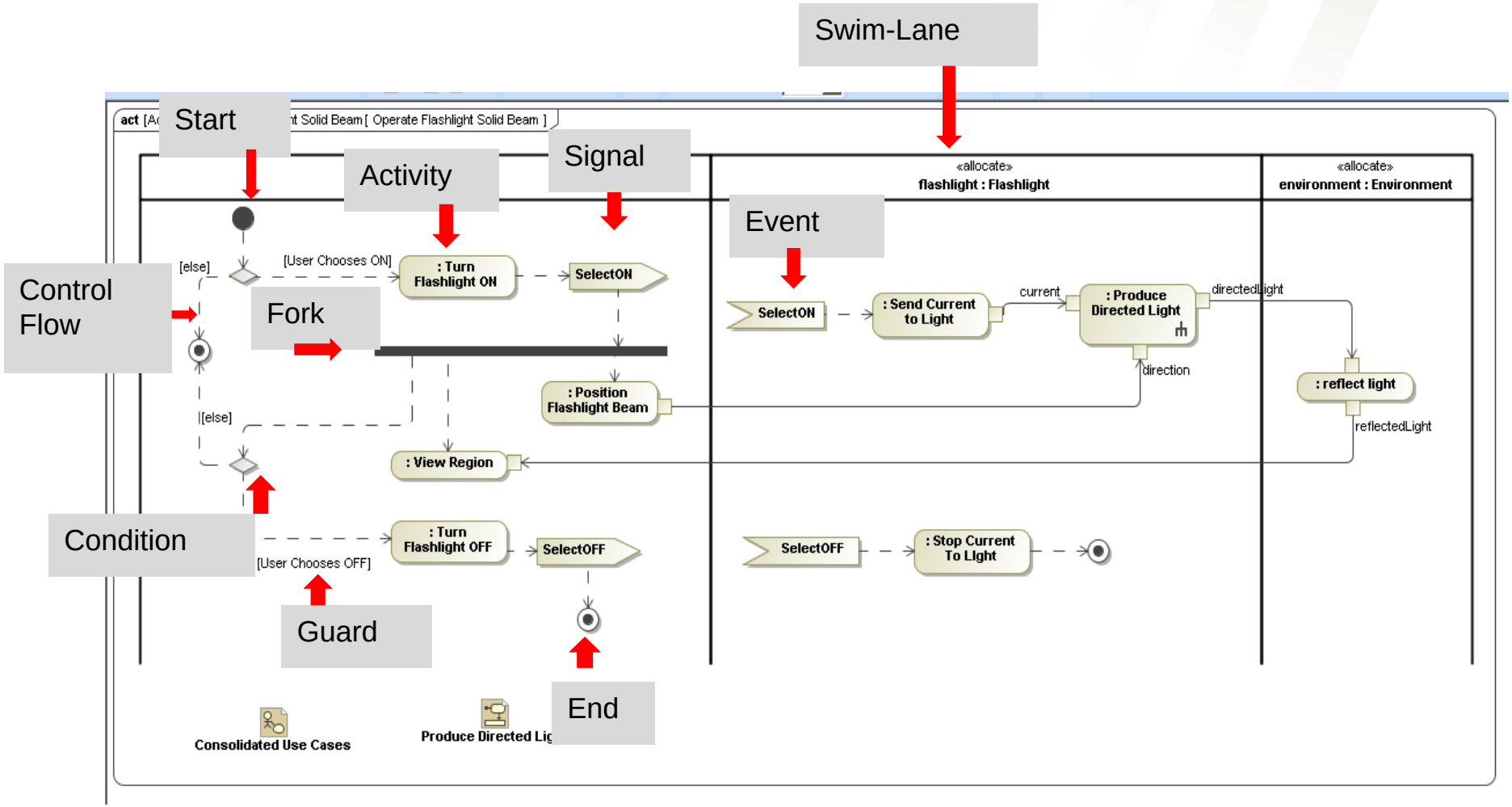
DIVER'S FLASHLIGHT MODEL – SYSTEM BEHAVIOR

This model requires 2 Activity Diagrams to show the essential required behaviors.



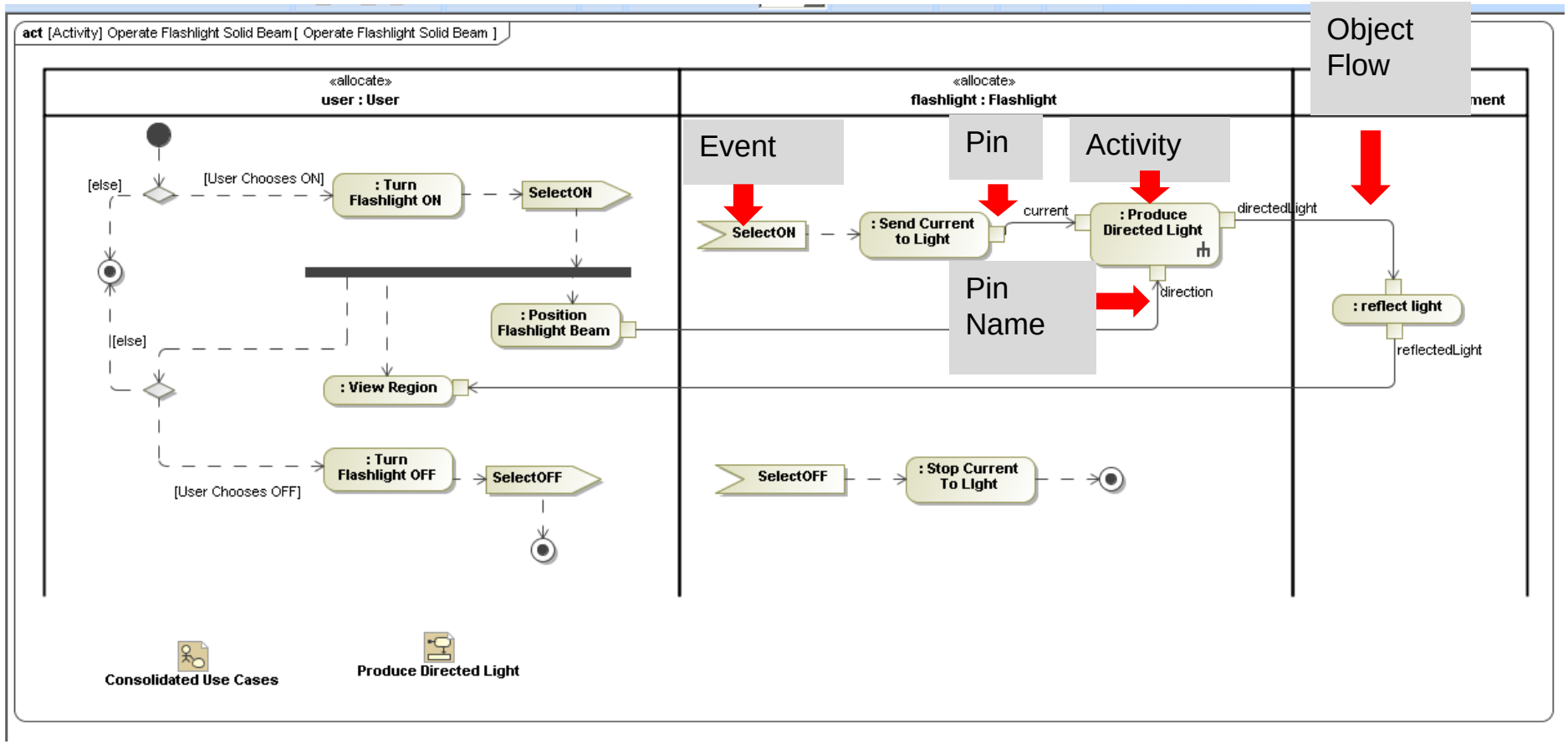
DIVER'S FLASHLIGHT MODEL – SYSTEM BEHAVIOR

Walking through the diagram you see the following elements.

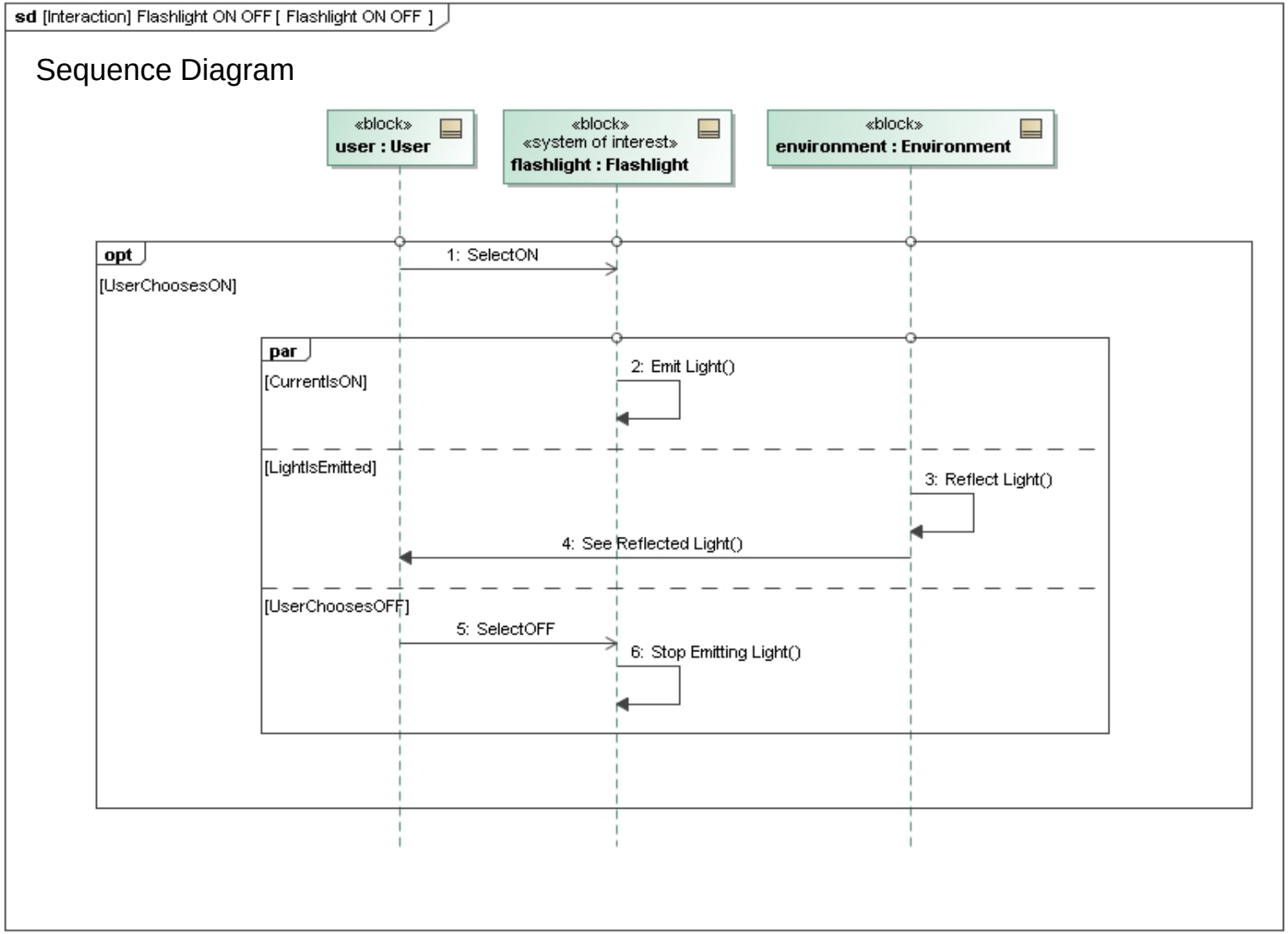


DIVER'S FLASHLIGHT MODEL – SYSTEM BEHAVIOR

Data flows and pins do not need to be fully typed for animation / simulation to be used to check diagram.



DIVERS FLASHLIGHT MODEL – EVENT SEQUENCES

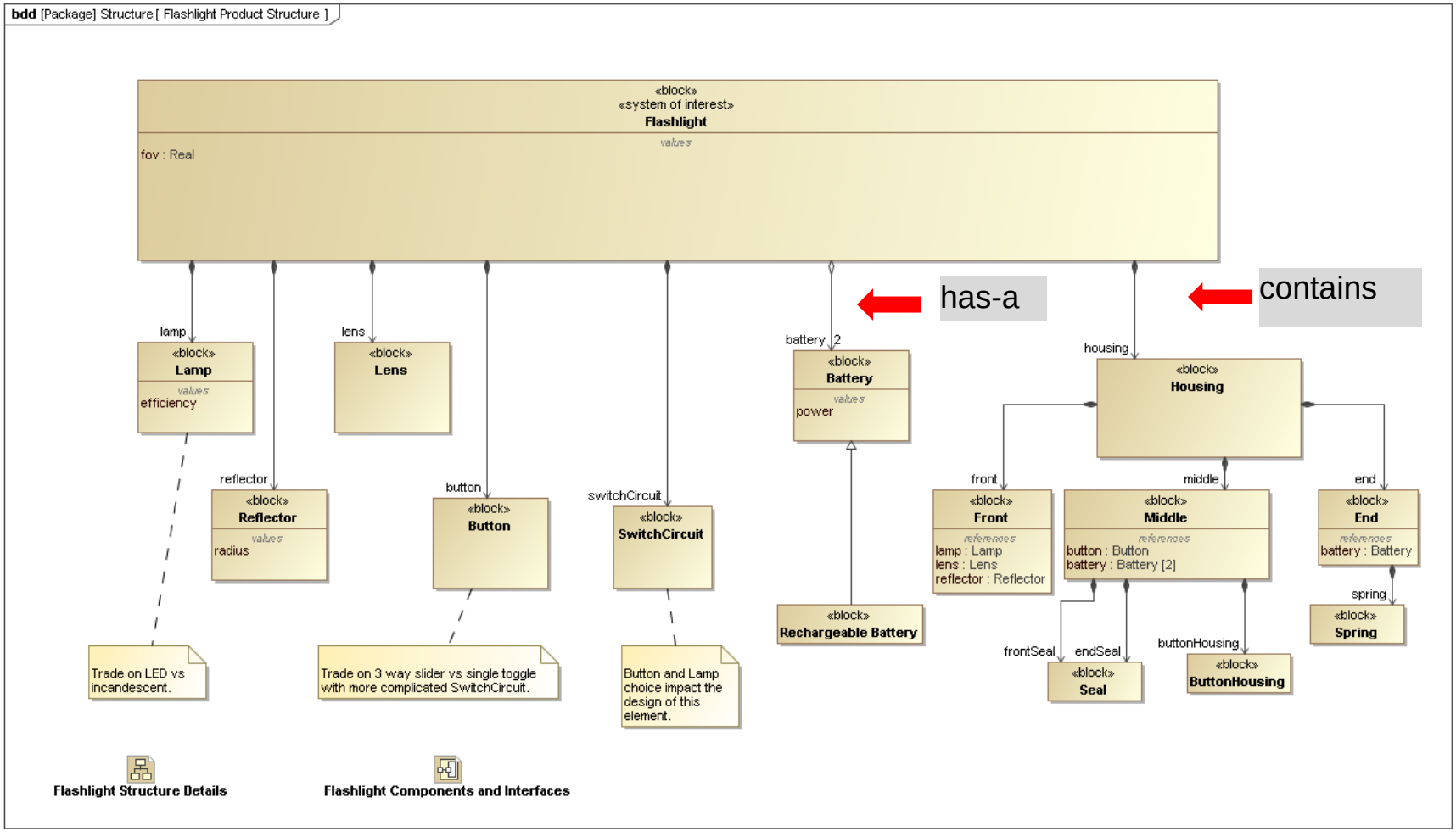


GO BEYOND

LAB 6 SYSTEM CONTEXT BEHAVIOR ACTIVITY DIAGRAMS

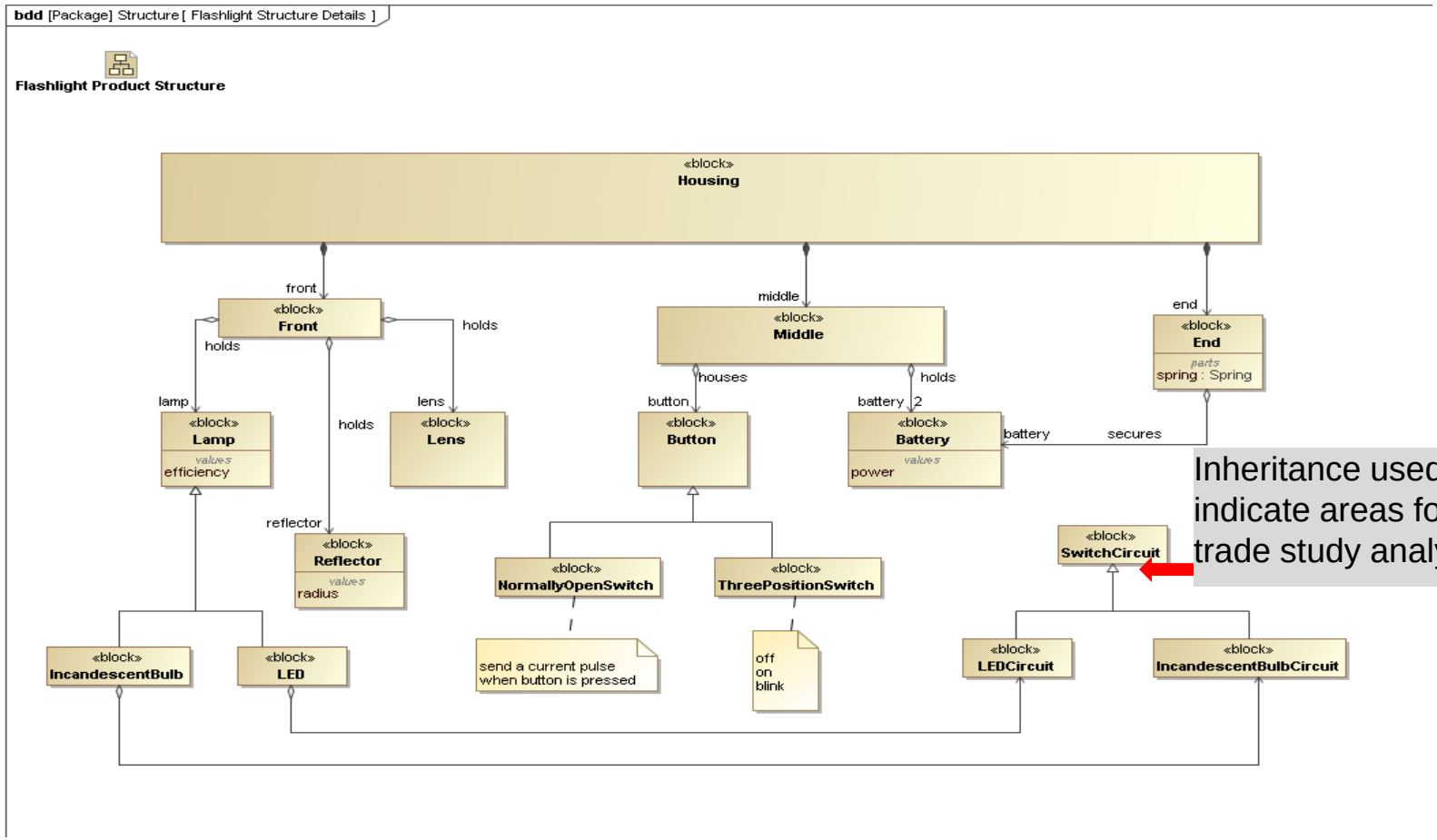
DIVER'S FLASHLIGHT MODEL – COMPONENTS BDD

The essential components of the system of interest (SOI) then need to be established.



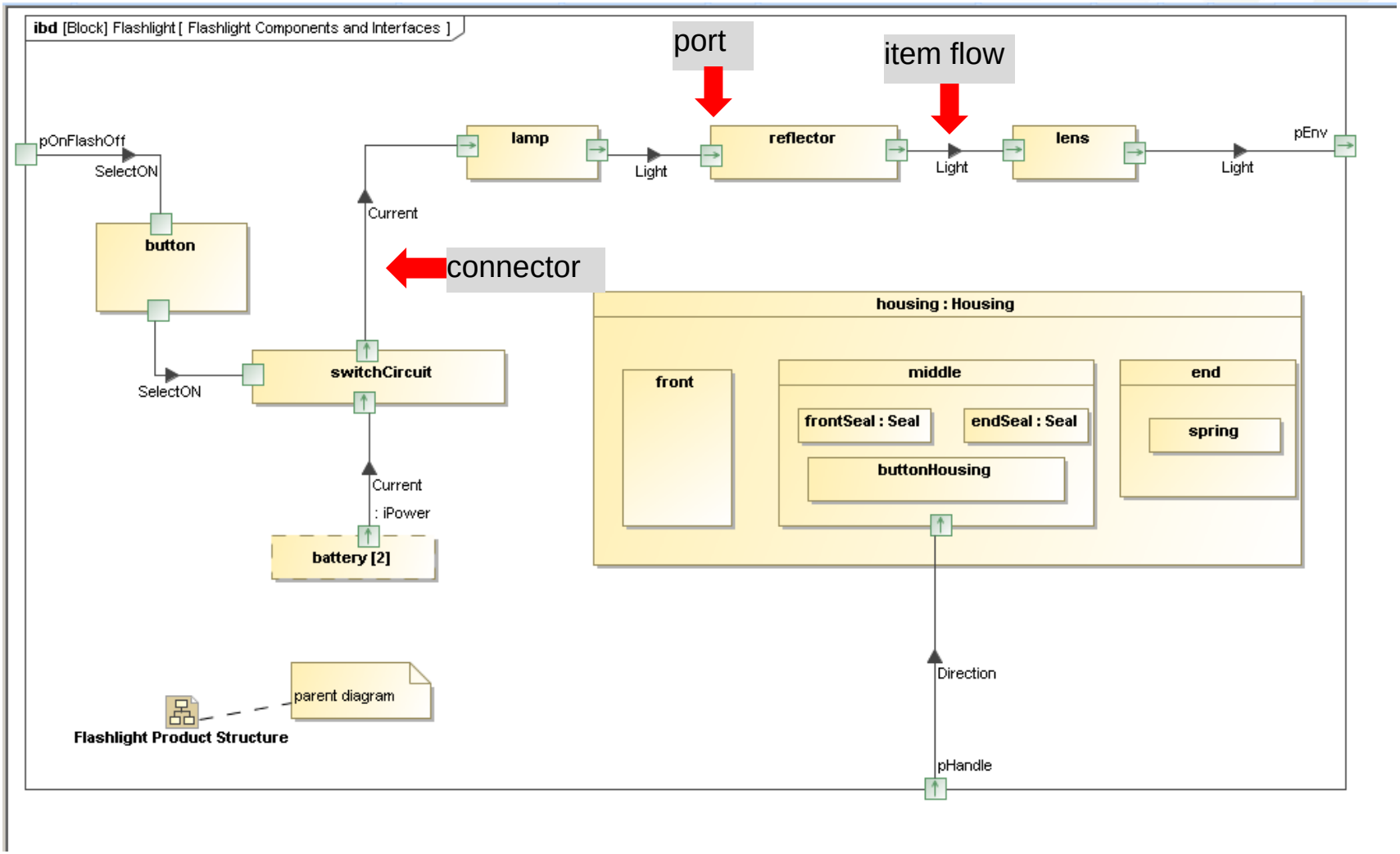
DIVER'S FLASHLIGHT MODEL – COMPONENTS BDD

The essential components of the system of interest (SOI) then need to be established.



DIVER'S FLASHLIGHT MODEL – COMPONENTS - IBD

The essential components of the system of interest (SOI) then need to be established and connected by typed interfaces.



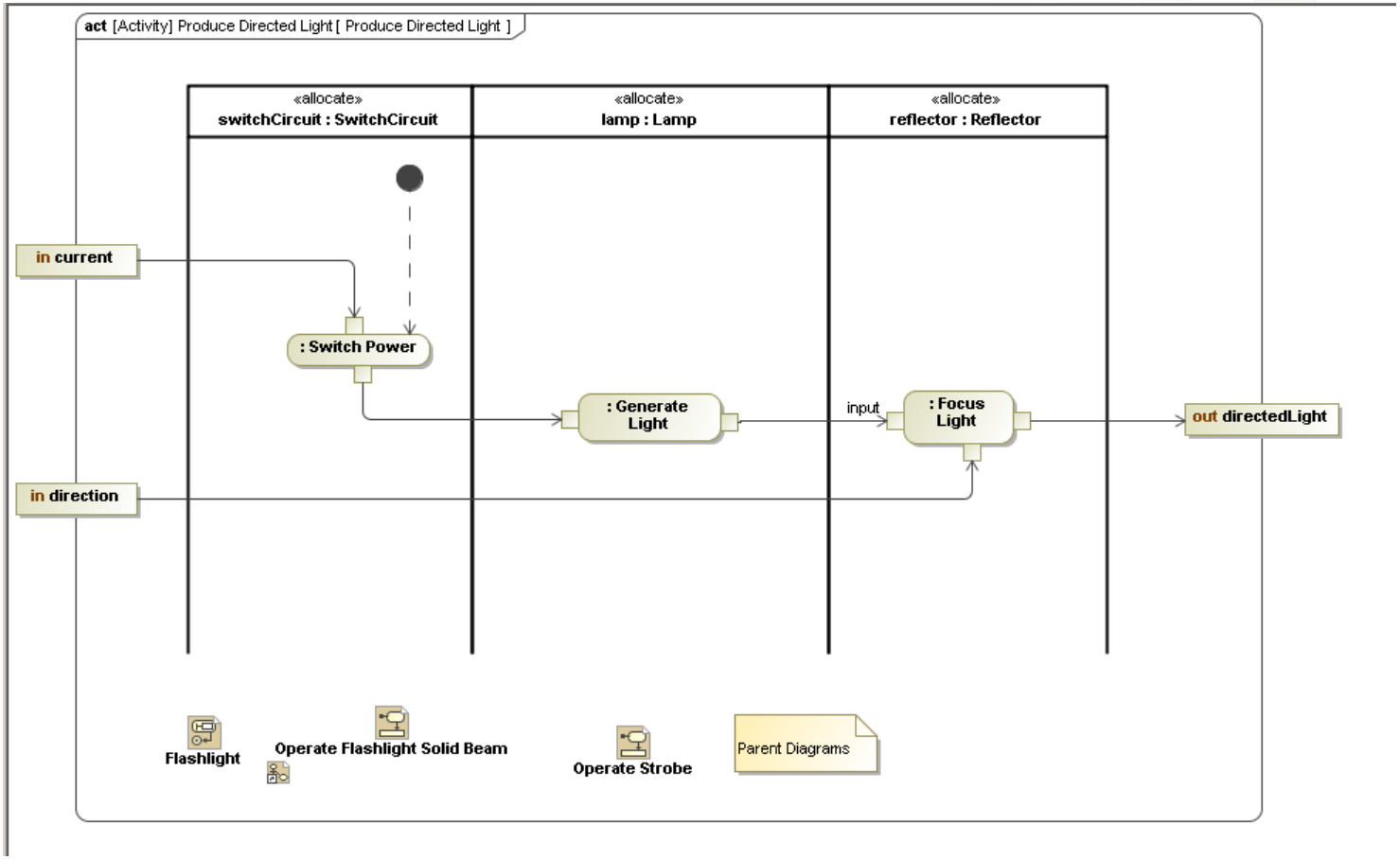
GO BEYOND

LAB 7 SYSTEM COMPONENT STRUCTURE

A UNITED TECHNOLOGIES COMPANY

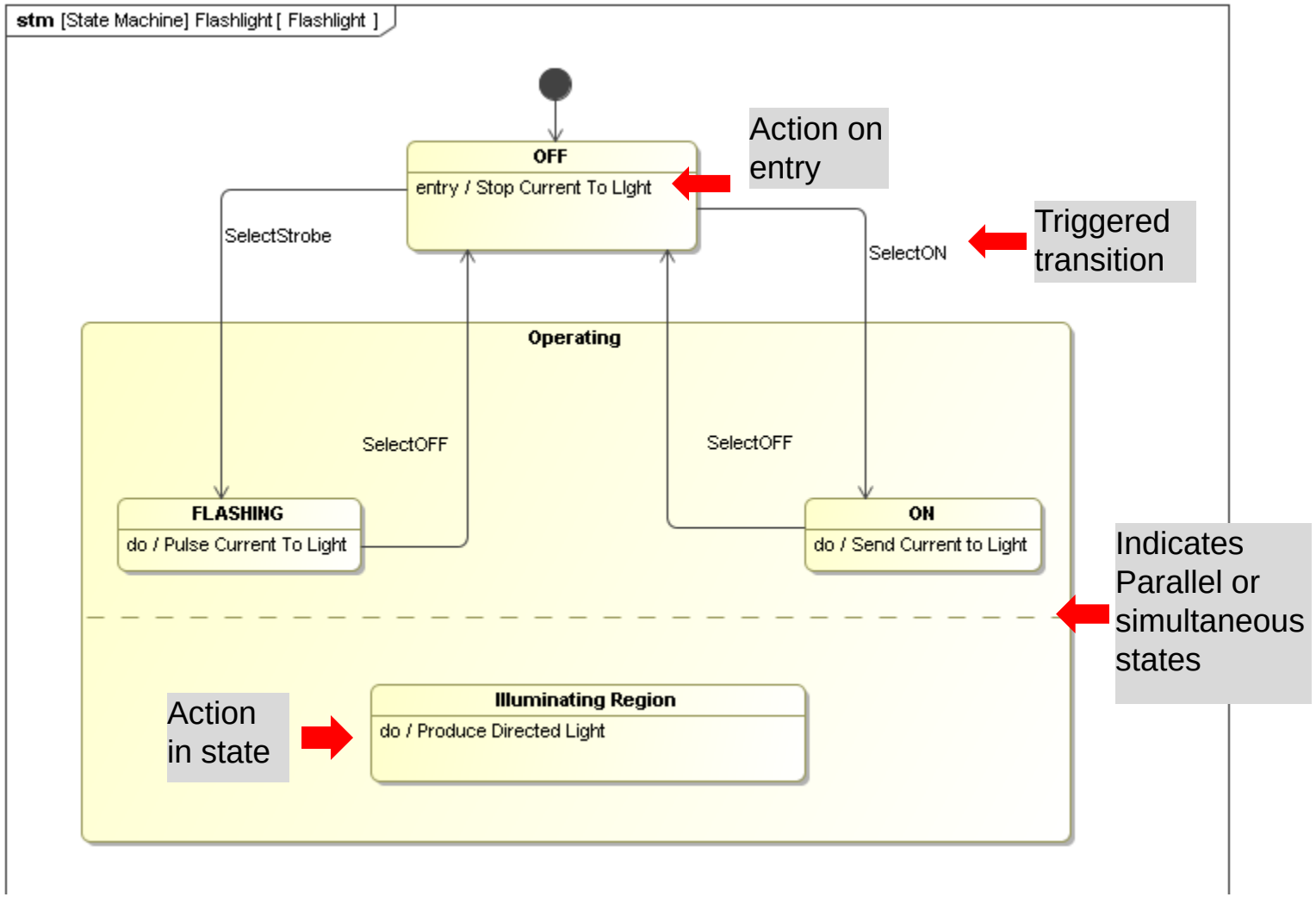
DIVER'S FLASHLIGHT MODEL – SOI BEHAVIOR

Next, the behavior of the components of the SOI needs to be established.



DIVER'S FLASHLIGHT – SOI STATES AND MODES

Next, states and modes of the system can be established.

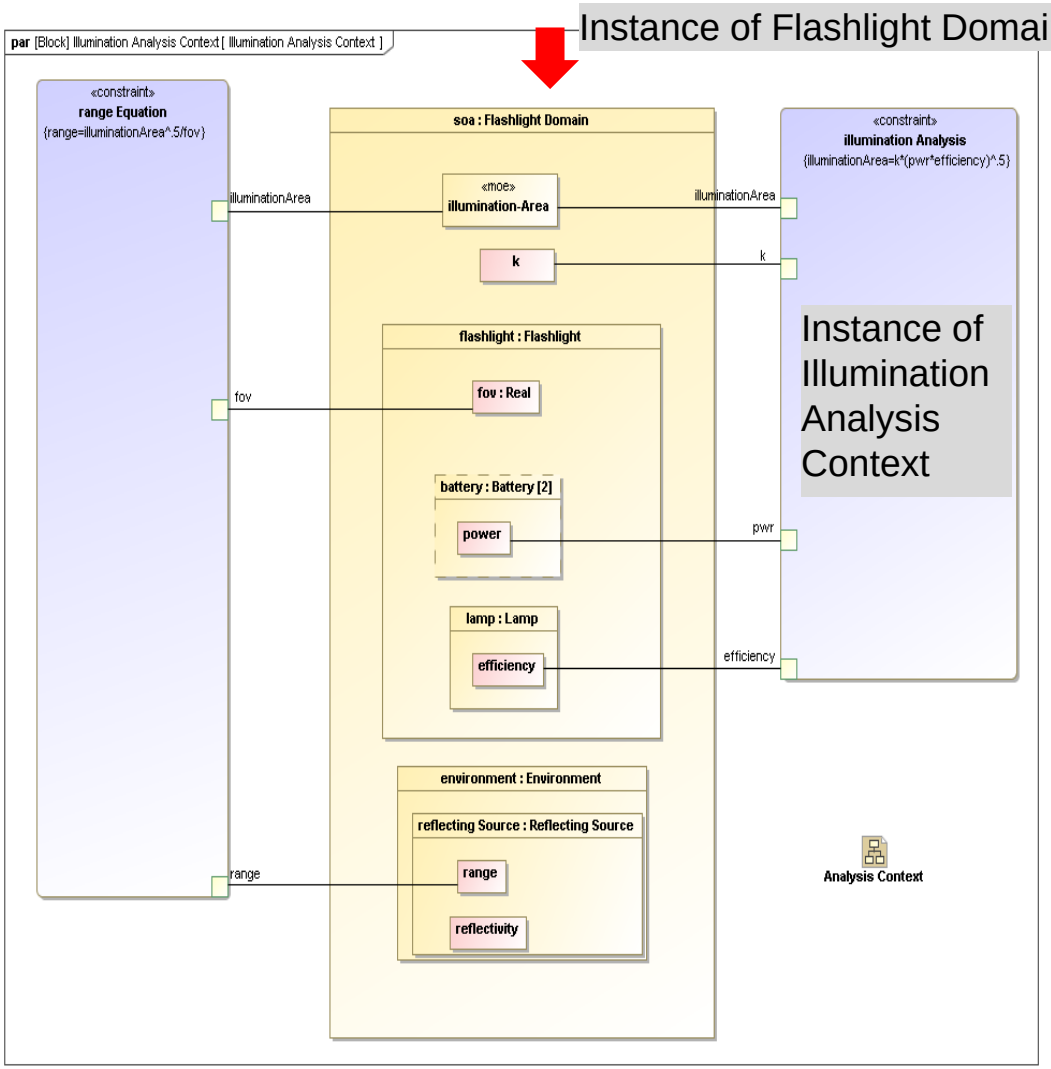


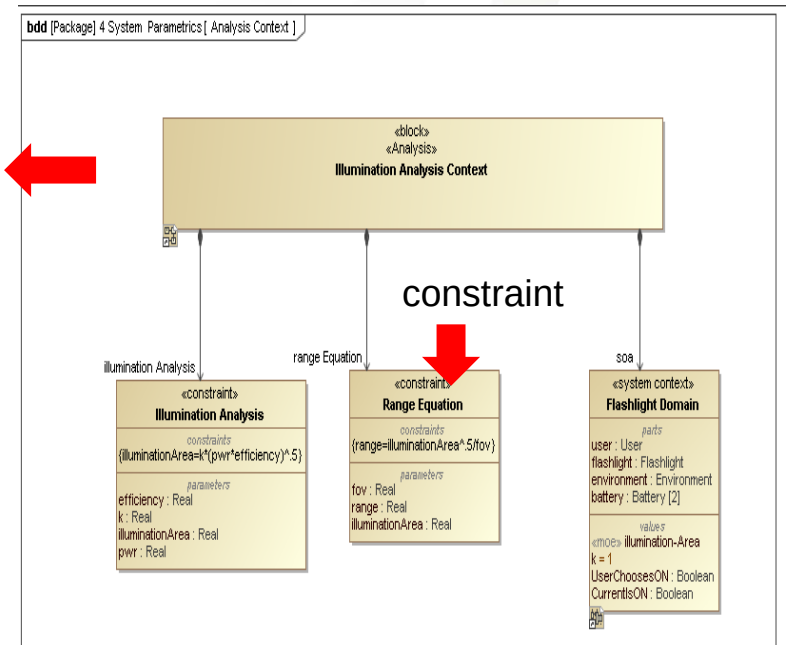
GO BEYOND

LAB 8 SYSTEM COMPONENT BEHAVIOR

A UNITED TECHNOLOGIES COMPANY

DIVERS FLASHLIGHT MODEL – PARAMETRIC RELATIONSHIPS





Parametric relationships are cross cutting, and are defined at the system context level.

GO BEYOND

LAB 9 PARAMETRIC RELATIONSHIPS

GO BEYOND

WHY SYSML? RECAP

SYSML - WHY SHOULD I USE IT?

- Breaking a system into Requirements, Structure, Behavior, Parametrics views allows for issues to become apparent early in design process
- The multiple views of the system lead to a better abstraction process. Eliminate details that cause confusion and extra work.
- Standard representation leads to unambiguous interpretation of design intent.
- Cross-cutting views encourage understanding of system environment, allowing emergent behavior to be caught early.
- Risk reduction, Cost reduction

SYSML – WHY SHOULD I USE IT?

- Strong SE discovers emergent behavior earlier in design process
- "Fail early and often" John C. Maxwell "Failing Forward"
- Enables tracking the *actual* implementation of stakeholder needs from early phases throughout the project with continuous integration (CI)
- Enables knowledge and process sharing. All phases of project
- Provides a stable industry standard for descriptive modeling for systems engineering replaces "ad hoc" schematics.
- Compliance with open standards mean reduced vendor / tool dependency and interoperation between various tools.
- Vendors are converging on providing full support to standard

SYSML - WHEN SHOULD I USE IT?

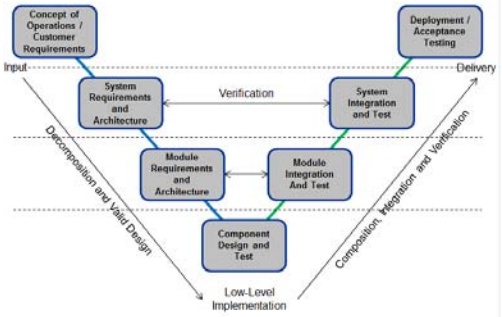
- Phase 0/1 Concept Initiation, Concepts Optimization
  - Functional understanding of requirements
  - Architecture studies
  - Strong handoff to next phases
- Phase 2/3 Preliminary Design, Detailed Design
  - Solution Defined, Requirements Valid and Mature
- Phase 4 Verification & Validation
  - Full understanding of how failed reqs affect system
- Phase 5/6 Delivery, Post Production & Service Support
  - Assist with root cause investigations

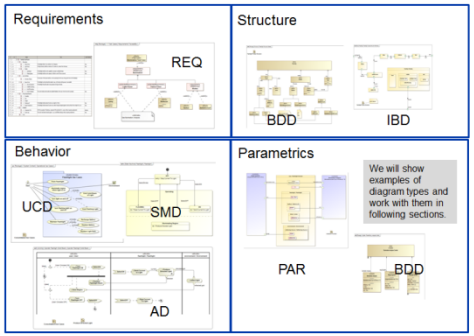
SysML has applications in all phases of engineering processes

SYSML - WHAT DO I NEED TO KNOW?

- To use SysML effectively… you need
- SysML Language – an effective subset of the full language
- SysML Tool Familiarity
- Methodology – What to do and when to do it.
- Systems Thinking – be able to abstract essential elements away from unessential details
- Specific Domain Knowledge – either be an expert or collaborate with an expert in the system or subsystem engineering area to be modeled.

WHY SYSML? QUESTIONS?





- Use SysML for your own work – individual tasks!
- You do not need wide agreement, multi organizational rollout, full infrastructure deployment, management buy in to start to see benefits.
- Try it!