

# Model Based System Engineering Best Practice Storyboard



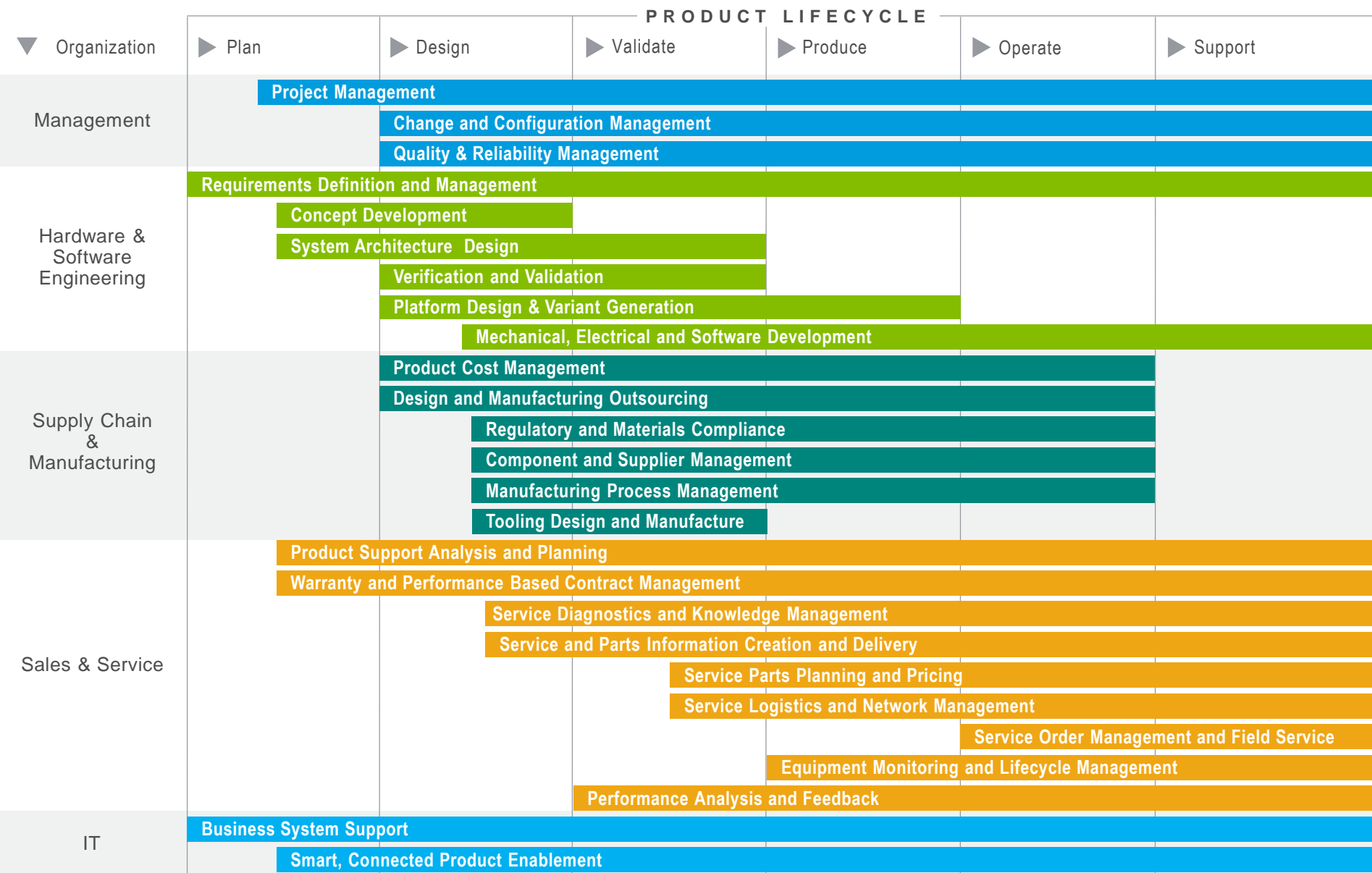
February 2015

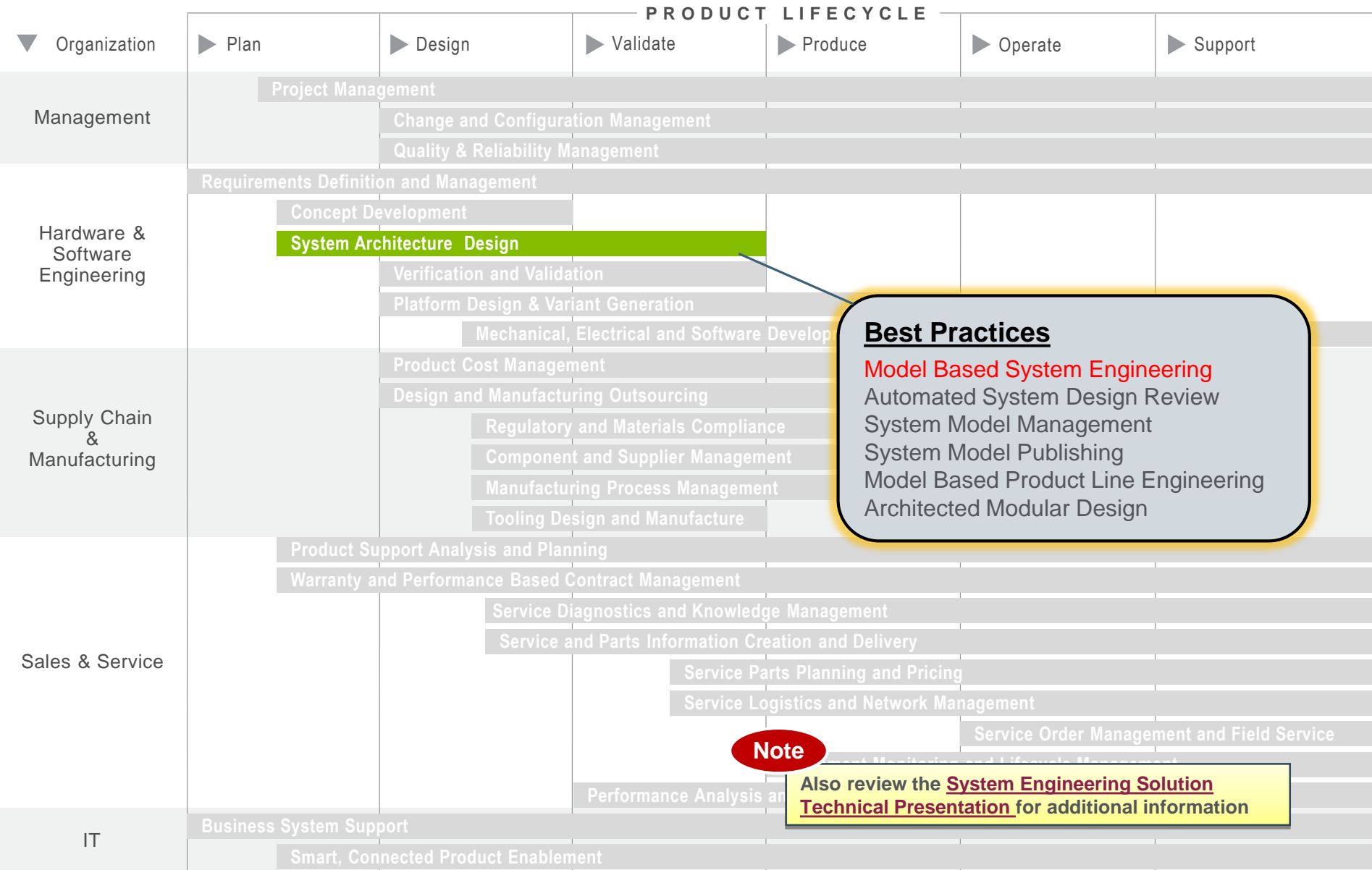
- Introduction
  - Key Concepts
  - Primary Challenges
- PTC Best Practice Storyboard
  - Model Based System Engineering

**Note**

It is recommended that the reader has some knowledge of SysML

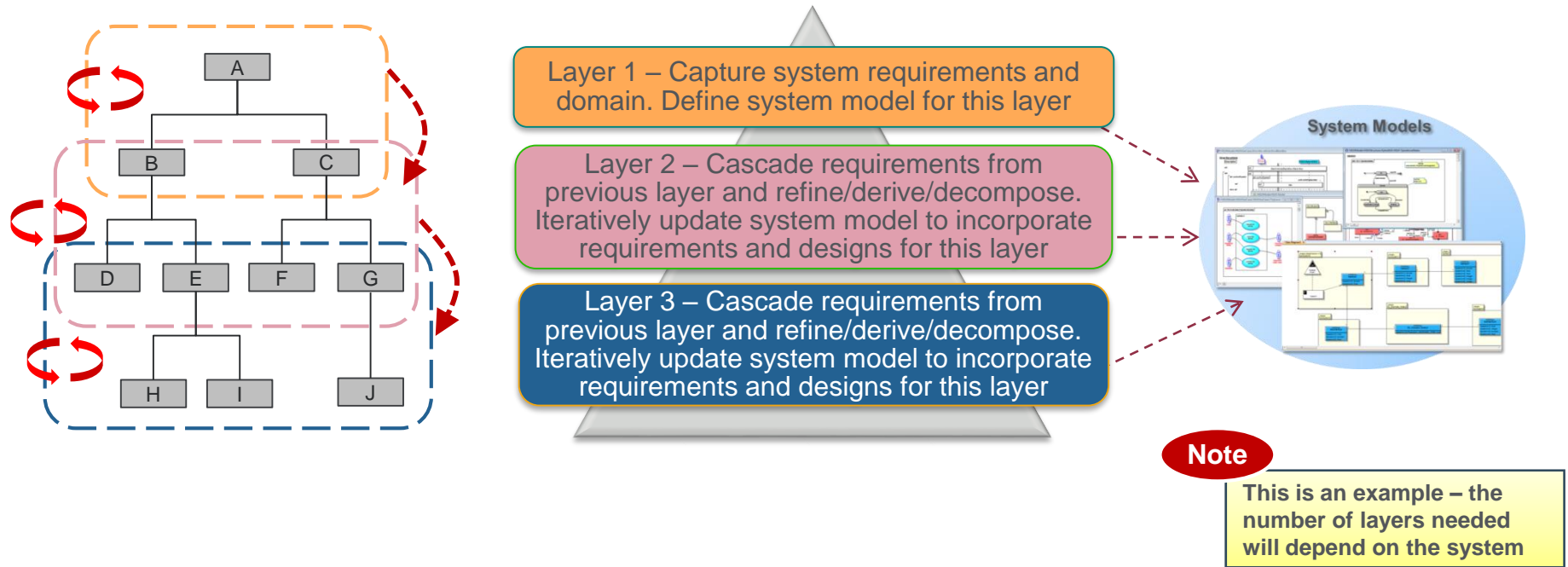
- OMG SysML Specification -  
<http://www.omg.org/spec/SysML/1.3/>





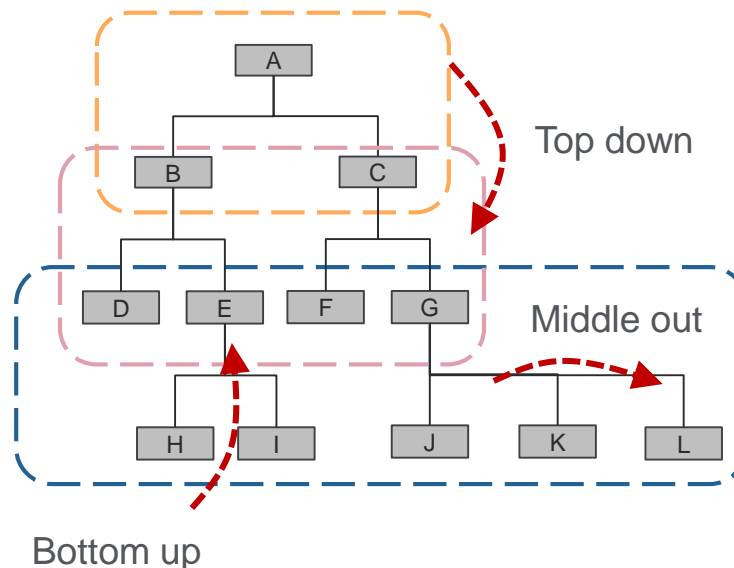


- A common process for constructing a SysML system model involves an iterative approach based around the SysML diagram taxonomy
  - Each iteration corresponds to a layer of abstraction of the system and involves the capture of requirements relevant to that layer, followed by the definition of structural and behavior elements necessary to meet those requirements
  - Work would then progress on the next layer(s) to model sub-systems
  - Work within the iteration such as defining the structural and behavioral elements can be done in parallel



## Considerations

- A top down modelling approach can produce a comprehensive definition of a product or system starting from the highest level down, but is not always practical or realistic
  - A bottom up approach should also be considered, where definition of critical aspects or sub-systems of a product are prioritized
  - Middle out is where additional detail is captured at the same level of abstraction
  - Often a combination of these approaches works best



## Considerations

- Often customers are looking to improve existing products, so model the parts of the system they plan to re-use only to the level of detail needed, enabling re-design of specific sub-systems
- Consider if product variability will be required
  - Refer to the Model Based Product Line Engineering storyboard for more information
- Large, complex systems can be managed using the Asset Library, allowing modular components to be created and re-used within multiple models. Requirements, use cases and interfaces are cascaded down when assets are re-used.
  - Refer to the Architected Modular Design storyboard for more information

- Introduction
  - Key Concepts
  - Primary Challenges
- PTC Best Practice Storyboard
  - Model Based System Engineering

## *Challenge: Designing complex systems within disconnected engineering disciplines*

### • Practice:

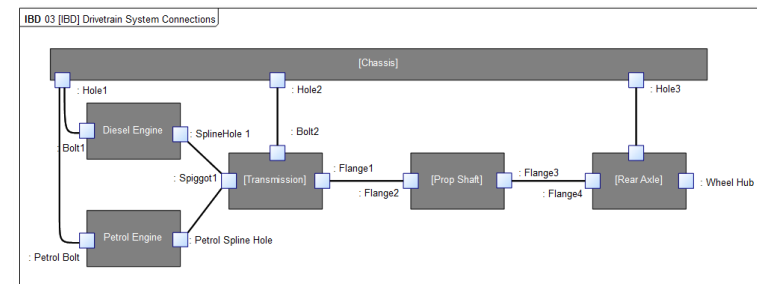
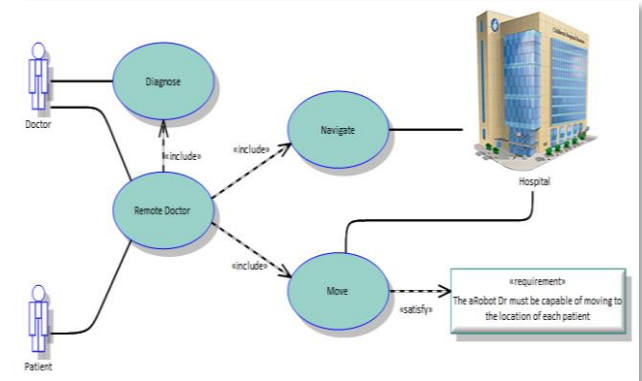
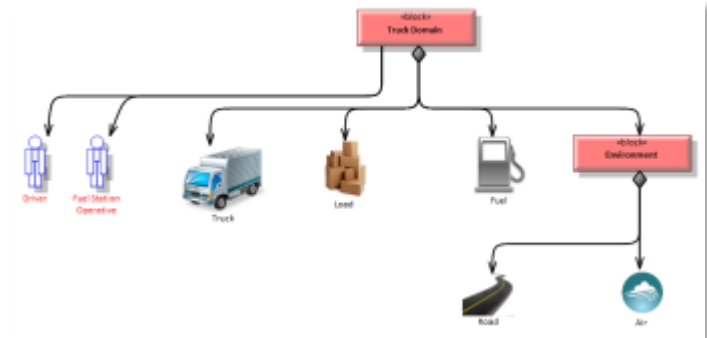
- Enable collaborative complex system engineering using a common industry standard modeling language

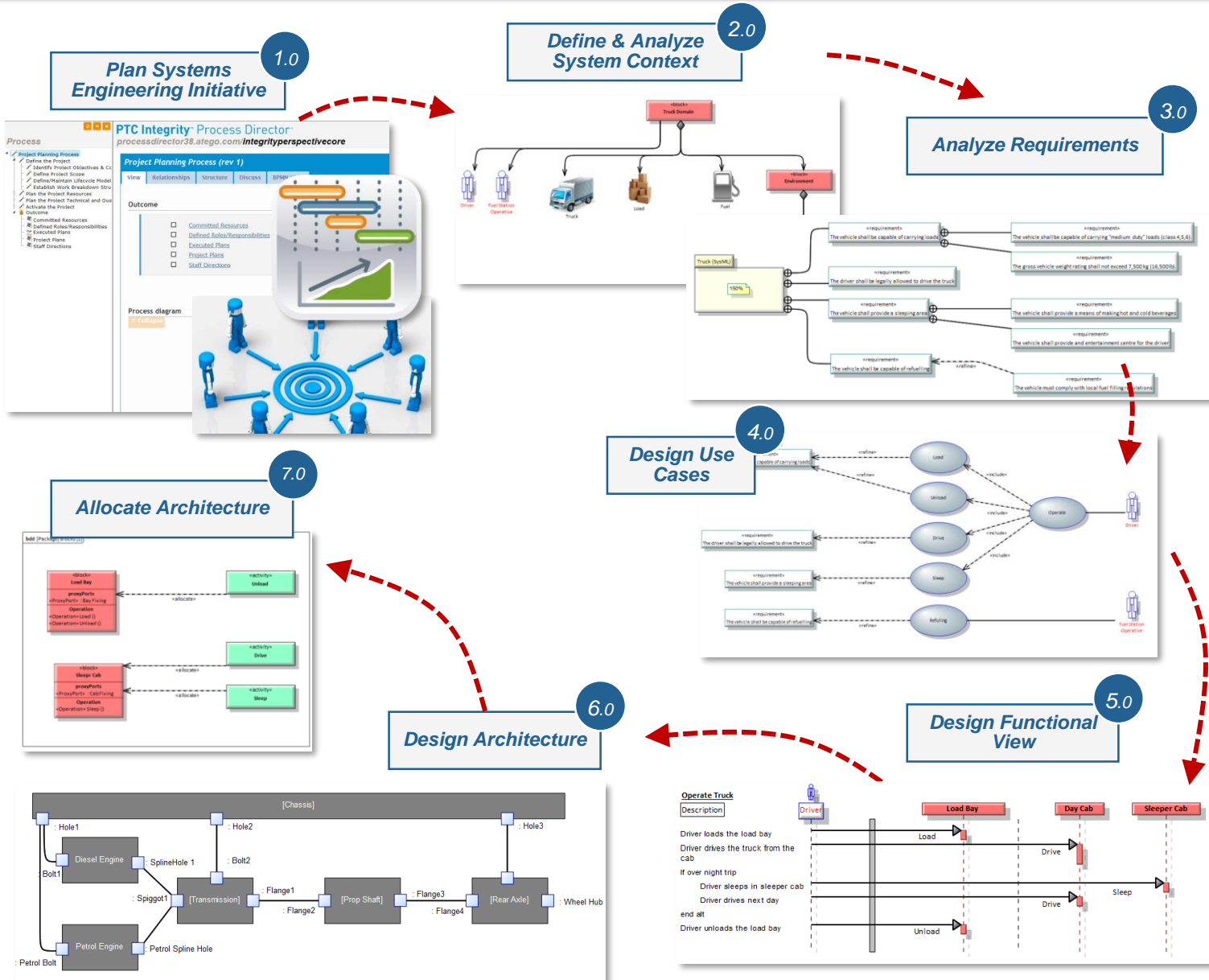
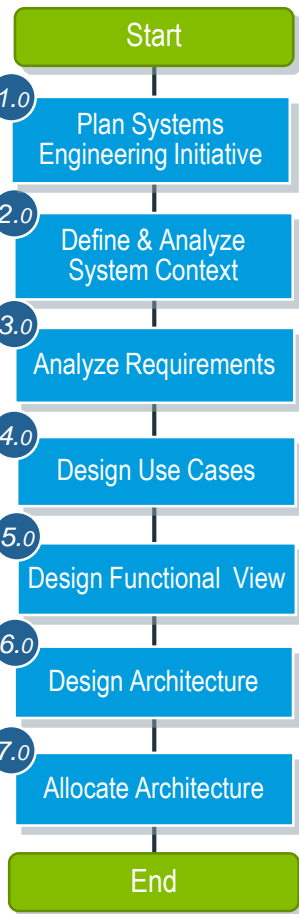
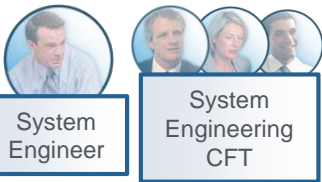
### • Capabilities:

- Graphically define product context and stakeholders
- Map and trace from requirements to the design
- Model architecture including system functions and structure

### • Value:

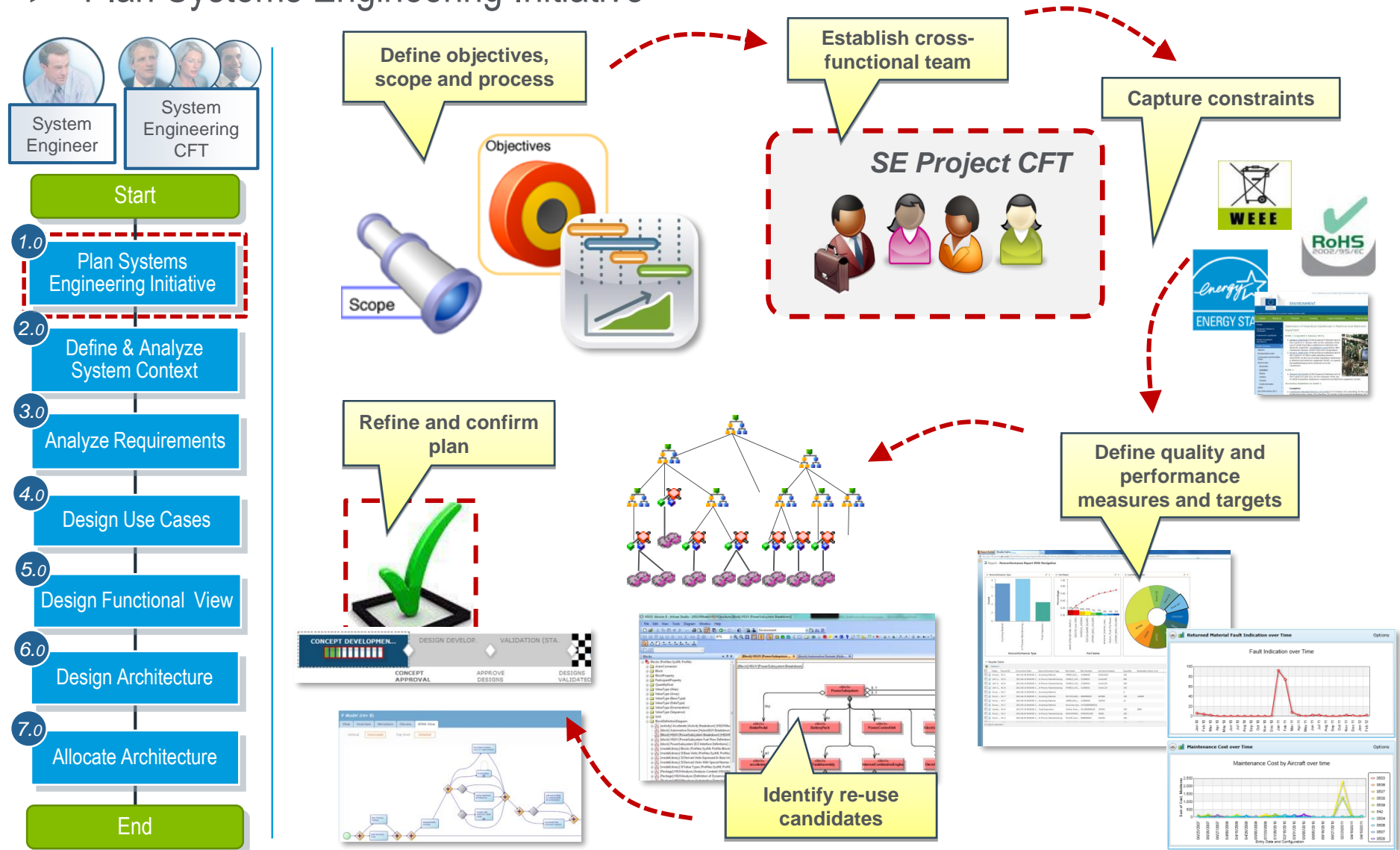
- Improve communication between stakeholders and engineering teams
- Improve product quality by early problem identification and enhanced design integrity
- Increase productivity by reuse of model elements and improved impact analysis
- Reduce risk with improved estimation and early requirements validation







## ► Plan Systems Engineering Initiative



## ► Define and Analyze System Context



Start

1.0 Plan Systems Engineering Initiative

2.0 Define & Analyze System Context

3.0 Analyze Requirements

4.0 Design Use Cases

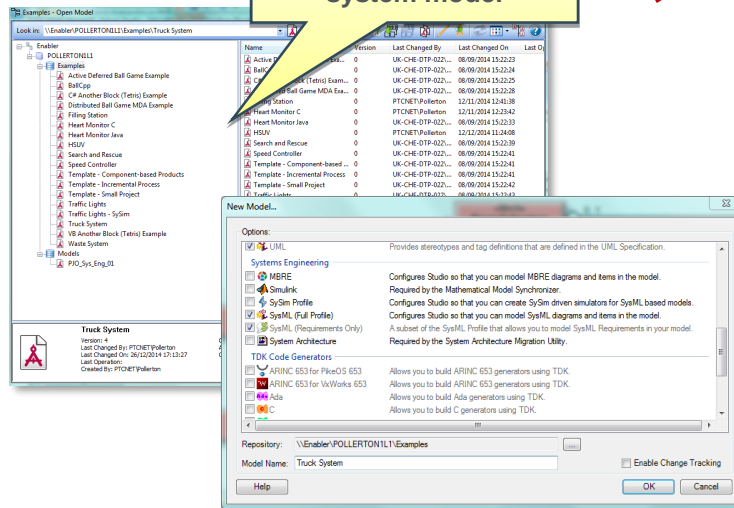
5.0 Design Functional View

6.0 Design Architecture

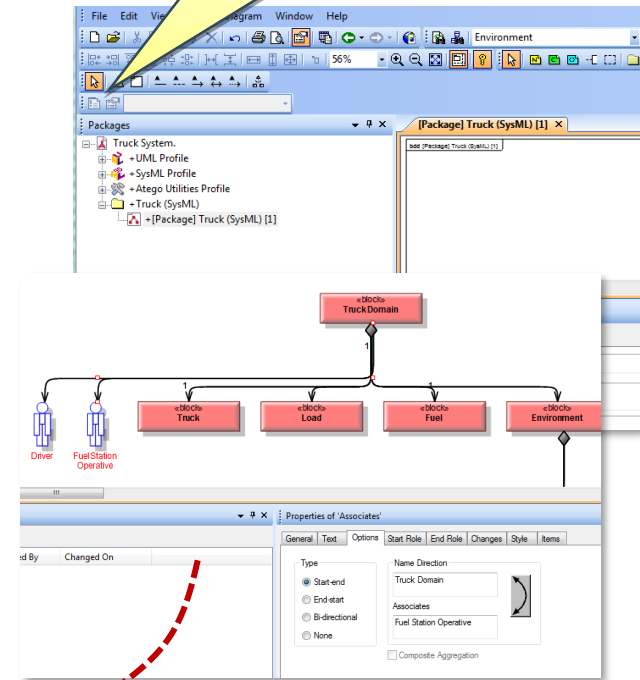
7.0 Allocate Architecture

End

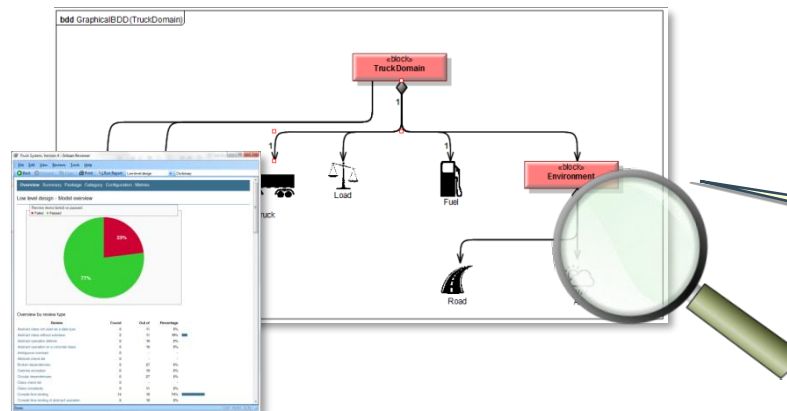
Create and configure system model



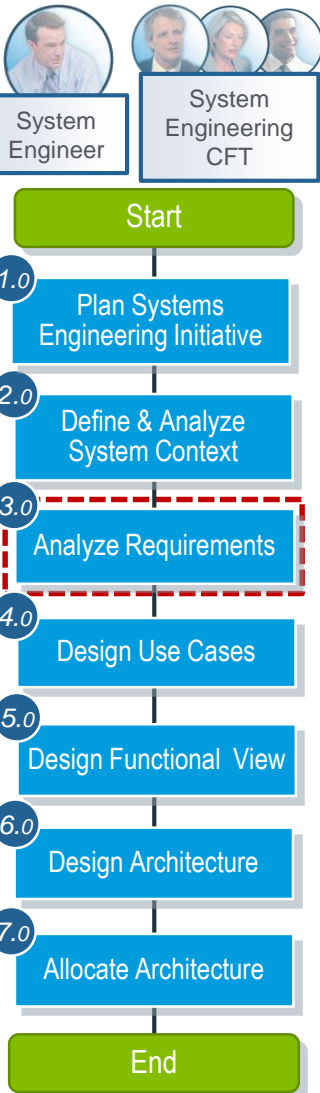
Capture system domain information in consultation with CFT



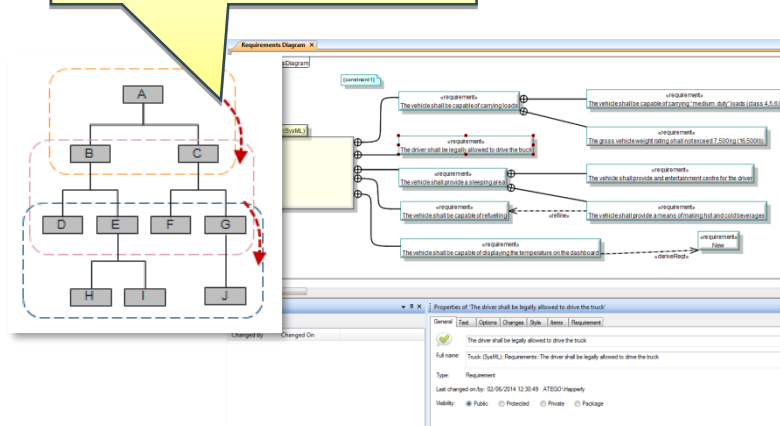
Analyze, refine and update system domain information



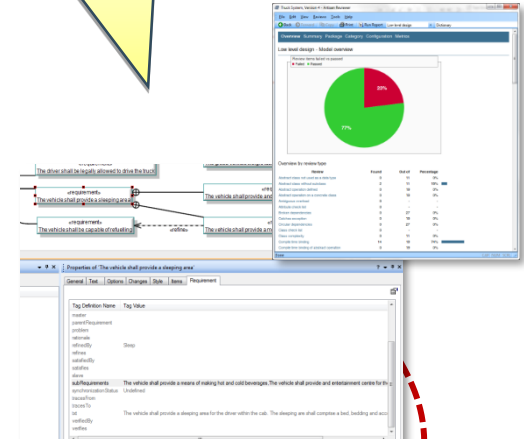
## ► Analyze Requirements



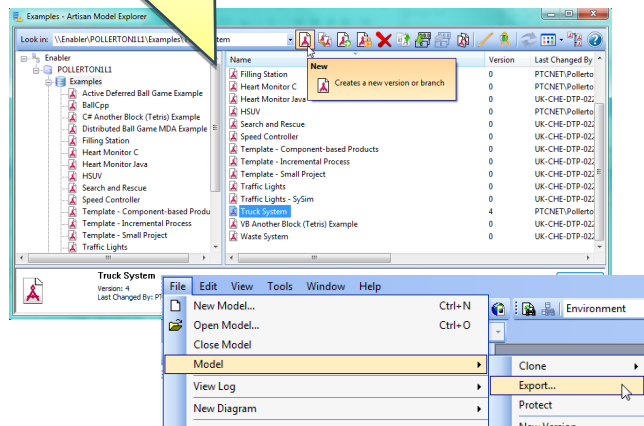
Capture system or sub-system requirements with CFT



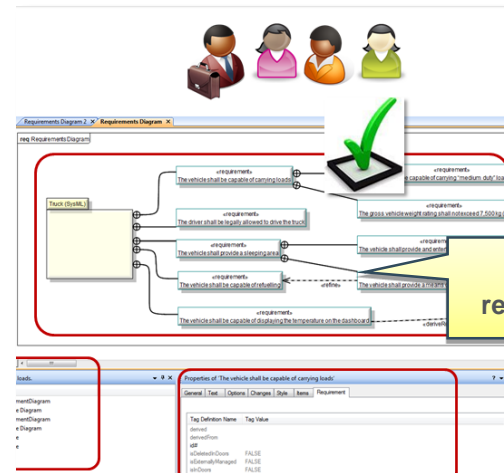
Analyze and refine requirements



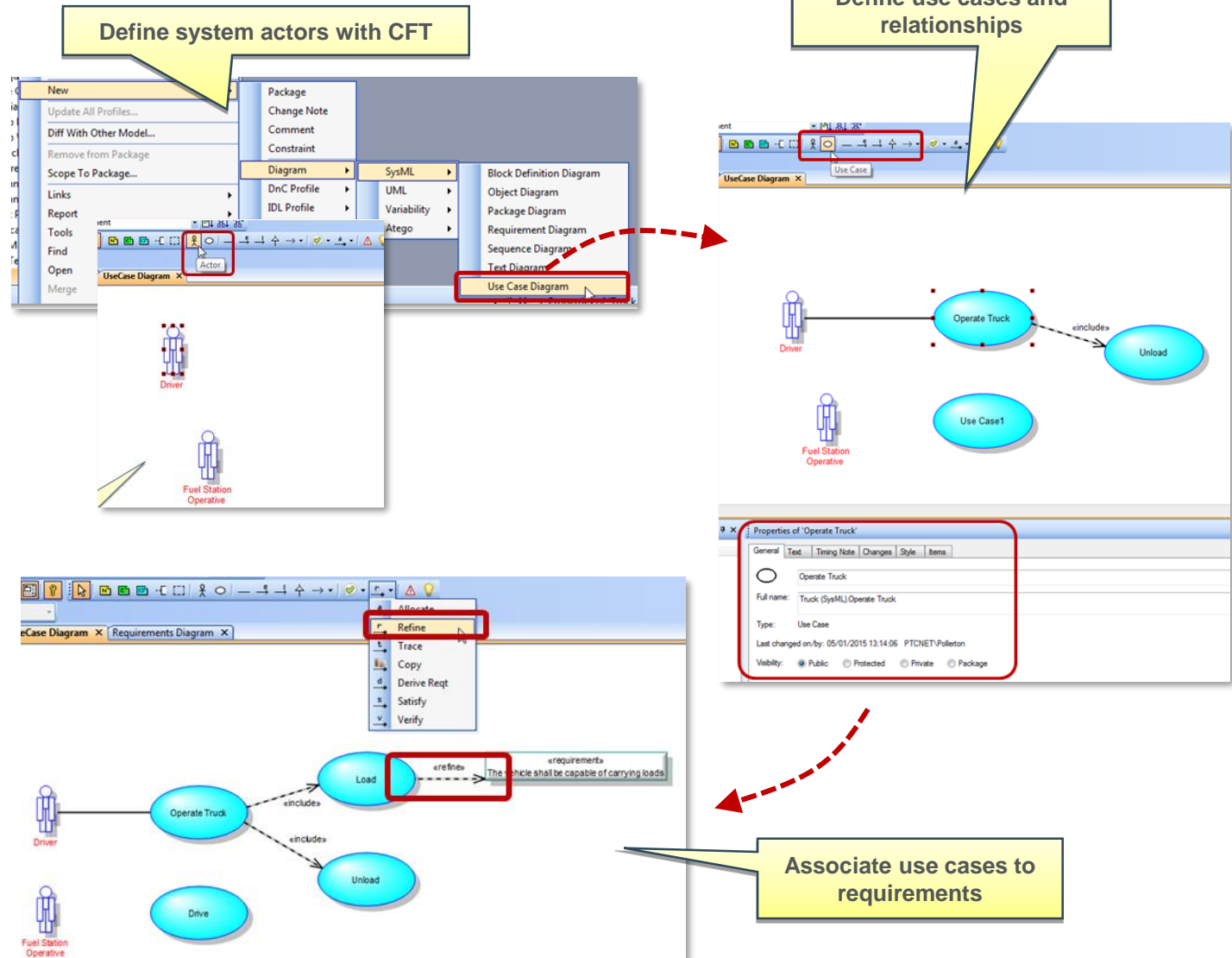
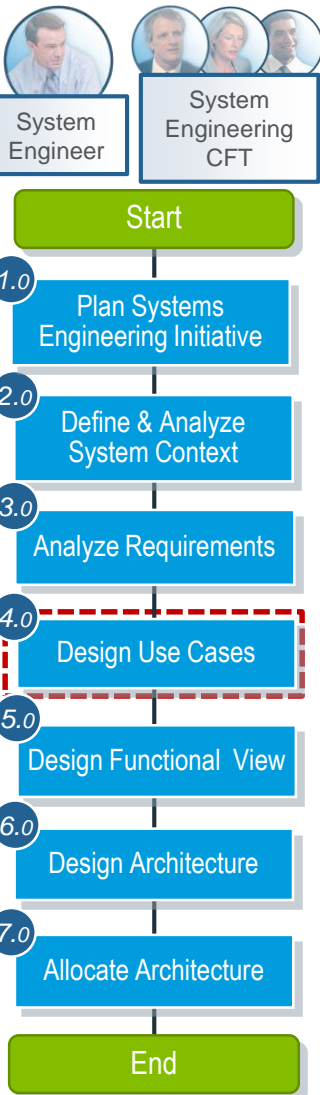
Update and baseline system model



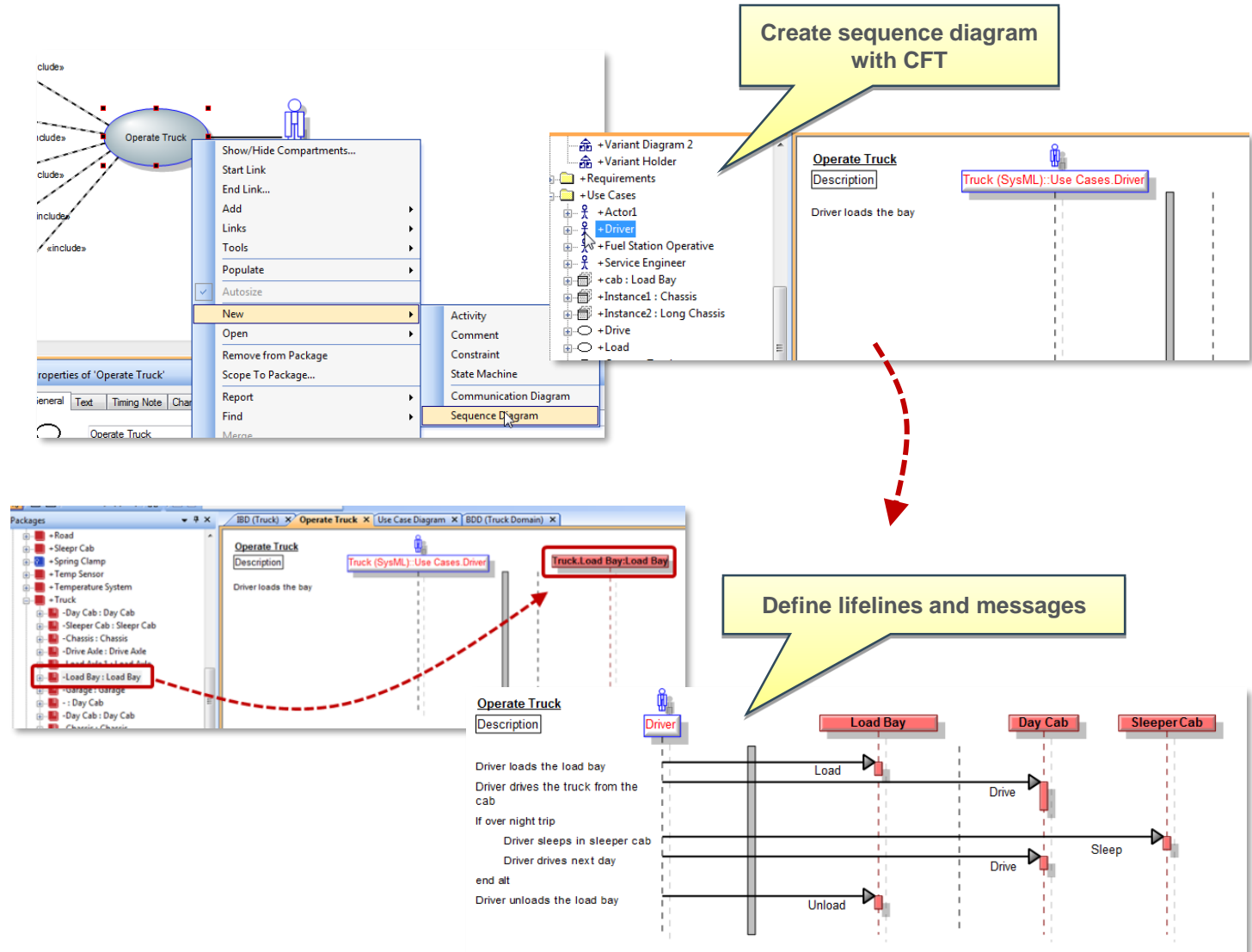
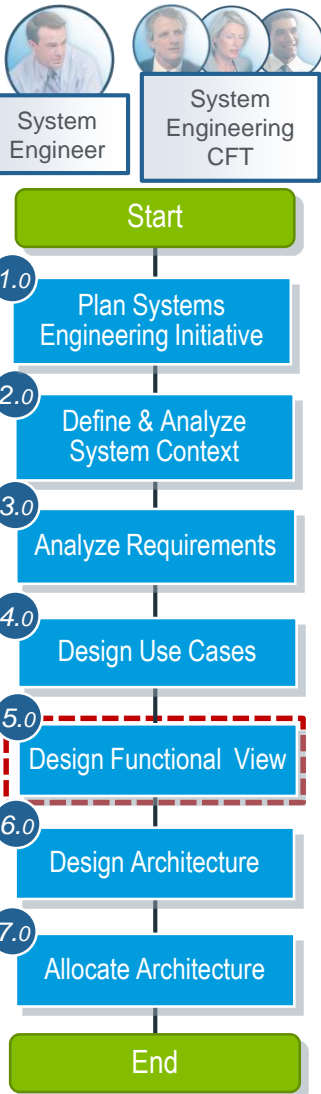
Confirm requirements



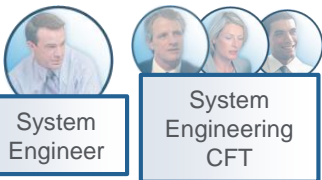
## ► Design Use Cases



## ► Design Functional View



## ► Design Architecture



Start

1.0 Plan Systems Engineering Initiative

2.0 Define & Analyze System Context

3.0 Analyze Requirements

4.0 Design Use Cases

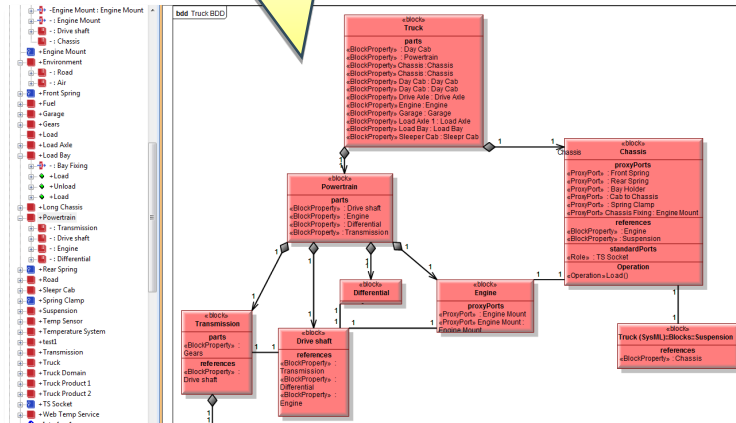
5.0 Design Functional View

6.0 Design Architecture

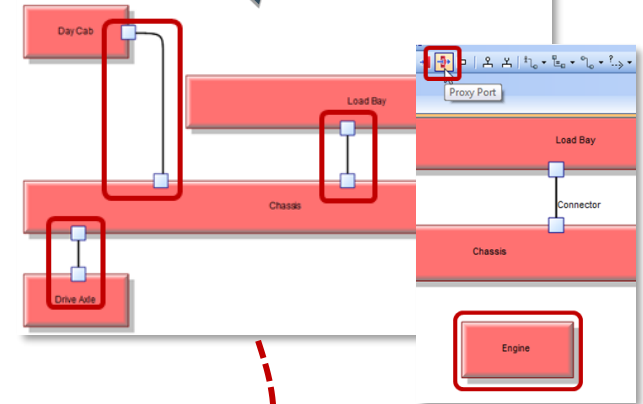
7.0 Allocate Architecture

End

Define system blocks



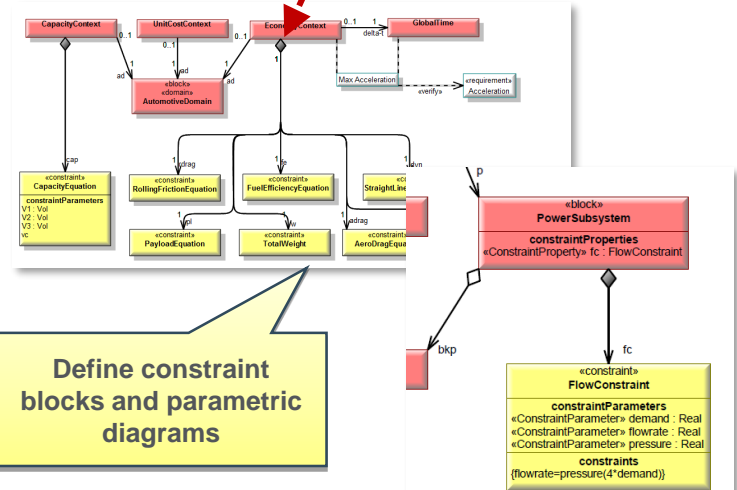
Define internal blocks, ports and item flows



Review system architecture design

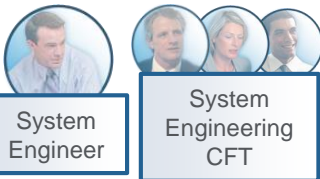


Define constraint blocks and parametric diagrams





## ► Allocate Architecture



Start

1.0 Plan Systems Engineering Initiative

2.0 Define & Analyze System Context

3.0 Analyze Requirements

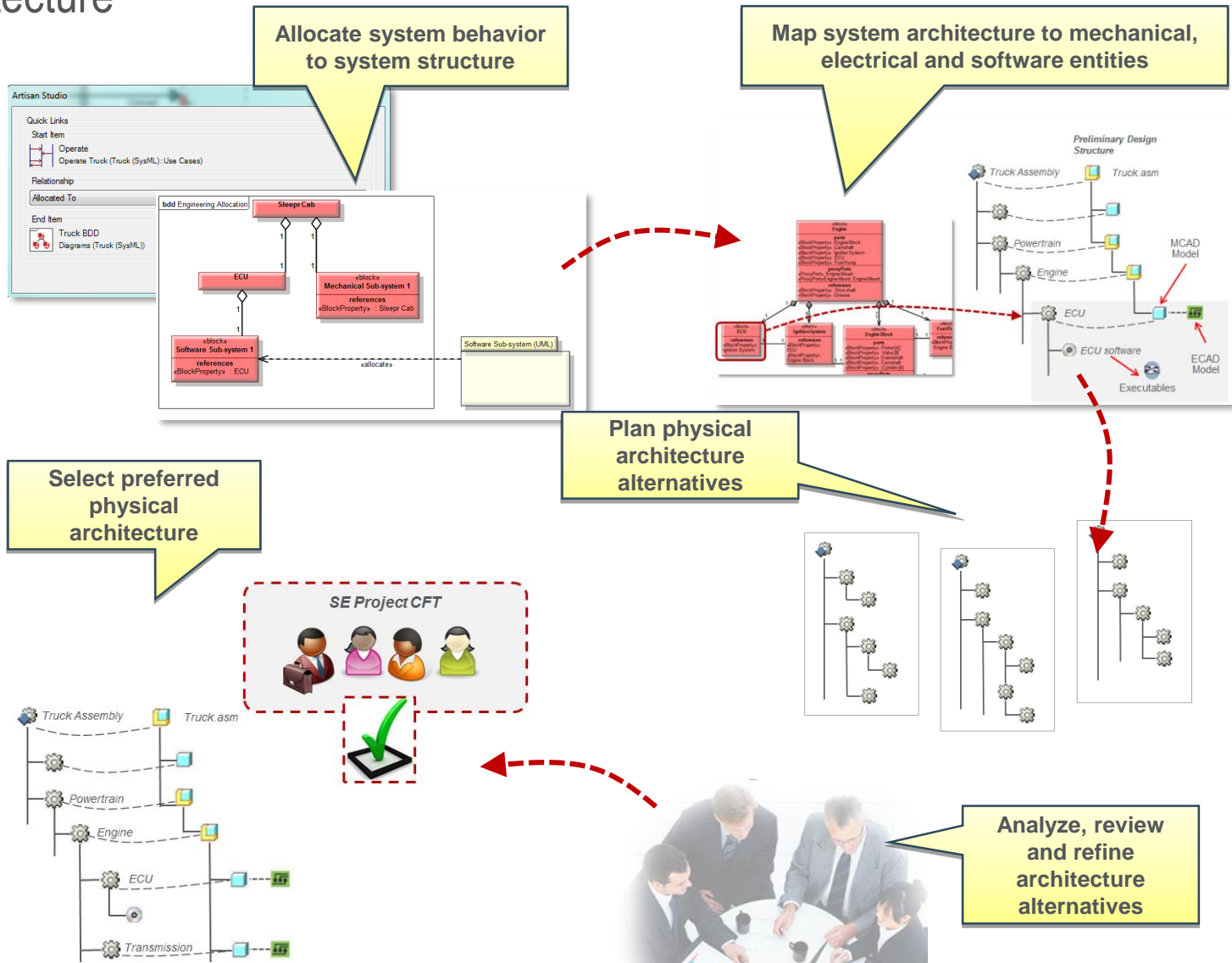
4.0 Design Use Cases

5.0 Design Functional View

6.0 Design Architecture

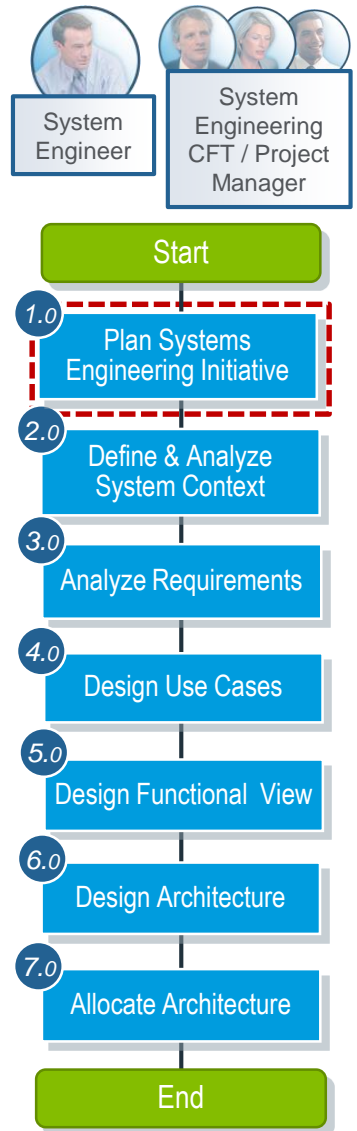
7.0 Allocate Architecture

End



# Plan Systems Engineering Initiative

## ► Plan Systems Engineering Initiative



- Objectives

- Define a plan for the systems engineering program, project or phase

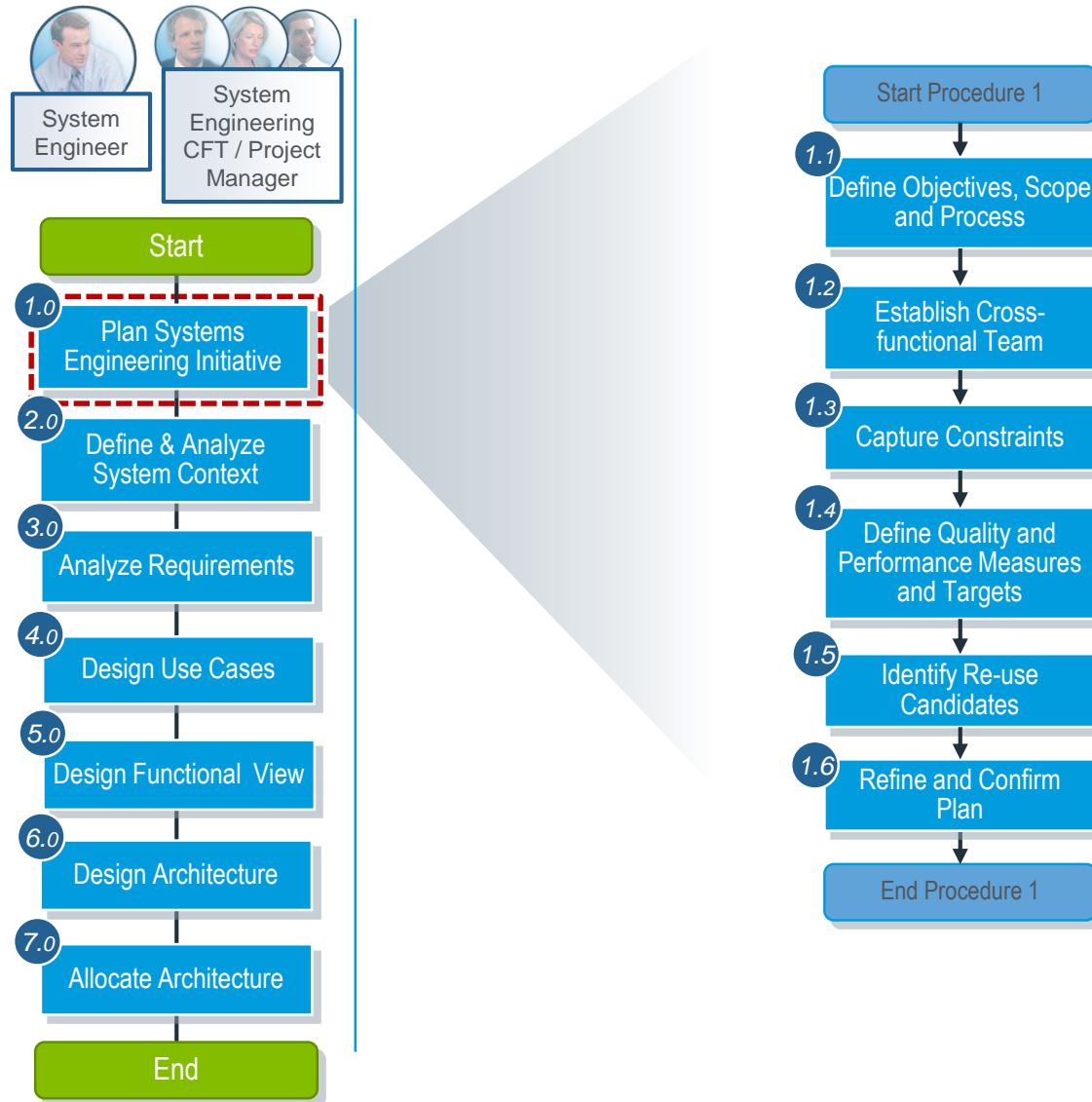
- Role

- Project/Program Manager
- System Engineer
- Cross-functional Team

- Outputs

- Plan for the systems engineering initiative covering objectives, scope, process, constraints, domain, resources, re-use opportunities and quality/performance measures and targets

## ► Plan Systems Engineering Initiative



## ► Define Objectives and Scope

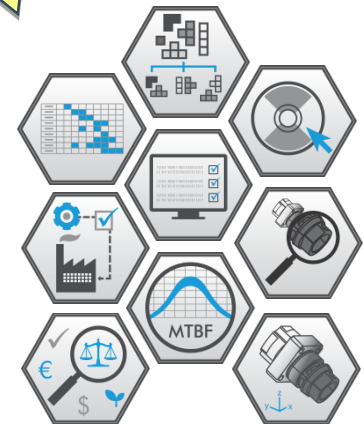
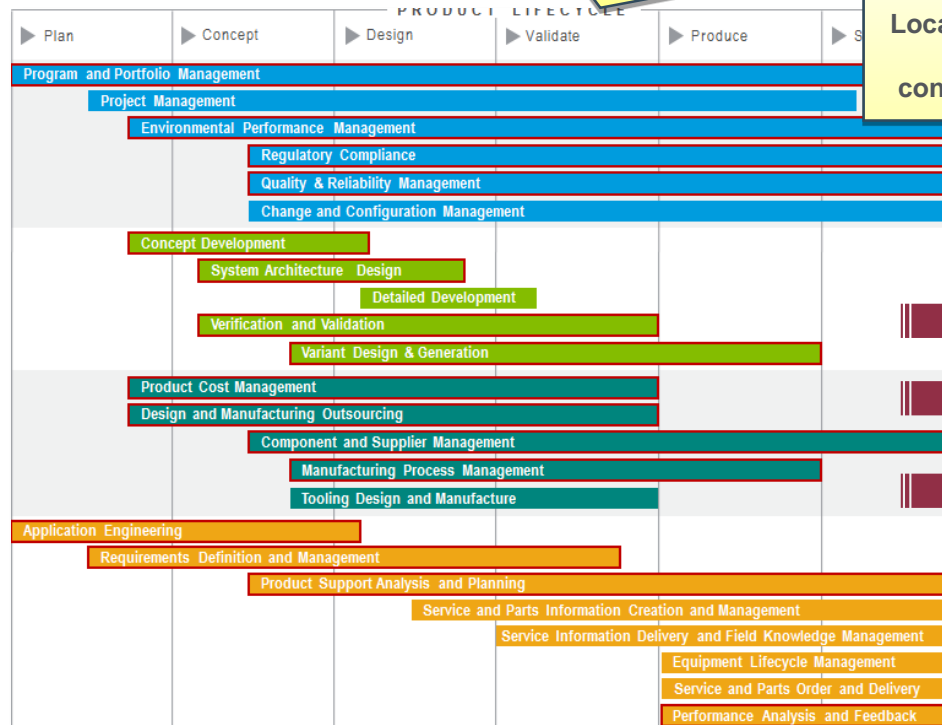


There are many different starting points for a Systems Engineering initiative:

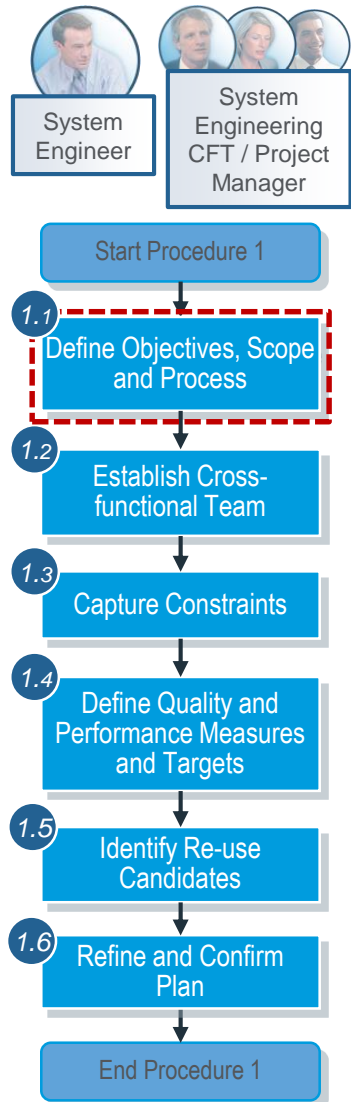
- Development of a new product or product platform
- Adapting existing product or platform
- New requirements or regulations

Therefore it is important to coordinate with related disciplines and processes

Locate and verify any existing requirements, objectives, scope, domain, process, constraints, performance measures/targets



## ► Define Objectives and Scope



In coordination with other processes, define draft objectives, scope and process for the systems engineering project



Refer to PTC System Engineering Process Governance practices for guidance on lifecycle process, PTC Integrity Process Perspective and Process Director

PTC Integrity™ Process Perspective™



Source: INCOSE, Wikipedia

responsibility is creating and executing an interdisciplinary process to ensure that the customer and stakeholder's needs are met in a compliant manner throughout a system's entire life cycle. This process is usually comprised of the following seven tasks: State the problem, Investigate alternatives, Model the system, Integrate, Launch the system, Assess performance, and Re-evaluate. These functions can be summarized with the acronym SIMILAR: State, Investigate, Model, Integrate, Launch, Assess and Re-evaluate. This Systems Engineering Process is shown in Figure 1. It is important to note that the Systems Engineering Process is not sequential. The functions are performed in a parallel and iterative manner.

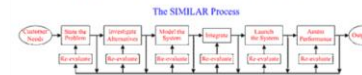
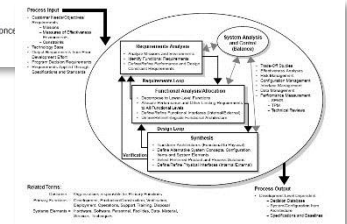


Figure 1. The Systems Engineering Process from A. T. Bahill and B. Glesing, Re-evaluating systems engineering concepts, Part C: Applications and Reviews, 28 (4), 516-527, 1998.

State the problem

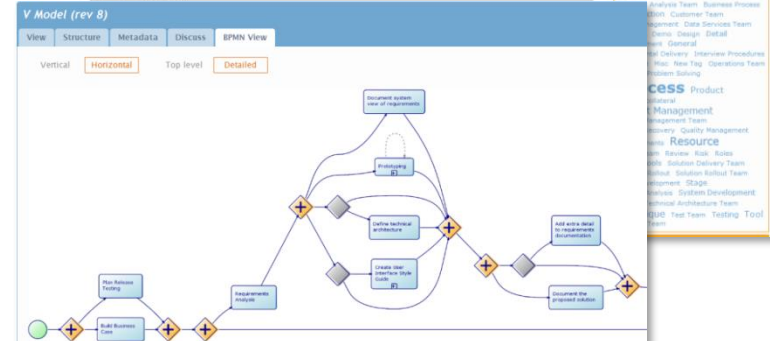


PTC Integrity™ Process Director  
uk-hom-ltp-003/ategoprocessdirectorwebdemo

Library summary

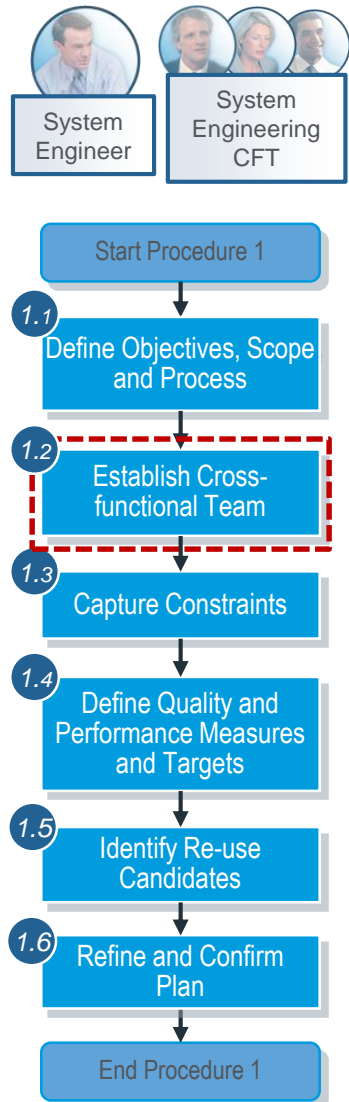
Process library maturity

- 2 definitions accessed this month
- 265 definitions created in total
- 262 definitions created or updated this month
- 0 metadata values entered this month





## ► Establish Cross-functional Team



Ensure representatives from all affected departments are members of the cross-functional team. This may include:

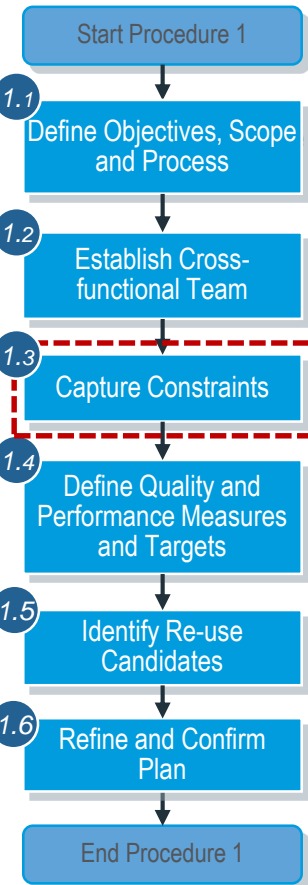
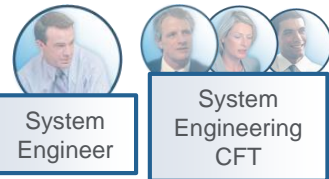
- Marketing / Sales
- System Engineering
- Software / Electrical / Mechanical Engineering
- Manufacturing
- Service

### Note

Additional input may also be obtained from external parties such as suppliers, partners, regulatory agencies and customers but these are not part of the core CFT.

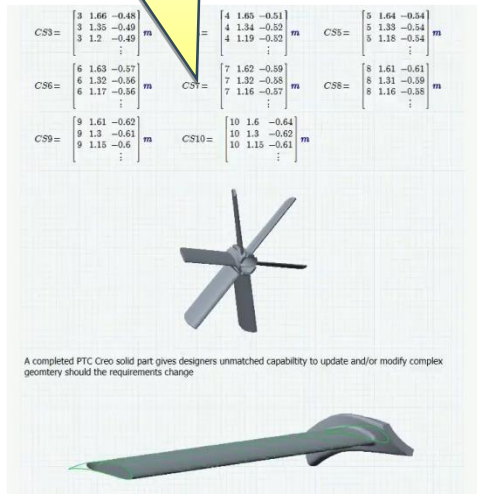


## ► Capture Constraints



**Confirm constraints such as regulatory compliance and company or industry standards**

**Include any known engineering limitations or constraints**



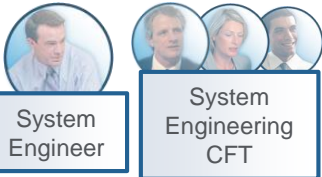
PTC Mathcad



**Note**

**Also consider business constraints such as project deadlines and milestones, or release/sales schedule**

## ► Define Quality and Performance Targets



Start Procedure 1

1.1 Define Objectives, Scope and Process

1.2 Establish Cross-functional Team

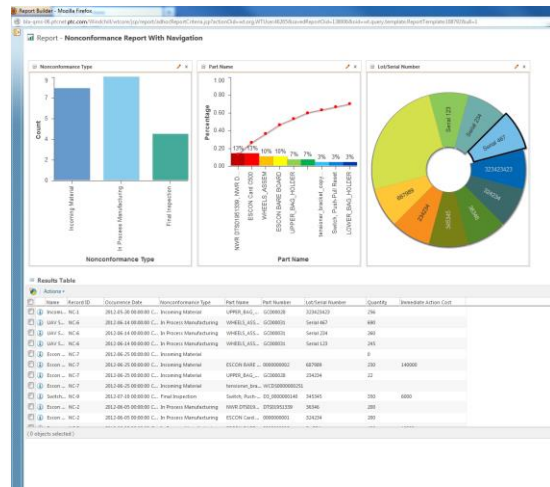
1.3 Capture Constraints

1.4 Define Quality and Performance Measures and Targets

1.5 Identify Re-use Candidates

1.6 Refine and Confirm Plan

End Procedure 1



Coordinate with Quality Mgmt processes and define quality and performance measures

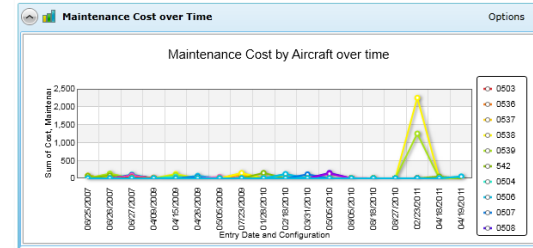
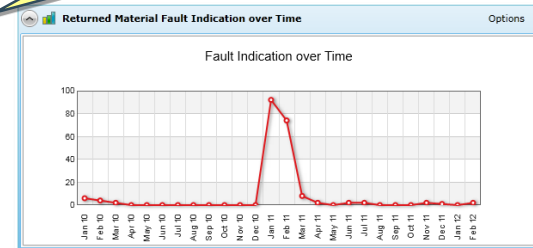
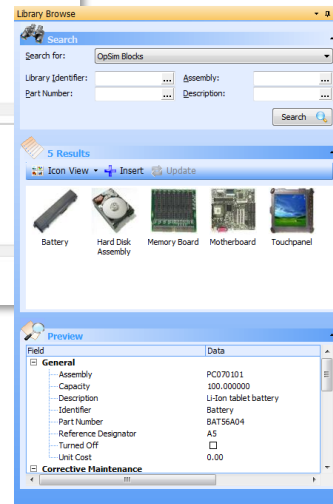
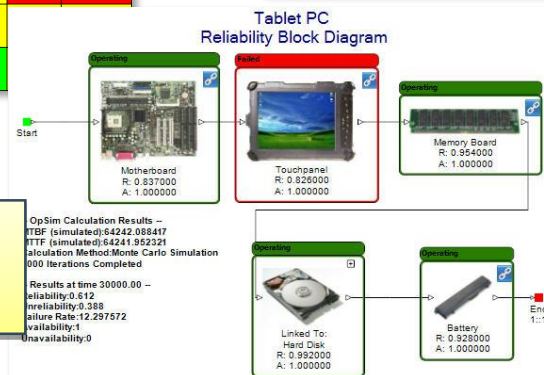


If possible also set quality and performance targets

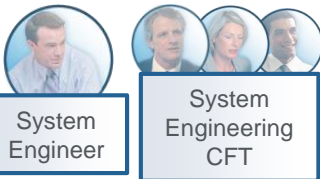
	Negligible effect	Minor effect	Moderate effect	Critical effect	Catastrophic effect
Frequent	Moderate Risk	Moderate Risk	High Risk	High Risk	High Risk
Probable	Low Risk	Moderate Risk	High Risk	High Risk	High Risk
Occasional	Low Risk	Moderate Risk	Moderate Risk	High Risk	High Risk
Remote	Low Risk	Low Risk	Moderate Risk		
Improbable	Low Risk	Low Risk	Low Risk		

Note

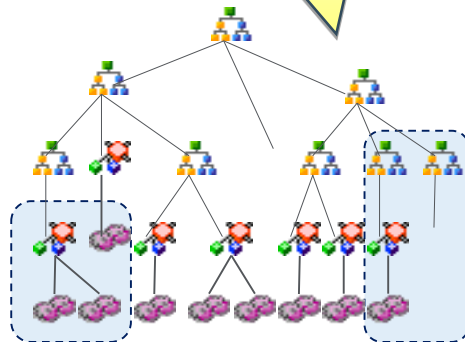
This is an advanced, optional technique



## ► Identify Re-use Candidates

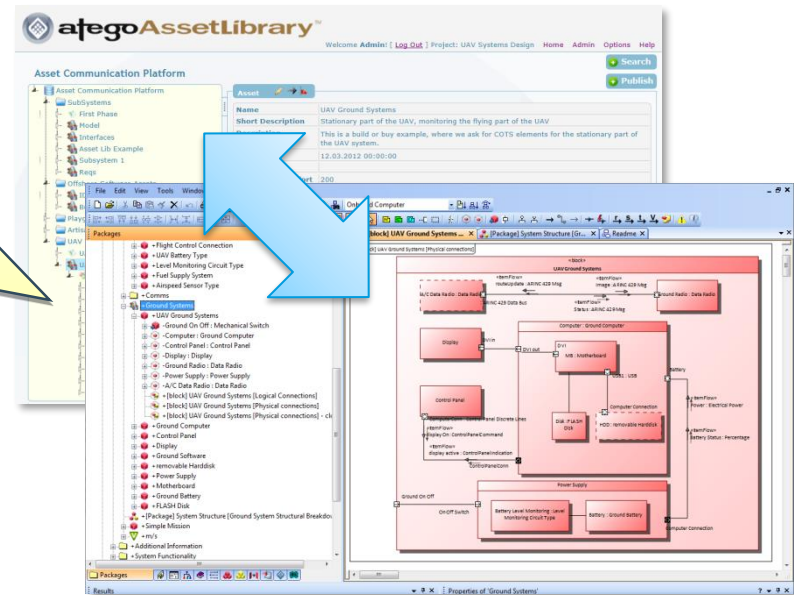
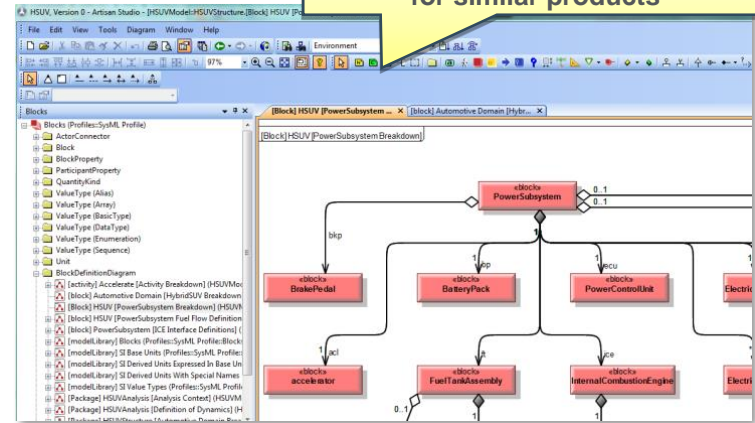


Identify re-use candidates from existing systems or designs and review with CFT



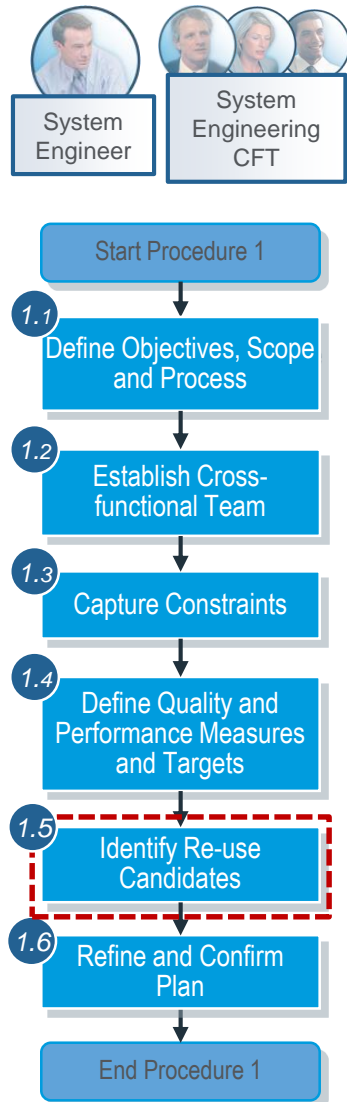
If using the Asset Library, evaluate any existing components or assets that may be re-used. Refer to the Architected Modular Design Best Practice for more information

Consider existing SysML models for similar products

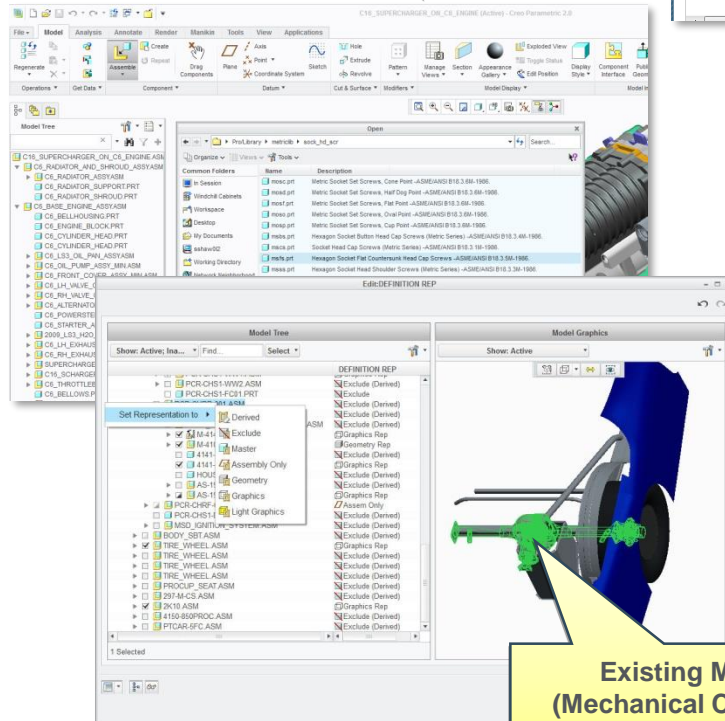




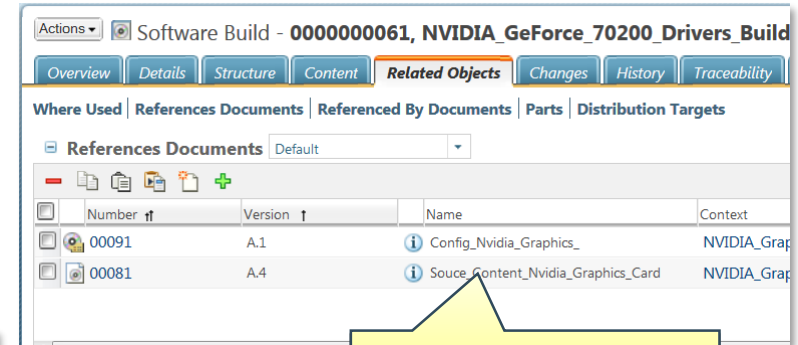
## ► Identify Re-use Candidates



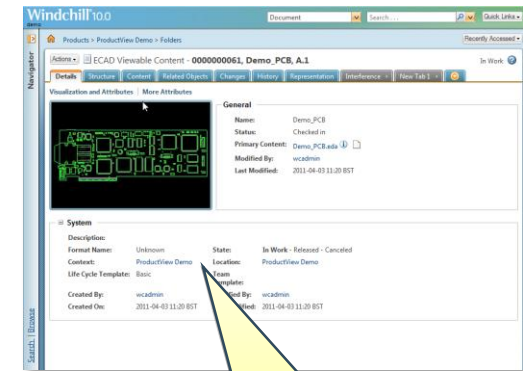
Evaluate existing mechanical, electrical of software designs for possible re-use



Existing MCAD (Mechanical Computer Aided Design) designs

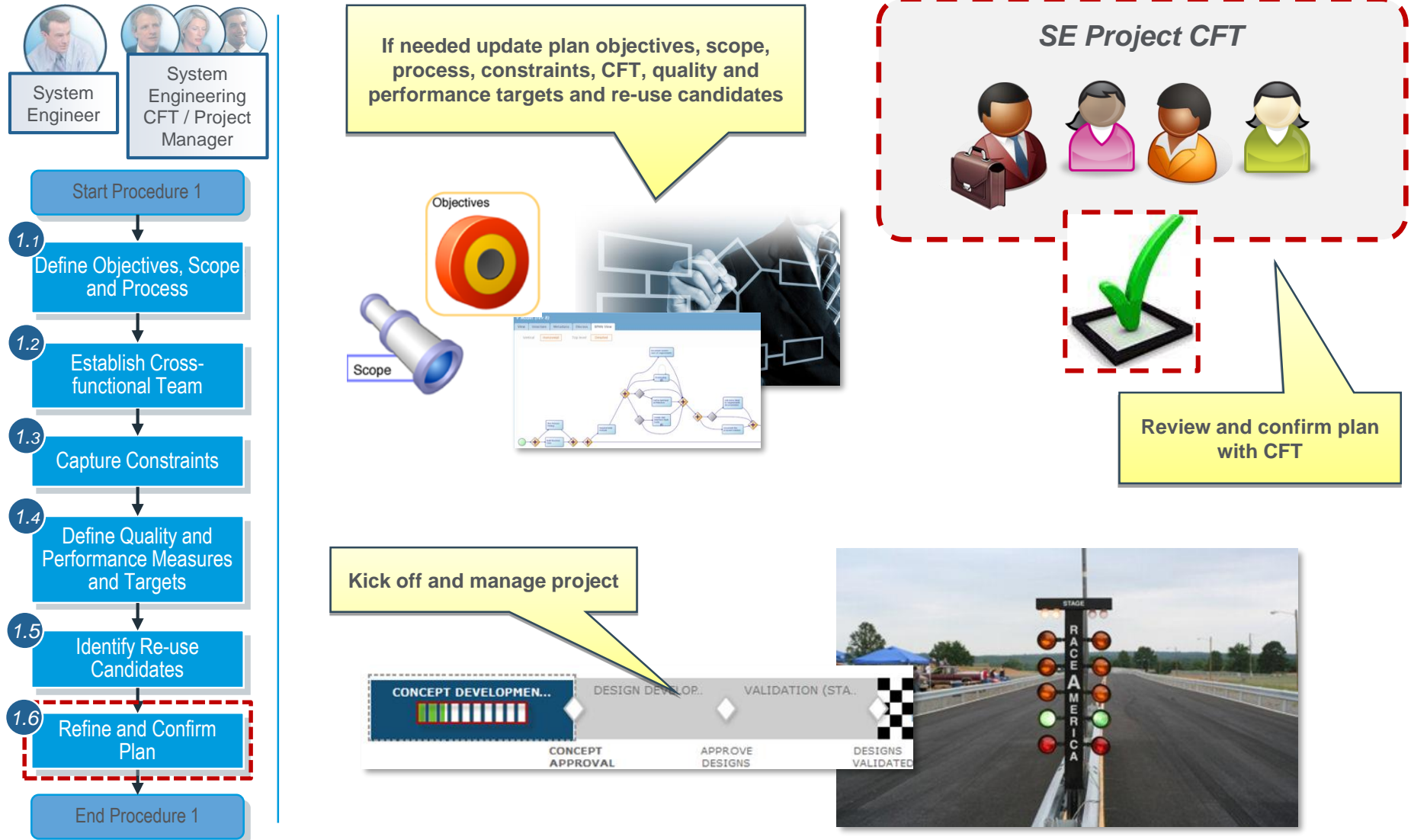


Existing software designs



Existing ECAD (Electrical Computer Aided Design) designs

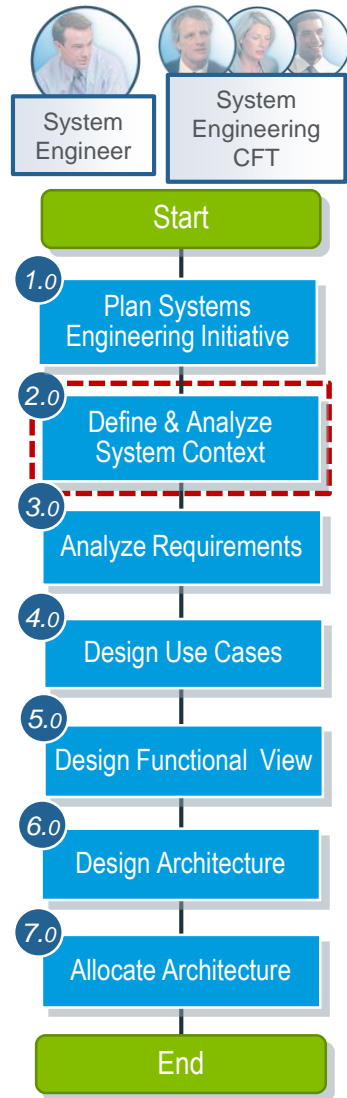
## ► Refine and Confirm Plan





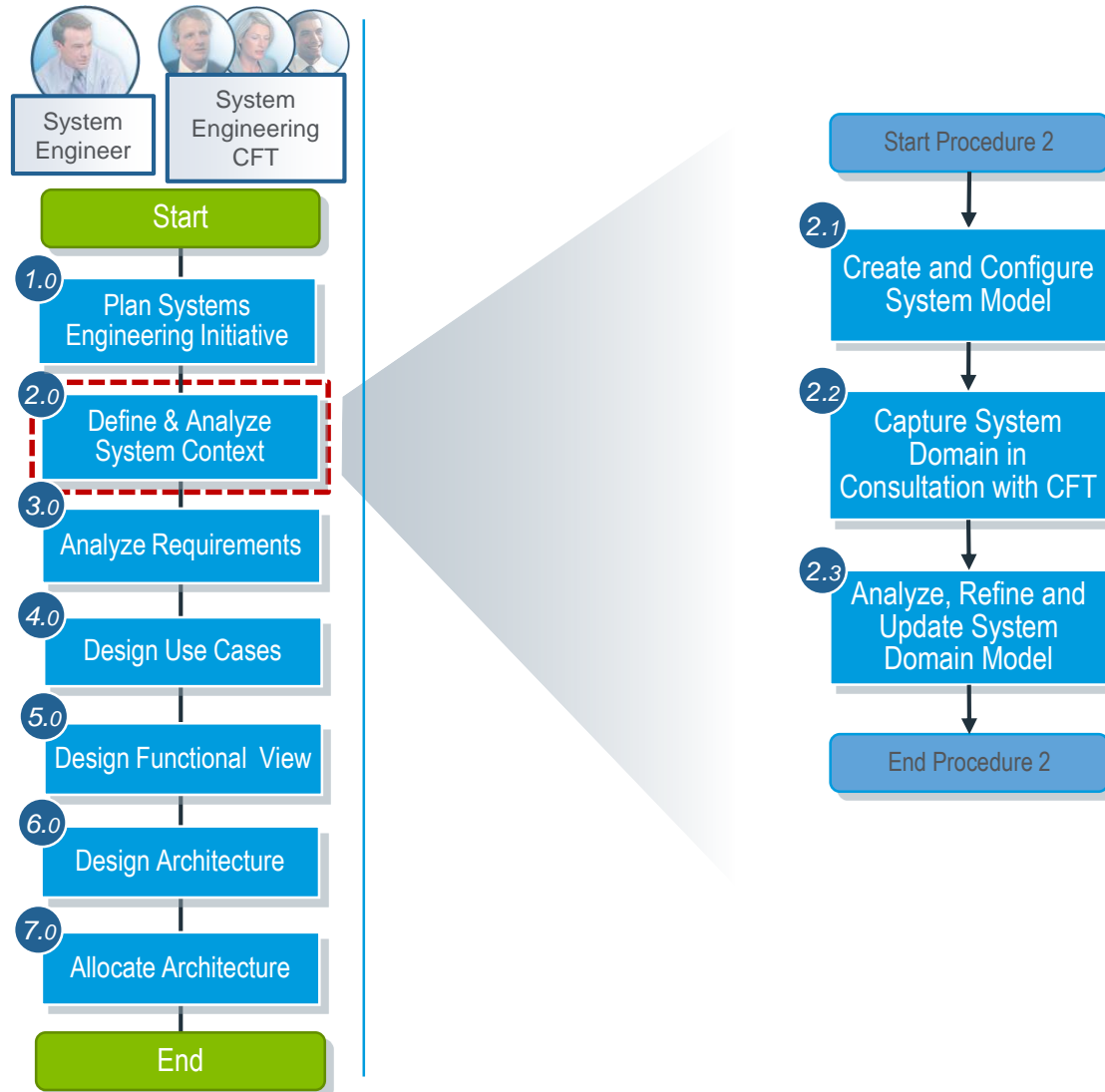
# Define and Analyze System Context

## ► Define and Analyze System Context

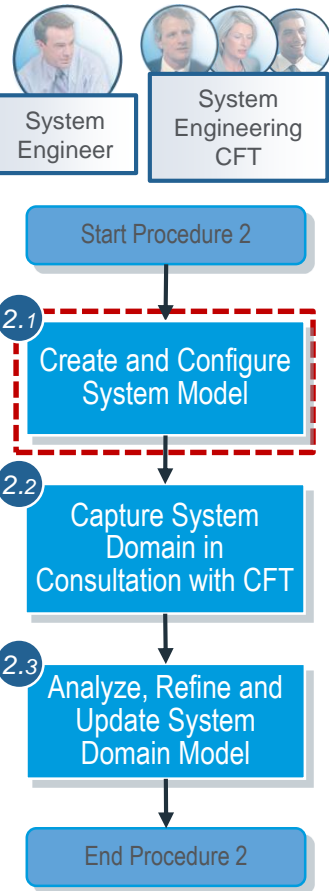


- Objectives
  - Capture, document and analyze system domain information
- Role
  - System Engineer
  - Cross-functional Team
- Outputs
  - System Domain Model

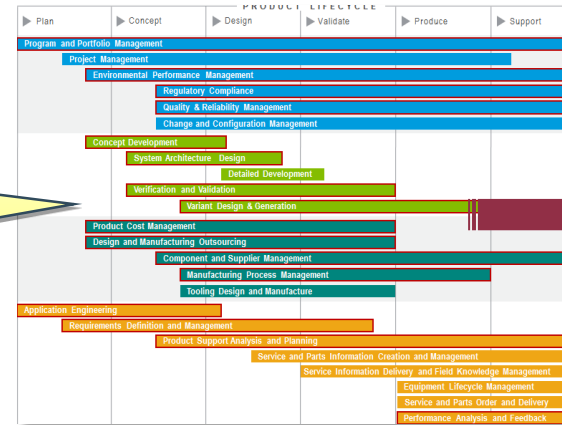
## ► Define and Analyze System Context



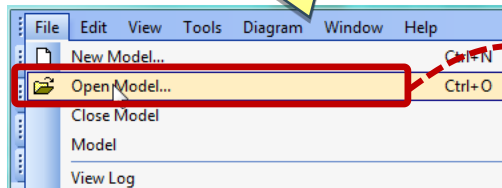
## ► Create and Configure System Model



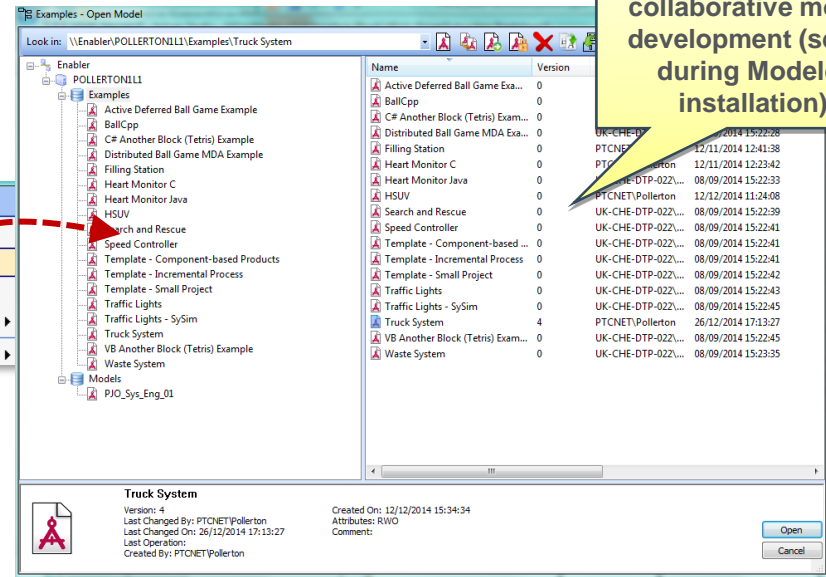
Coordinate with related processes such as Requirements Management, Quality Mgmt and identify any relevant existing information



Select File > Open Model... to open Enabler and review existing Repositories and Models



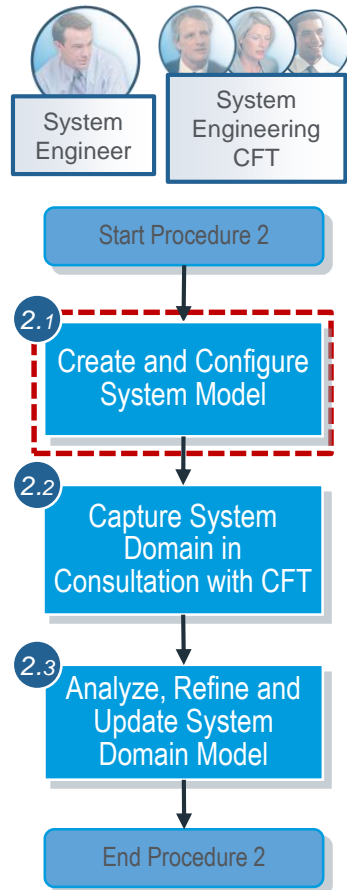
Enabler supports collaborative model development (setup during Modeler installation)



**Note**

If using Asset Library, this may be a collection of linked models for systems and sub-systems

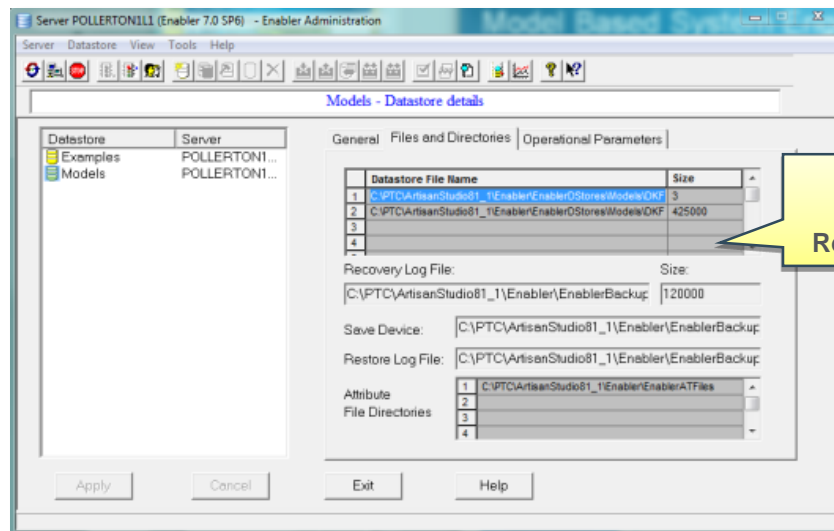
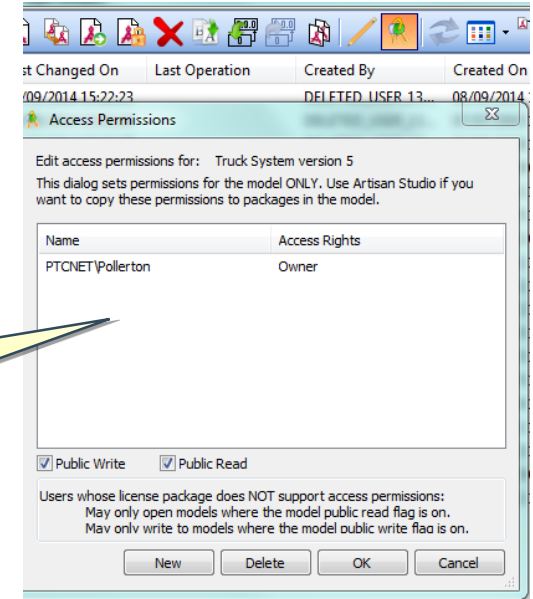
## ► Create and Configure System Model



Notify those involved in the modelling activity which model(s) and version is being used and how to access

Traffic Lights - Sysim	0	UK-CHE-DTP-022\...	08/09/2014 15:22:45
Truck System	4	PTCNET\Pollerton	26/12/2014 17:13:27
VB Another Block (Tetris) Exam	0	UK-CHE-DTP-022\...	08/09/2014 15:22:45

Access permissions on models can be defined

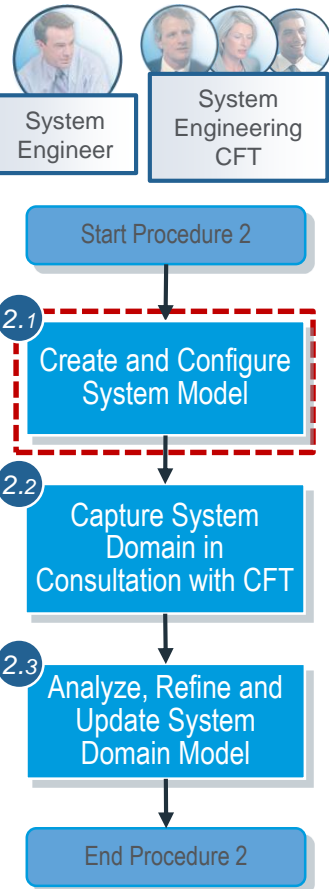


Repositories can be managed using the Repository Administrator

**Note**

Refer to System Model Management Best Practice for more information

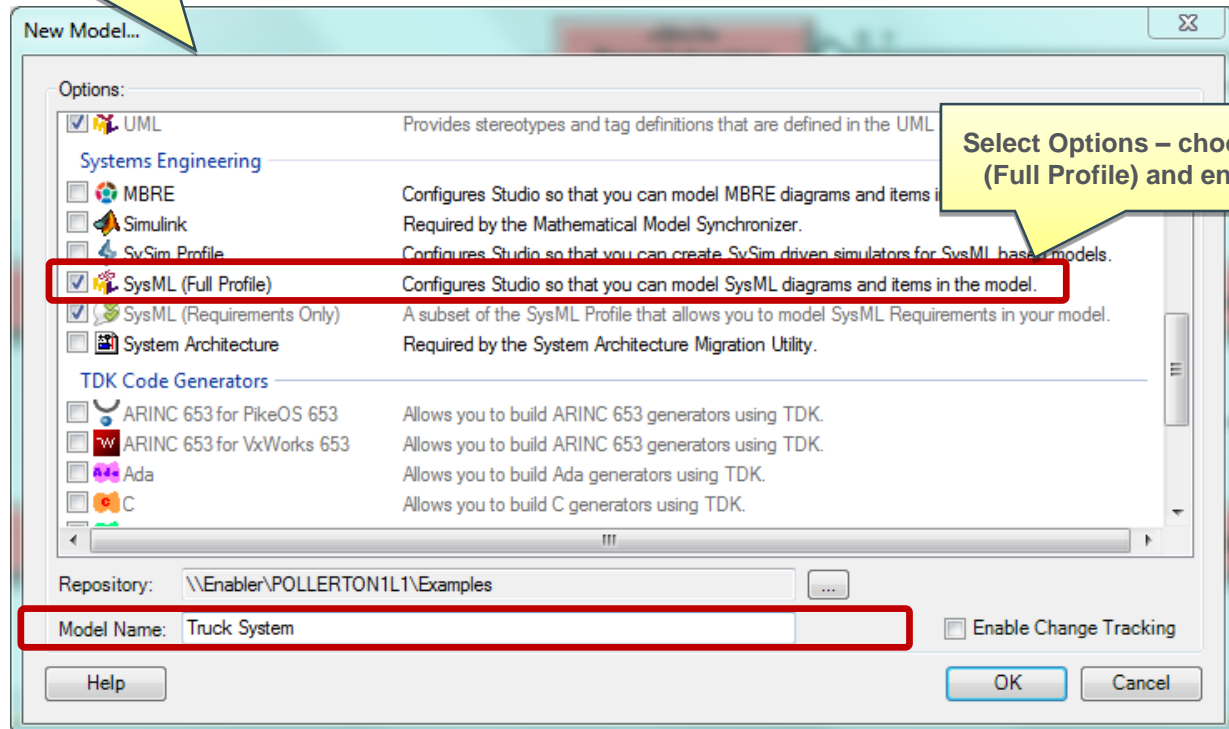
## ► Create and Configure System Model



Select File > New Model to create new system model

### Note

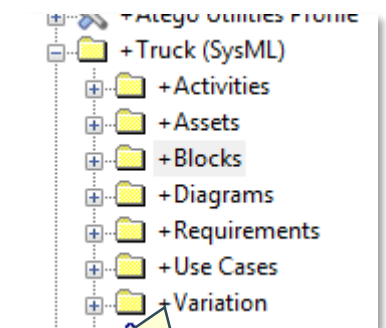
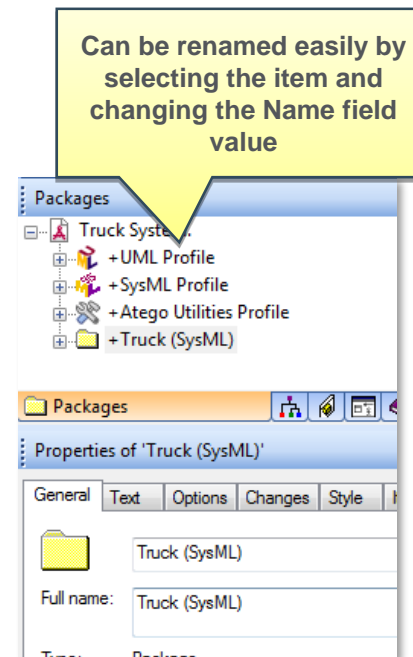
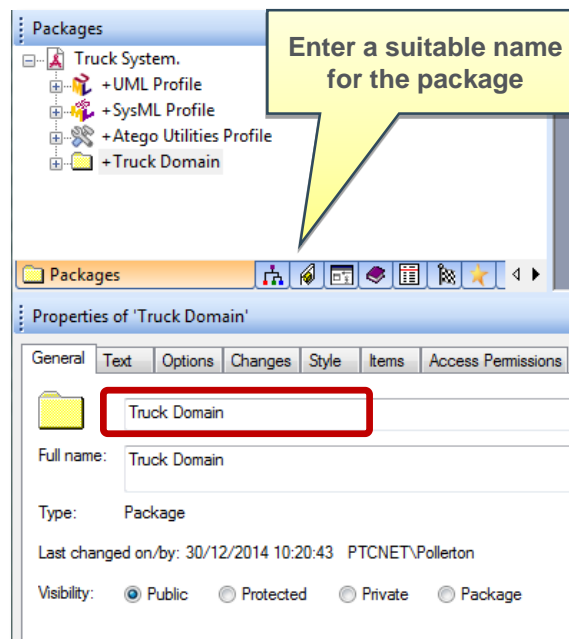
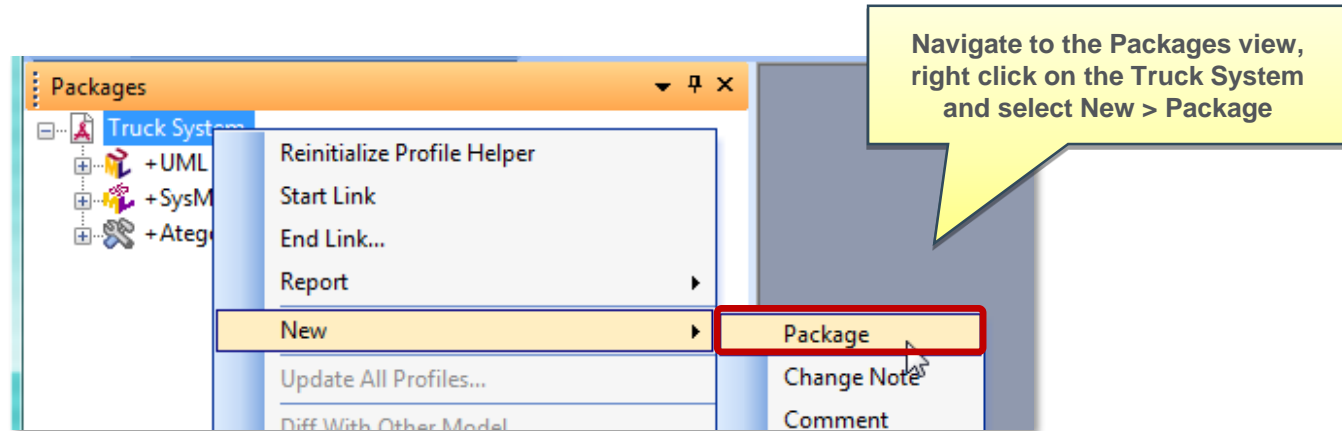
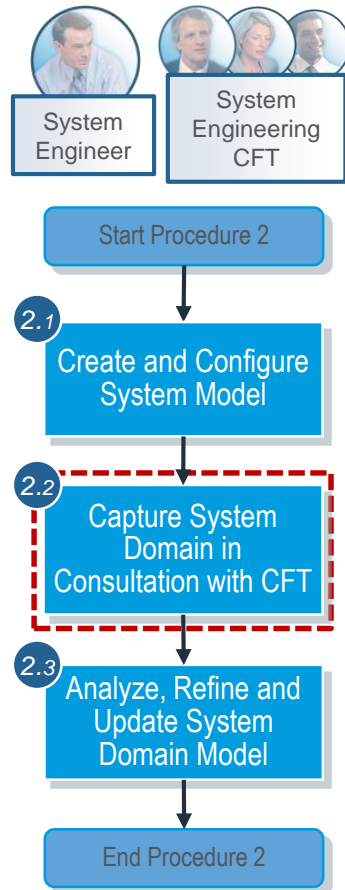
If needed, import the SysML Profile (Tools > Add Profile)



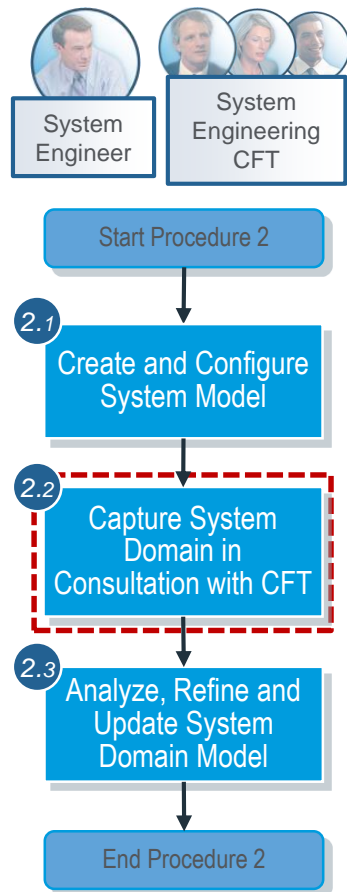
### Note

Profile extensions can be used to support a customer specific data model

## ► Capture System Domain in Consultation with CFT



## ► Capture System Domain in Consultation with CFT



Right click on a package and select New > Diagram > SysML > Block Definition Diagram

New diagram created within the new package

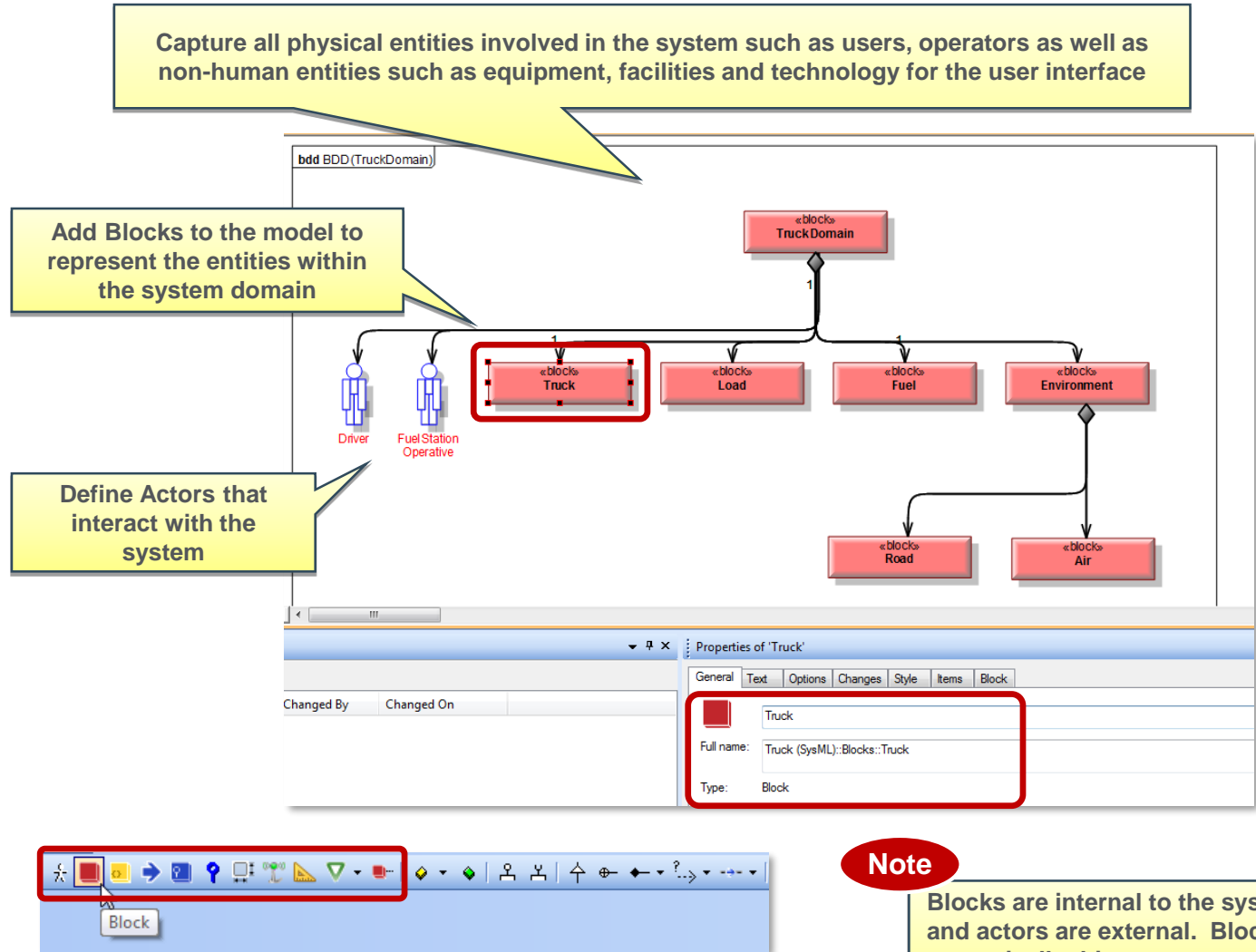
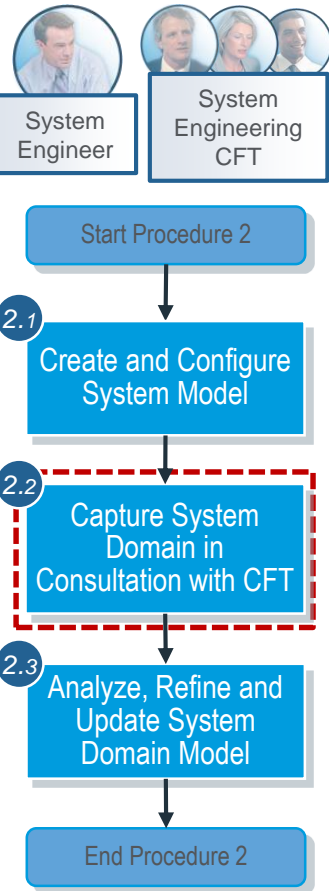
**Note**

Blocks, in SysML, provide a means of describing a system as a hierarchy of modular components, each of which can have a set of structural and/or behavioral features

The screenshot shows the SysML software interface. A context menu is open over the 'Truck (SysML)' package in the 'Packages' tree. The menu path 'New > Diagram > SysML > Block Definition Diagram' is highlighted. A new diagram window titled '[Package] Truck (SysML) [1]' is open, showing a new 'bdd' (Block Definition Diagram) created within the package. The 'Properties' panel at the bottom shows the details of the new diagram, including its full name and page reference.



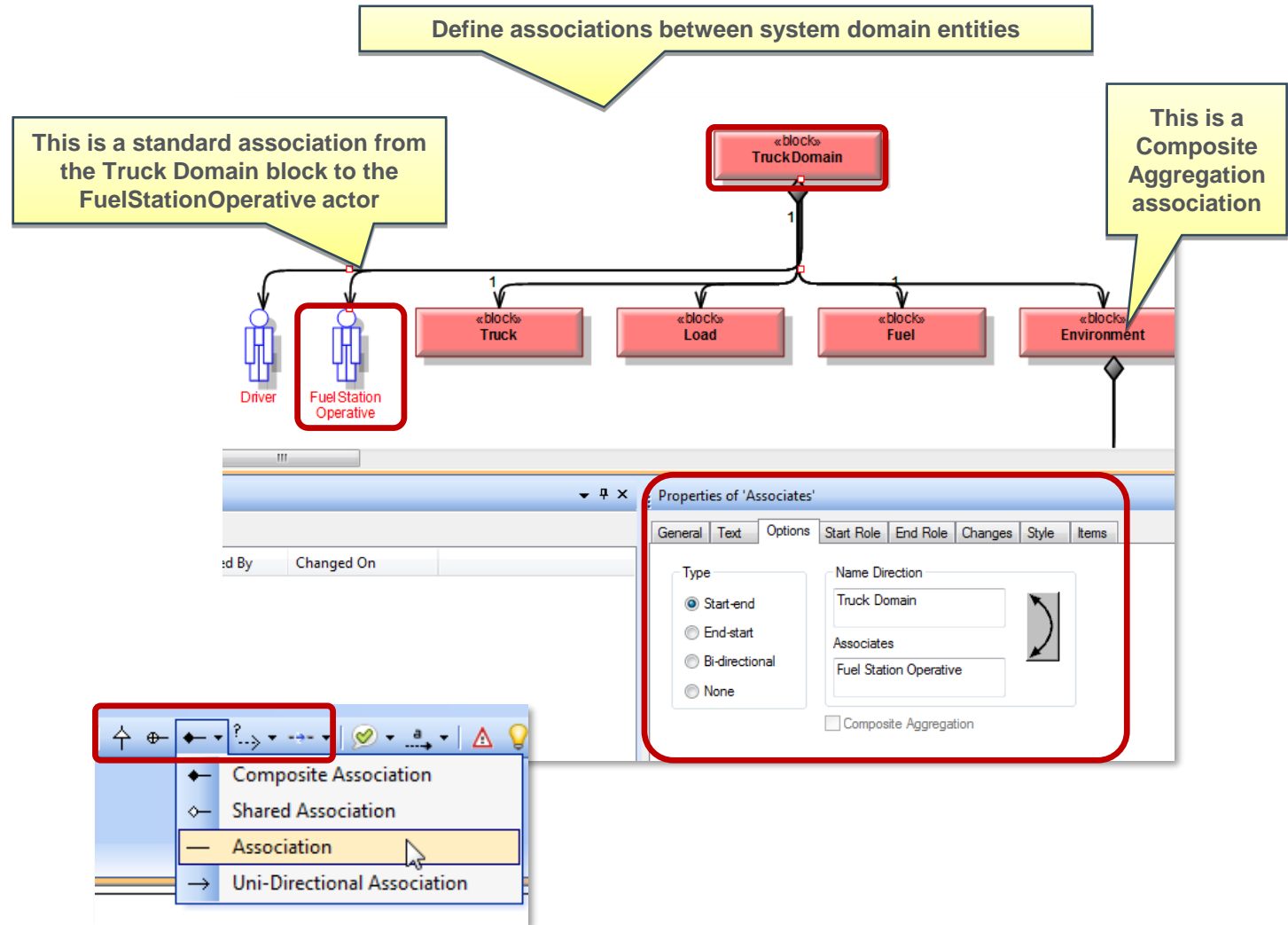
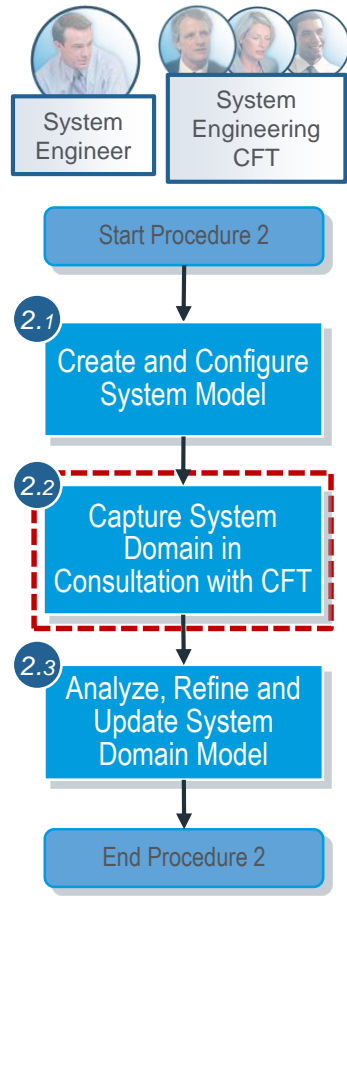
## ► Capture System Domain in Consultation with CFT



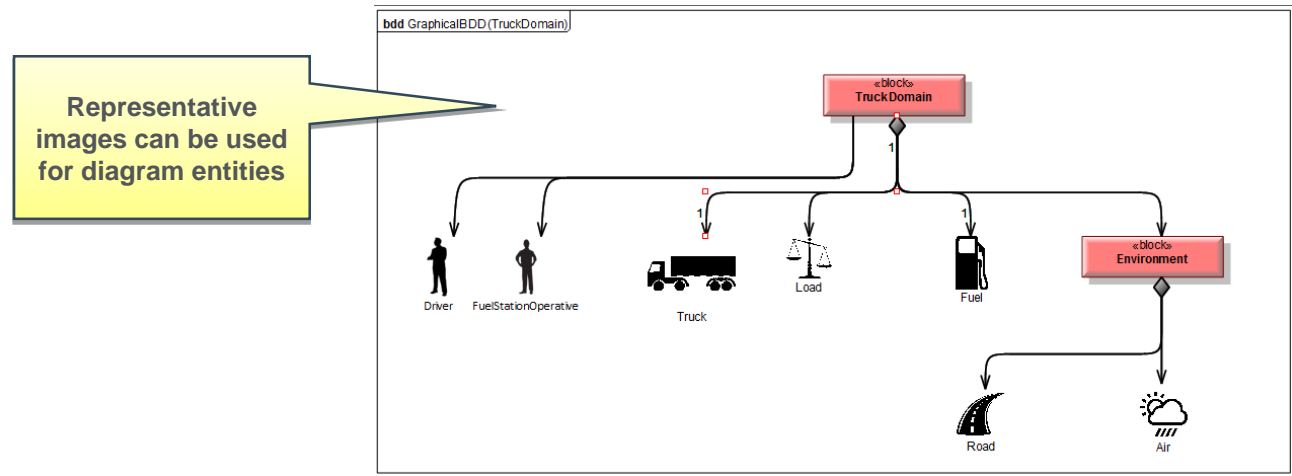
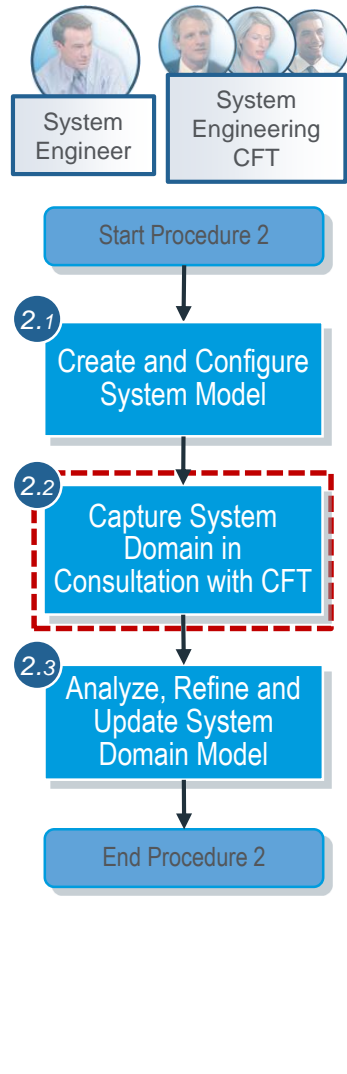
### Note

Blocks are internal to the system and actors are external. Blocks are typically things we can manage or control

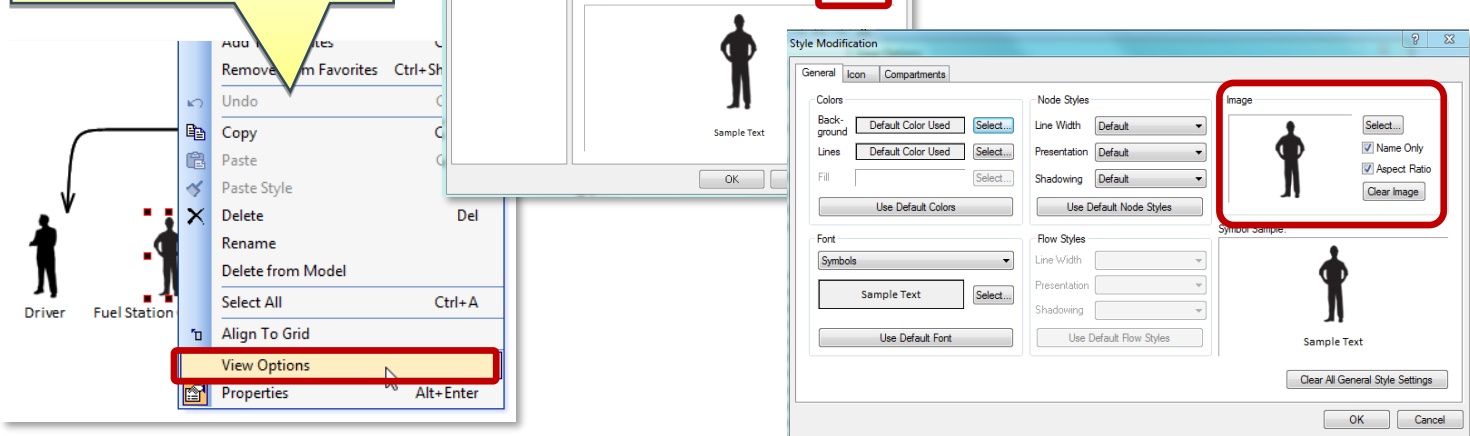
## ► Capture System Domain in Consultation with CFT



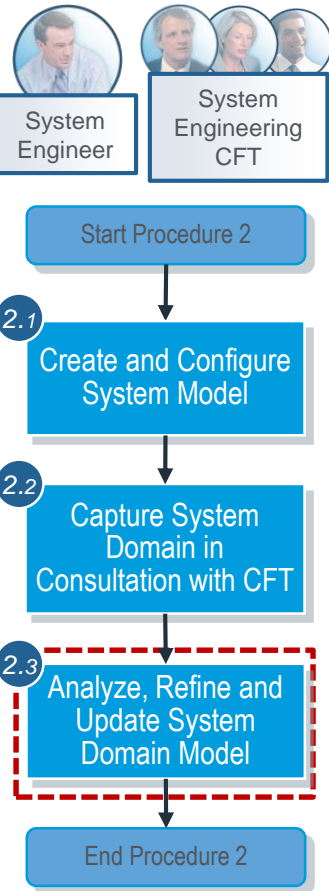
## ► Capture System Domain in Consultation with CFT



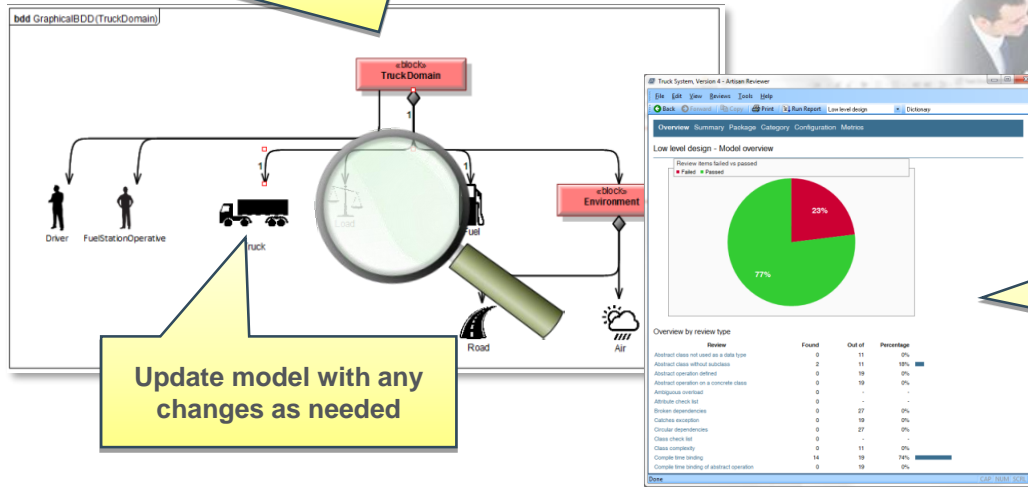
Right click on the entity and select View Options > Edit > Select Image



## ► Analyze, Refine and Update System Model



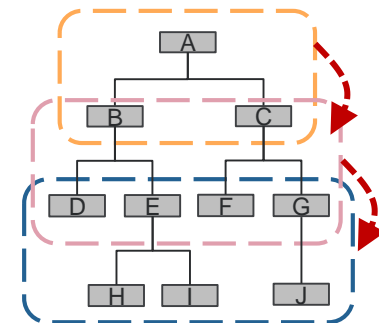
Conduct a review of the system domain model with the CFT to confirm that the model accurately represents the domain in focus



Perform analysis to confirm model is complete and correct (Refer to Automated System Design Review Best Practice)

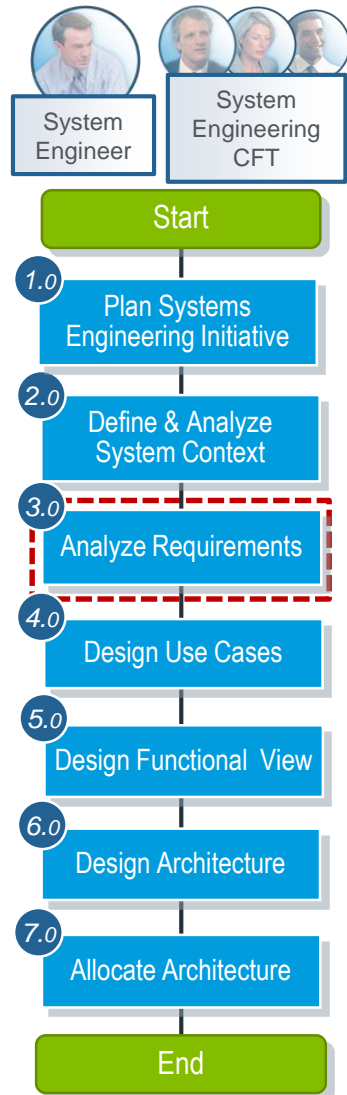
### Note

There may be incomplete information available at this point so the creation of block and internal block diagrams should occur iteratively and recursively as more system and sub-system requirements are identified. Further iterations occur in the following procedures



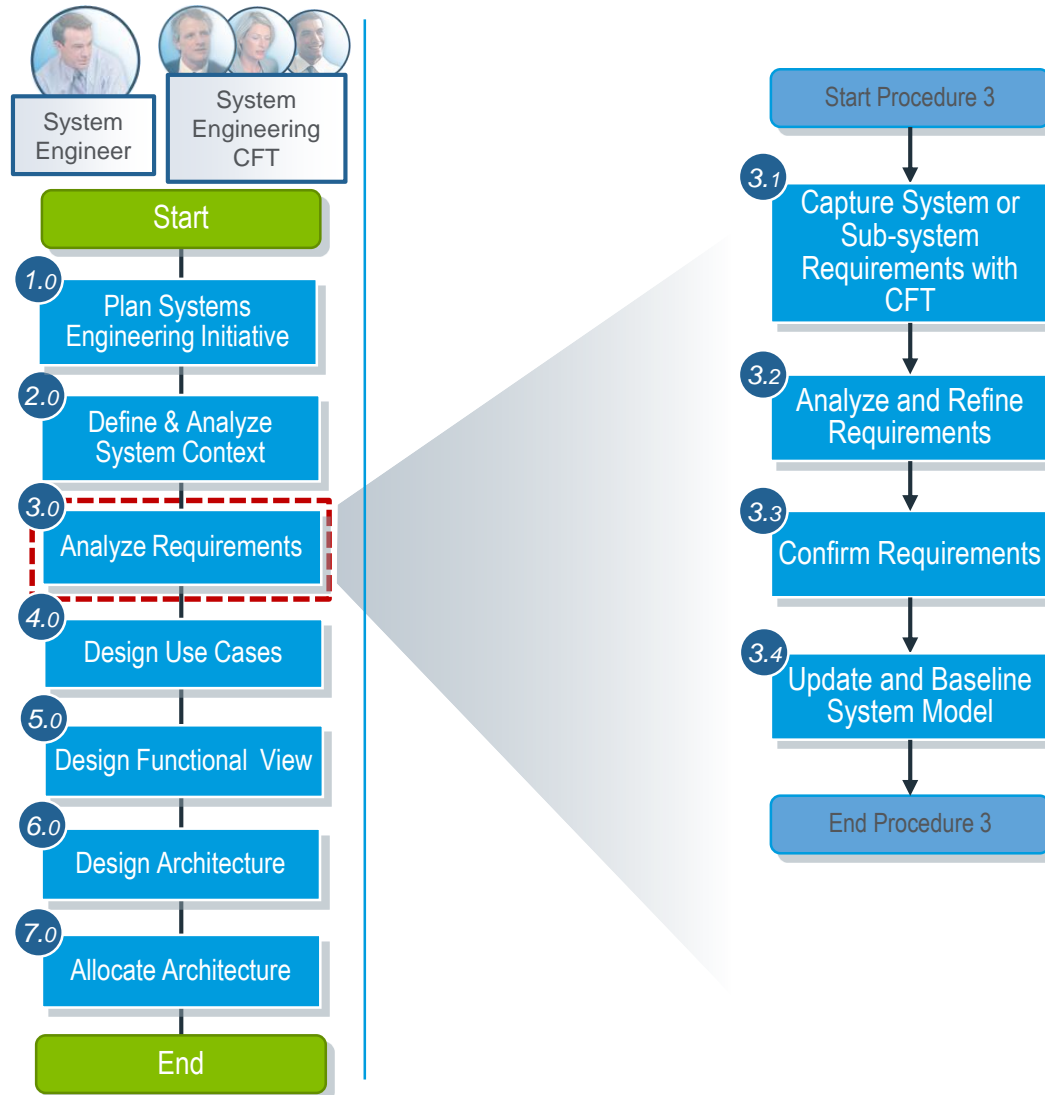
# Analyze Requirements

## ► Analyze Requirements



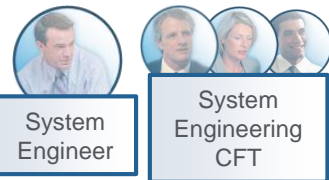
- Objectives
  - Capture, analyze and refine system and sub-system requirements
- Role
  - System Engineer
  - Cross-functional Team
- Outputs
  - Requirements Model

## ► Analyze Requirements

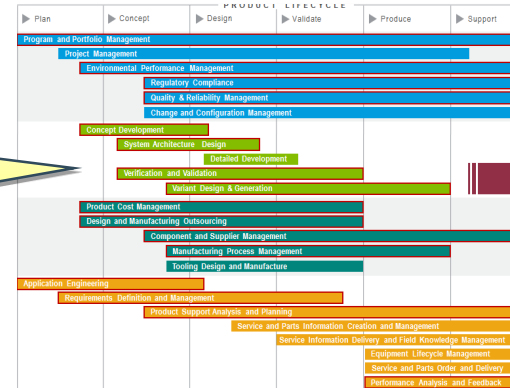




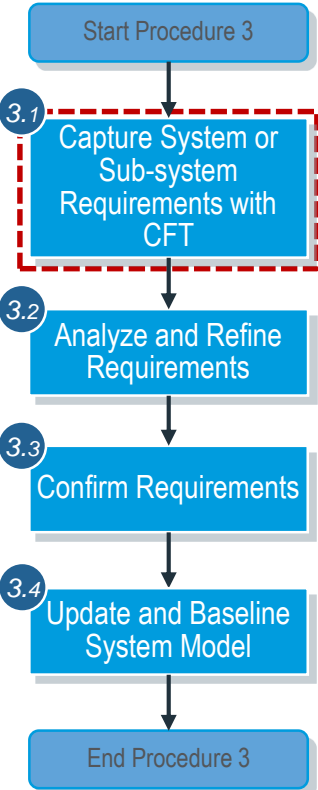
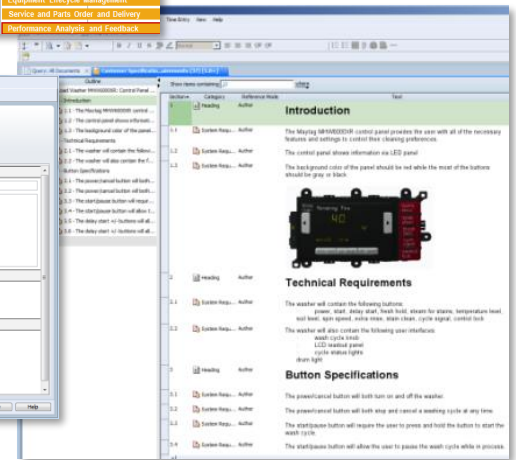
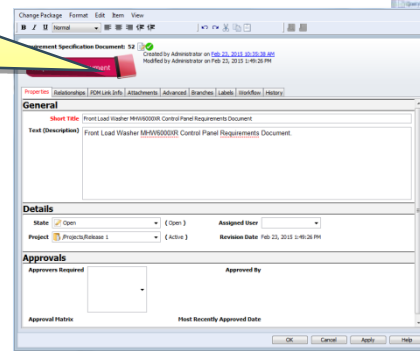
## ► Capture System or Sub-system Requirements with CFT



Coordinate with related processes such as Requirements Management, Quality Mgmt and identify any relevant existing information



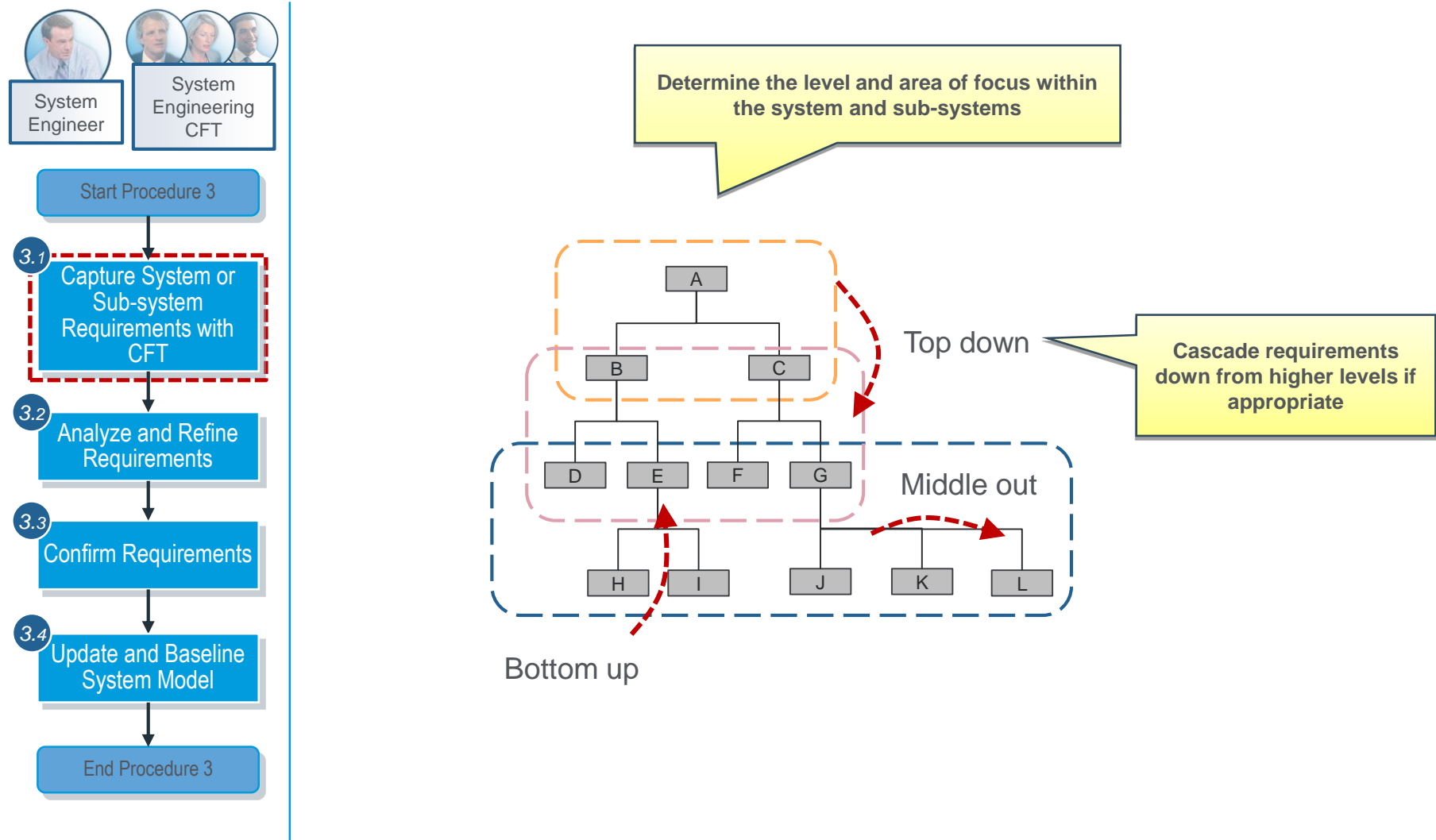
Requirements may be stored in Lifecycle Manager or other requirements management tools



**Note**

This storyboard covers a MBSE approach to defining requirements. Lifecycle Manager provides structured text based requirements management and can be used in conjunction with Modeler

## ► Capture System or Sub-system Requirements with CFT



## ► Capture System or Sub-system Requirements with CFT



System Engineer

Start Procedure 3

3.1 Capture System or Sub-system Requirements with CFT

3.2 Analyze and Refine Requirements

3.3 Confirm Requirements

3.4 Update and Baseline System Model

End Procedure 3

Collect drivers for the new system from the Requirements Management system

Note

For more details on how to capture requirements in Lifecycle Manager, please refer to [Collaborative Requirements Definition Best Practice Storyboard BP Storyboard](#)

Section	Text	Category	Trace Status	Priority	ID	ID
1	PTC Medium Truck System Requirements	Heading	none		1679	1678
1.1	This document details the system requirements for the PTC Truck Mark 2. The truck shall be offered in variant suitable for a range of established and emerging markets and offer varying levels of load and driver options.	System Requirement	downstream		1681	1678
1.2	Market	Heading	none		1683	1678
1.2.1	Europe	Heading	none		1685	1678
1.2.1.1	The vehicle shall be sold into the European market.	System Requirement	downstream	High	1687	1678
1.2.1.2	The vehicle shall be available in left and right-hand driver versions.	System Requirement	downstream	High	1689	1678
1.2.2	USA	Heading	none		1691	1678
1.2.2.1	The vehicle shall be sold into the US market.	System Requirement	downstream	High	1693	1678
1.2.3	China	Heading	none		1695	1678
1.2.3.1	The vehicle shall be sold into the Chinese market.	System Requirement	downstream	Medium	1697	1678
1.3	Functional	Heading	none		1699	1678
1.3.1	Load	Heading	none		1701	1678
1.3.1.1	The vehicle shall be capable of carrying "medium duty" loads (class 4.5.6).	System Requirement	downstream	Critical	1703	1678
1.3.1.2	The gross vehicle weight rating shall not exceed 7,500 kg (16,500lb).	System Requirement	downstream	Critical	1705	1678
1.3.1.3		Heading	none		1707	1678
1.3.1.4		Heading	none		1709	1678
1.3.1.5		Heading	downstream	Critical	1711	1678

Note

Synchronization of requirements between Lifecycle Manager and Modeler is often iterative and includes additions, deletions and modifications

## ► Capture System or Sub-system Requirements with CFT



System Engineer

Start Procedure 3

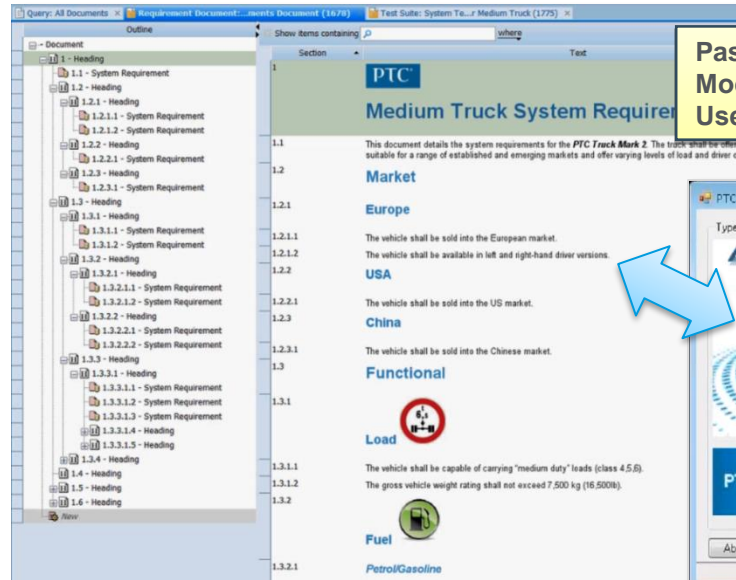
3.1 Capture System or Sub-system Requirements with CFT

3.2 Analyze and Refine Requirements

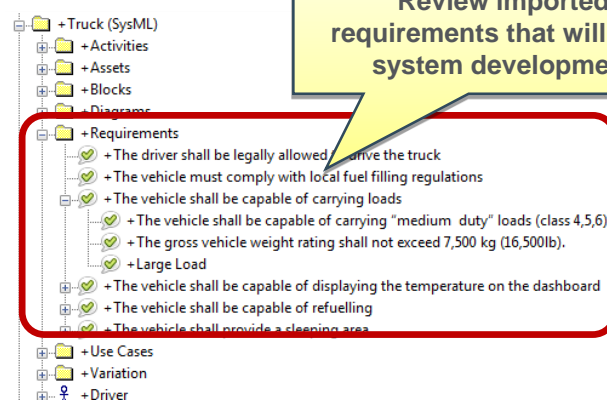
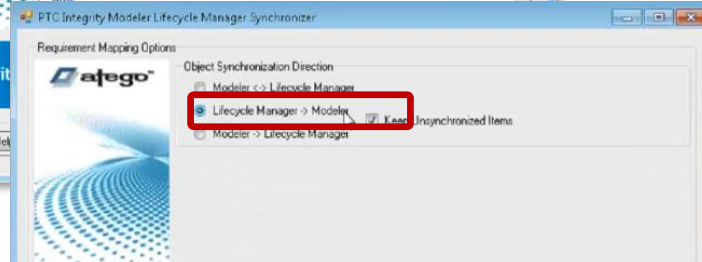
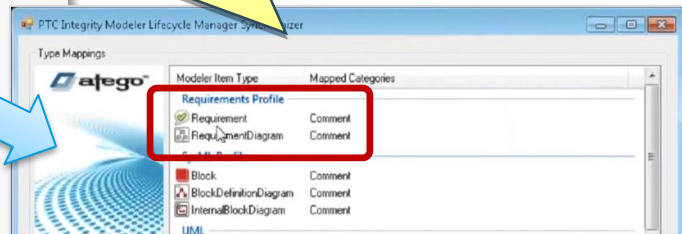
3.3 Confirm Requirements

3.4 Update and Baseline System Model

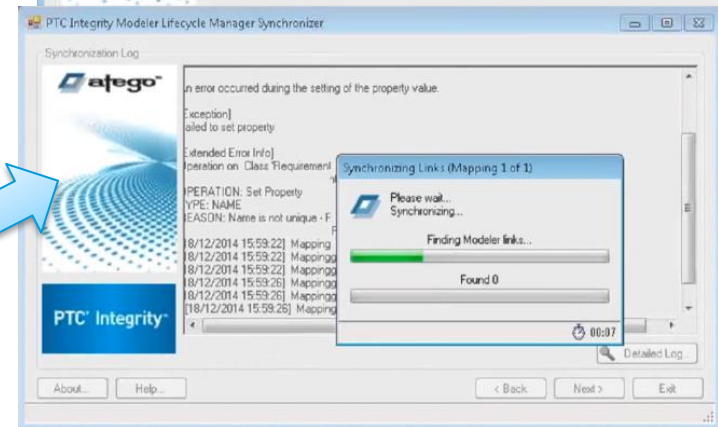
End Procedure 3



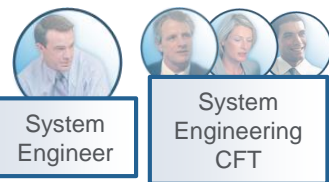
Pass requirements from Lifecycle Manager to Modeler and vice versa.  
Use the 'Integration for Lifecycle Manager.'



Review imported requirements that will drive system development



## ► Capture System or Sub-system Requirements with CFT

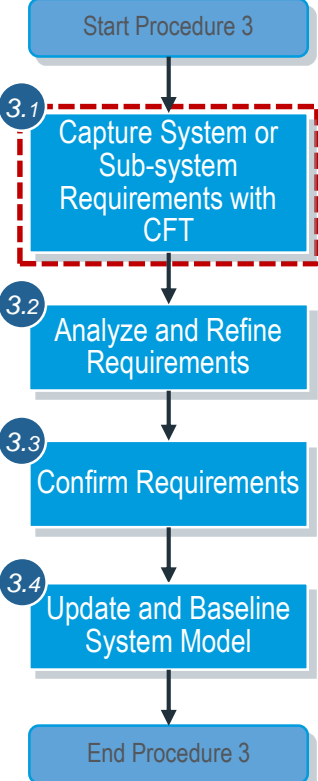


Perform system requirements gathering sessions with the cross-functional team

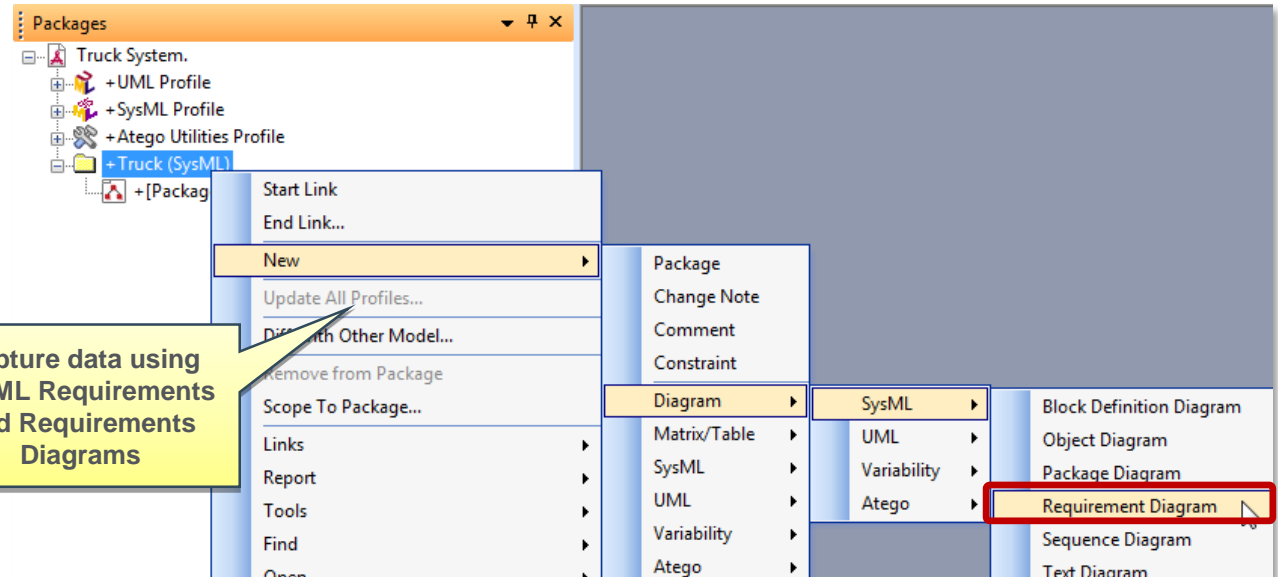


### Note

Include any other relevant subject matter experts, potentially including customers and end product users



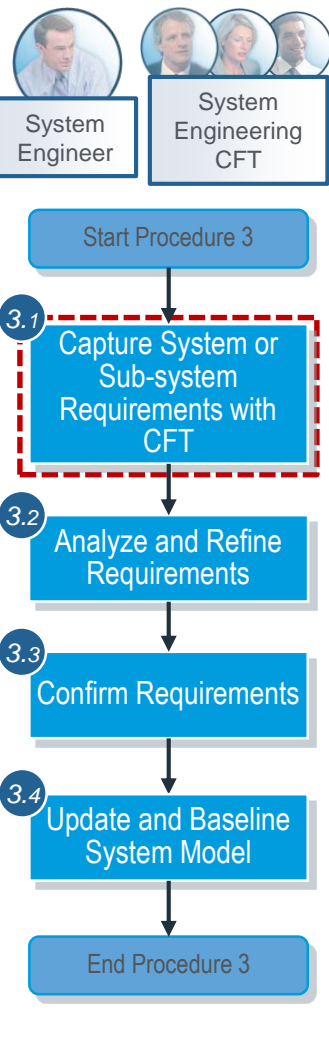
Capture data using SysML Requirements and Requirements Diagrams



### Note

A requirement specifies a capability or condition that must (or should) be satisfied. A requirement may specify a function that a system must perform or a performance condition a system must achieve

## ► Capture System or Sub-system Requirements with CFT



**Requirements toolbar**

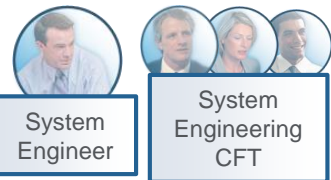
**Create new Requirement**

**Define Requirement properties. Text tab allows additional textual information to be recorded**

The screenshot displays the PTC Model Based System Engineering software interface. At the top, a 'Requirements toolbar' is shown with a red box highlighting the 'Create new Requirement' icon (a green heart with a checkmark). A callout points to this icon with the text 'Create new Requirement'. Below the toolbar, the 'Packages' pane on the left shows a tree structure with 'Truck System.' expanded, containing '+UML Profile', '+SysML Profile', '+Atego Utilities Profile', '+Truck (SysML)', '+Requirement1', and two instances of '[Package] Truck (SysML)'. The main editor area shows a diagram of a requirement element labeled 'req [Package] Truck (SysML) [2]' with a red box around the text '«requirement» Requirement1'. At the bottom, the 'Properties of Requirement1' dialog is open, with the 'Text' tab selected. A red box highlights the 'Text' tab and the 'Requirement1' text field. A callout points to this area with the text 'Define Requirement properties. Text tab allows additional textual information to be recorded'. The 'Text' tab shows the following information: 'Requirement1', 'Full name: Truck (SysML)::Requirement1', 'Type: Requirement', 'Last changed on/by: 01/01/2015 18:01:19 PTCNET\Pollerton', and 'Visibility: Public' (selected).



## ► Capture System or Sub-system Requirements with CFT



Start Procedure 3

3.1 Capture System or Sub-system Requirements with CFT

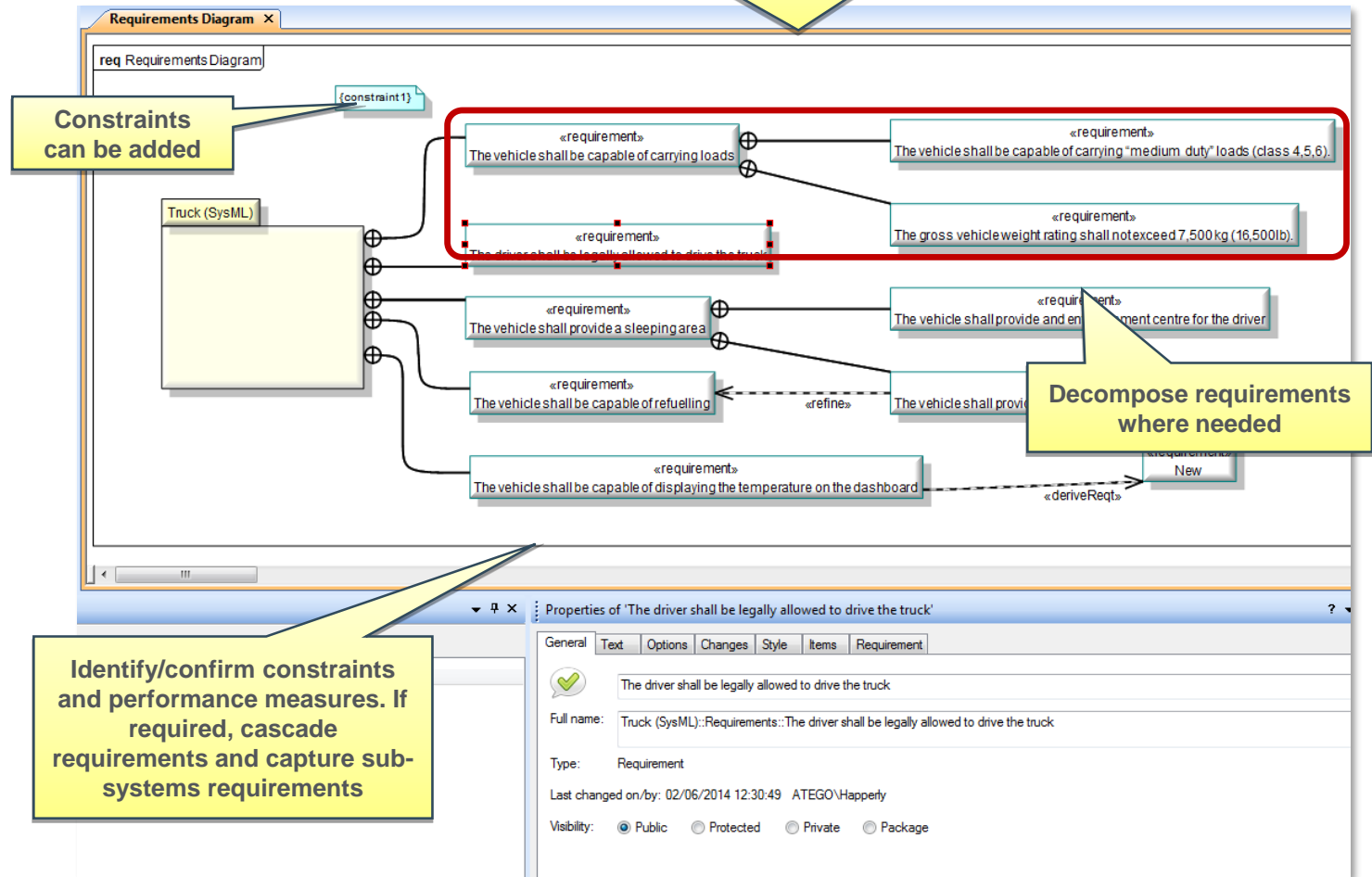
3.2 Analyze and Refine Requirements

3.3 Confirm Requirements

3.4 Update and Baseline System Model

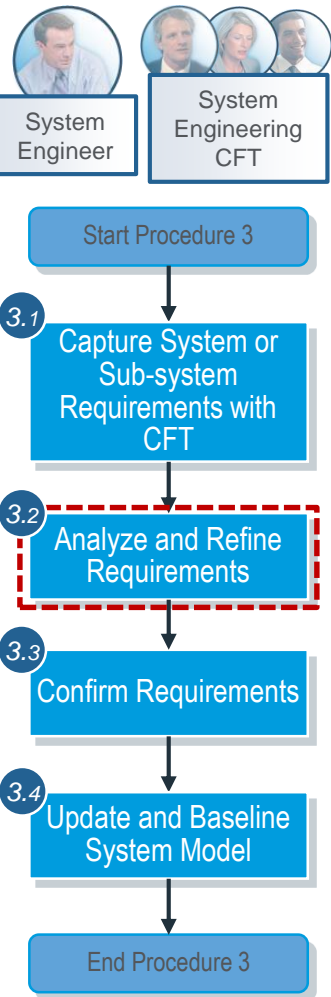
End Procedure 3

Capture what capabilities, functions or services the system or sub-system must perform. Identify any actors that have an interaction with the system





## ► Analyze and Refine Requirements



Review, analyze and refine and requirements with the CFT

Review requirements definitions and associations to other requirements

The screenshot shows the 'Properties of 'The vehicle shall provide a sleeping area'' dialog box. The 'Requirement' tab is active, displaying a table of tag definitions and values.

Tag Definition Name	Tag Value
master	
parentRequirement	
problem	
rationale	Sleep
refinedBy	
refines	
satisfiedBy	
satisfies	
slave	
subRequirements	The vehicle shall provide a means of making hot and cold beverages. The vehicle shall provide and entertainment centre for the driver.
synchronizationStatus	Undefined
tracesFrom	
tracesTo	
bt	
verifiedBy	The vehicle shall provide a sleeping area for the driver within the cab. The sleeping area shall comprise a bed, bedding and accessories.
verifies	

**Note**

Analyze and review requirements for sub-systems within the context of the overall system

Validate correctness and eliminate overlap

The screenshot shows the 'Overview' tab of the 'Model overview' window. It features a pie chart and a table of review results.

Review	Found	Out of	Percentage
Abstract class defined in a hierarchy	0	11	0%
Abstract class without methods	0	11	0%
Abstract interface defined	0	18	0%
Abstract operation on a concrete class	0	19	0%
Anonymous method	0	1	0%
Attribute check list	0	27	0%
Boolean expression	0	27	0%
Boolean expression	0	19	0%
Class check list	0	27	0%
Class constant	0	11	0%
Class check list	0	19	0%
Class check list	0	19	0%

Perform analysis to confirm model is complete and correct (Refer to Automated System Design Review Best Practice)

**Note**

Refer to the System Engineering Process Governance and Efficient Design Review practices for additional information on system engineering reviews

## ► Confirm Requirements



Start Procedure 3

3.1 Capture System or Sub-system Requirements with CFT

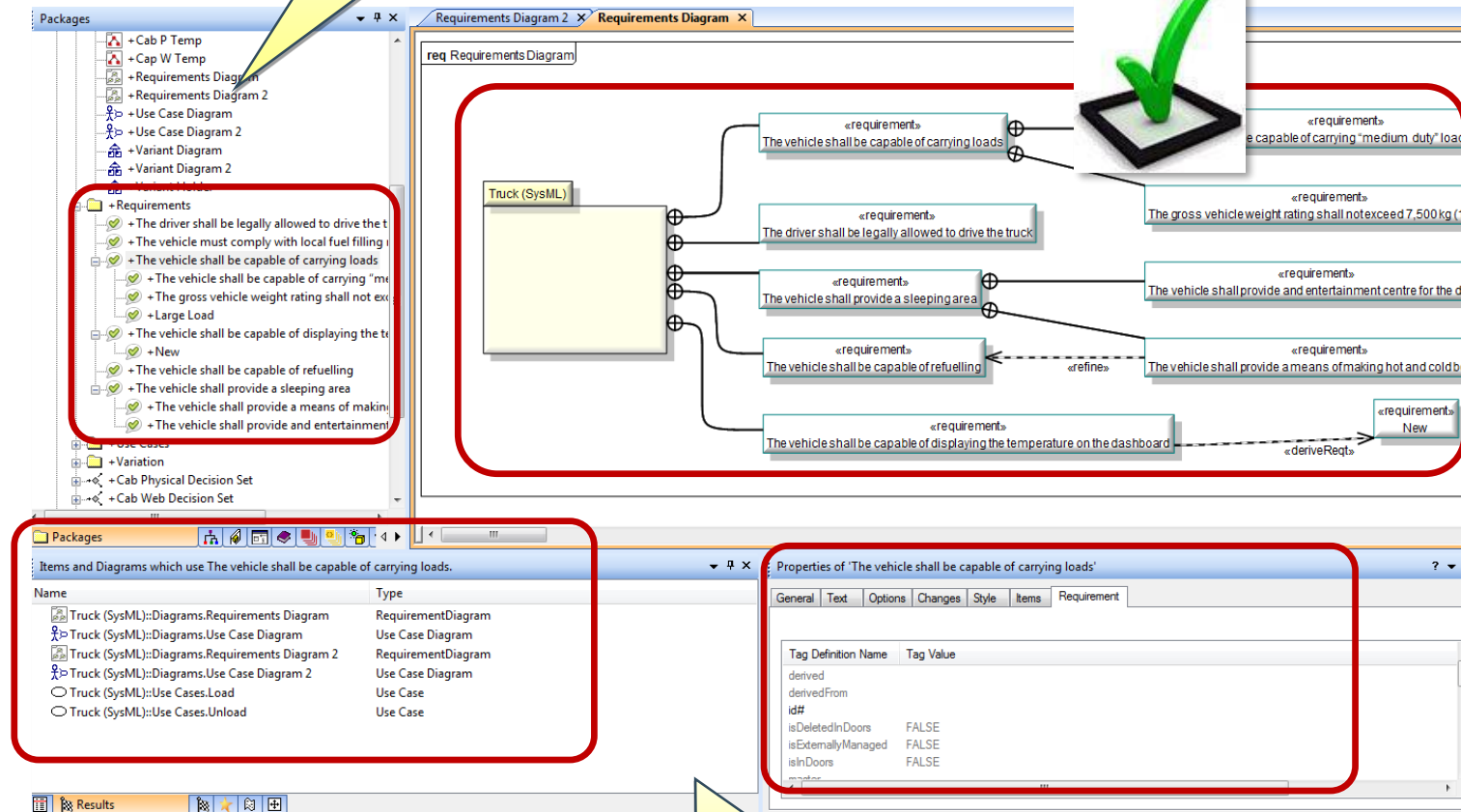
3.2 Analyze and Refine Requirements

3.3 Confirm Requirements

3.4 Update and Baseline System Model

End Procedure 3

Confirm and approve all requirements with the CFT



Update requirements model if needed

## ► Confirm Requirements



System Engineer

Start Procedure 3

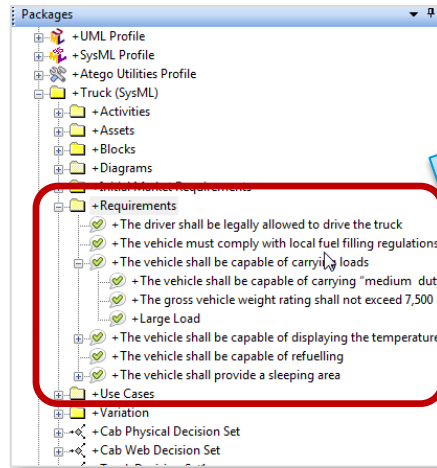
3.1 Capture System or Sub-system Requirements with CFT

3.2 Analyze and Refine Requirements

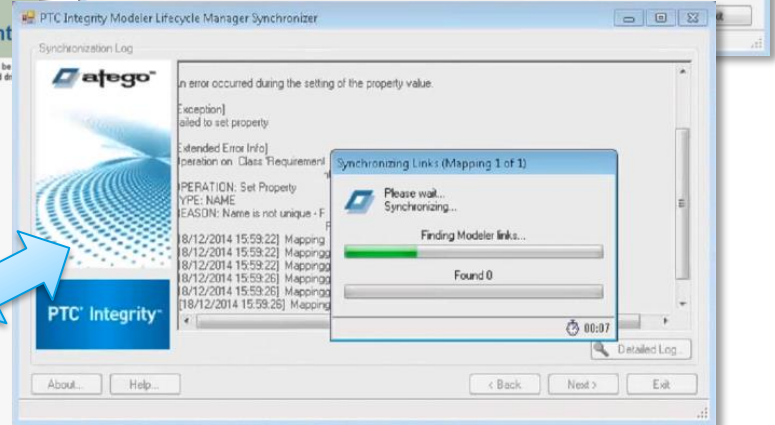
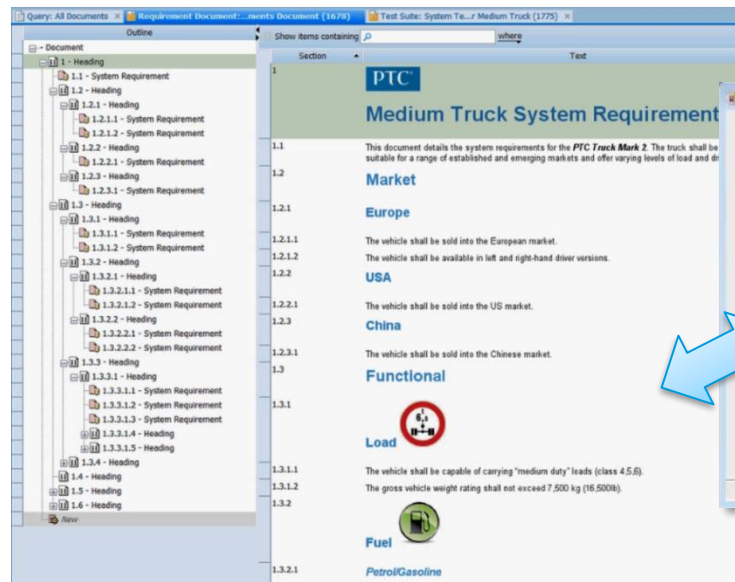
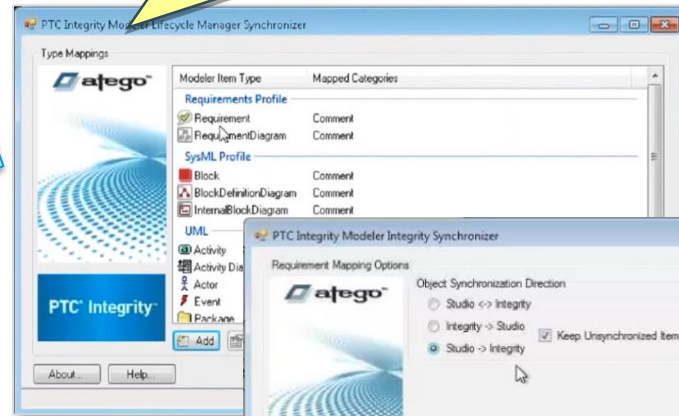
3.3 Confirm Requirements

3.4 Update and Baseline System Model

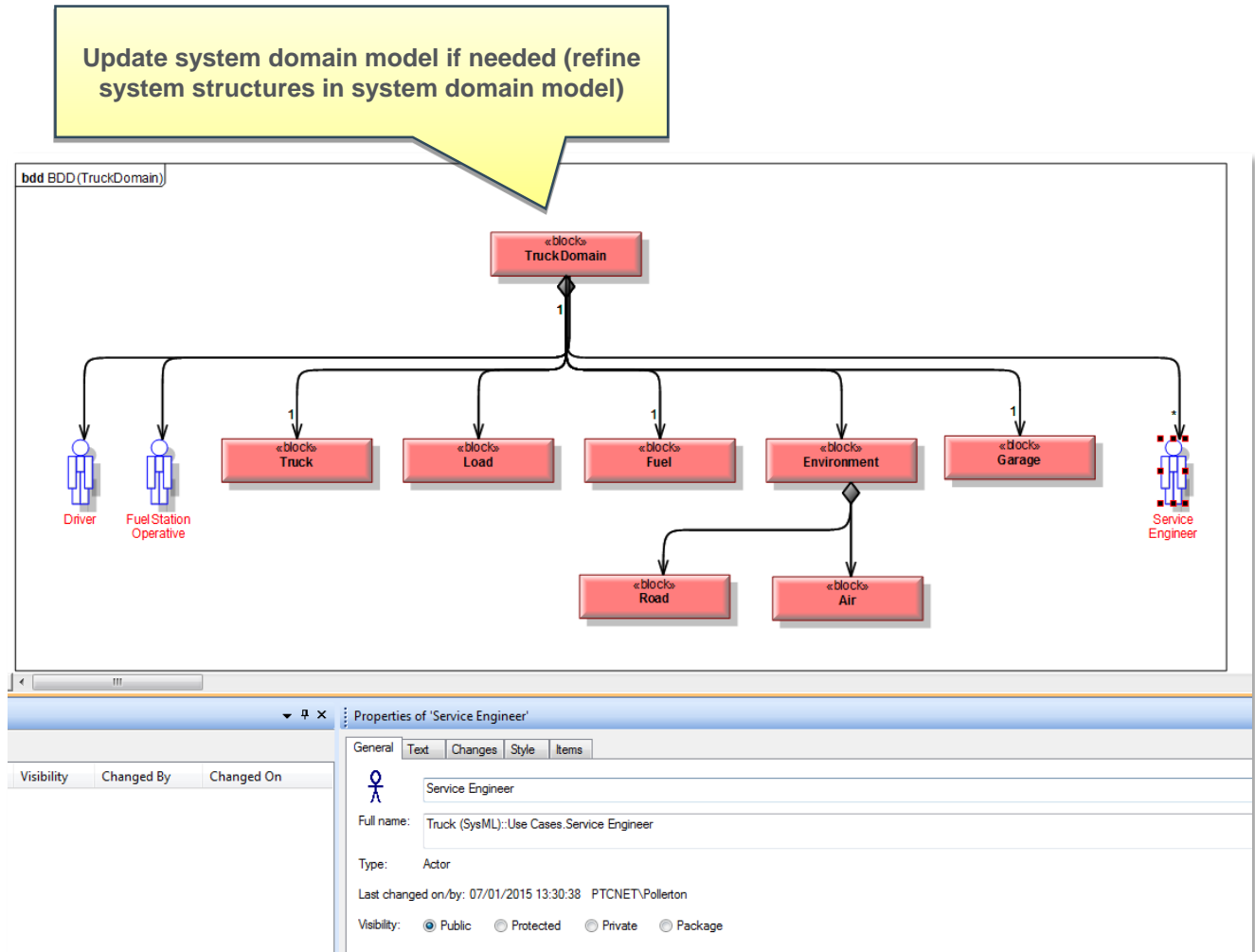
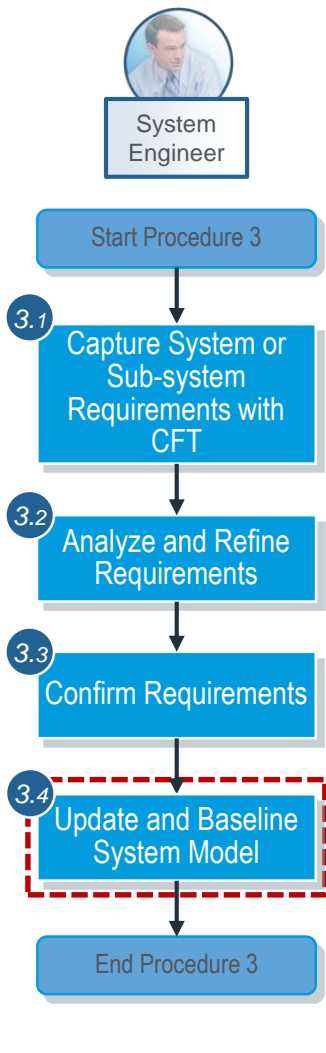
End Procedure 3



Pass System Level Requirements, Model Elements and Trace Links to Lifecycle Modeler



## ► Update System Model



## ► Update System Model



System Engineer

Start Procedure 3

3.1 Capture System or Sub-system Requirements with CFT

3.2 Analyze and Refine Requirements

3.3 Confirm Requirements

3.4 Update and Baseline System Model

End Procedure 3

Creating a new version is a common way to baseline the model. Launch the Model Explorer (or select File > Open from with Modeler)

Select the model and click New to create a new version

The screenshot shows the 'Examples - Artisan Model Explorer' window. The 'Look in' path is '\\Enabler\\POLLERTON11\\Examples\\Truck System'. The left pane shows a tree view with 'Truck System' selected. The right pane shows a list of models with their versions and last changed by information. A red box highlights the 'New' button in the toolbar, and a callout box explains its function. Another red box highlights the 'Truck System' entry in the list.

Name	Version	Last Changed By
Filling Station	0	PTCNET\\Pollerto
Heart Monitor C	0	PTCNET\\Pollerto
Heart Monitor Java	0	UK-CHE-DTP-022
HSUV	0	PTCNET\\Pollerto
Search and Rescue	0	UK-CHE-DTP-022
Speed Controller	0	UK-CHE-DTP-022
Template - Component-based Products	0	UK-CHE-DTP-022
Template - Incremental Process	0	UK-CHE-DTP-022
Template - Small Project	0	UK-CHE-DTP-022
Traffic Lights	0	UK-CHE-DTP-022
Traffic Lights - Sysim	0	UK-CHE-DTP-022
<b>Truck System</b>	4	PTCNET\\Pollerto
VB Another Block (Tetris) Example	0	UK-CHE-DTP-022
Waste System	0	UK-CHE-DTP-022

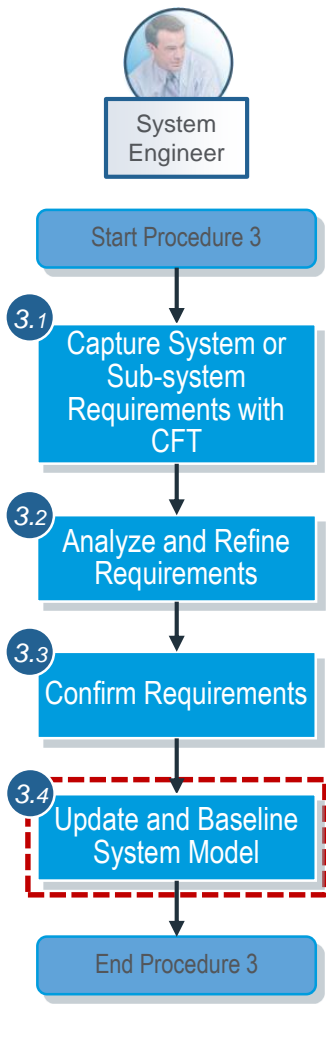
**Truck System**  
Version: 4  
Last Changed By: PTCNET\\Pollerton

Last Changed On: 20/01/2015 10:50:12  
Last Operation:

Created By: PTCNET\\Pollerton  
Created On: 12/12/2014 15:34:34

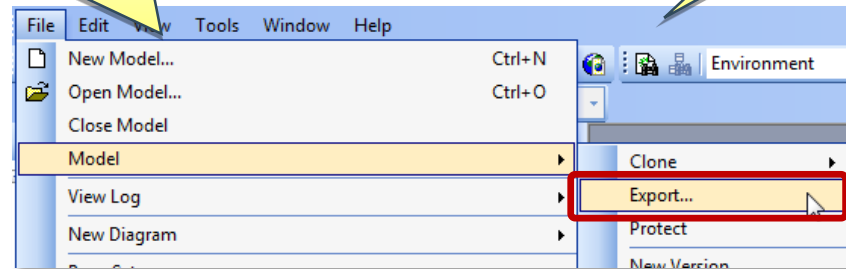
Open  
Close

## ► Update System Model

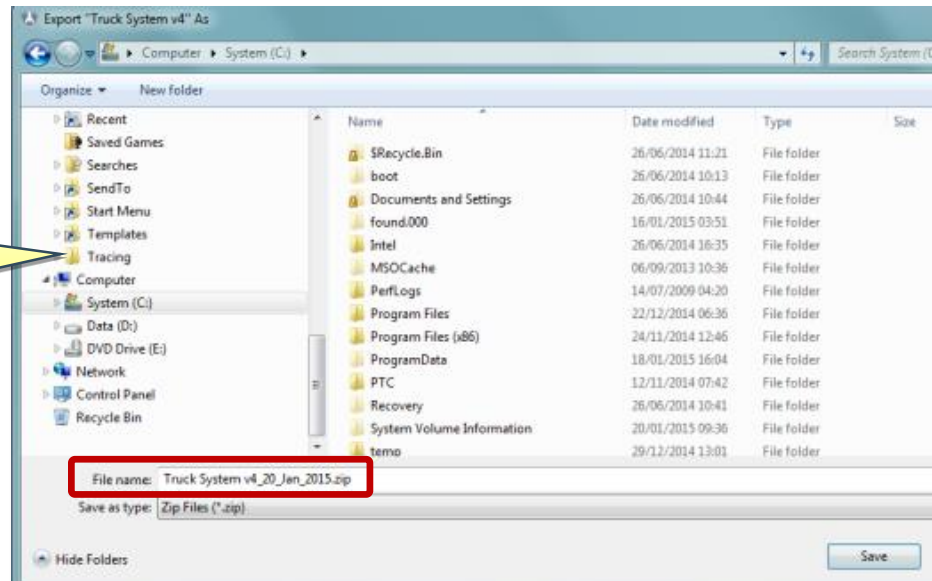


If a copy of the file needs to be stored in another location, Export can be used.

Choose File > Model > Export



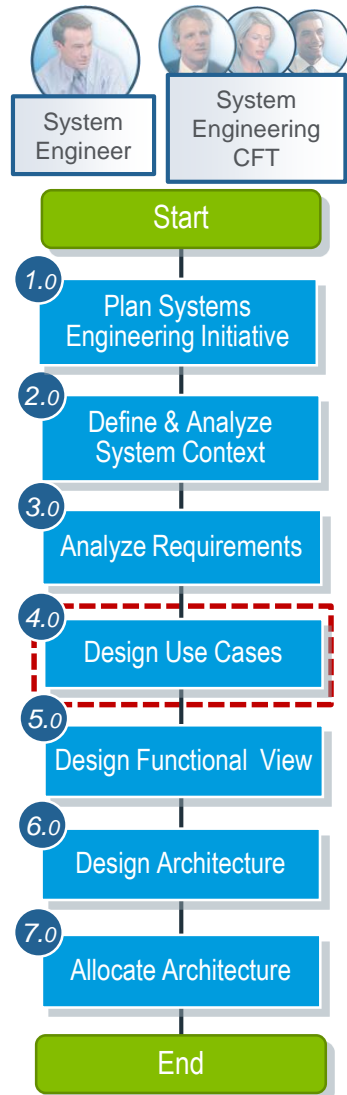
Select a location and enter a name for exported model (.zip format)





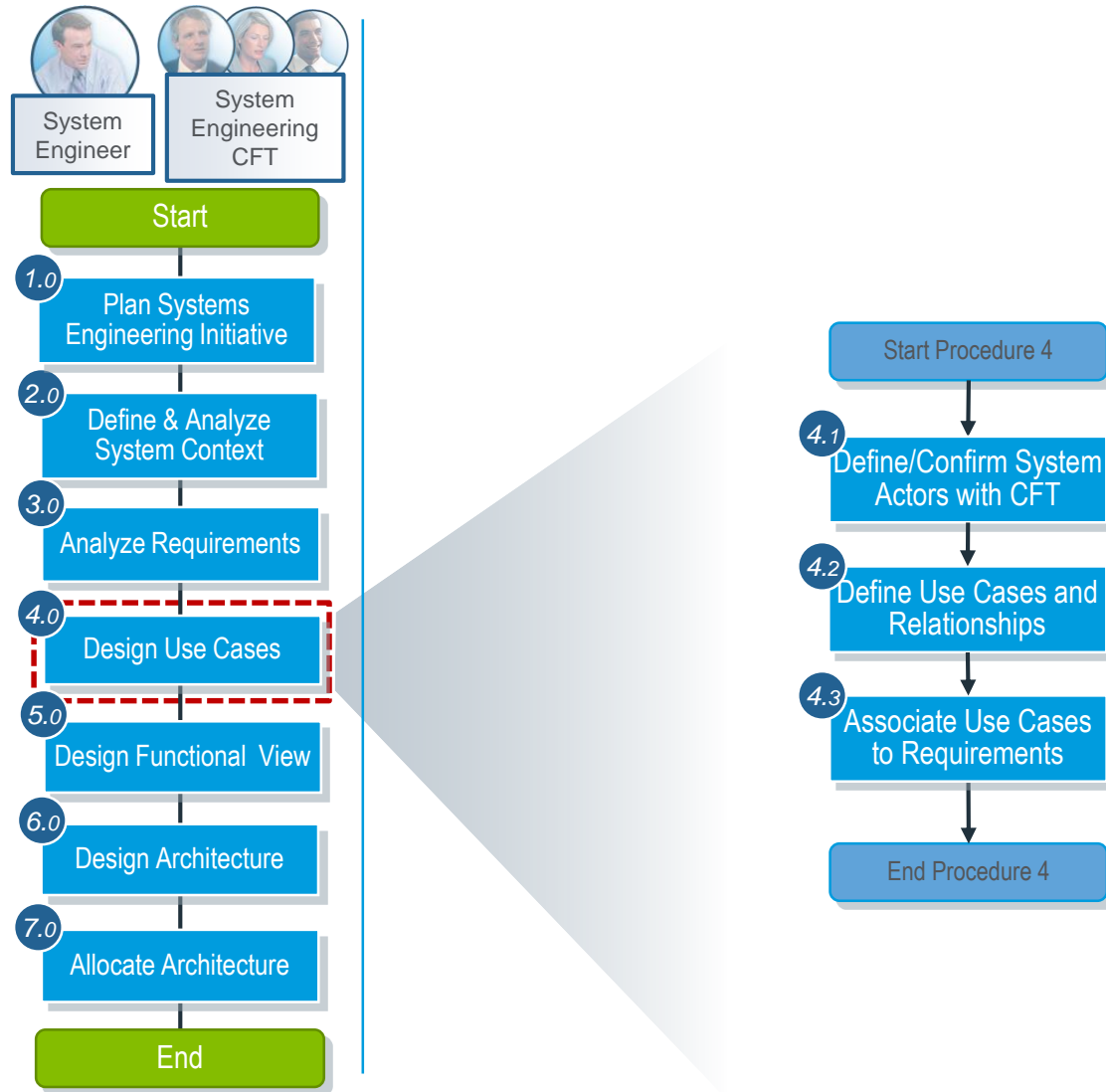
# Design Use Cases

## ► Design Use Cases

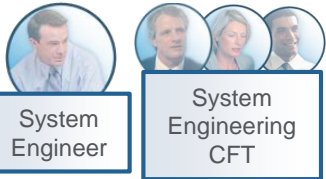


- Objectives
  - Capture and document use cases and relationships
- Role
  - System Engineer
  - Cross-functional Team
- Outputs
  - Use Case Model

## ► Design Use Cases



## ► Design System Actors with CFT

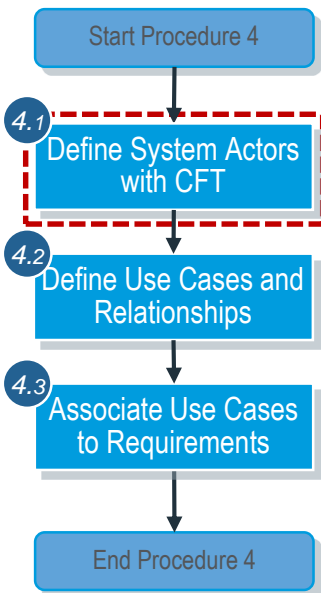


Work with the CFT to identify and document use cases

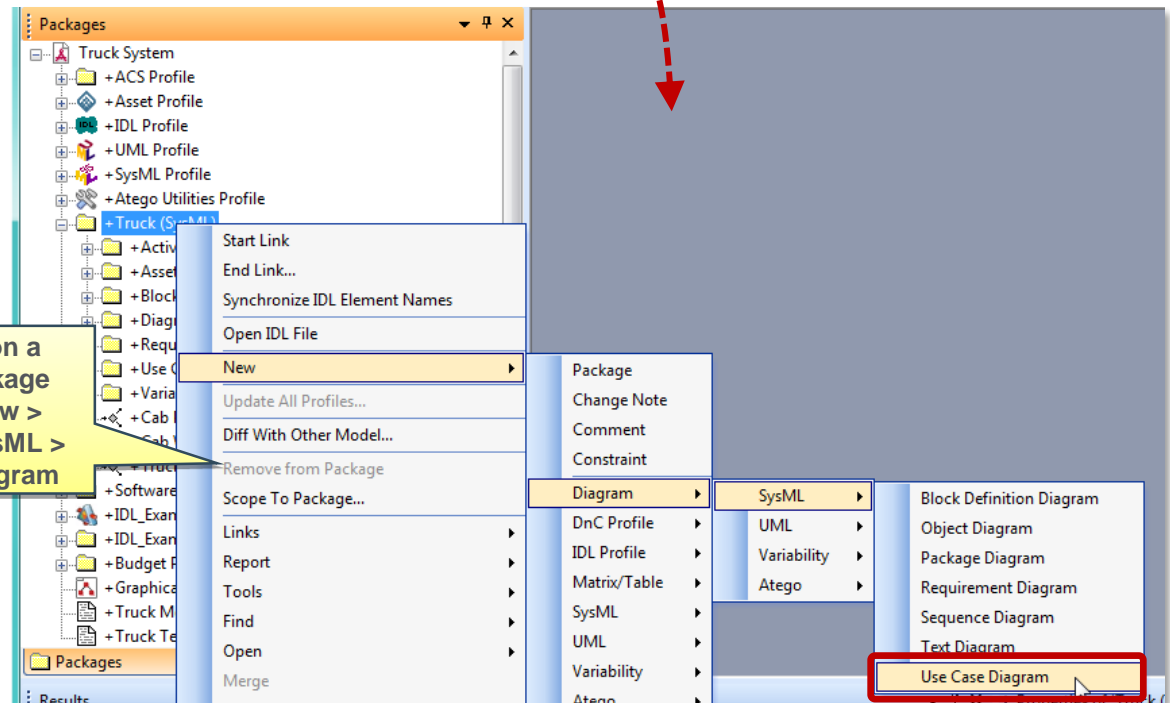


Note

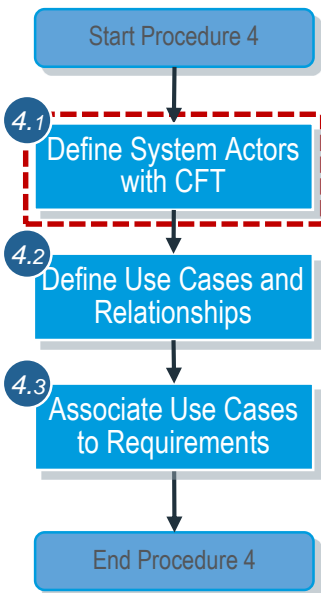
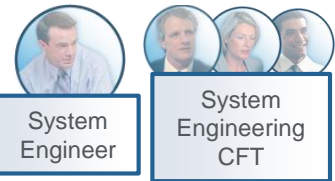
Use case diagrams focus on system behavior. They describe the usage of a system (subject) by its actors (environment) to achieve a goal, which is realized by the subject providing a set of services to selected actors



Right click on a suitable package and click New > Diagram > SysML > Use Case Diagram



## ► Design System Actors with CFT



Confirm the name of the new diagram and specify any other options

Properties of 'UseCase Diagram'

General Text Changes Style Items

UseCase Diagram

Full name: Truck (SysML).UseCase Diagram

Page reference: uc

Type: Use Case Diagram

Last changed on/by: 05/01/2015 13:01:05 PTCNET\Pollerton

Visibility: ☒ Public ☐ Protected ☐ Private ☐ Package

Actor

Driver

Fuel Station Operative

Add Actors to represent any entity that will interact with the system

Properties of 'Driver'

General Text Changes Style Items

Driver

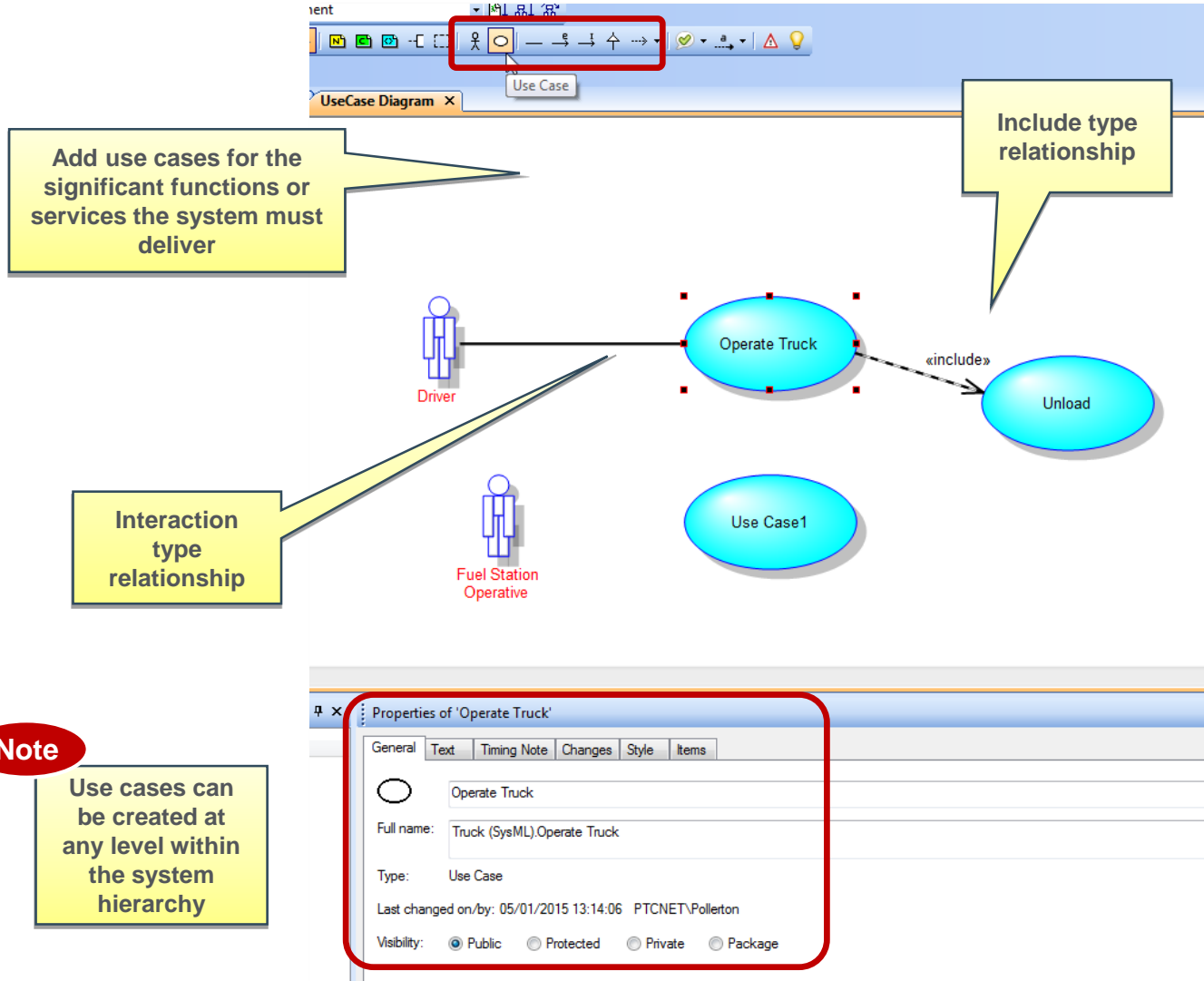
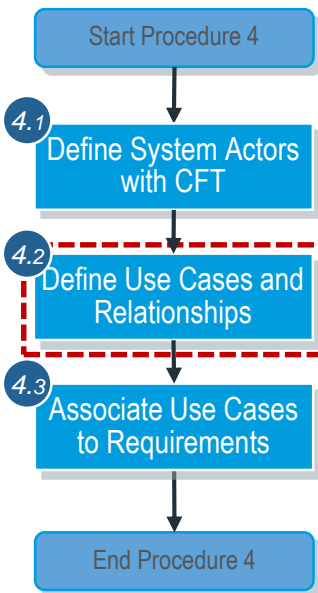
Full name: Truck (SysML).Driver

Type: Actor

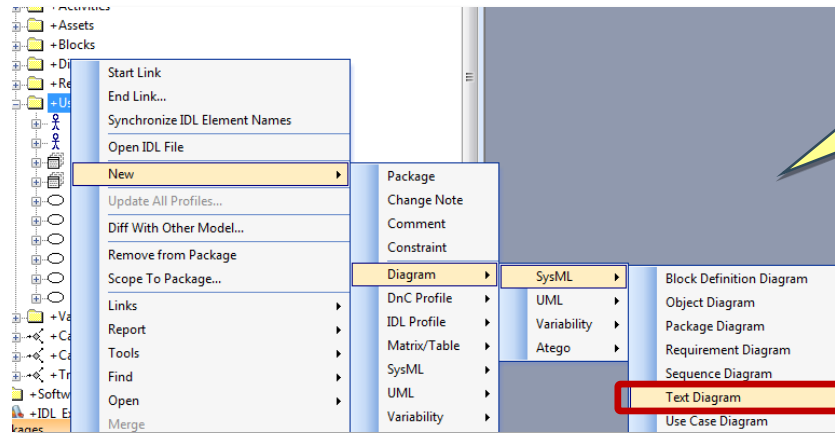
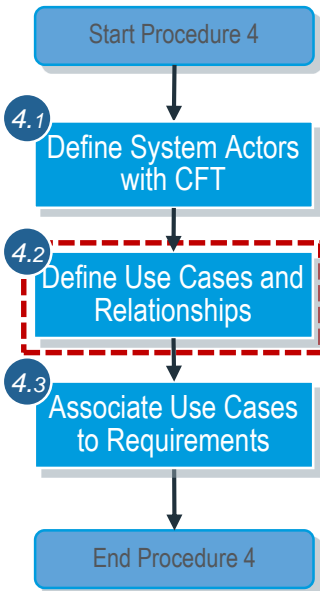
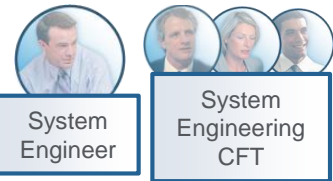
Last changed on/by: 05/01/2015 13:12:55 PTCNET\Pollerton

Visibility: ☒ Public ☐ Protected ☐ Private ☐ Package

## ► Define Use Cases and Relationships



## ► Define Use Cases and Relationships



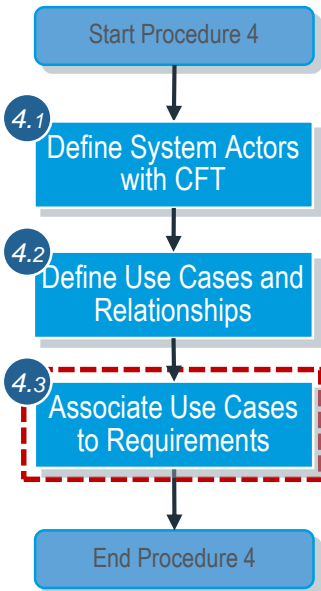
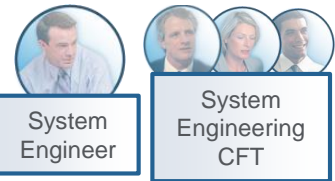
Use case descriptions can also be created to provide further information and also capture multiple scenarios

Create Text diagrams to capture textual descriptions of use cases

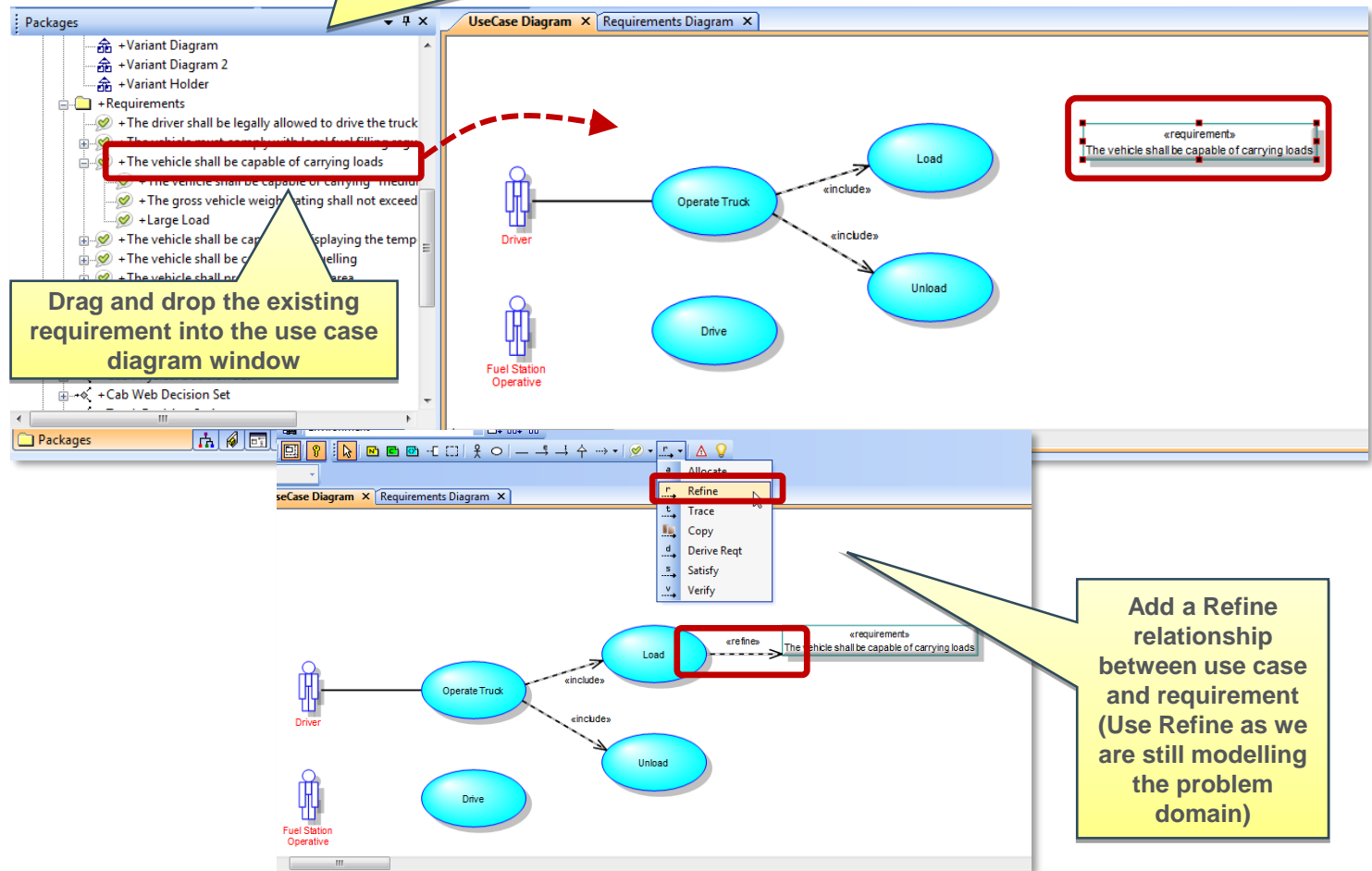
Load UC Description* x	
Use Case ID	LT001
Use Case Name	Load Truck
Actors	Driver
Description	The driver confirms load suitability and loads securely onto truck.
Preconditions	1. Load available and weight correct 2. Loading mechanism available
Postconditions	1. Truck loaded 2. Collection recorded in electronic tracking system 3. Receipt or invoice paperwork provided
Normal Flow	1. Driver inspects load to ensure weight is correct and packaged ready for loading 2. Driver utilises loading mechanism to load onto truck 3. Driver ensures load is secured as appropriate 4. Driver records collection in electronic tracking system 5. Driver provides receipt or invoice paperwork
Alternative Flows	1. Driver inspects load to ensure weight is correct and packaged ready for loading 2. Driver manually loads onto truck 3. Driver ensures load is secured as appropriate 4. Driver records collection in electronic tracking system 5. Driver provides receipt or invoice paperwork
Exceptions	Load too heavy or too large for truck capacity



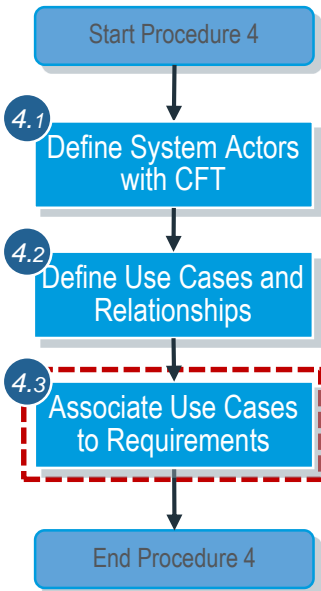
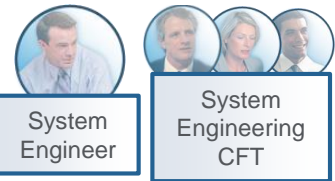
## ► Associate Use Cases to Requirements



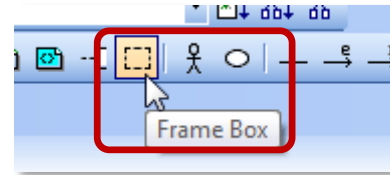
Use cases provide additional details on the behavior needed to implement a requirement. Therefore it can be useful to link a requirement to it's related use case



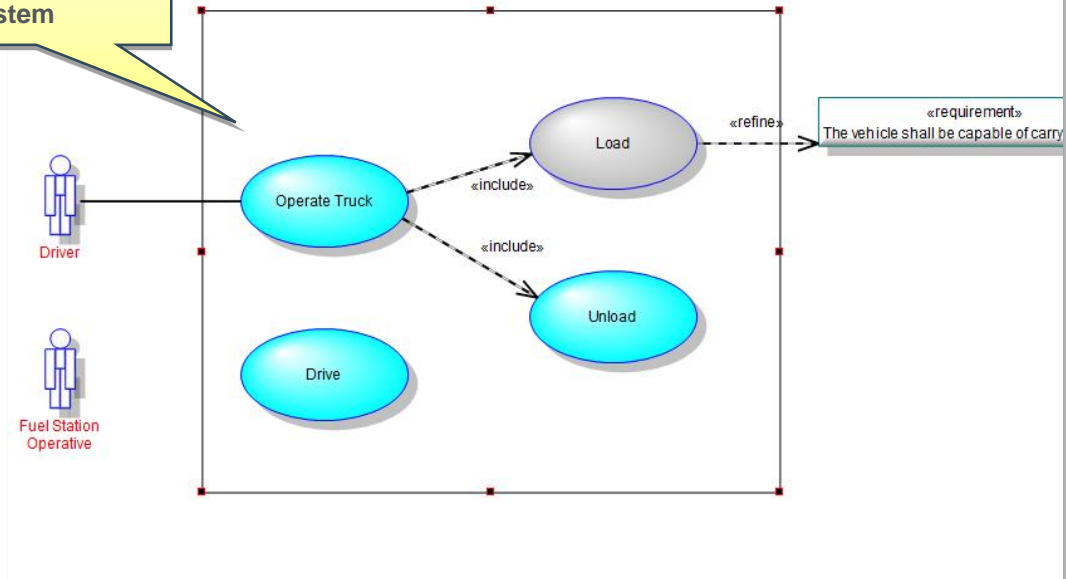
## ► Associate Use Cases to Requirements



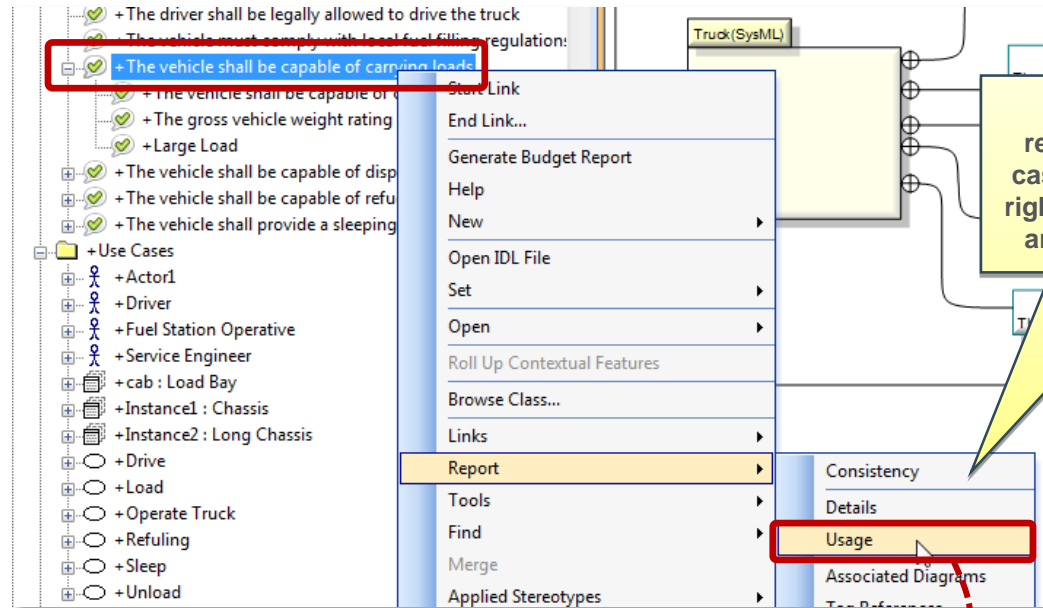
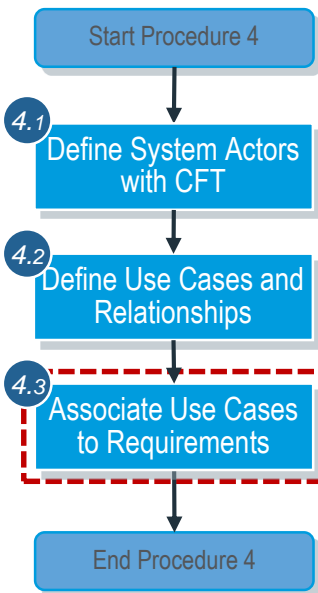
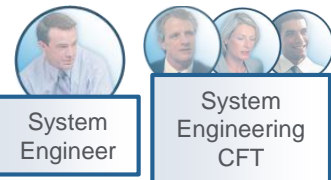
As with Block Definition Diagrams, it is possible to put a system boundary onto a use case diagram



Select Frame Box and drag the box around the use cases in the system



## ► Associate Use Cases to Requirements



To navigate from a requirement to related use cases or use case diagrams, right click on the requirement and select Report > Usage

Related use cases and use case diagrams are displayed in the Results window. Related diagrams can be opened by double clicking

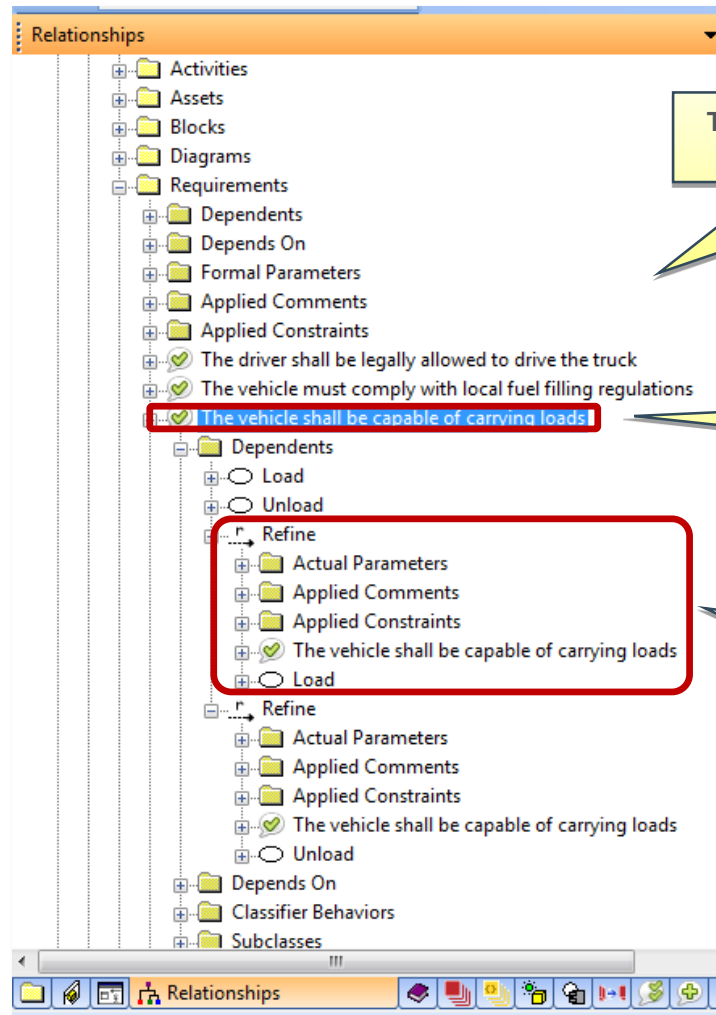
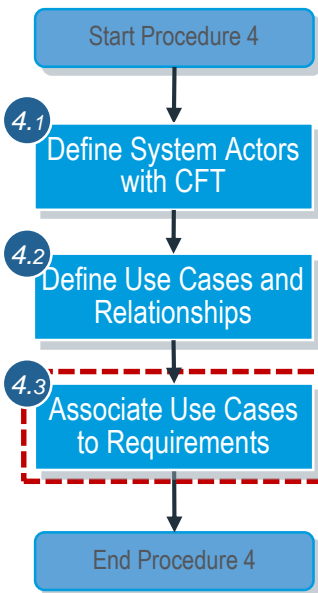
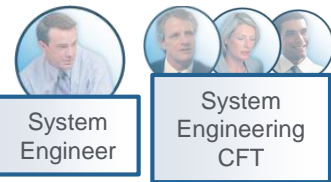
Items and Diagrams which use The vehicle shall be capable of carrying loads.

Name	Type
Truck (SysML).Use Case Diagram	Use Case Diagram
Truck (SysML)::Diagrams.Use Case Diagram	Use Case Diagram
Truck (SysML)::Diagrams.Use Case Diagram 2	Use Case Diagram
Truck (SysML).Load	Use Case
Truck (SysML)::Use Cases.Load	Use Case
Truck (SysML)::Use Cases.Unload	Use Case
Truck (SysML)::Diagrams.Requirements Diagram	RequirementDiagram
Truck (SysML)::Diagrams.Requirements Diagram 2	RequirementDiagram
Truck (SysML)::Blocks::Engine.[Block] Engine [1]	InternalBlockDiagram

**Note**

This view shows usage, it does not show actual links between model item. Refer to the next slide for more information

## ► Associate Use Cases to Requirements



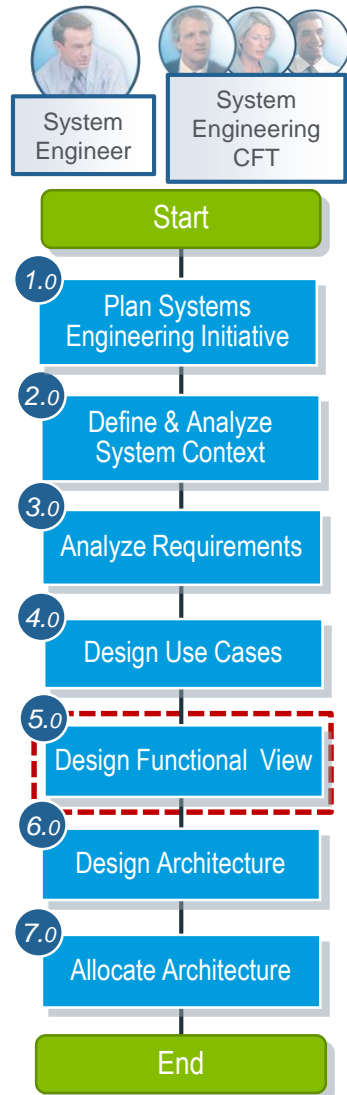
To display all relationships between model items use the Relationship browser

Locate the requirement and expand the Dependents folder

Relationships are displayed, in this case the Refine link between requirement and the Load use case

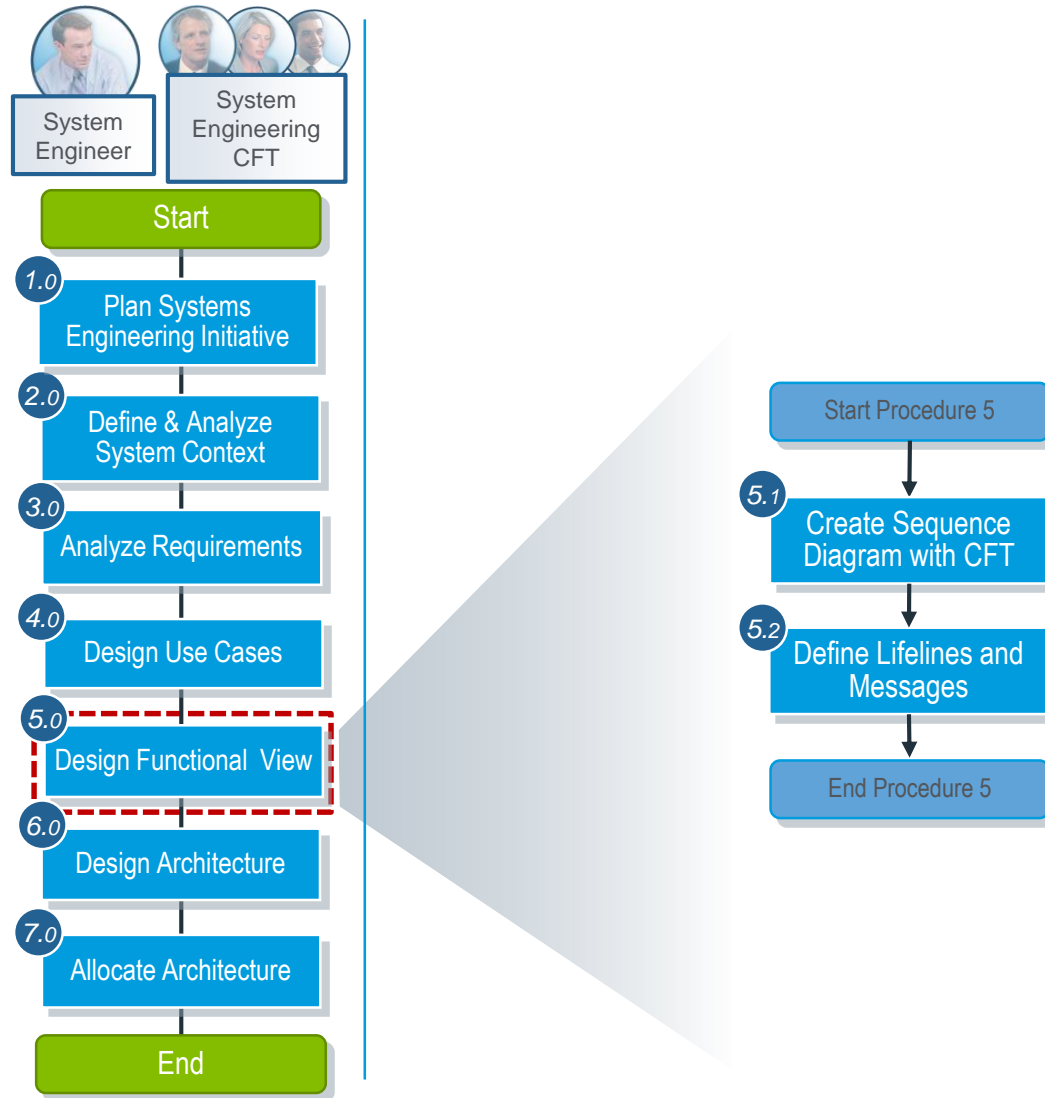
# Design Functional View

## ► Design Functional View




- Objectives
  - Define functional interactions and preliminary functional architecture
- Role
  - System Engineer
  - Cross-functional Team
- Outputs
  - Interaction Model
  - Functional Architecture Model

## ► Design Functional View





## ► Create Sequence Diagram with CFT



System Engineer

System Engineering CFT

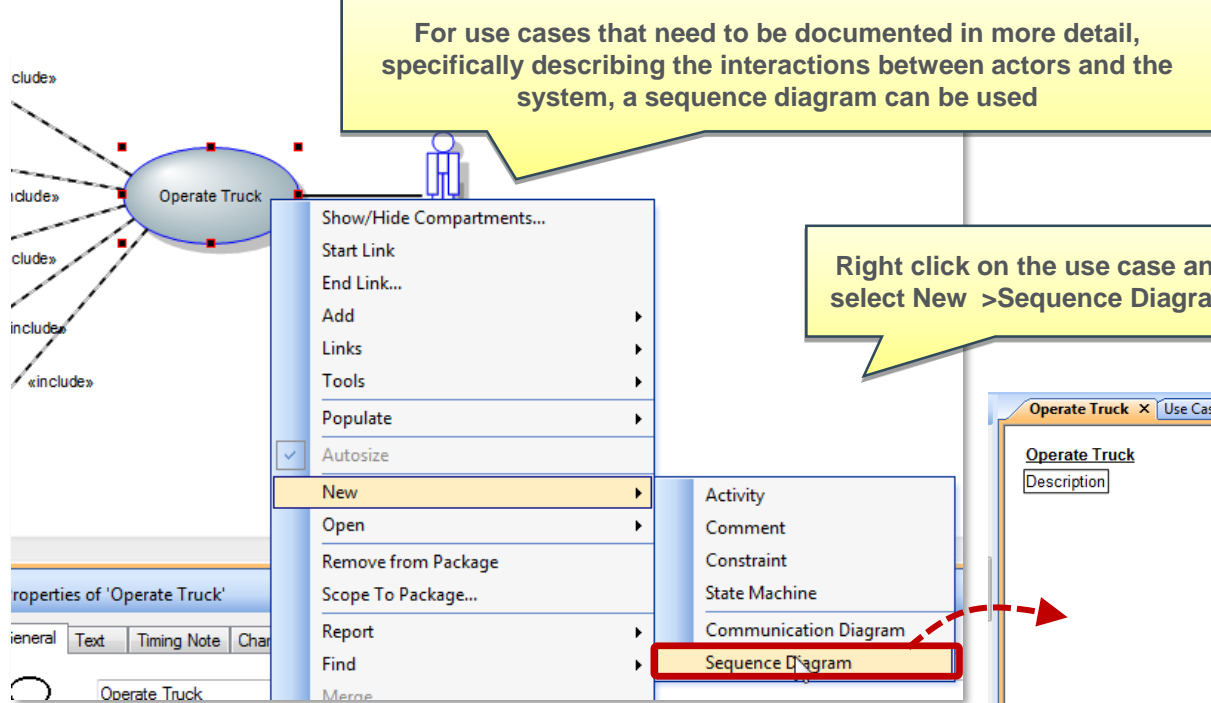
Start Procedure 5

5.1 Create Sequence Diagram with CFT

5.2 Define Lifelines and Messages

End Procedure 5

For use cases that need to be documented in more detail, specifically describing the interactions between actors and the system, a sequence diagram can be used

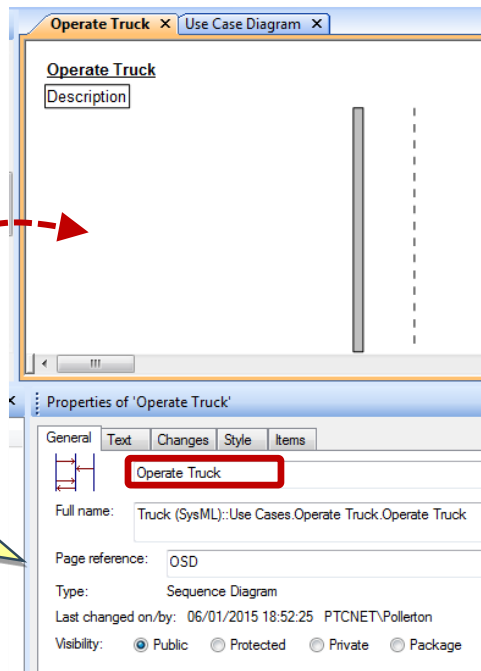


Right click on the use case and select New >Sequence Diagram

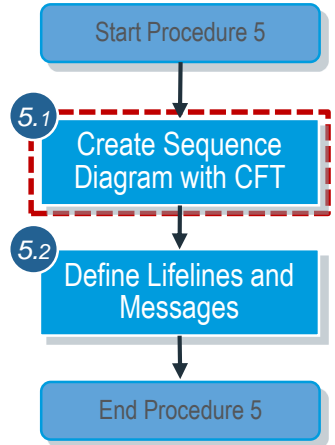
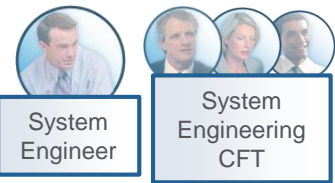
**Note**

The Sequence diagram describes the flow of control between actors and systems (blocks) or between parts of a system. This diagram represents the sending and receiving of messages between the interacting entities called lifelines, where time is represented along the vertical axis

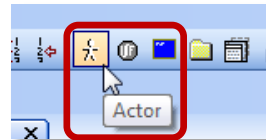
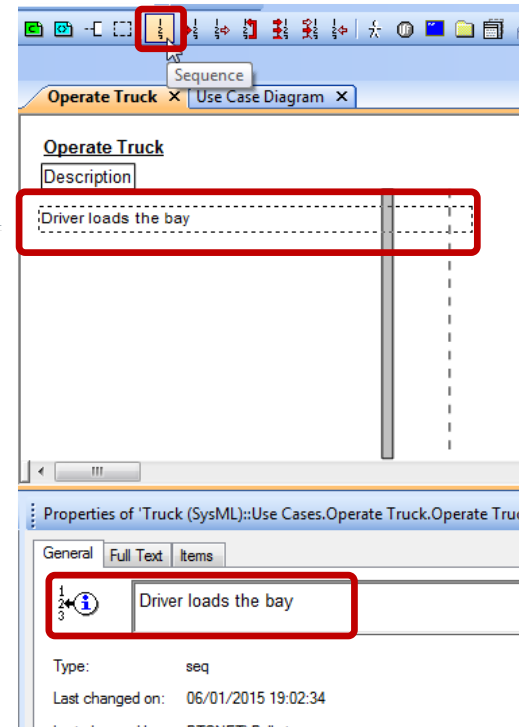
Enter name for diagram and select options



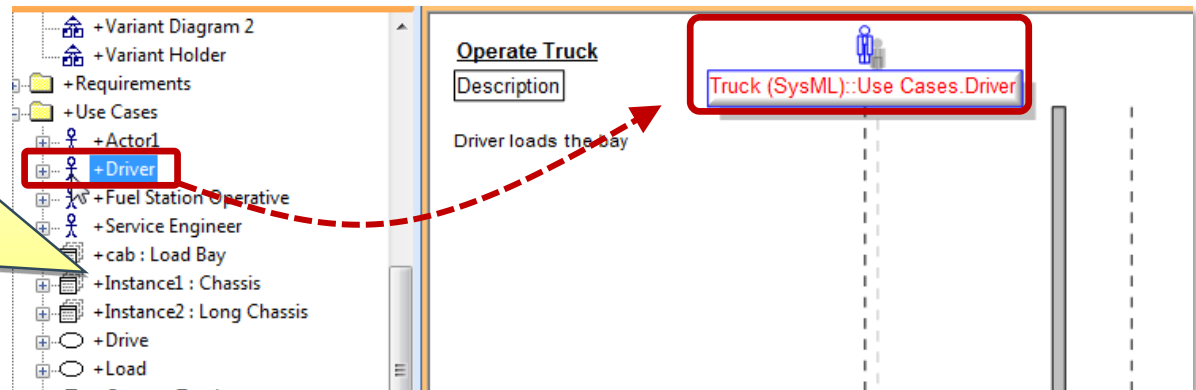
## ► Create Sequence Diagram with CFT



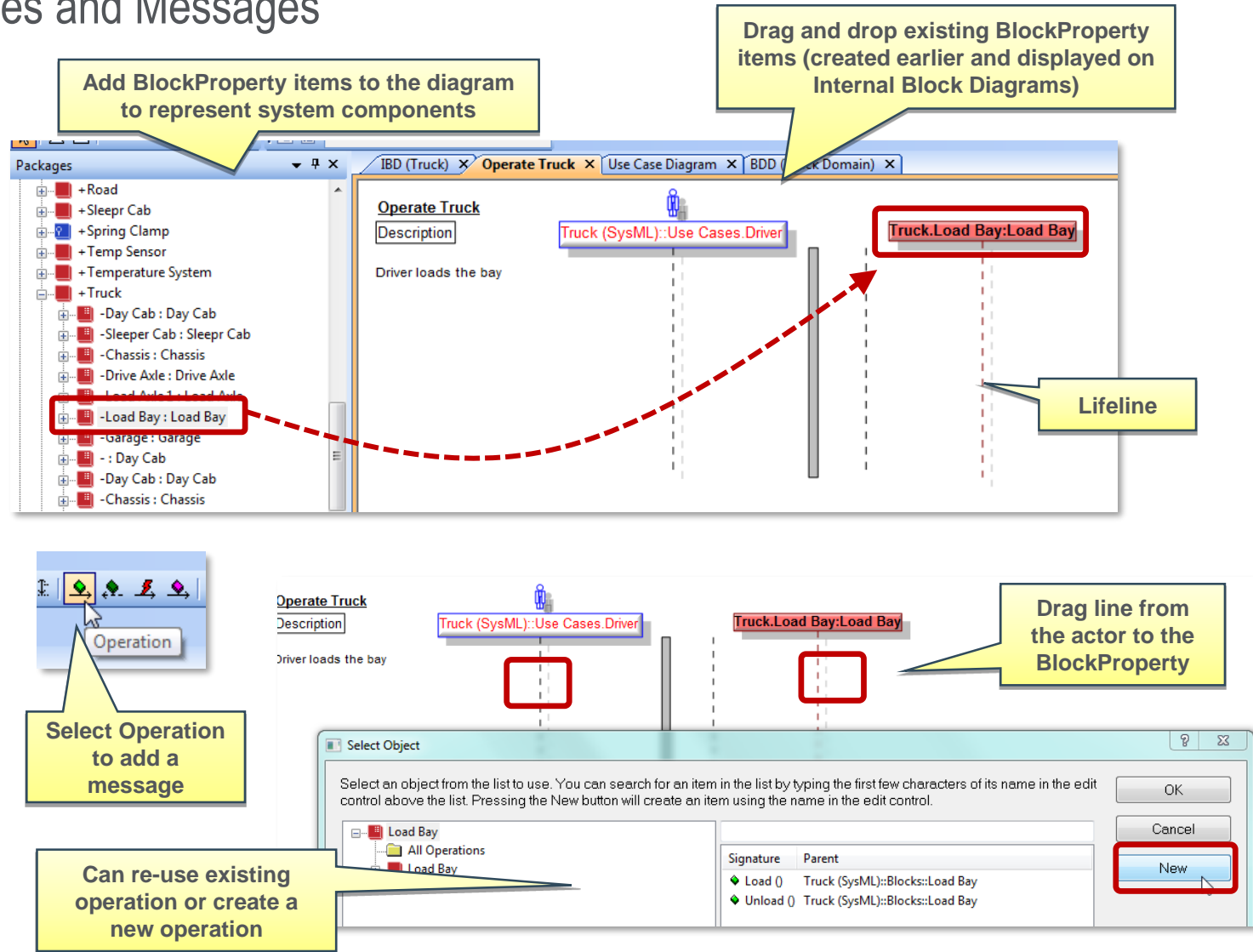
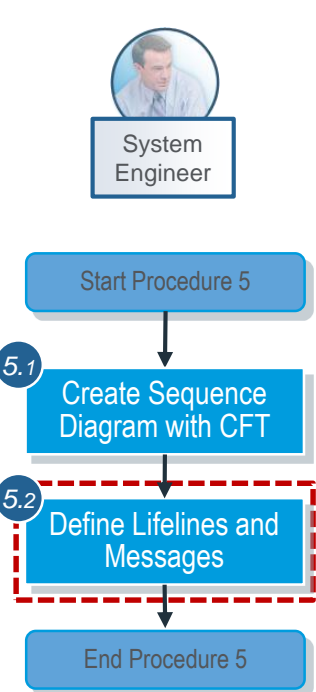
Click on Sequence and then click in diagram window below the text Description. Add a sequence to the diagram and enter a name



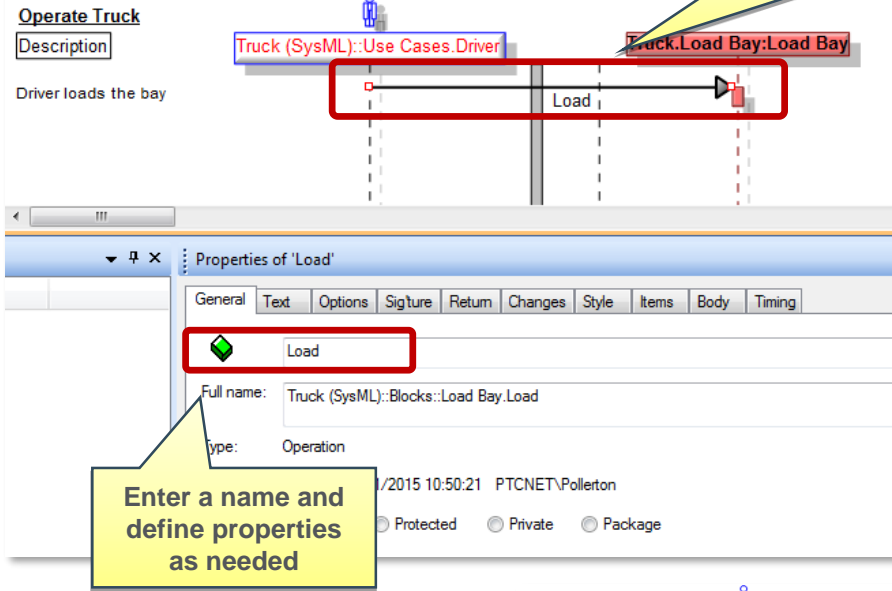
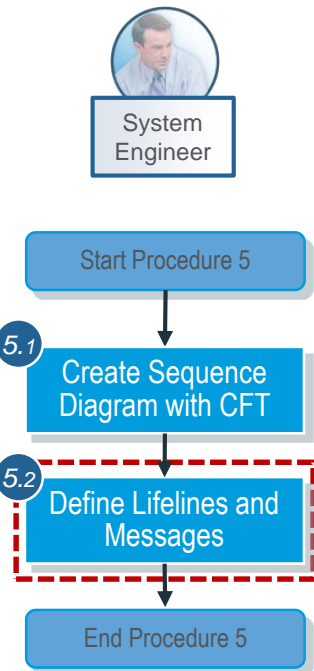
A new actor can be added, but if the correct actor already exists in the model, drag and drop into the diagram as shown below



## ► Define Lifelines and Messages



## ► Define Lifelines and Messages



**Note**

Operations describe functions that should be provided by the system

Continue to add operations to define the sequence of interactions

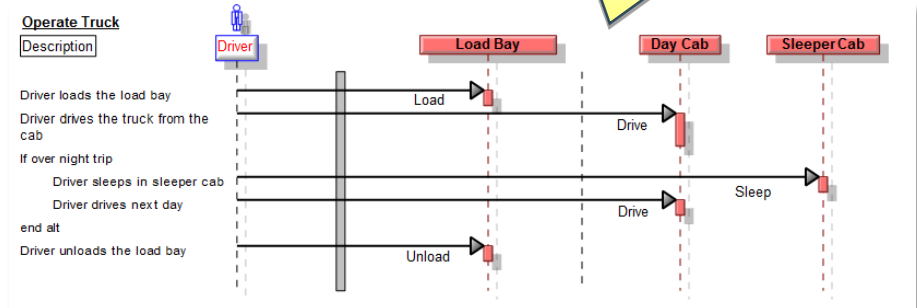
### Drive the Vehicle

#### Description

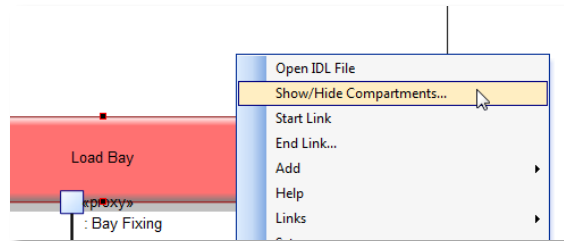
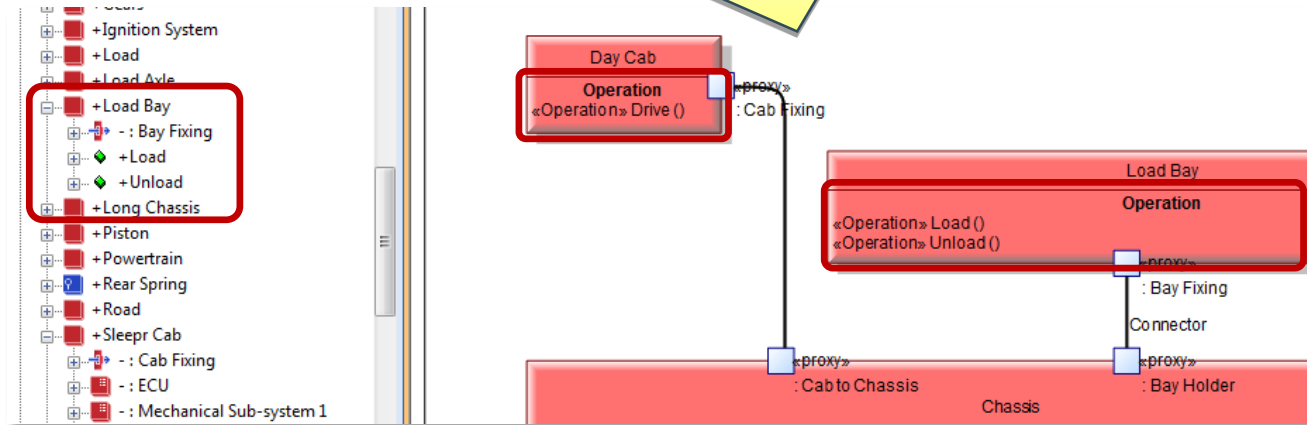
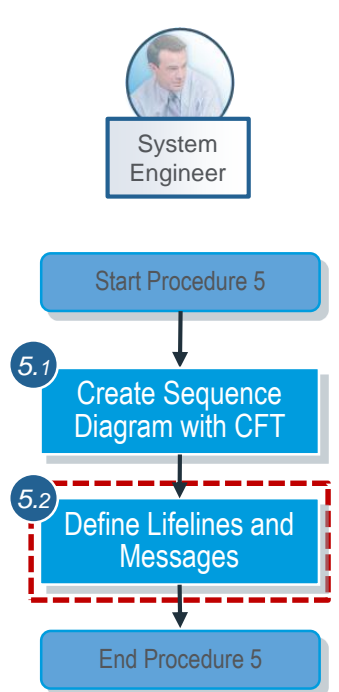
```

if engine not running
  startup
end if
loop
  accelerate
  brake
end loop
    
```

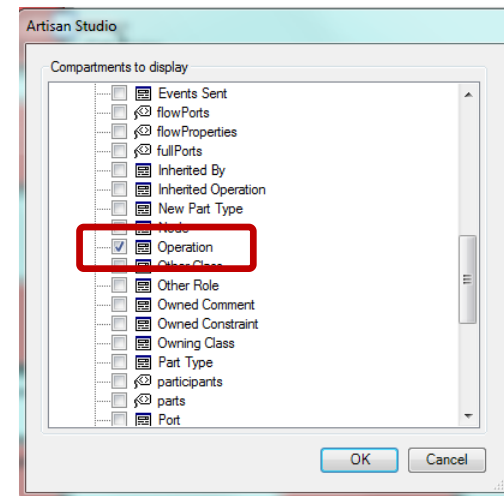
Logical constructs can be defined within the sequence



## ► Define Lifelines and Messages

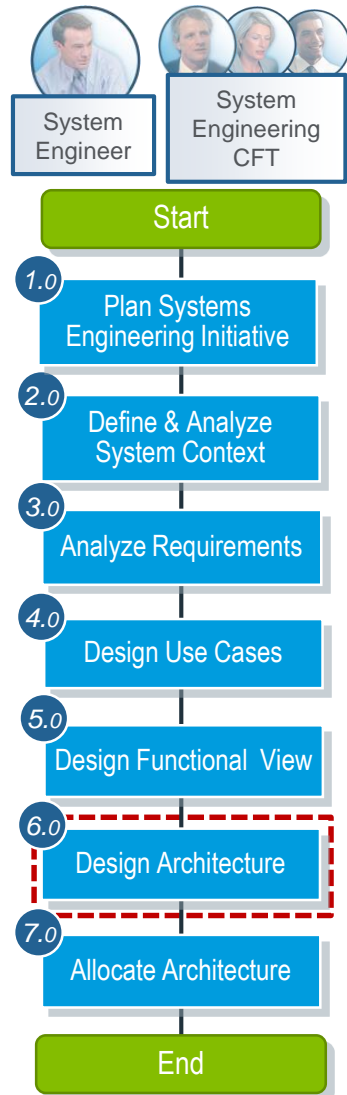


Right click on the block and select Show/Hide Compartments...



# Design Architecture

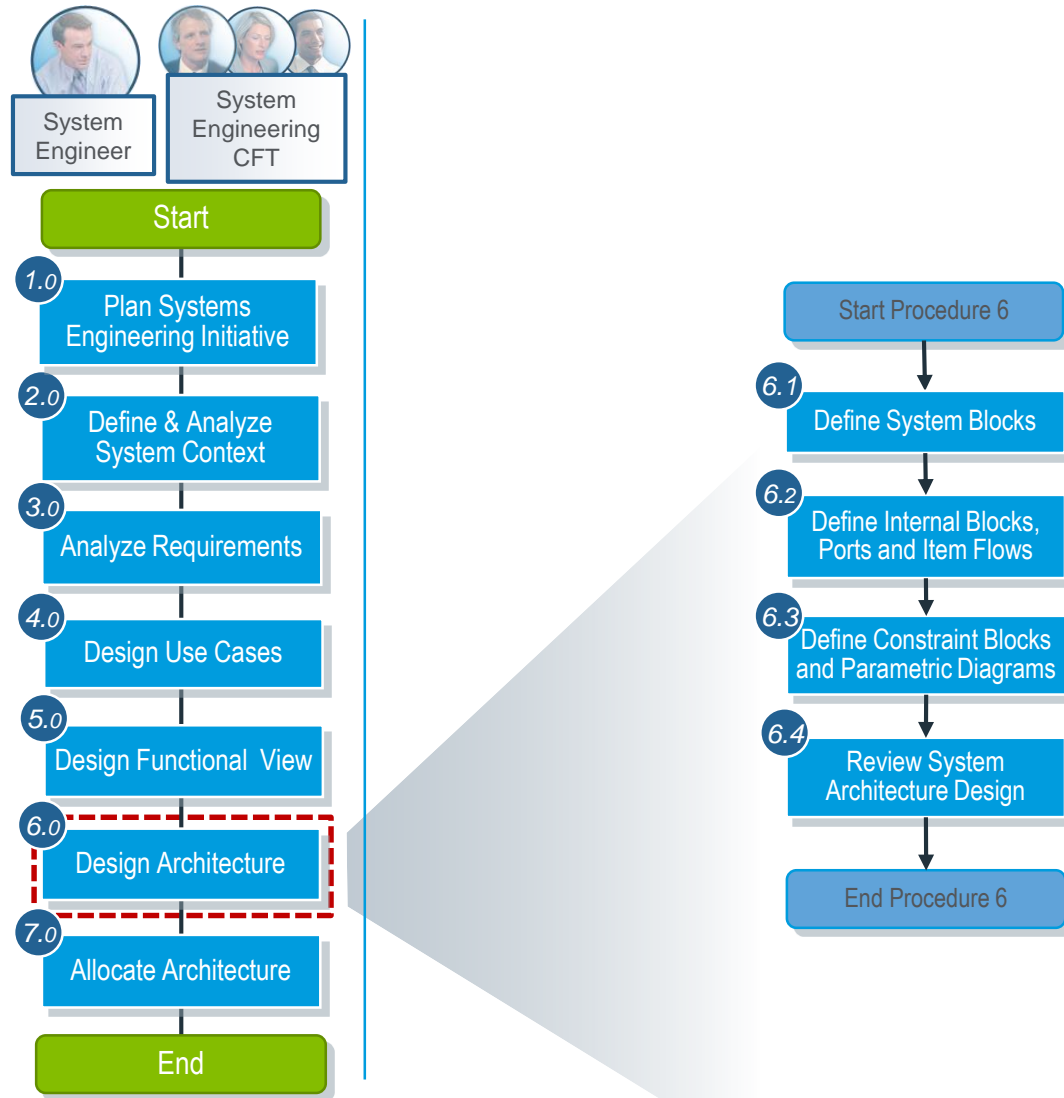
## ► Design Architecture



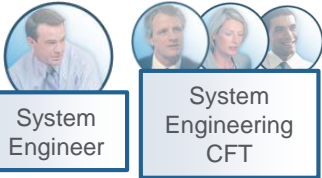
- Objectives
  - Design, analyze and review system architecture
- Role
  - System Engineer
  - Cross-functional Team
- Outputs
  - System Architecture Model



## ► Design Architecture



## ► Update System Model



### Note

Before commencing the detailed design of the system, create another baseline of the model. System requirements should be frozen and only be modified via a managed change process

Start Procedure 6

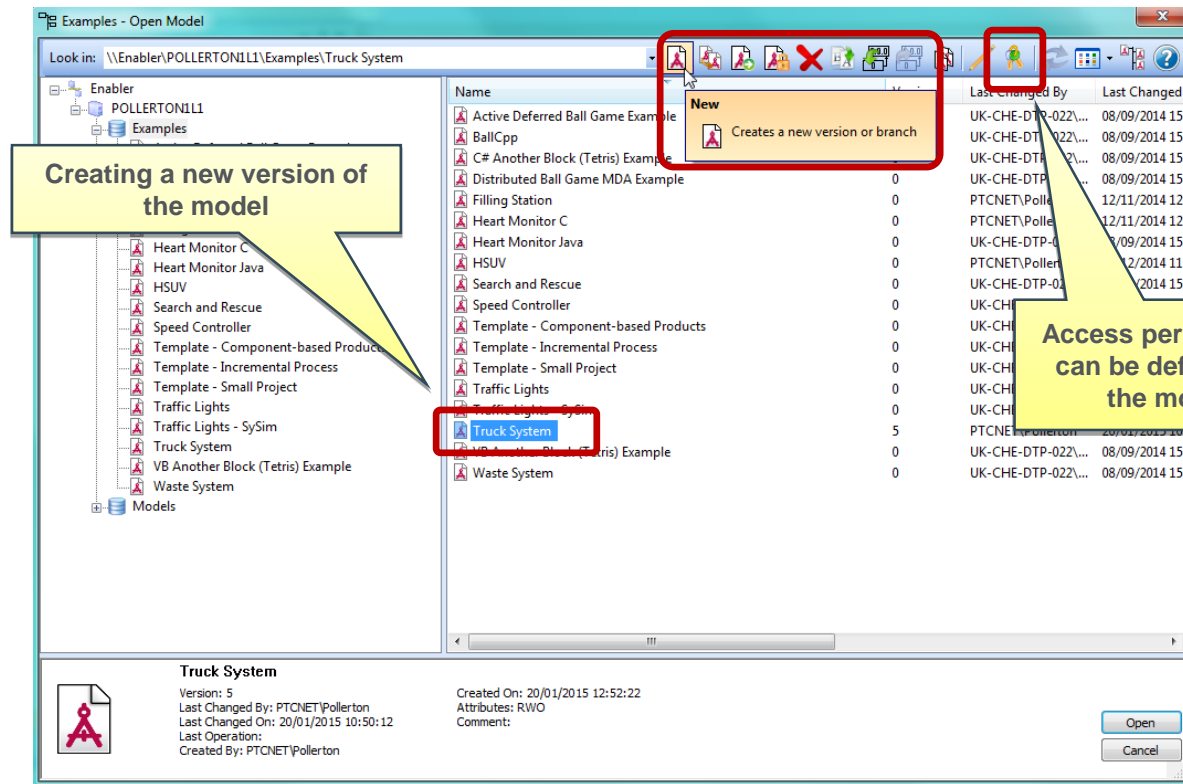
6.1 Define System Blocks

6.2 Define Internal Blocks, Ports and Item Flows

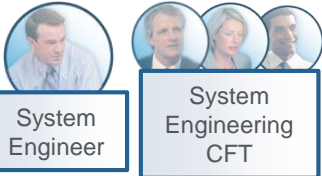
6.3 Define Constraint Blocks and Parametric Diagrams

6.4 Review System Architecture Design

End Procedure 6



## ► Define System Blocks



Start Procedure 6

6.1 Define System Blocks

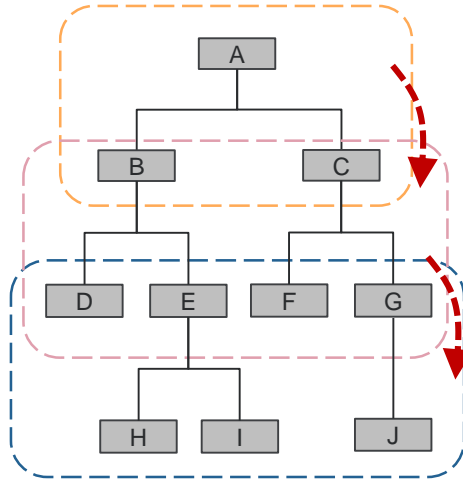
6.2 Define Internal Blocks, Ports and Item Flows

6.3 Define Constraint Blocks and Parametric Diagrams

6.4 Review System Architecture Design

End Procedure 6

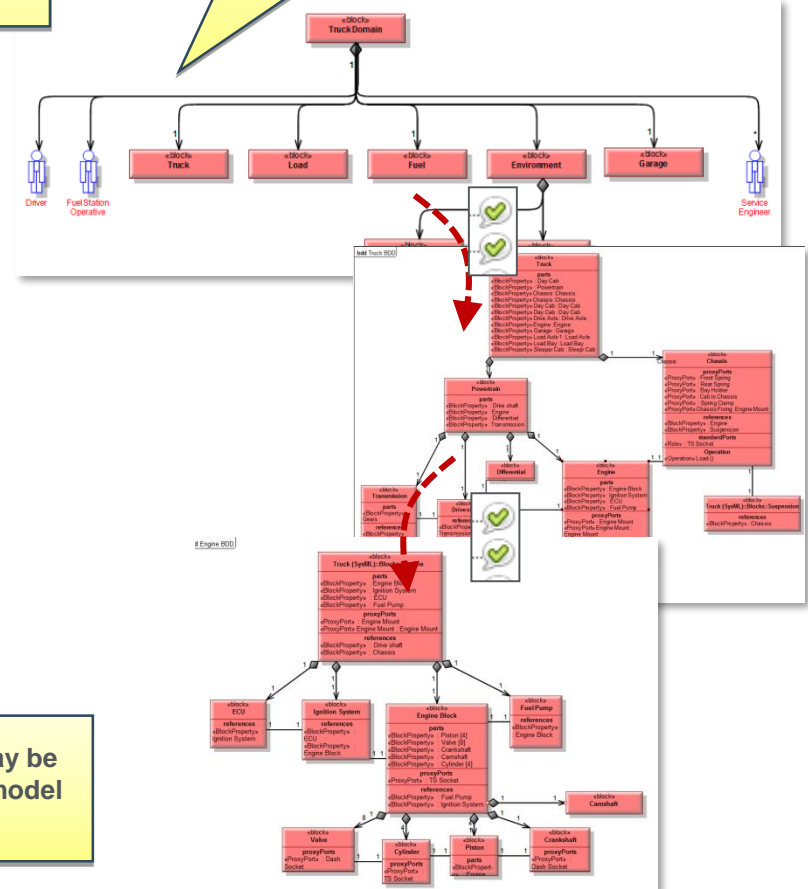
After creating the various models in previous steps, system requirements will have been refined and elaborated and there will be a greater understanding of the needed system/sub-system structure as well as system functions



Note

Sub-systems that can stand-alone may be better defined as a separate system model and then linked using Asset Library

Cascade requirements down from higher levels and define Block Definition Diagrams and Internal Block Diagrams to describe the various sub-systems



## ► Define System Blocks



Start Procedure 6

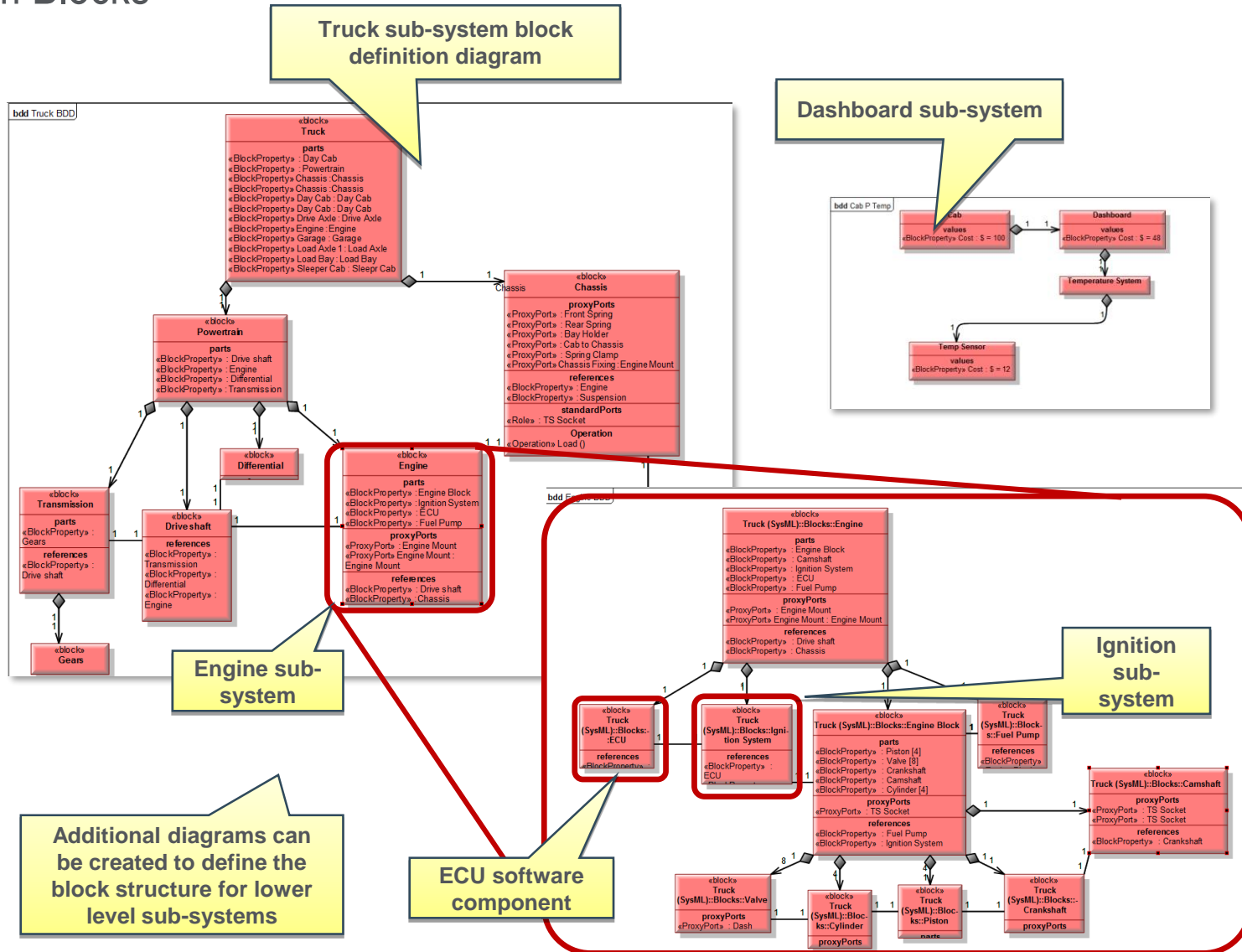
6.1 Define System Blocks

6.2 Define Internal Blocks, Ports and Item Flows

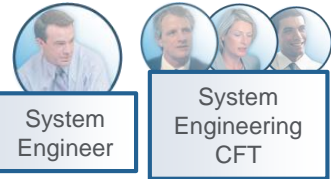
6.3 Define Constraint Blocks and Parametric Diagrams

6.4 Review System Architecture Design

End Procedure 6



## ► Define System Blocks



Start Procedure 6

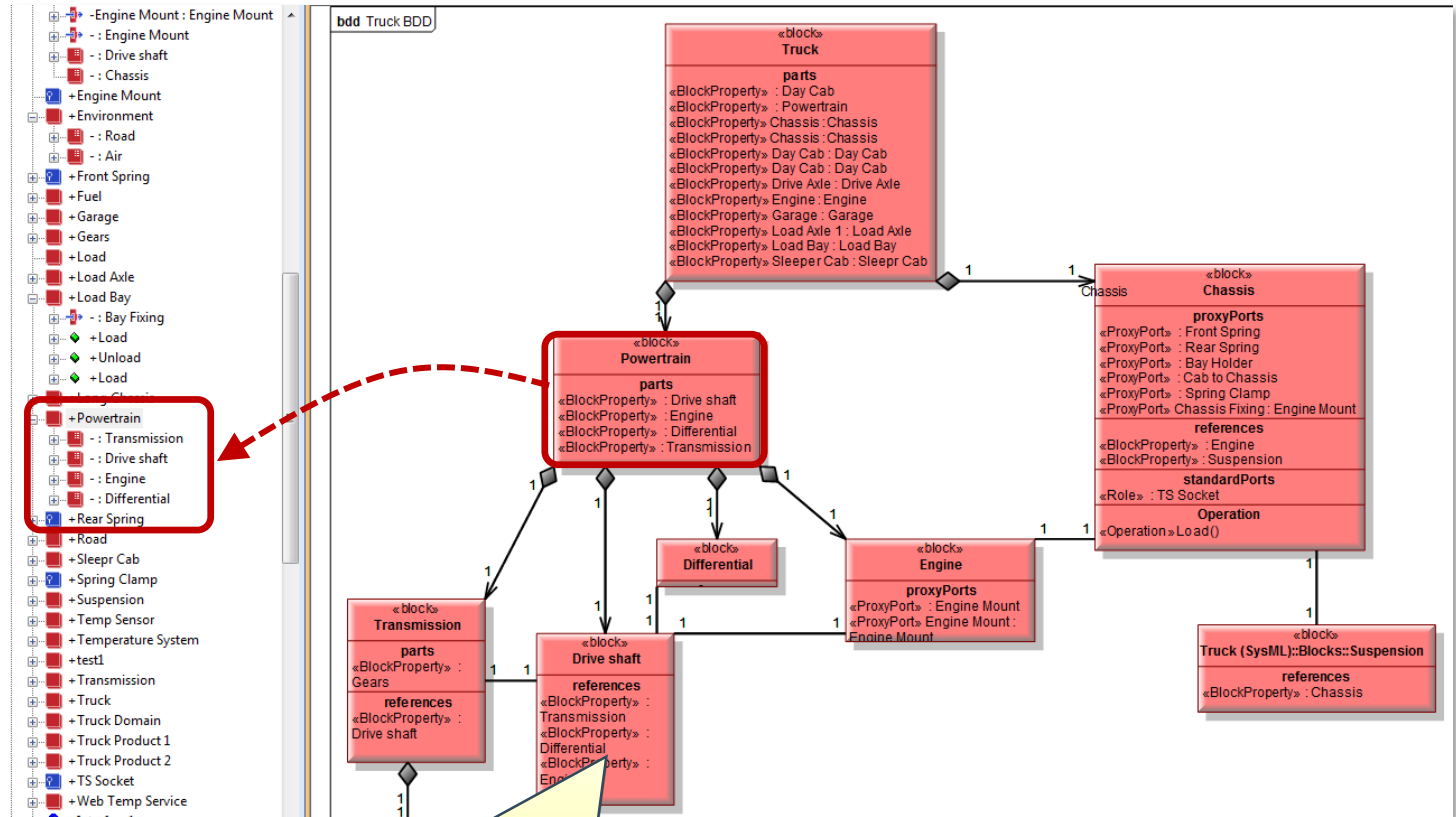
6.1 Define System Blocks

6.2 Define Internal Blocks, Ports and Item Flows

6.3 Define Constraint Blocks and Parametric Diagrams

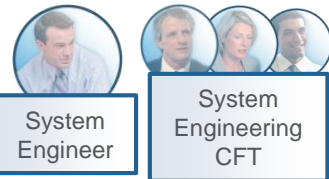
6.4 Review System Architecture Design

End Procedure 6



Note that BlockProperty entities are created where blocks are associated. (Composite aggregations are defined as parts on the parent block). This is also displayed in the Package view

## ► Define System Blocks



Start Procedure 6

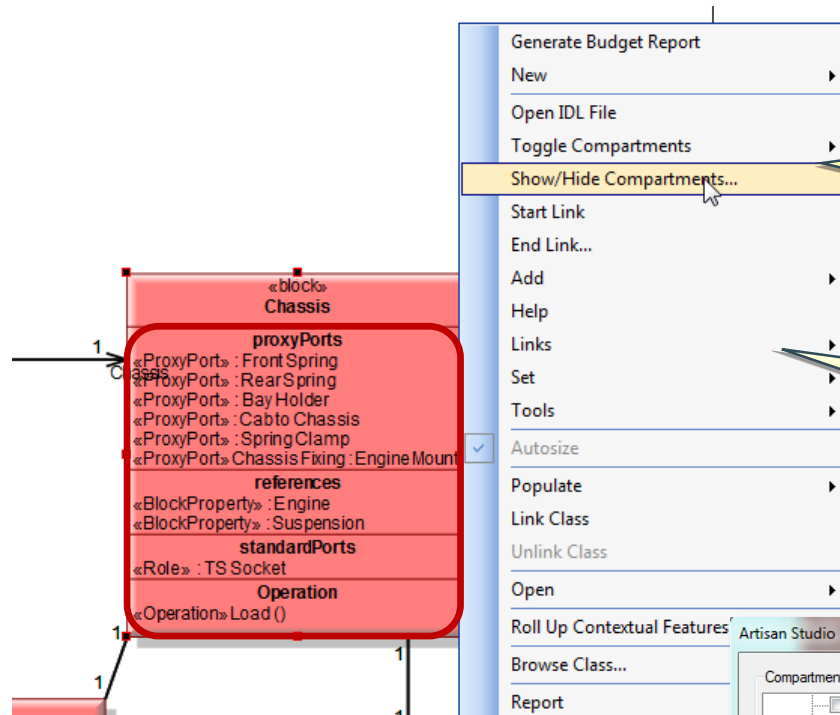
6.1 Define System Blocks

6.2 Define Internal Blocks, Ports and Item Flows

6.3 Define Constraint Blocks and Parametric Diagrams

6.4 Review System Architecture Design

End Procedure 6



Generate Budget Report

New

Open IDL File

Toggle Compartments

Show/Hide Compartments...

Start Link

End Link...

Add

Help

Links

Set

Tools

Autosize

Populate

Link Class

Unlink Class

Open

Roll Up Contextual Features

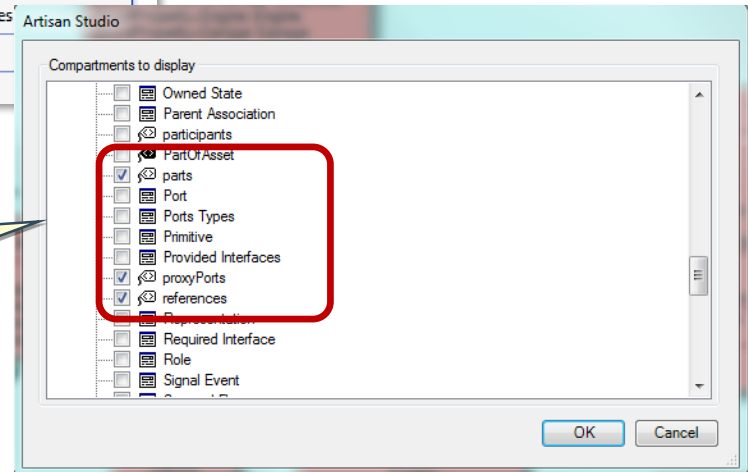
Browse Class...

Report

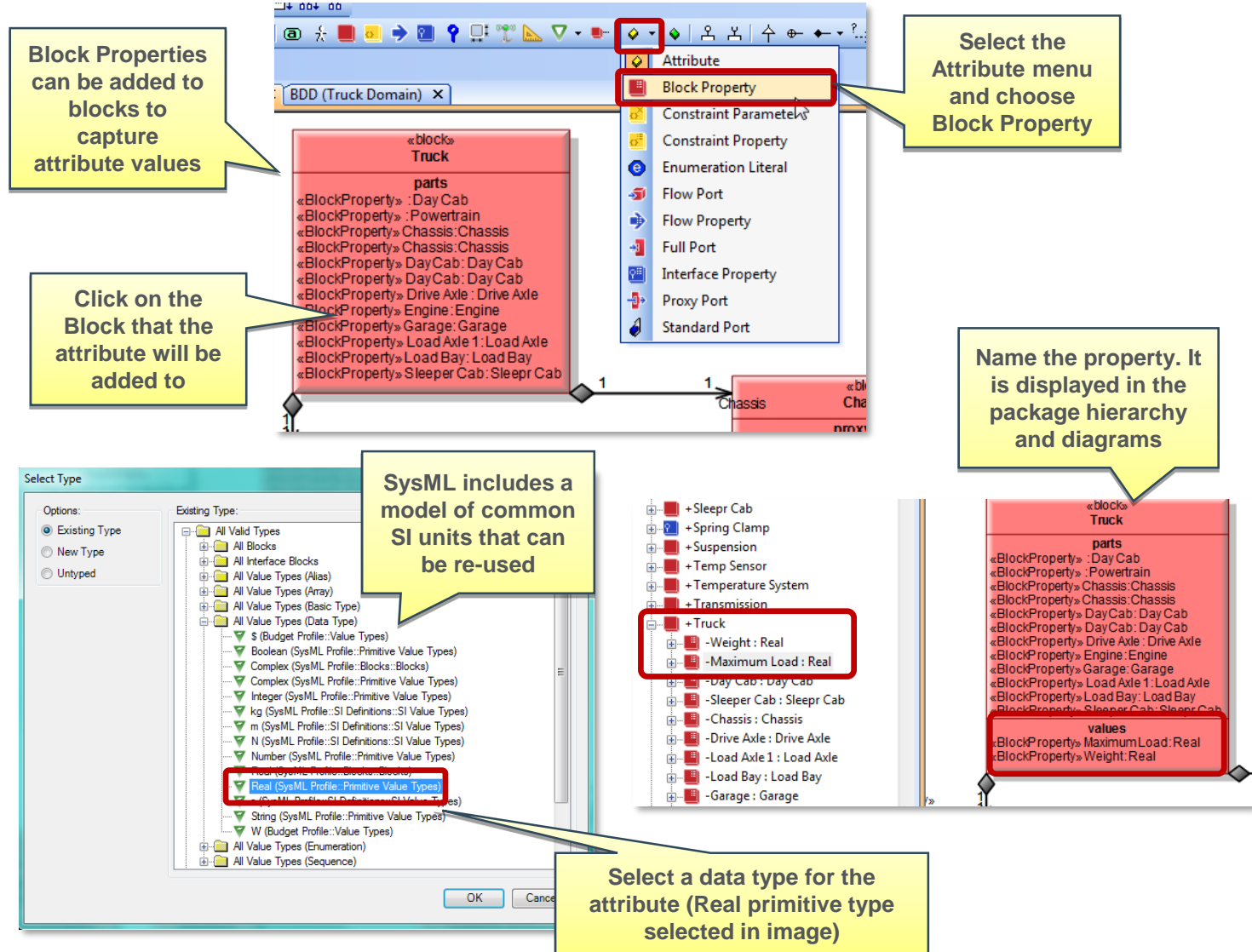
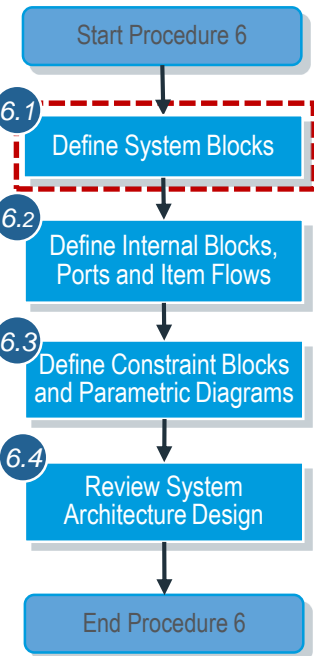
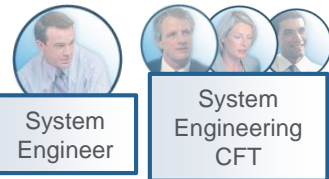
Various properties such as attribute values, parts, references and associations can be displayed or hidden

Right click on a block and select Show/Hide Compartments

Choose what information to display

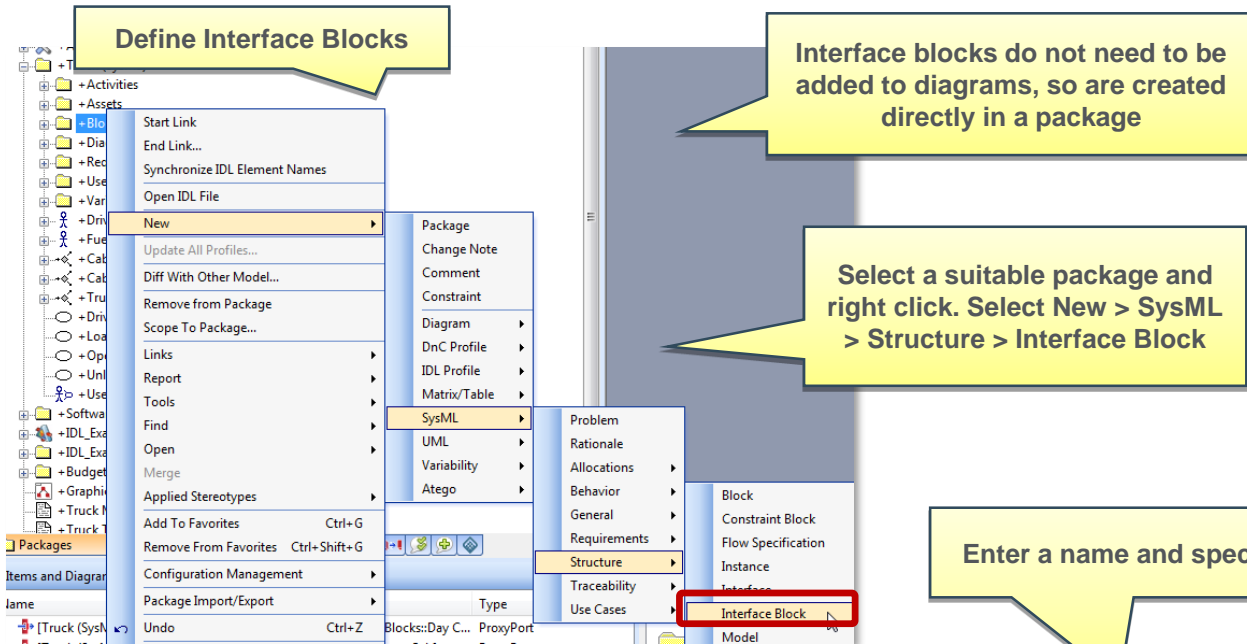
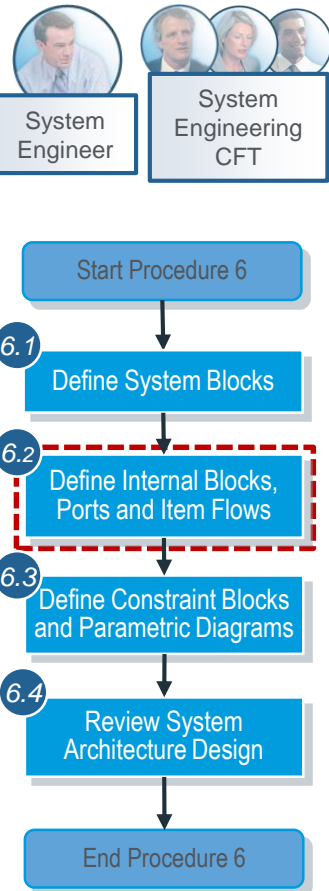


## ► Define System Blocks



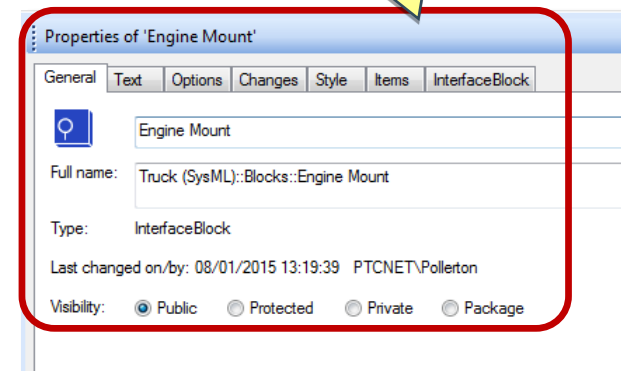


## ► Define Internal Blocks, Ports and Item Flows

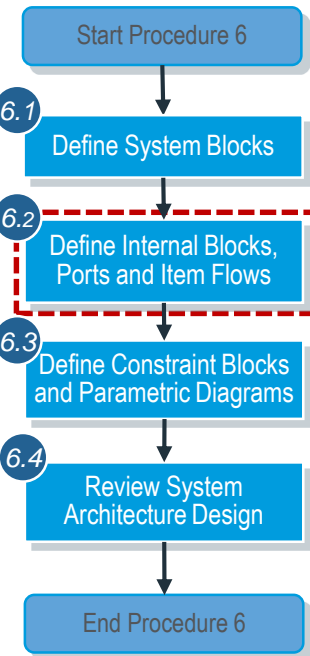
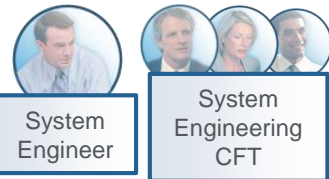


### Note

Interface blocks are specialized blocks that have no behaviors or internal parts and are used to type Proxy Ports (used in Internal Block Diagrams)



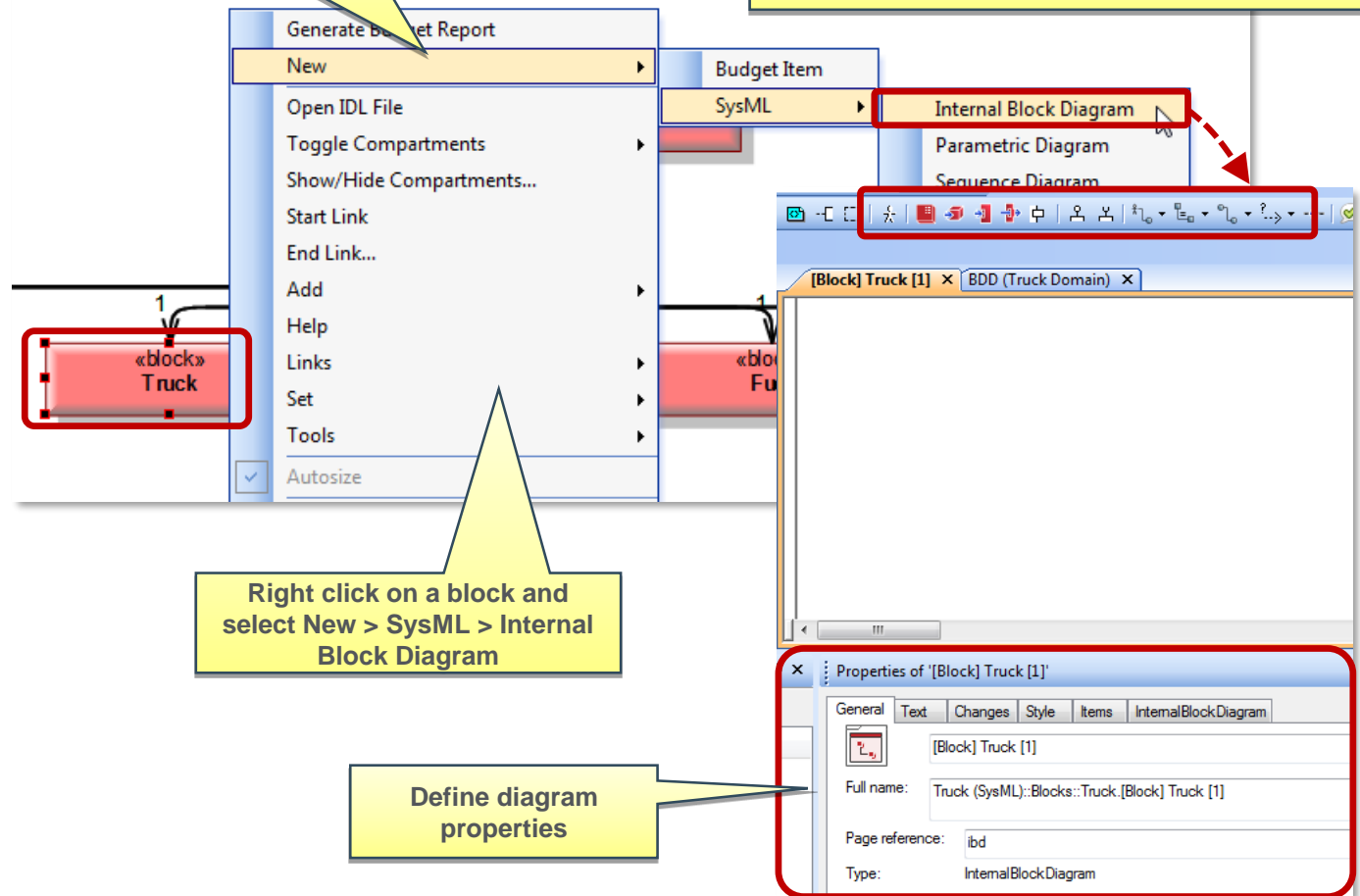
## ► Define Internal Blocks, Ports and Item Flows



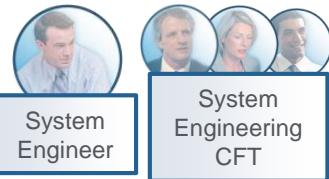
Capture more details on the internal structure of each block using an Internal Block Diagram

Note

Block Definition Diagrams (BDDs) show the structure of the system functions. Internal Block Diagrams (IBDs) look inside the blocks to define the data/information/item flows and the ports



## ► Define Internal Blocks, Ports and Item Flows



Start Procedure 6

6.1 Define System Blocks

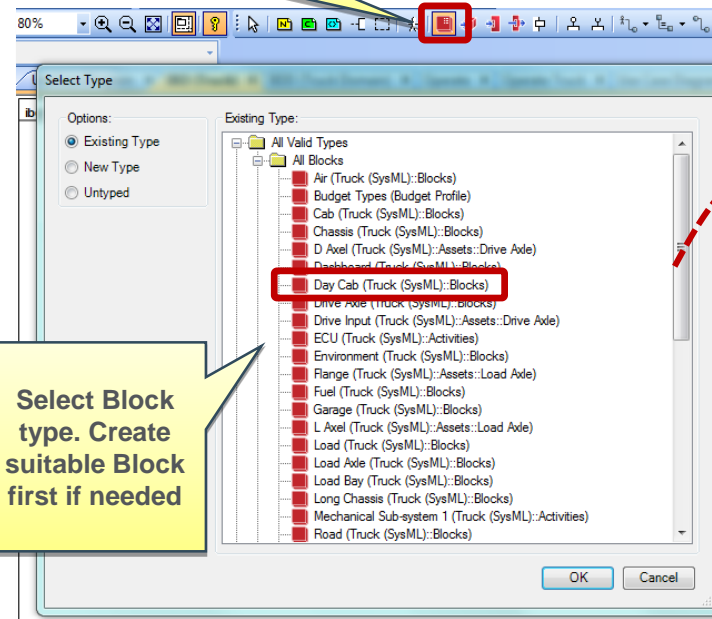
6.2 Define Internal Blocks, Ports and Item Flows

6.3 Define Constraint Blocks and Parametric Diagrams

6.4 Review System Architecture Design

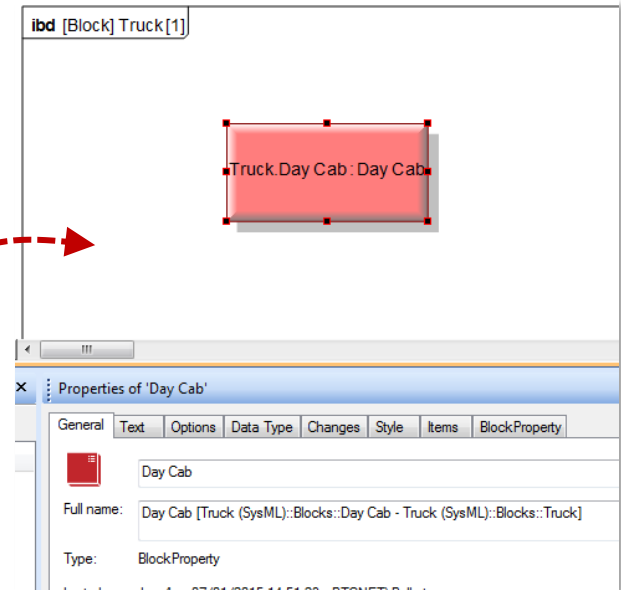
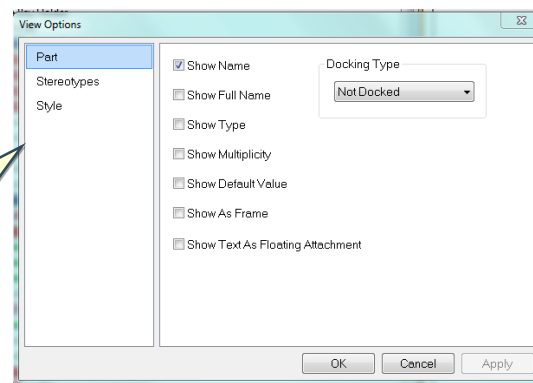
End Procedure 6

Add a Block Property



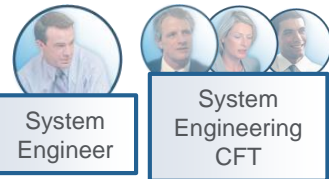
Select Block type. Create suitable Block first if needed

Right click on the Block Property and select View Options



Optionally show name, full name, type, multiplicity, etc.

## ► Define Internal Blocks, Ports and Item Flows



Start Procedure 6

6.1 Define System Blocks

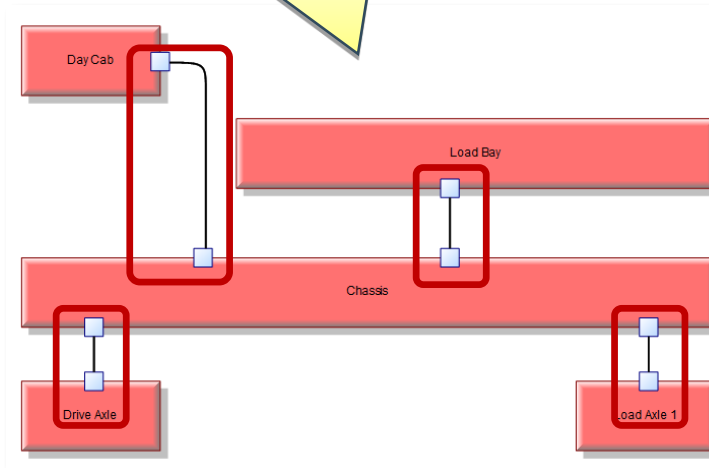
6.2 Define Internal Blocks, Ports and Item Flows

6.3 Define Constraint Blocks and Parametric Diagrams

6.4 Review System Architecture Design

End Procedure 6

Define ports and connectors

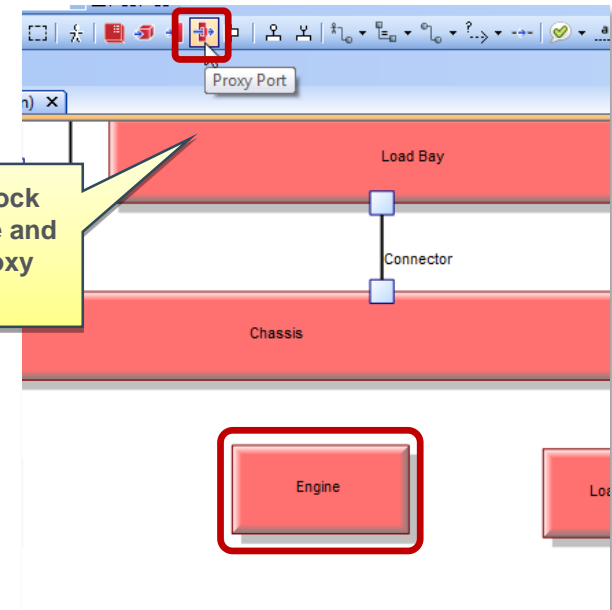


Ports are points at which external entities can connect to and interact with a block in different or more limited ways than connecting directly to the block itself. They are properties with a type that specifies features available to the external entities via connectors to the ports

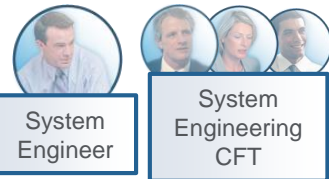
**Note**

Ports and flows can be used to enable design of modular, reusable blocks with clearly defined ways of connecting and interacting with their context of use

Add new Block called Engine and click on Proxy Port



## ► Define Internal Blocks, Ports and Item Flows



Start Procedure 6

6.1 Define System Blocks

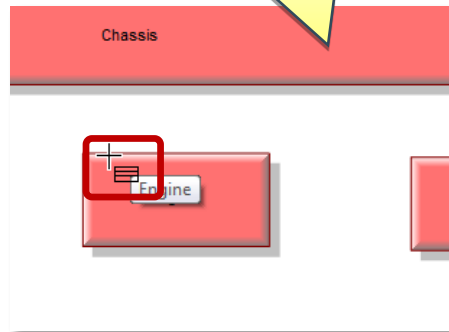
6.2 Define Internal Blocks, Ports and Item Flows

6.3 Define Constraint Blocks and Parametric Diagrams

6.4 Review System Architecture Design

End Procedure 6

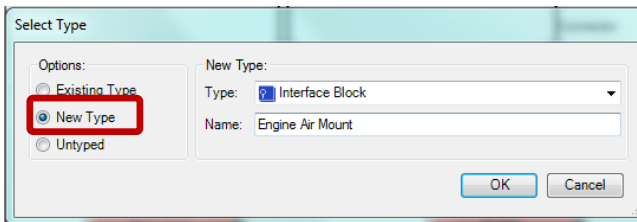
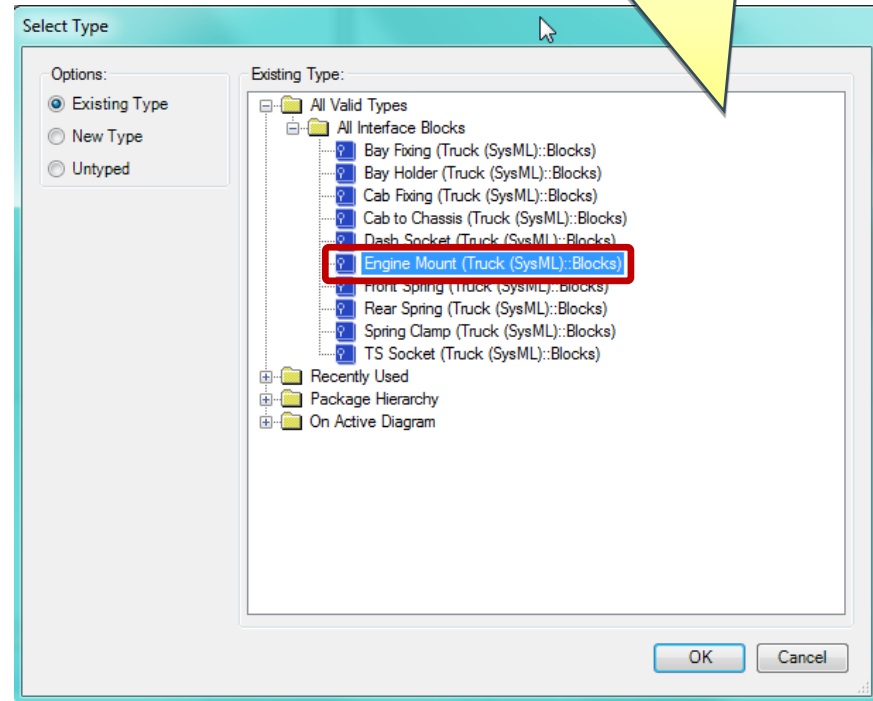
Click on the top edge of the Engine block



**Note**

Can also create a new Interface Block type

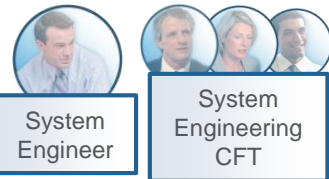
Select type for new port – we will use the Engine Mount Interface Block created earlier



**Note**

Port definitions may be driven by system requirements (e.g. communication network protocols)

## ► Define Internal Blocks, Ports and Item Flows



Start Procedure 6

6.1 Define System Blocks

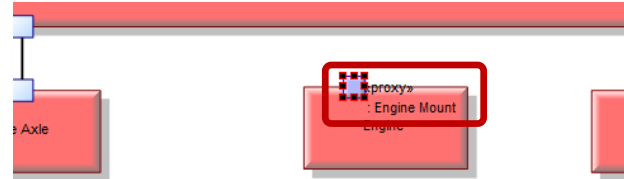
6.2 Define Internal Blocks, Ports and Item Flows

6.3 Define Constraint Blocks and Parametric Diagrams

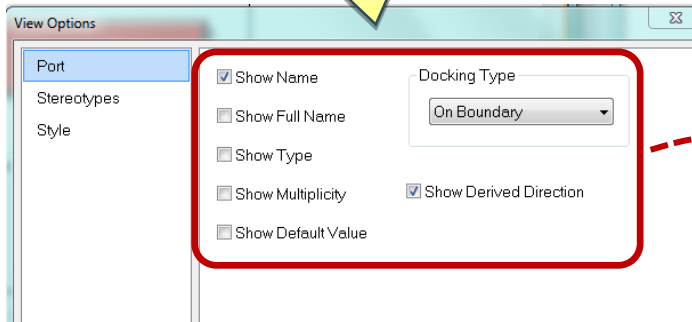
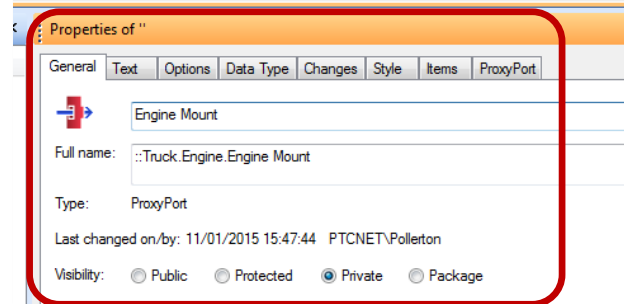
6.4 Review System Architecture Design

End Procedure 6

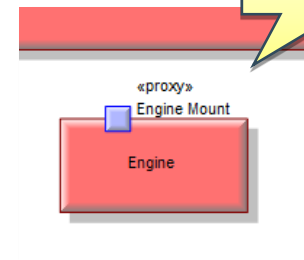
Name the port and choose options



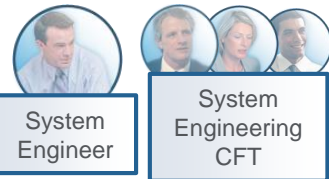
Right click on the port and select View Options



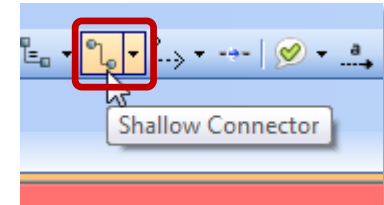
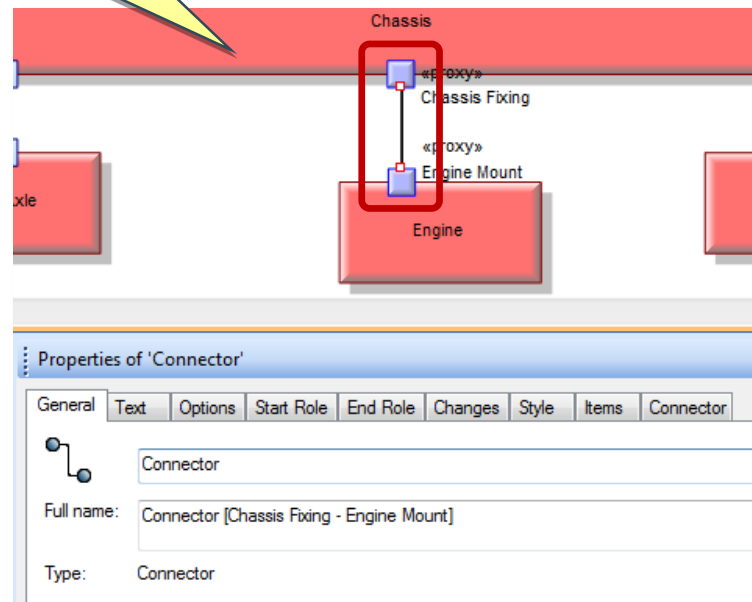
Text can be repositioned



## ► Define Internal Blocks, Ports and Item Flows



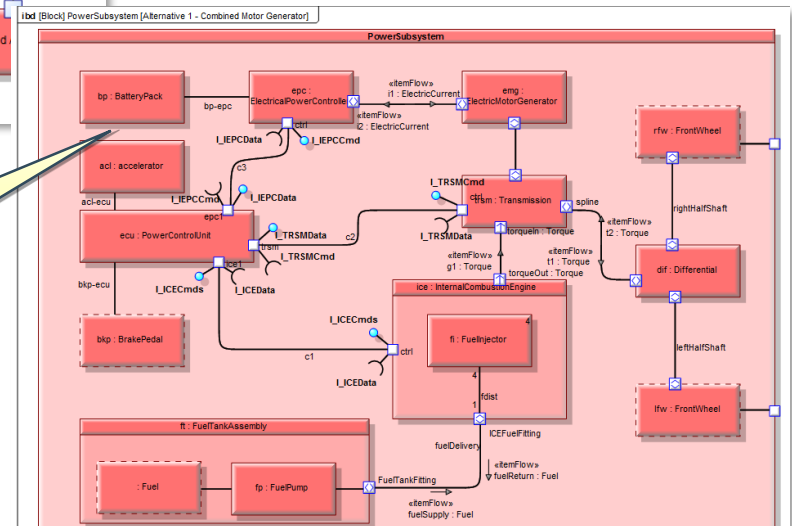
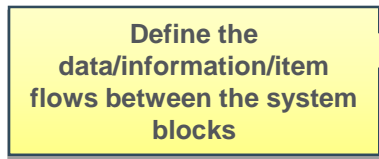
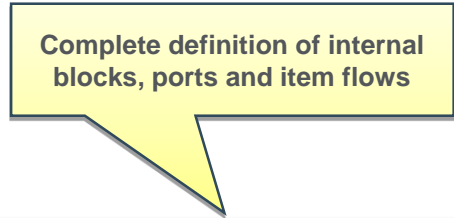
Add a port to the Chassis block and then add a Shallow Connector



### Note

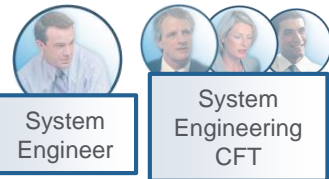
Ports and connectors represent an interface between system entities. This concept is explained further in a later procedure.

The diagram shows two roles: 'System Engineer' and 'System Engineering CFT'. The 'System Engineer' role is represented by a single circular portrait of a man. The 'System Engineering CFT' role is represented by a group of four circular portraits: a man, a woman, and two other men.





## ► Define Constraint Blocks and Parametric Diagrams



Start Procedure 6

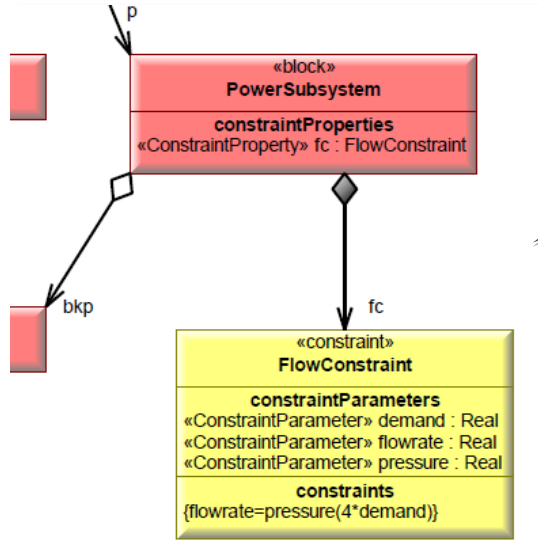
6.1 Define System Blocks

6.2 Define Internal Blocks, Ports and Item Flows

6.3 Define Constraint Blocks and Parametric Diagrams

6.4 Review System Architecture Design

End Procedure 6

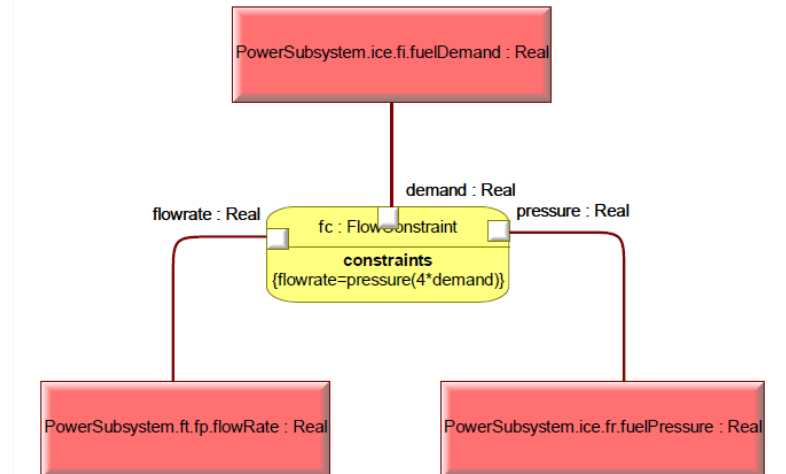


Constraint Blocks can be optionally defined within BDDs to represent system properties such as cost, power output, weight, flow rate etc.

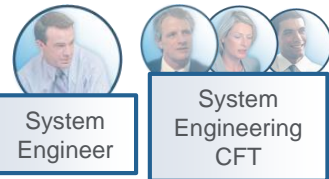
Parametric diagrams are used to define how the parameters for the constraint relate to specific value properties of the sub-system block

**Note**

Parametric diagrams and constraint block should be used to capture non-functional requirements



## ► Define Constraint Blocks and Parametric Diagrams



Start Procedure 6

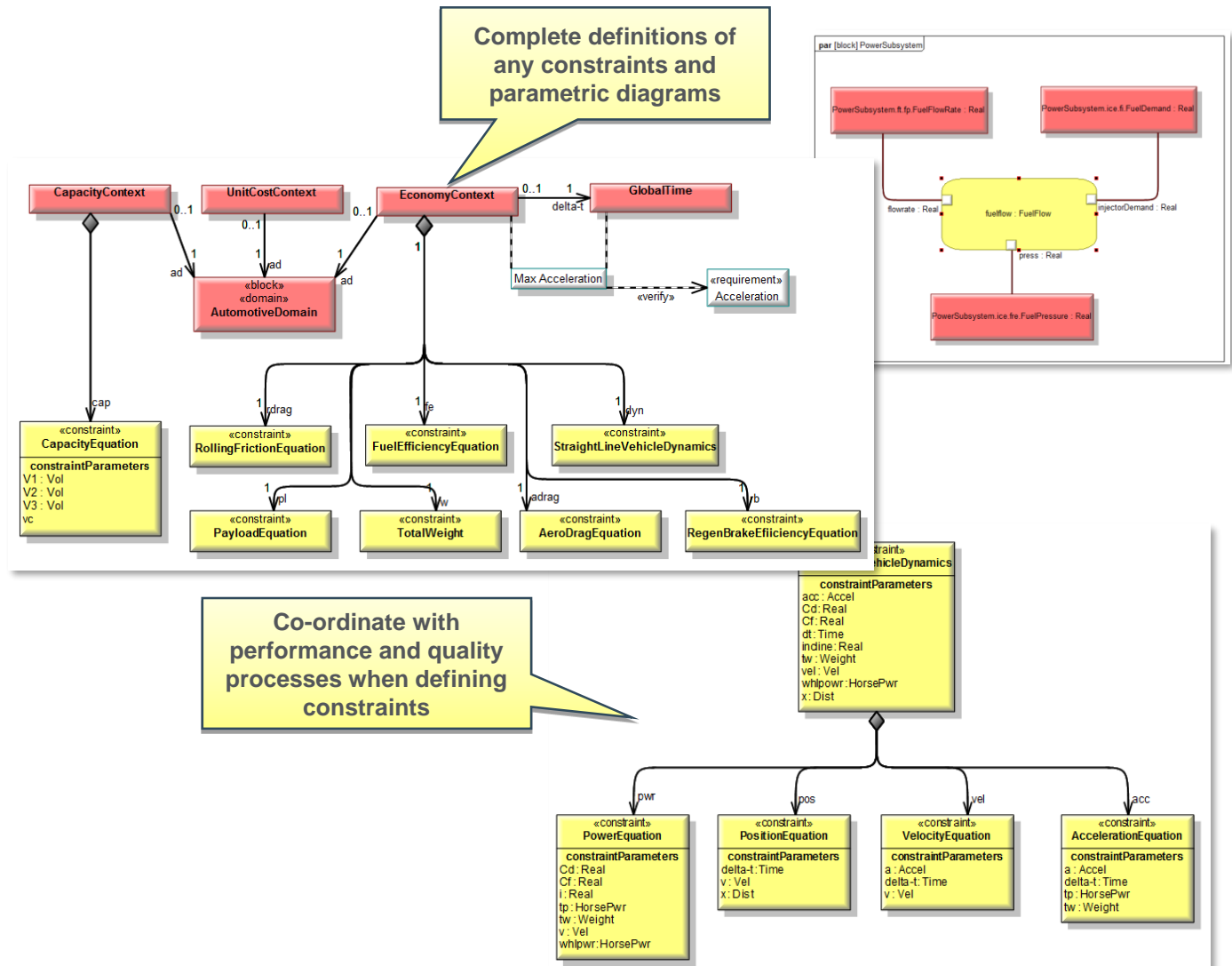
6.1 Define System Blocks

6.2 Define Internal Blocks, Ports and Item Flows

6.3 Define Constraint Blocks and Parametric Diagrams

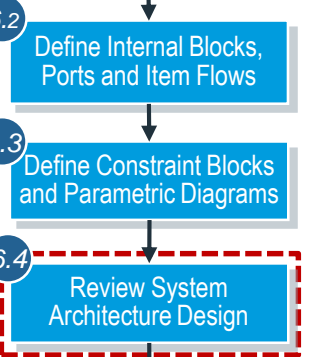
6.4 Review System Architecture Design

End Procedure 6



System Engineer

System Engineering CFT



The Gantt chart illustrates the project lifecycle with the following tasks and their durations:

- Plan** (Phase 1)
- Commit** (Phase 2)
- Design** (Phase 3)
- Release** (Phase 4)
- Support** (Phase 5)

**Project Management** (Spans Plan to Commit)

- Environmental Performance Management** (Spans Plan to Commit)
- Regulatory Compliance** (Spans Plan to Commit)
- Quality & Reliability Management** (Spans Plan to Commit)
- Change and Configuration Management** (Spans Plan to Commit)

**Concept Development** (Spans Commit to Design)

- System Architecture Design** (Spans Commit to Design)
- Product Development** (Spans Commit to Design)

**Verifications and Validations** (Spans Design to Release)

- Variant Design & Generation** (Spans Design to Release)

**Product Cost Management** (Spans Release to Support)

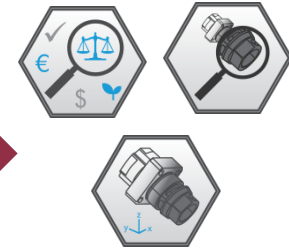
- Design and Manufacturing Outsourcing** (Spans Release to Support)

**Component and Supplier Management** (Spans Release to Support)

- Manufacturing Process Management** (Spans Release to Support)
- Tooling Design and Manufacture** (Spans Release to Support)

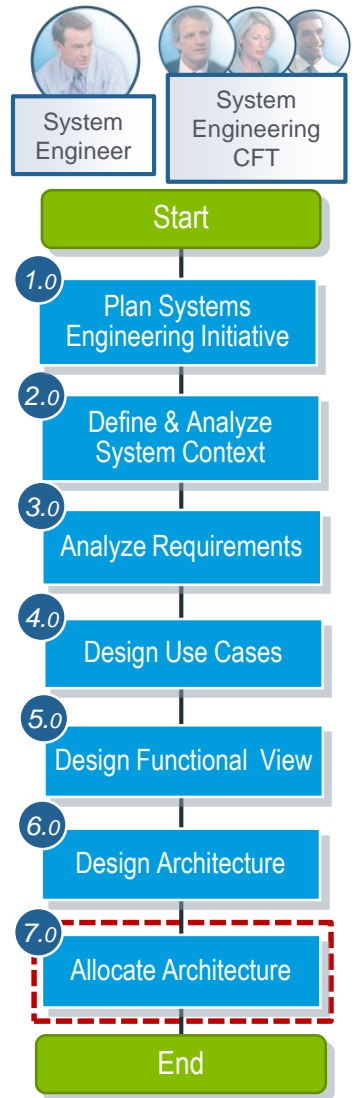
**Requirements Definition and Management** (Spans Plan to Support)

- Threat, Request Analysis, and Planning** (Spans Plan to Support)
- Service and Parts Information Creation and Management** (Spans Release to Support)
- Service Information Delivery and Field Knowledge Management** (Spans Release to Support)
- Equipment Lifecycle Management** (Spans Release to Support)
- Service and Parts Sales and Delivery** (Spans Release to Support)
- Performance Analysis and Feedback** (Spans Release to Support)

[illegible]

# Allocate Architecture

## ► Allocate Architecture



- Objectives

- Define traceability from requirements to system architecture. Plan physical architecture

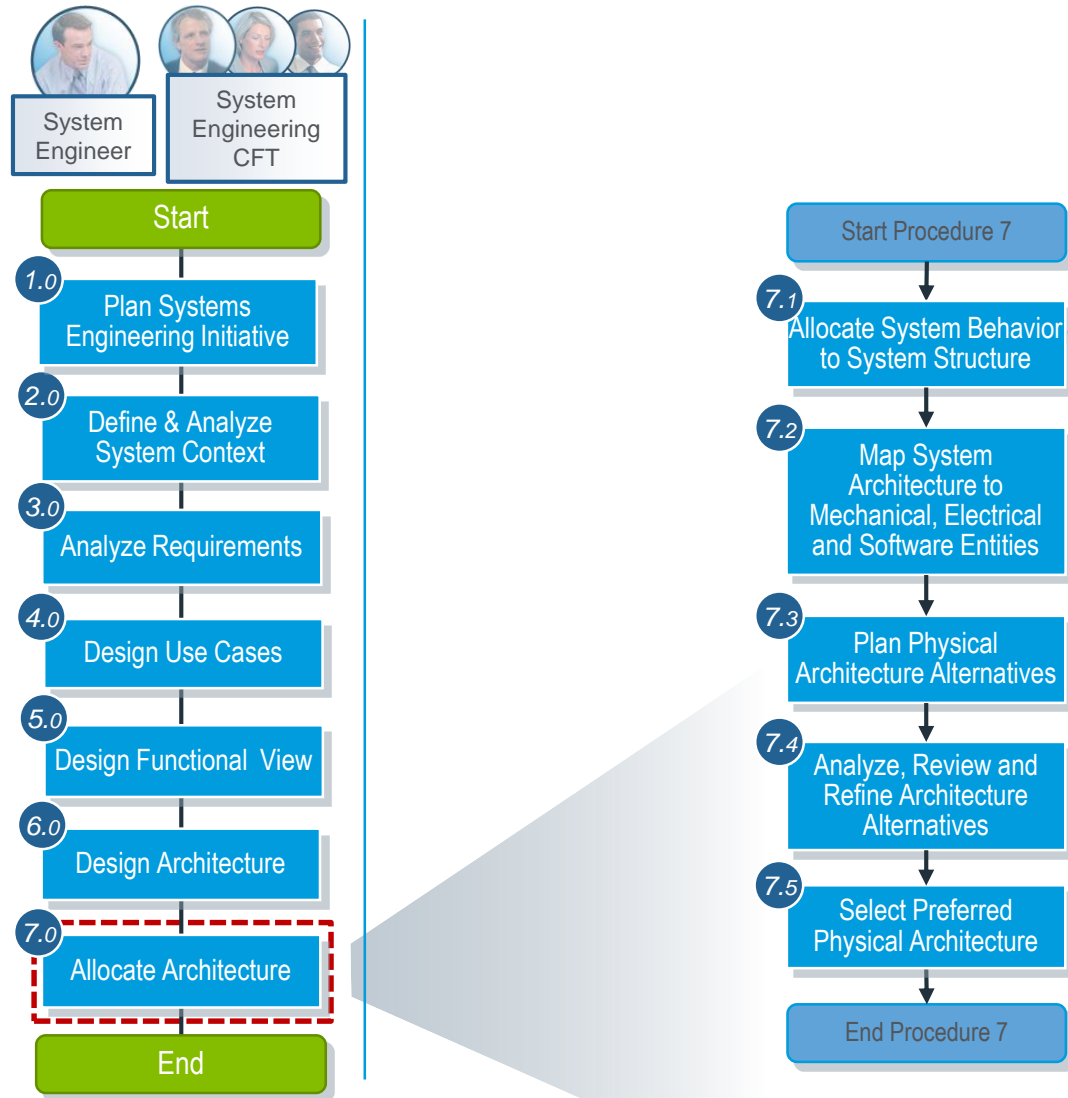
- Role

- System Engineer
- Cross-functional Team

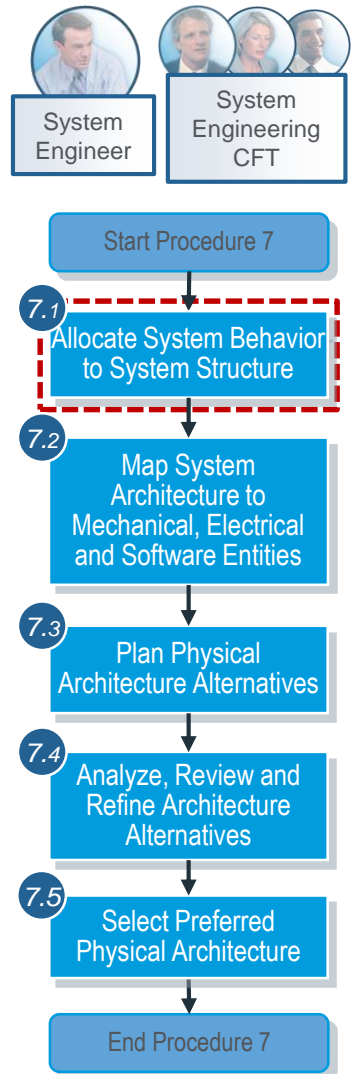
- Outputs

- Associations from Requirements to System Architecture Model
- Physical Architecture plans

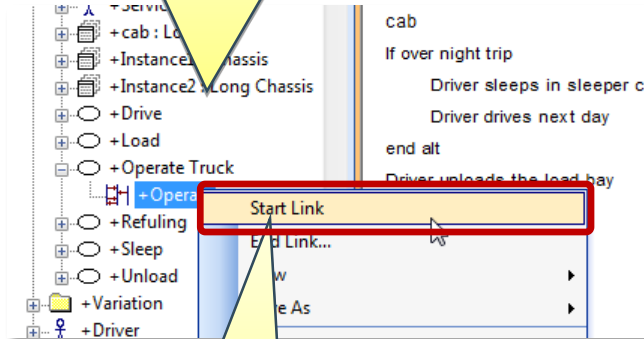
## ► Allocate Architecture



## ► Link System Behavior to System Structure

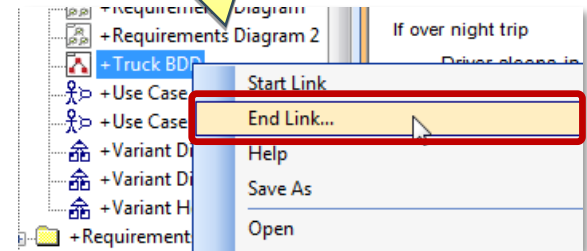


Allocation can be used to link system behavior to system structure, in this case linking the Operate Truck Sequence diagram with the Truck Block Definition Diagram

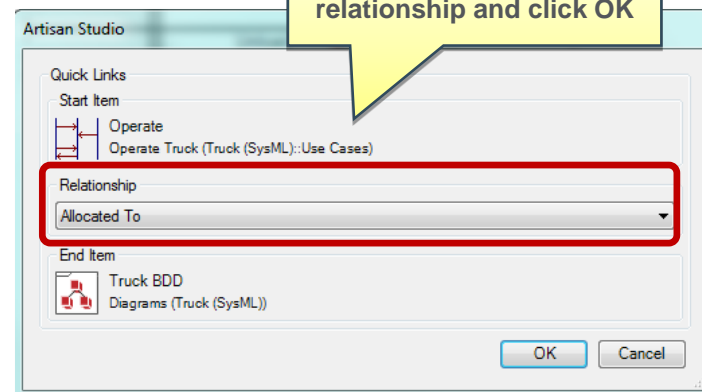


Right click on a sequence diagram and select Start Link...

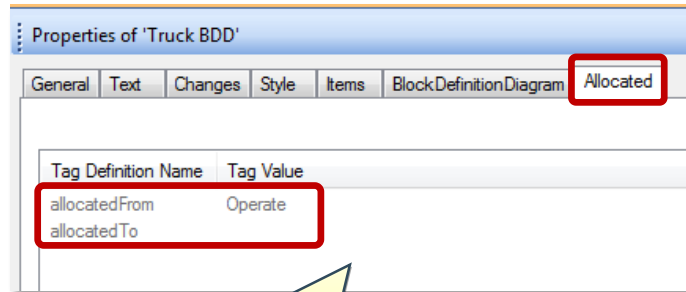
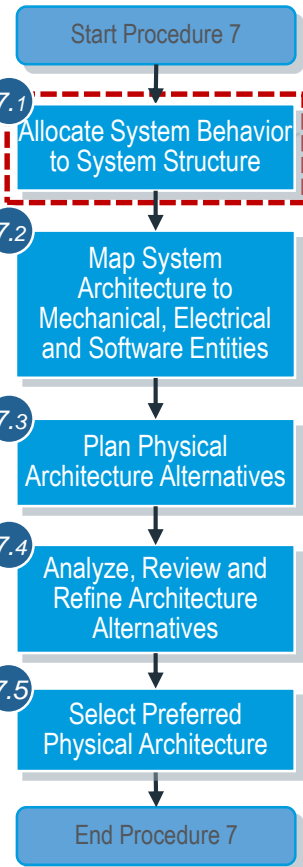
Right click on the block definition diagram to be linked and select End Link...



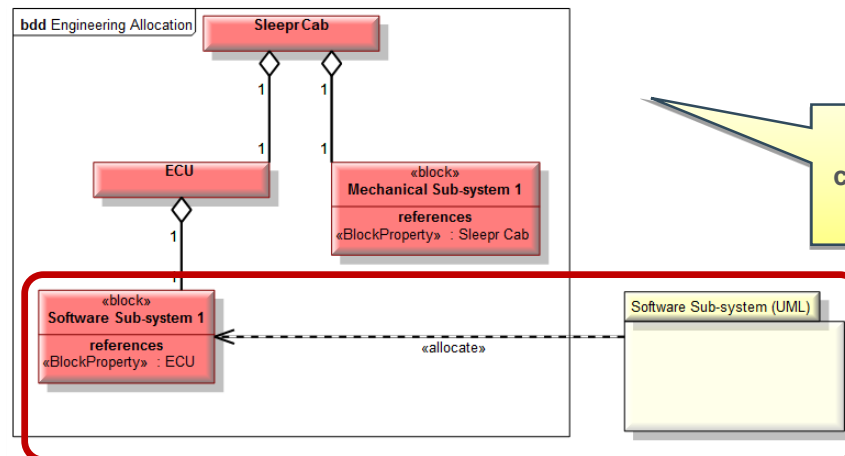
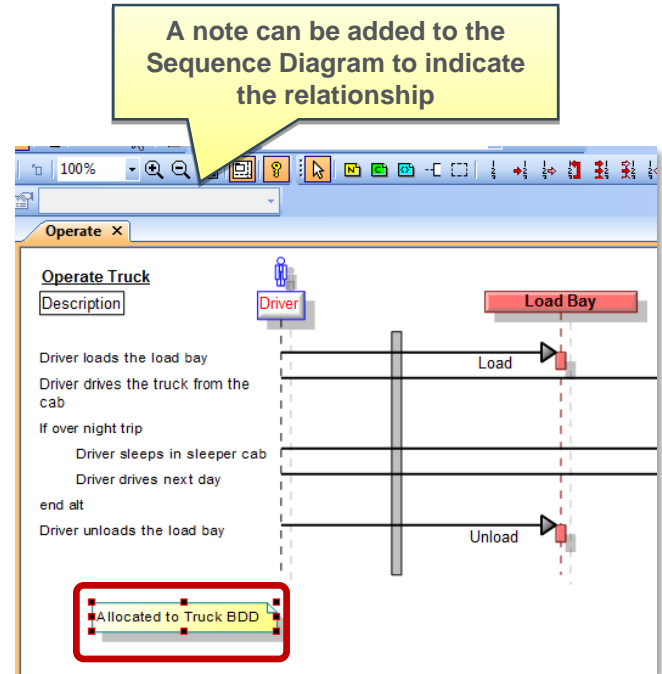
Select Allocated To as the relationship and click OK



## ► Link Requirements to System Architecture



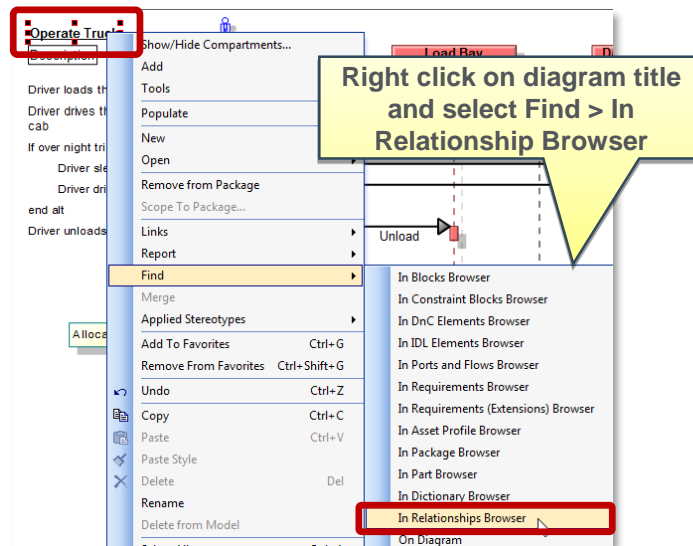
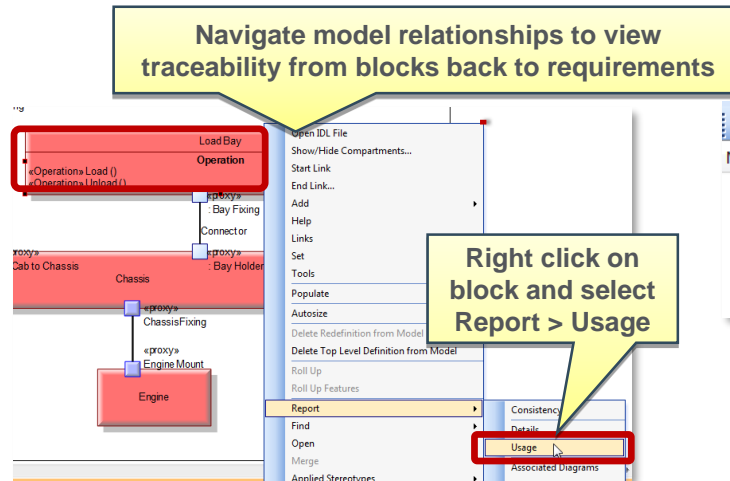
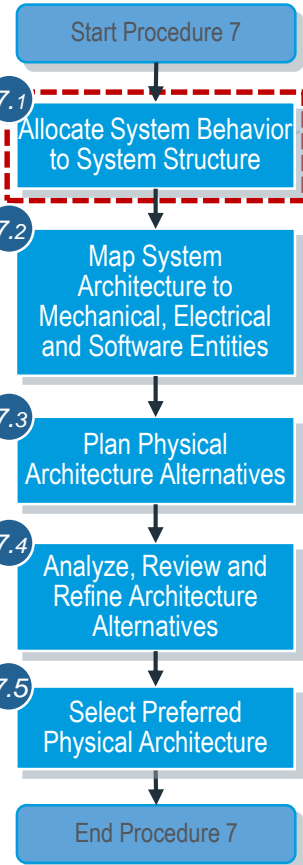
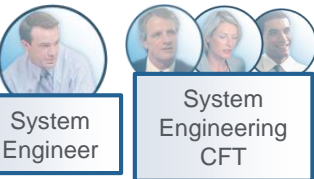
This allocation can be seen in the Properties of the linked items



Software designs can also be allocated to system blocks



## ► Link Requirements to System Architecture

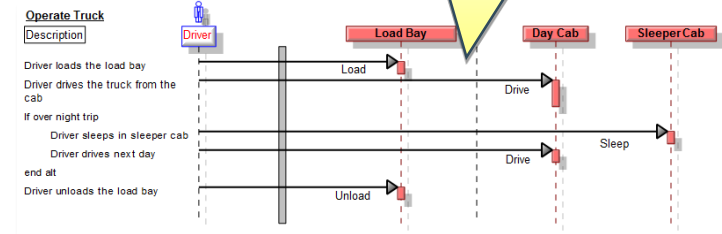


Review results to identify sequence diagram

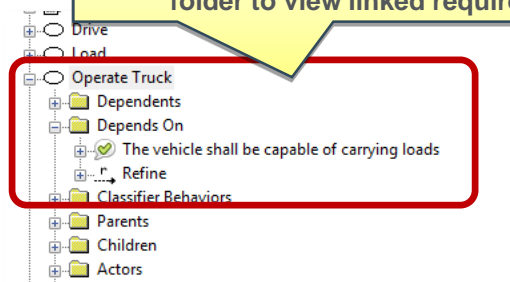
Items and Diagrams which use Load Bay.

Name	Type
Truck (SysML)::Blocks::Truck.IBD (Truck)	InternalBlockDiagram
Truck (SysML)::Blocks::Truck Product 1.[Block] Truck Product 1 [1]	InternalBlockDiagram
Truck (SysML)::Blocks::Truck Product 2.[Block] Truck Product 2 [1]	InternalBlockDiagram
Truck (SysML)::Use Cases:Operate Truck:Operate	Object Sequence Diagram
Associates [Truck - Load Bay]	Association
Load Bay [Truck (SysML)::Blocks::Load Bay - Truck (SysML)::Bloc...	BlockProperty

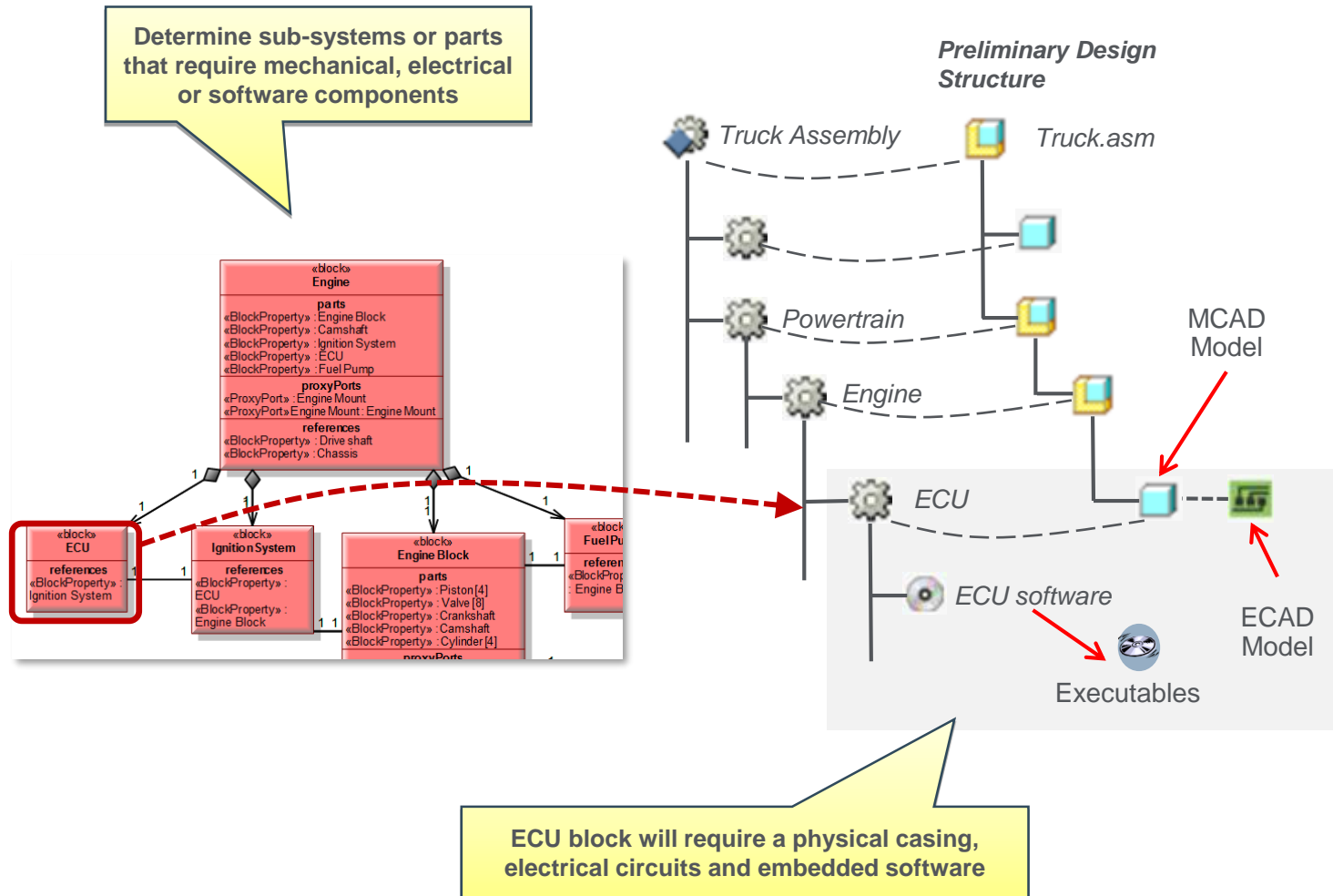
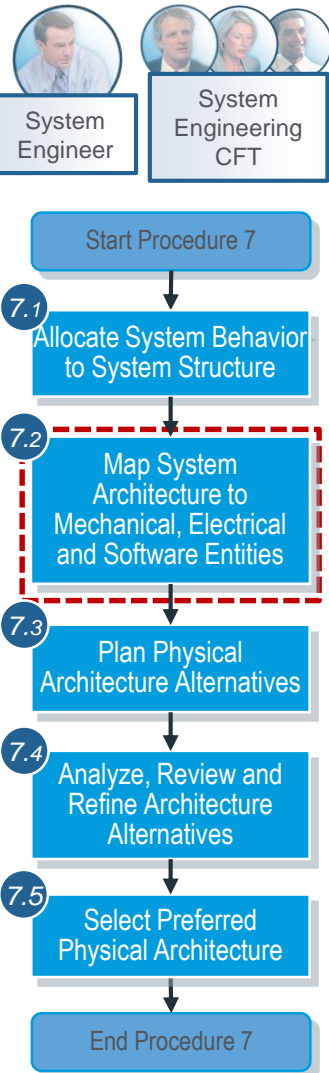
Double click to open sequence diagram



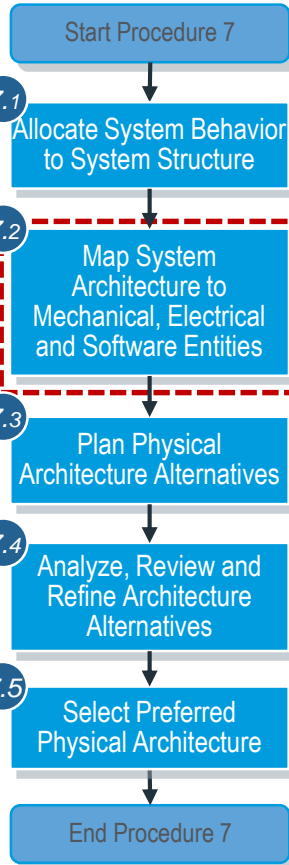
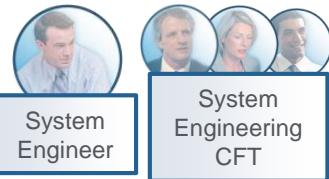
Use case will be highlighted in Relationship browser. Expand and select Depends On folder to view linked requirements



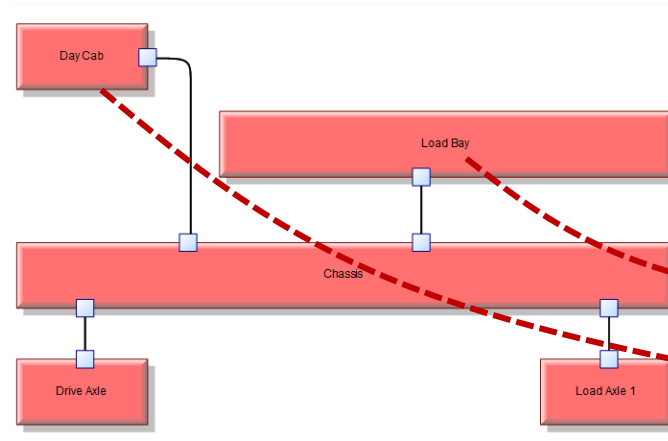
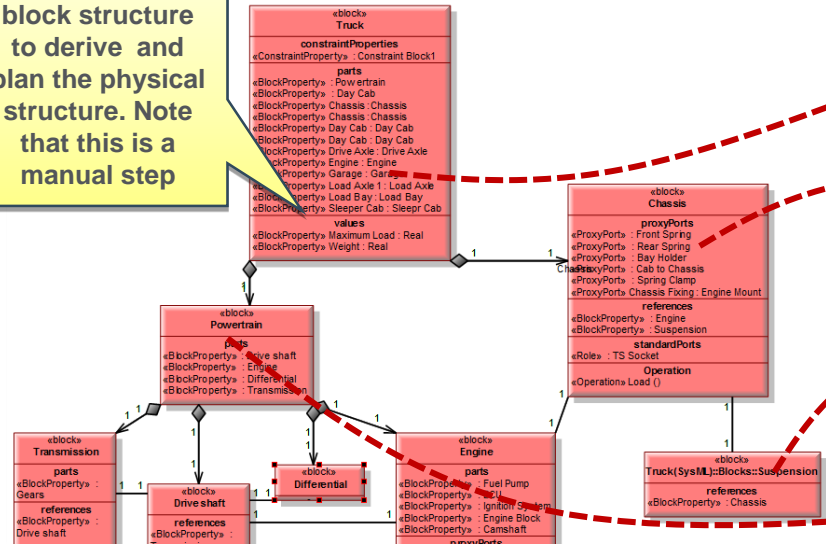
## ► Map System Architecture to Mechanical, Electrical and Software Entities



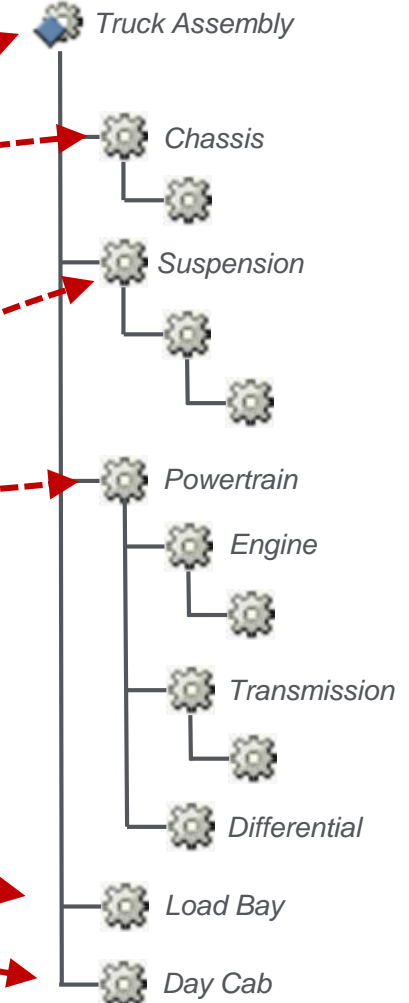
## ► Map System Architecture to Mechanical, Electrical and Software Entities



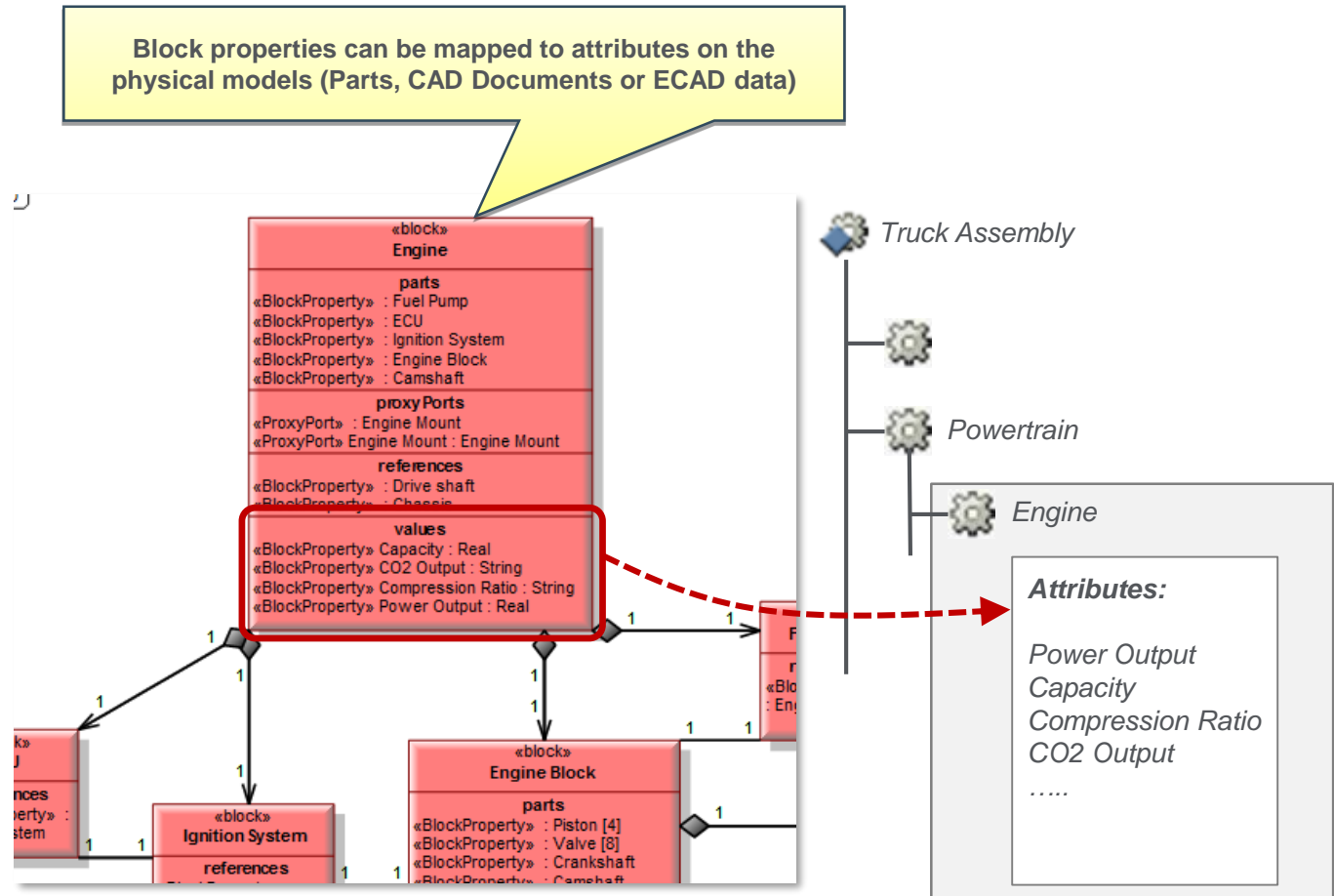
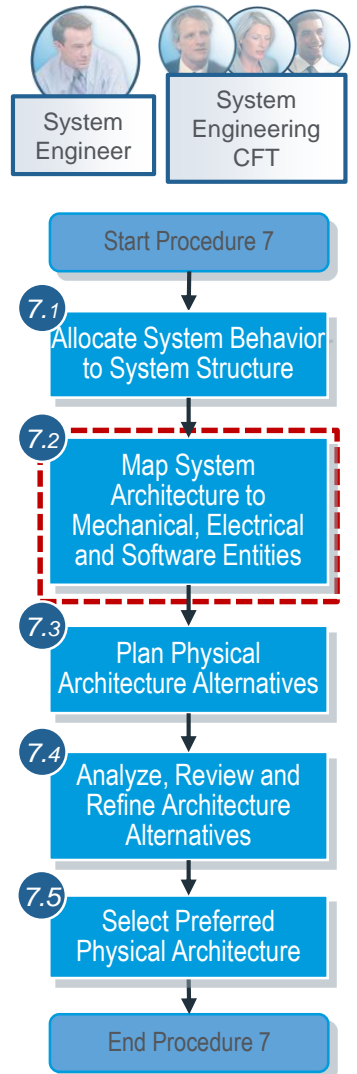
Use the system block structure to derive and plan the physical structure. Note that this is a manual step



Preliminary PDMLink Product Structure

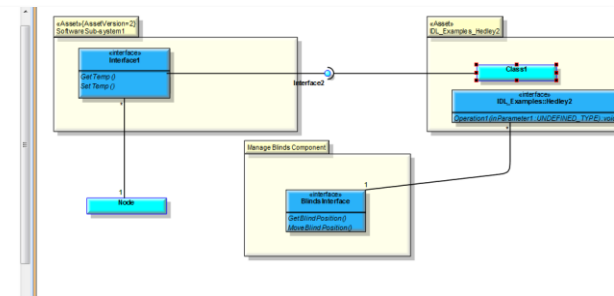
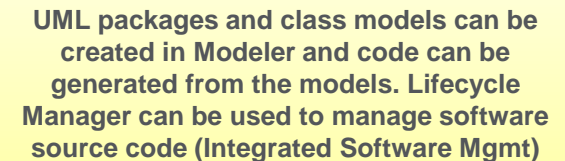
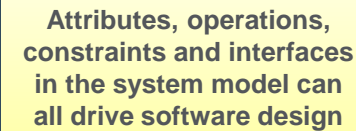


## ► Map System Architecture to Mechanical, Electrical and Software Entities

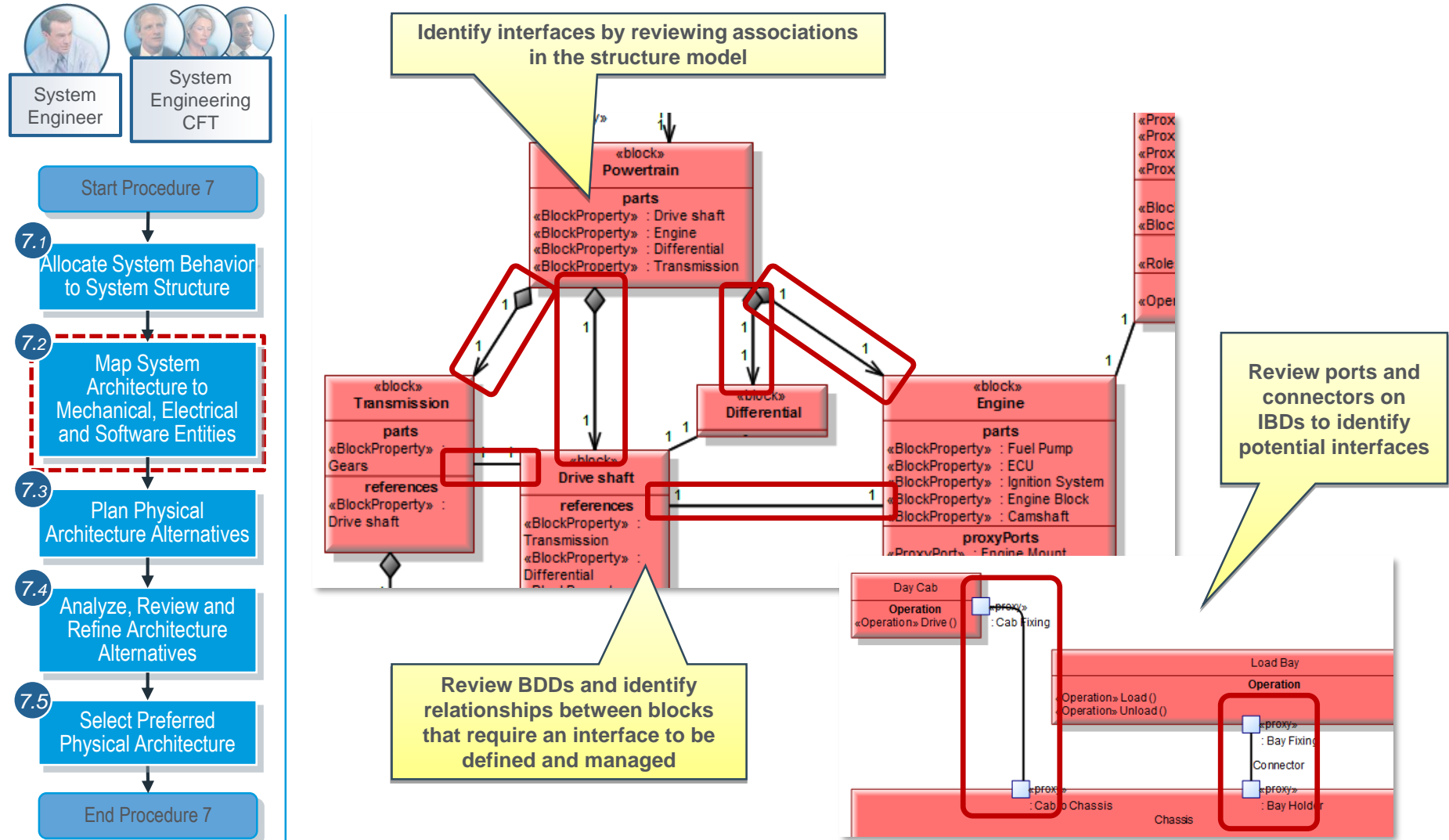


System Engineer

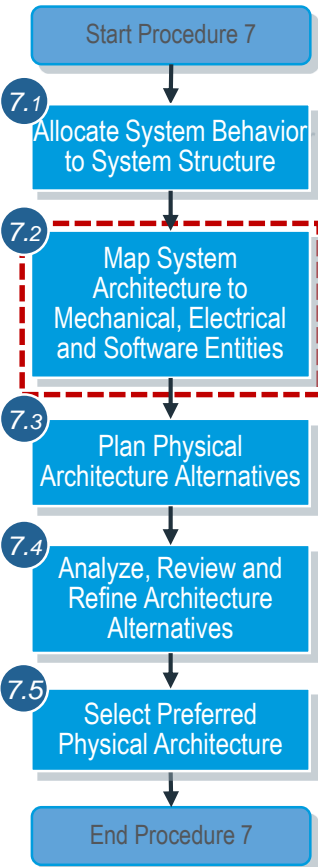
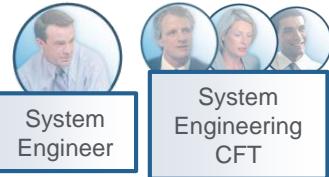
System Engineering CFT



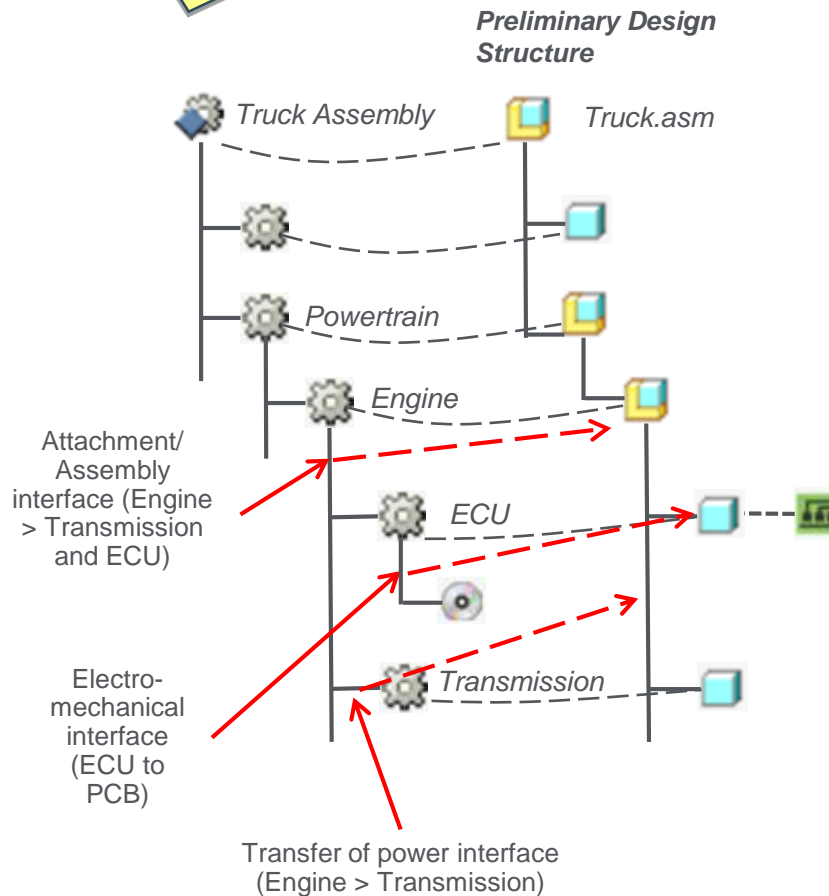
## ► Map System Architecture to Mechanical, Electrical and Software Entities




## ► Map System Architecture to Mechanical, Electrical and Software Entities



Plan the interfaces needed in the physical architecture and related design models

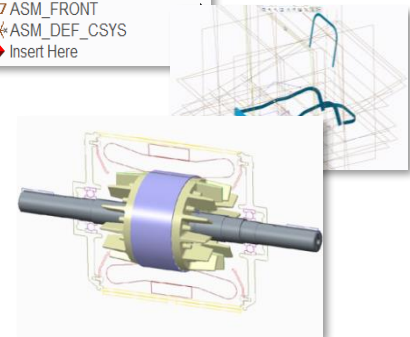


INTERFACE SPECIFICATION				
				
Interface Number: GC901-200-200				
Description: TRANSMISSION TO ENGINE				
Revision History				
Rev	Date	Change #	Description of Change	
A				
Interface Detailed Description				
Module Number	Module Name	Module Specification Number		
GC201-0000-001	Go-cart Transmission	GC201_SPEC		
GC202-0000-005	Go-cart Engine	GC202_SPEC		
Interface Description				
Attachment 4 M6 bolts to secure coupling bracket				
Spatial Interdependent space class (see interface part)				
Comments				
Co				
Define and agree interface				

Define and agree interface specifications and assign ownership

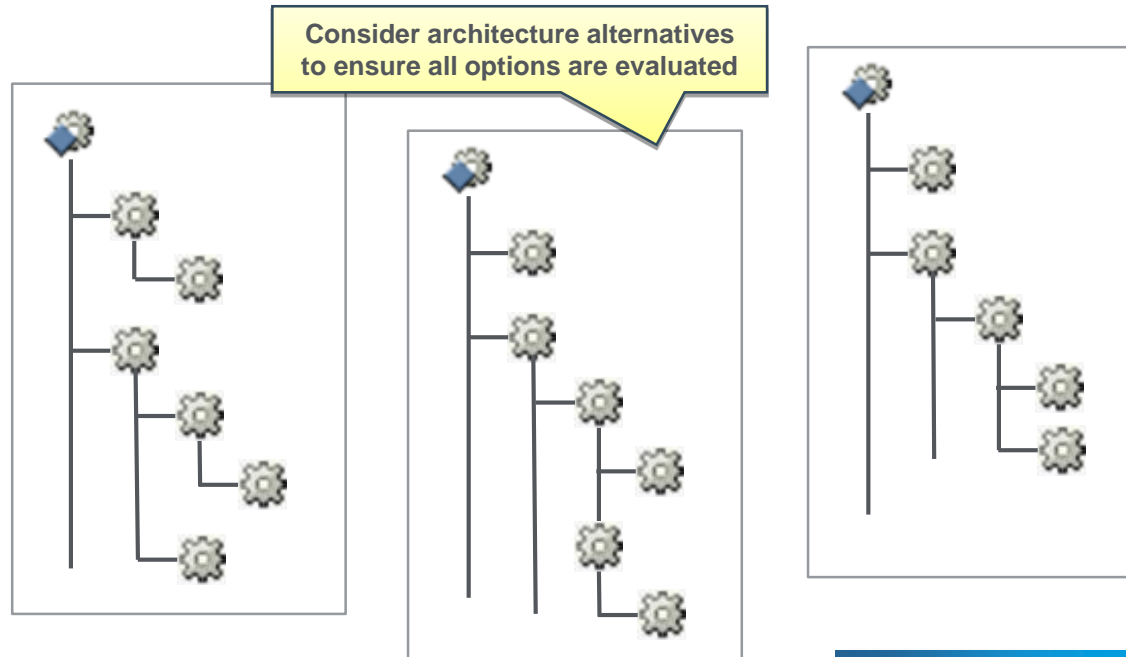
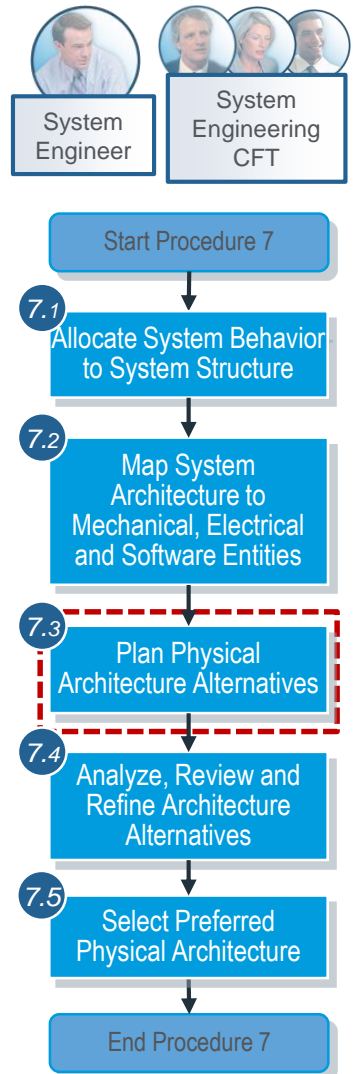
Plan the realization of interfaces in design models (Refer to Interface Definition and Mgmt best practice)

109\_1001\_PTC.ASM  
109-1001K01\_P16\_SKEL.PRT  
ASM\_RIGHT  
ASM\_TOP  
ASM\_FRONT  
ASM\_DEF\_CSYS  
Insert Here

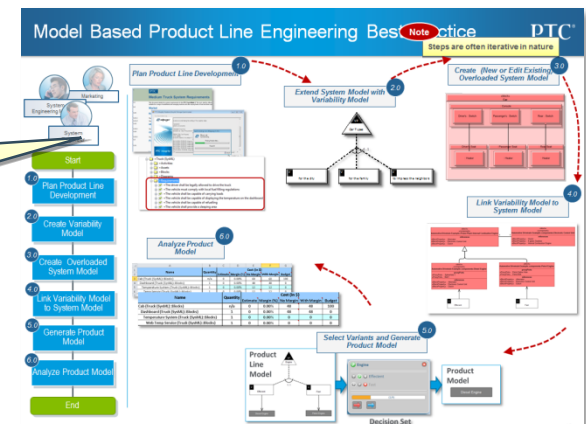




## ► Plan Physical Architecture Alternatives

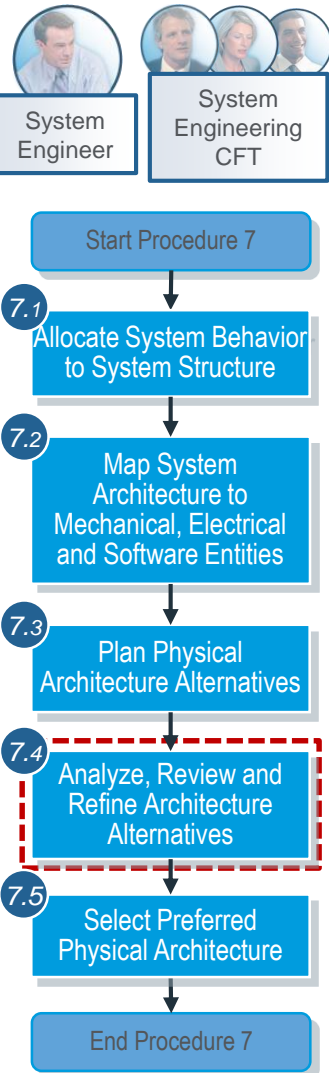


Also consider product options and variants – refer to the Model Based Product Line Engineering best practice for more information

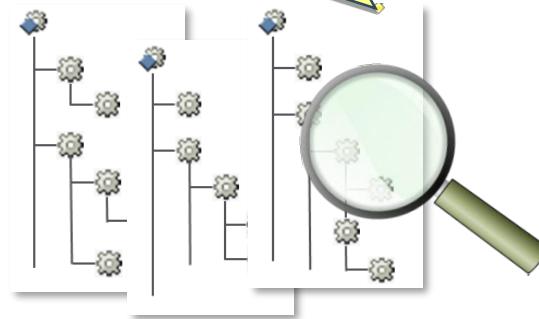




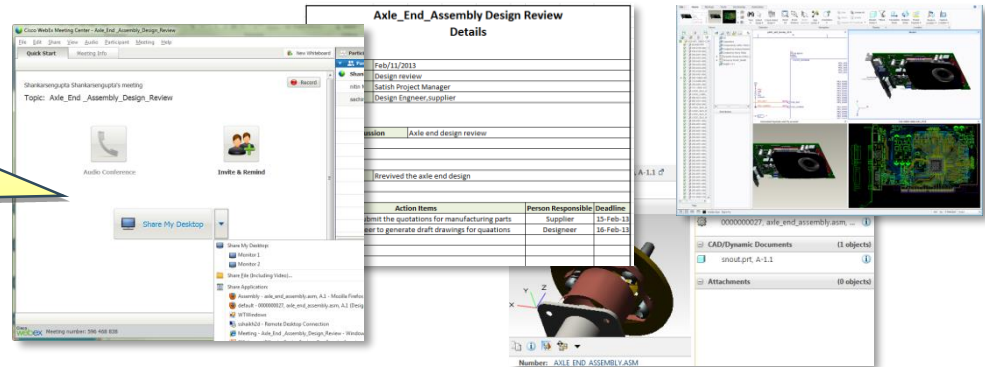
## ► Analyze, Review and Refine Architecture Alternatives



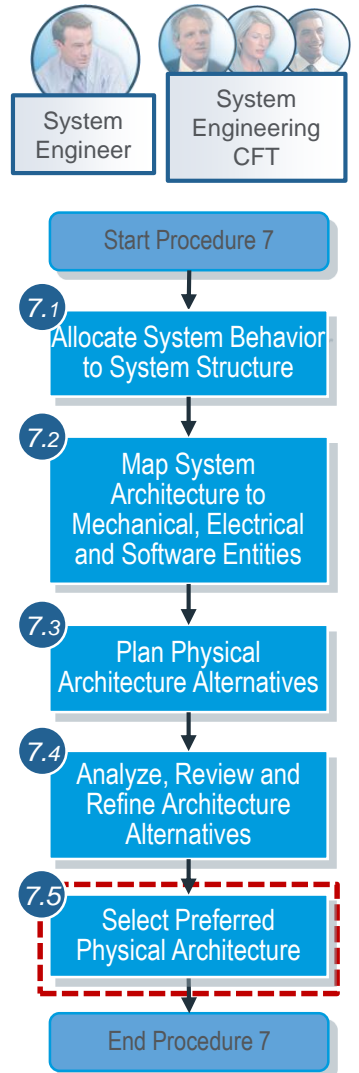
Review and refine architecture alternatives with CFT



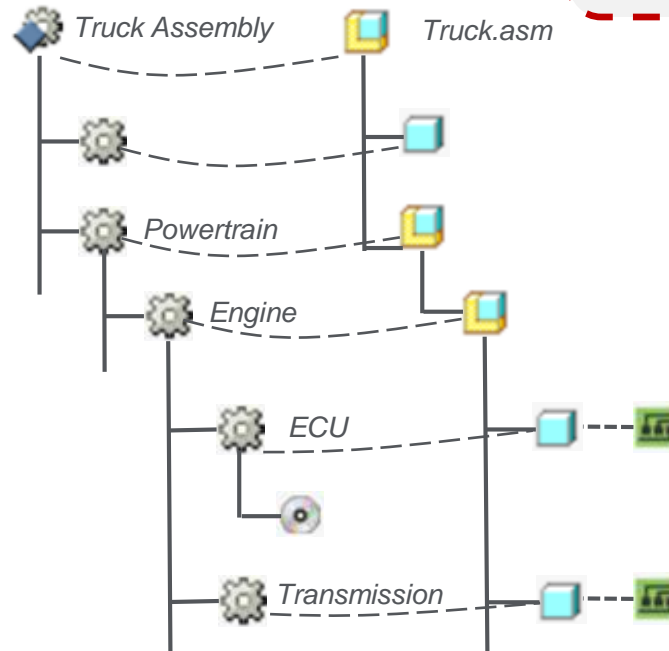
Refer to the Efficient Design Review best practice for more information



## ► Select Preferred Physical Architecture



CFT will then select the optimum physical architecture to best meet the system requirements



PTC<sup>®</sup> PRODUCT & SERVICE  
ADVANTAGE

## Document Properties

File Name	Status
MBSE_BestPractice_Storyboard.pptx	Accepted

## Change History

Date	Name of Author	Version	Description
05/02/2015	Patrick Ollerton	1.0	