# MODEL BASED SYSTEMS ENGINEERING

# BEST PRACTICE STORYBOARD

Patrick Ollerton

# AGENDA

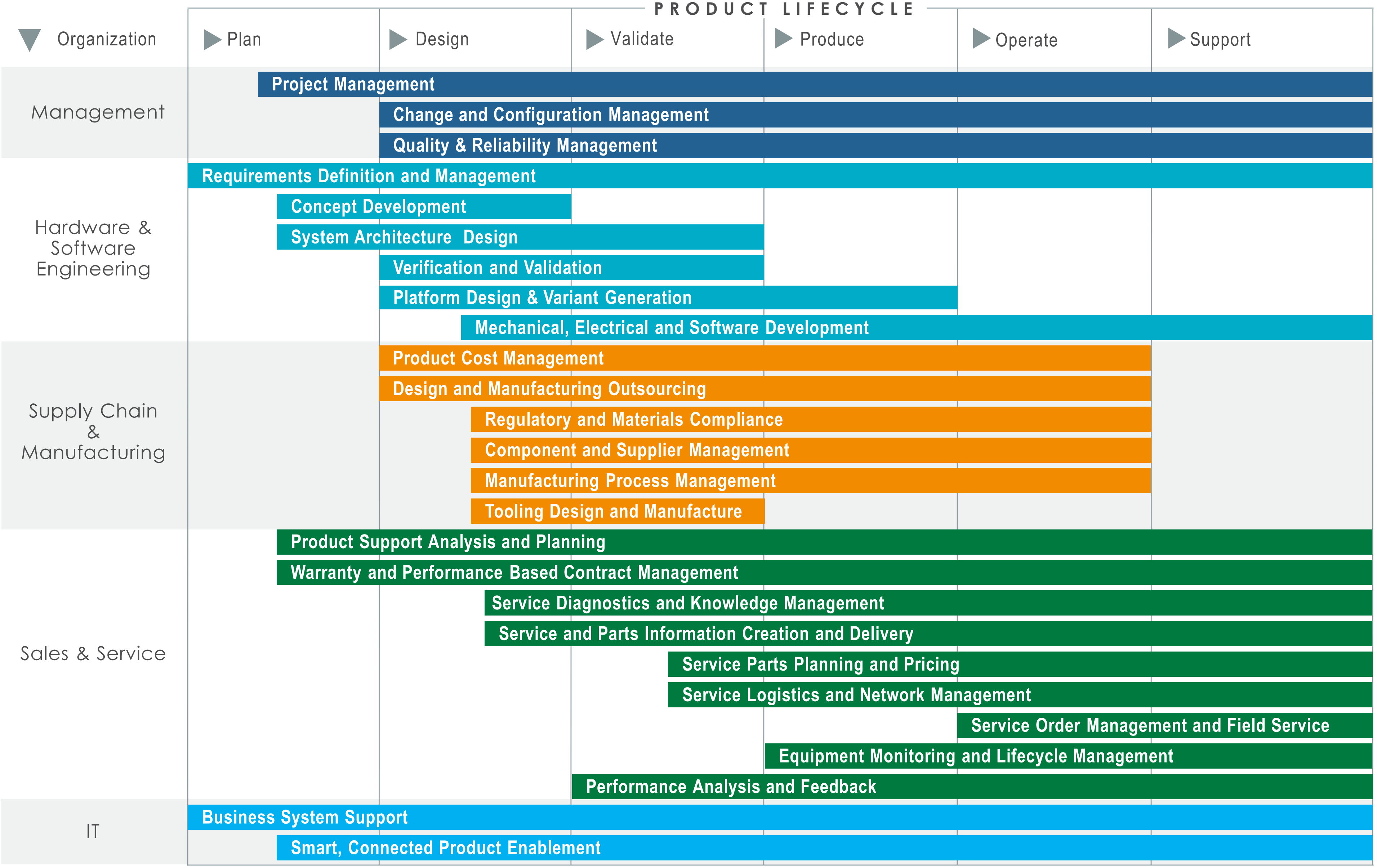
Note

It is recommended that the reader has some knowledge of SysML

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

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

# PTC PROCESS LANDSCAPE



PRODUCT LIFECYCLE

Organization

Plan

Design

Validate

Produce

Operate

Support

Management

Project Management

Change and Configuration Management

Quality & Reliability Management

Hardware & Software Engineering

Requirements Definition and Management

Concept Development

System Architecture  Design

Verification and Validation

Platform Design & Variant Generation

Mechanical, Electrical and Software Development

Supply Chain & Manufacturing

Product Cost Management

Design and Manufacturing Outsourcing

Regulatory and Materials Compliance

Component and Supplier Management

Manufacturing Process Management

Tooling Design and Manufacture

Sales & Service

Product Support Analysis and Planning

Warranty and Performance Based Contract Management

Service Diagnostics and Knowledge Management

Service and Parts Information Creation and Delivery

Service Parts Planning and Pricing

Service Logistics and Network Management

Service Order Management and Field Service

Equipment Monitoring and Lifecycle Management

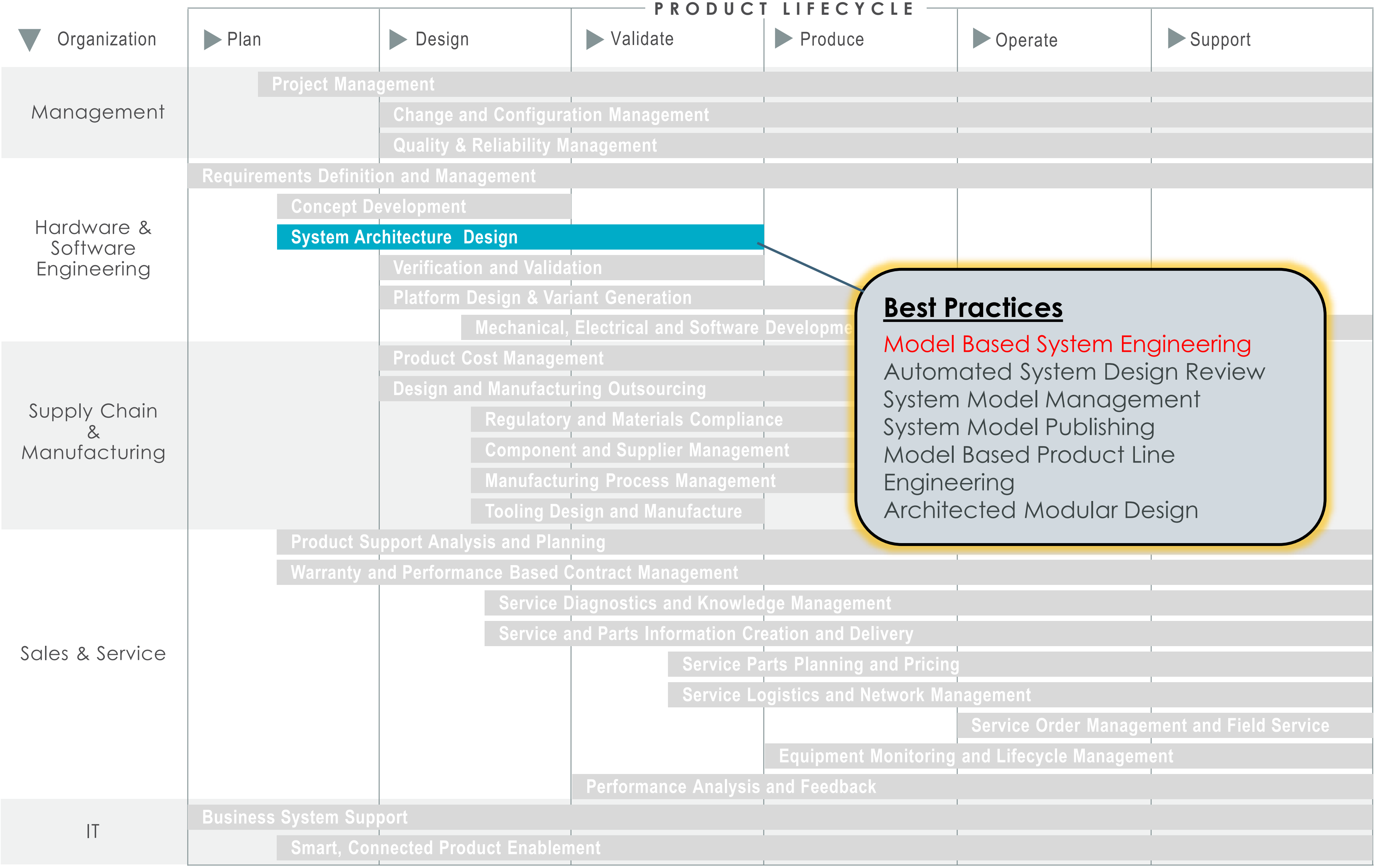
Performance Analysis and Feedback

IT

Business System Support

Smart, Connected Product Enablement

# PTC PROCESS LANDSCAPE



PRODUCT LIFECYCLE

Organization | Plan | Design | Validate | Produce | Operate | Support

**Management**
- Project Management
- Change and Configuration Management
- Quality & Reliability Management

**Hardware & Software Engineering**
- Requirements Definition and Management
- Concept Development
- System Architecture Design
- Verification and Validation
- Platform Design & Variant Generation
- Mechanical, Electrical and Software Developme

**Supply Chain & Manufacturing**
- Product Cost Management
- Design and Manufacturing Outsourcing
- Regulatory and Materials Compliance
- Component and Supplier Management
- Manufacturing Process Management
- Tooling Design and Manufacture

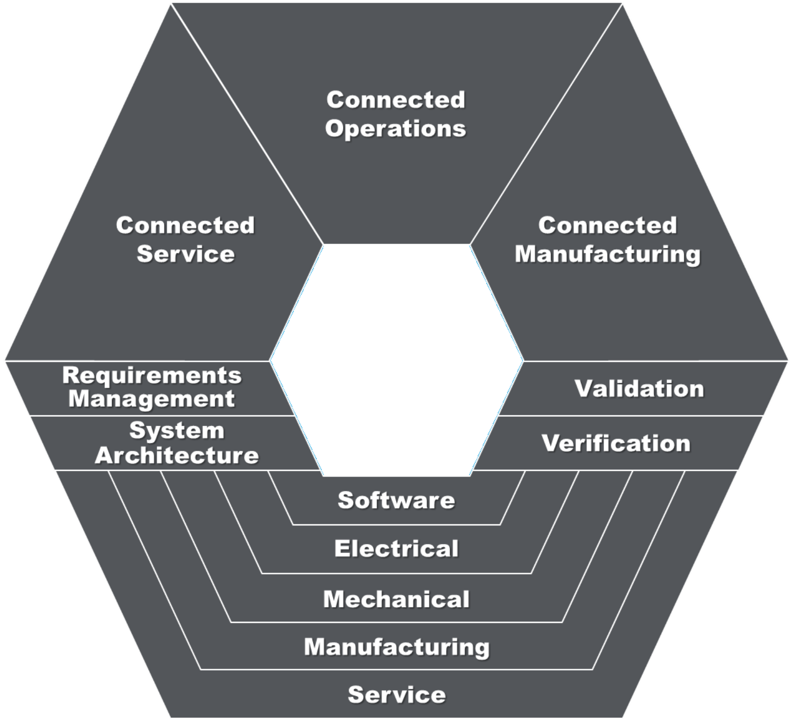
**Sales & Service**
- Product Support Analysis and Planning
- Warranty and Performance Based Contract Management
- Service Diagnostics and Knowledge Management
- Service and Parts Information Creation and Delivery
- Service Parts Planning and Pricing
- Service Logistics and Network Management
- Service Order Management and Field Service
- Equipment Monitoring and Lifecycle Management
- Performance Analysis and Feedback

**IT**
- Business System Support
- Smart, Connected Product Enablement

**Best Practices**

Model Based System Engineering
Automated System Design Review
System Model Management
System Model Publishing
Model Based Product Line Engineering
Architected Modular Design

# PTC VISION HIGHLIGHTS



## Closed-Loop Lifecycle Management

**Closes the loop** with smart, connected products operation to understand how product, users, and the environment truly interact

**Provides visibility** into manufacturing and service feedback to inform next generation development

**Validates requirements** cradle-to-grave to ensure customer needs and quality expectations are met

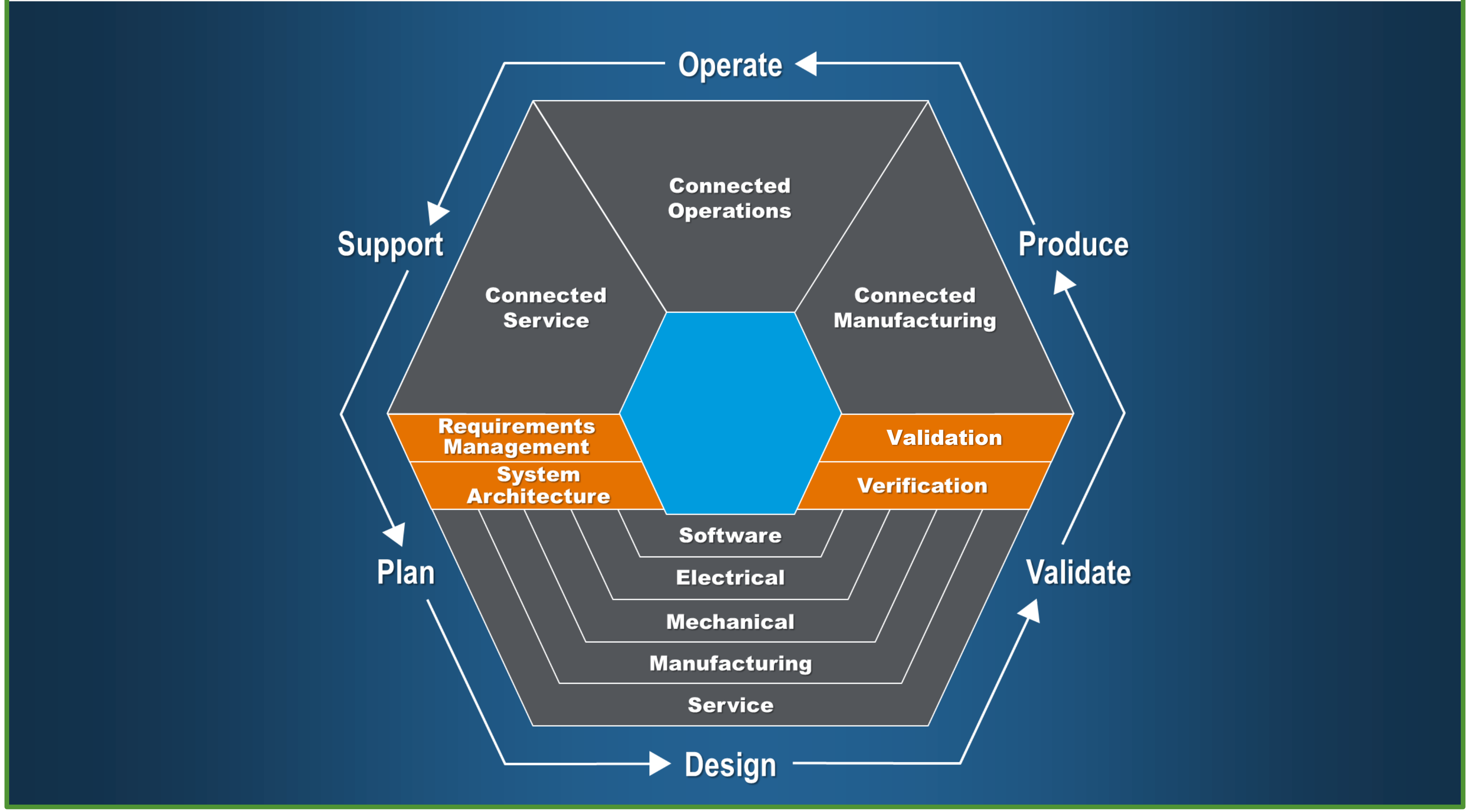
**Fully integrates systems engineering** with product engineering in a single source of truth

**Manages collaboration and change** within and across different disciplines

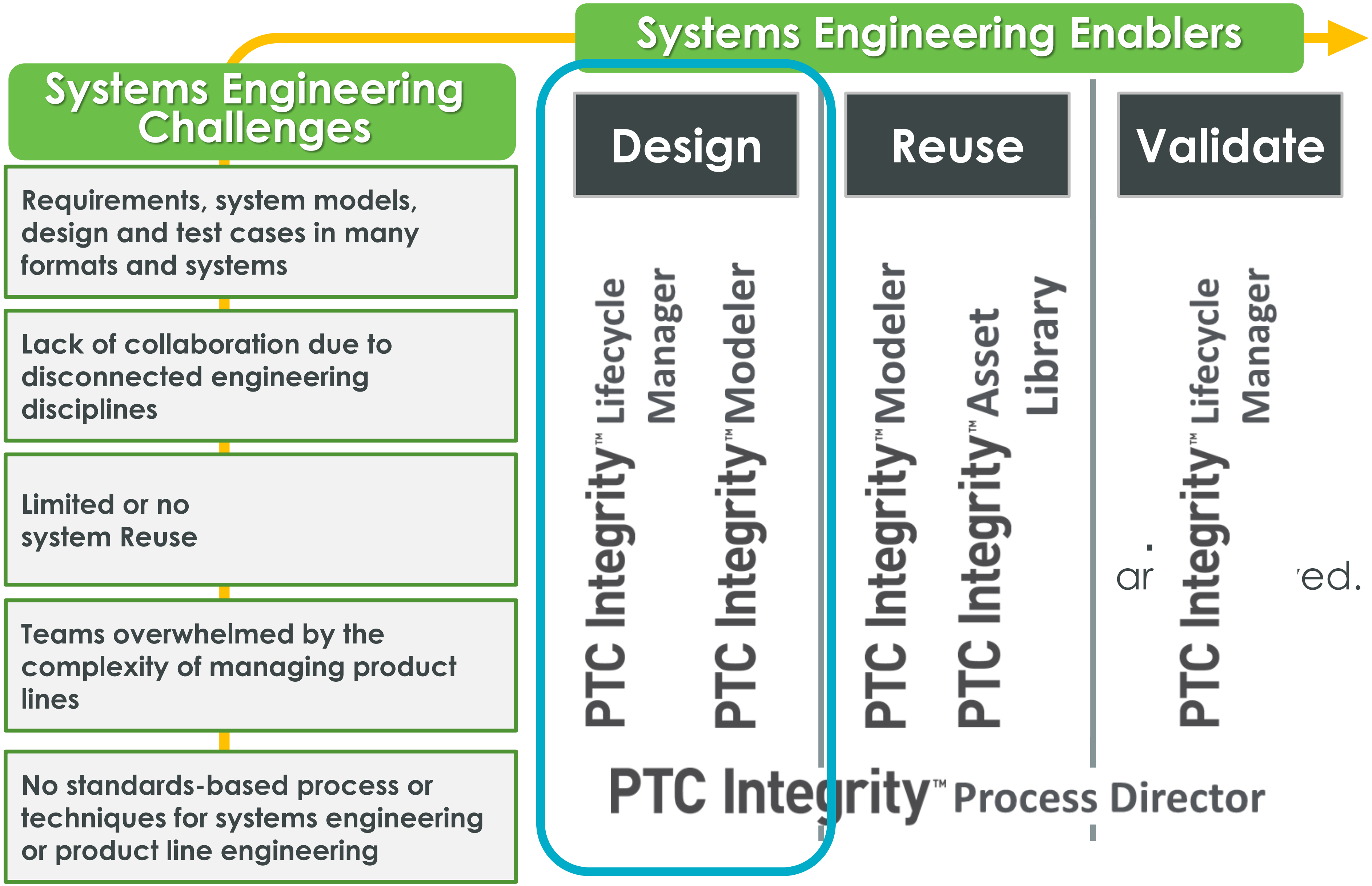
**Enables holistic system definition** and design, including early manufacturing and service plans

# SYSTEMS ENGINEERING IS…

A **holistic**, **multi-disciplinary** and collaborative approach to designing and maintaining **complex** systems.



# HOW WE DO IT WITH PTC INTEGRITY



## Systems Engineering Challenges

- Requirements, system models, design and test cases in many formats and systems
- Lack of collaboration due to disconnected engineering disciplines
- Limited or no system Reuse
- Teams overwhelmed by the complexity of managing product lines
- No standards-based process or techniques for systems engineering or product line engineering

## Systems Engineering Enablers

**Design**

- PTC Integrity™ Lifecycle Manager
- PTC Integrity™ Modeler

**Reuse**

- PTC Integrity™ Modeler
- PTC Integrity™ Asset Library

**Validate**

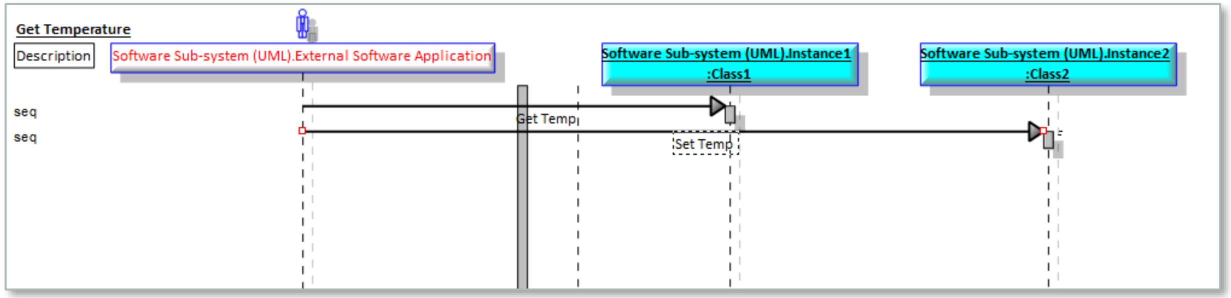
- PTC Integrity™ Lifecycle Manager

ar · 'ed.

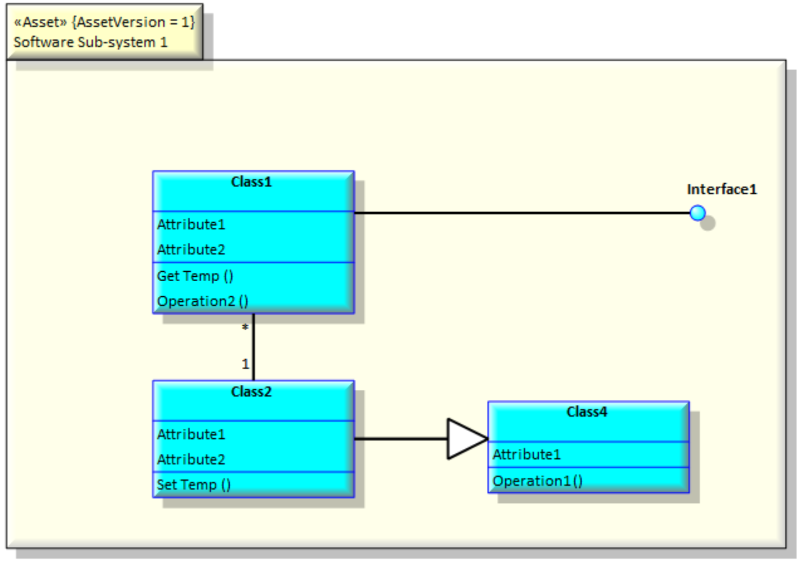
PTC Integrity™ Process Director

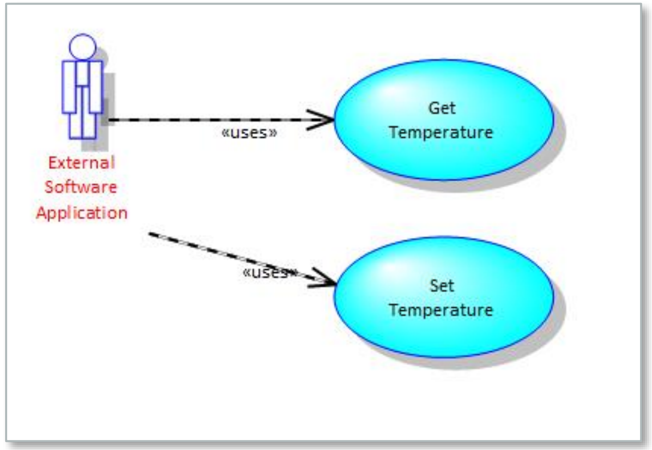
# WHAT IS UML?

- The **Unified Modeling Language** (**UML**) is an OMG Standard, general-purpose modeling language in the field of software engineering, which is designed to provide a standard way to visualize the design of a system.


Get Temperature
Description
Software Sub-system (UML).External Software Application
Software Sub-system (UML).Instance1
:Class1
Software Sub-system (UML).Instance2
:Class2
seq
seq
Get Temp
Set Temp

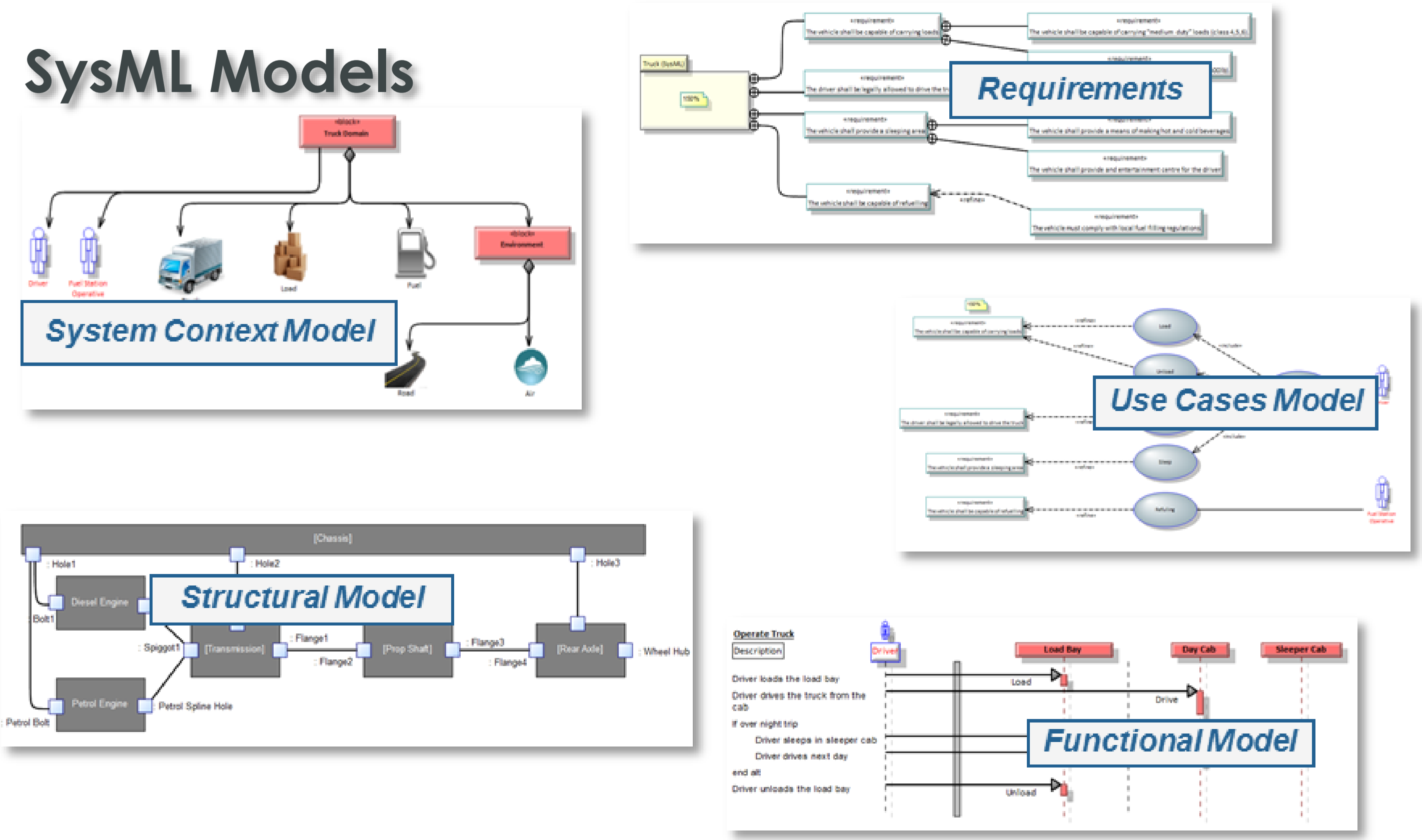
**UML Models**





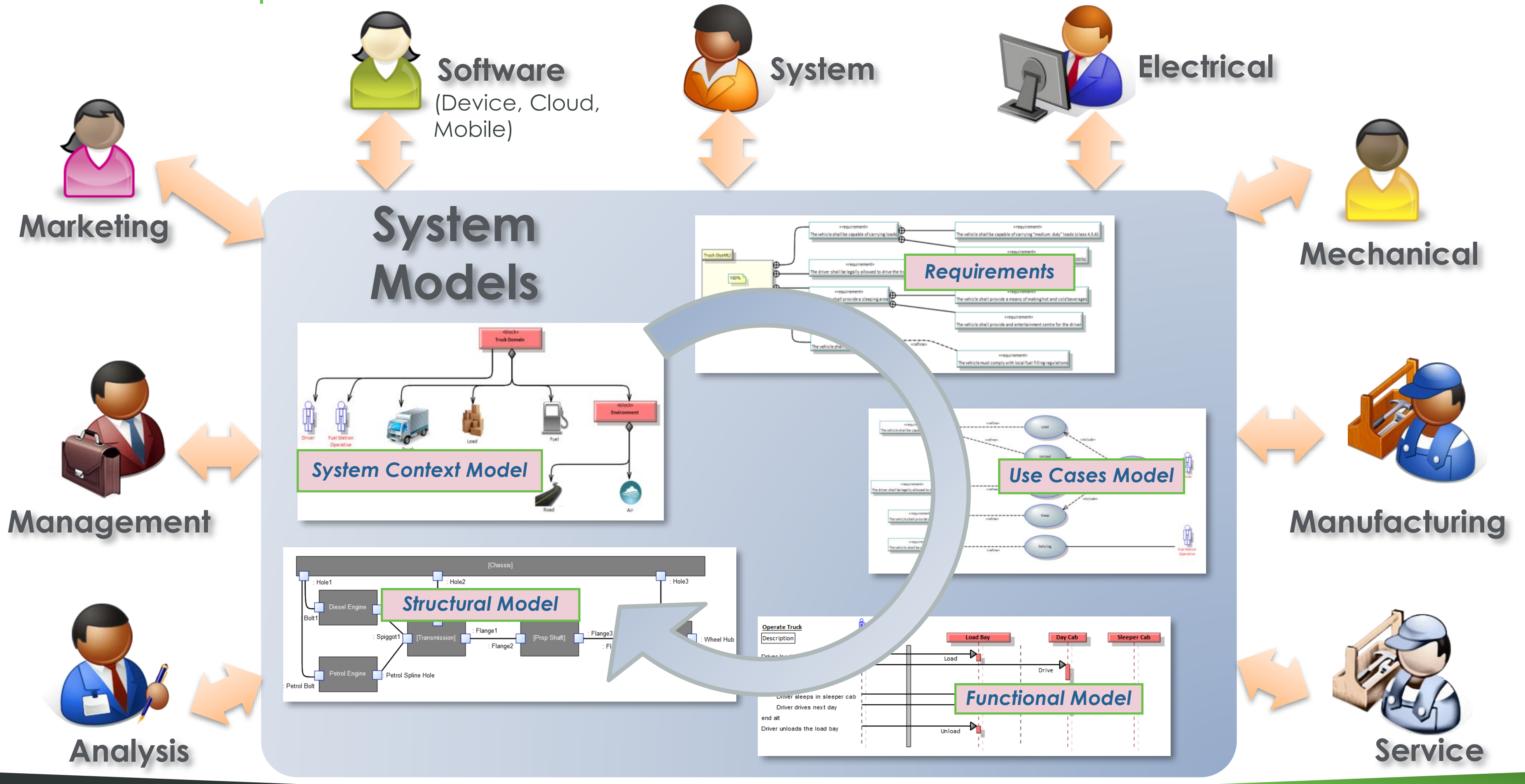
# WHAT IS SYSML?

- The **Systems Modeling Language (SysML)** is an OMG standard (extension of **UML**), general-purpose modeling language for systems engineering. It enables the specification, analysis, design, verification and validation of a broad range of systems and systems-of-systems.



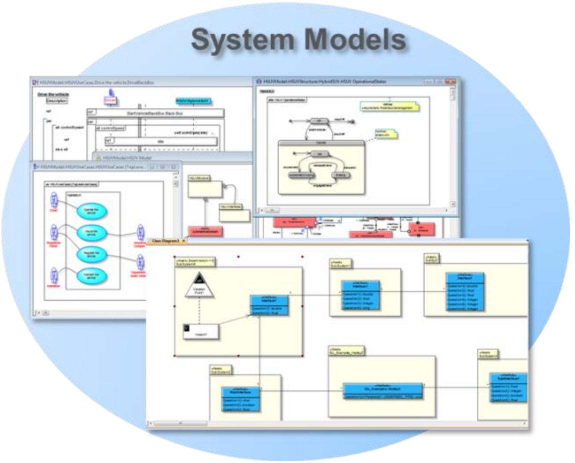
# WHAT IS MODEL BASED SYSTEM ENGINEERING?

- **MBSE** is a model-based approach to Systems Engineering, typically applying the SysML modeling language to deal with system complexity and enabling unambiguous communication amongst interested parties



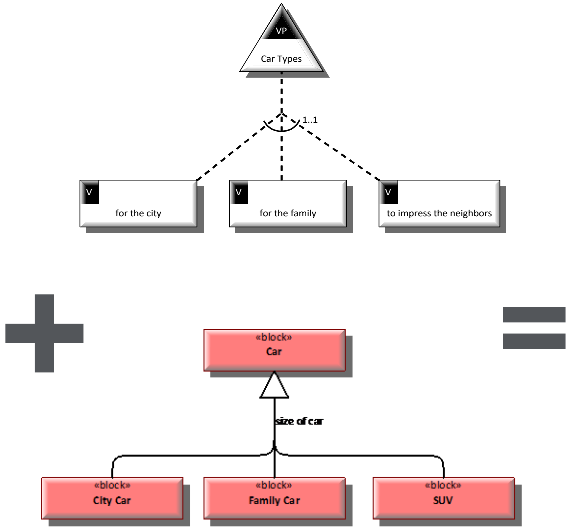
# WHAT IS MODEL BASED PRODUCT LINE ENGINEERING?

- **MBPLE** is a model-based approach to the analysis and design of product families (or product lines), to capitalize on the commonality & variation between the products (systems, software, etc.) in these families



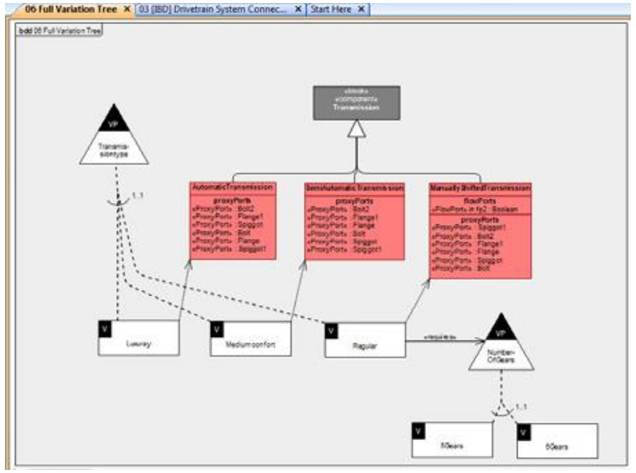
+

## Variability



=

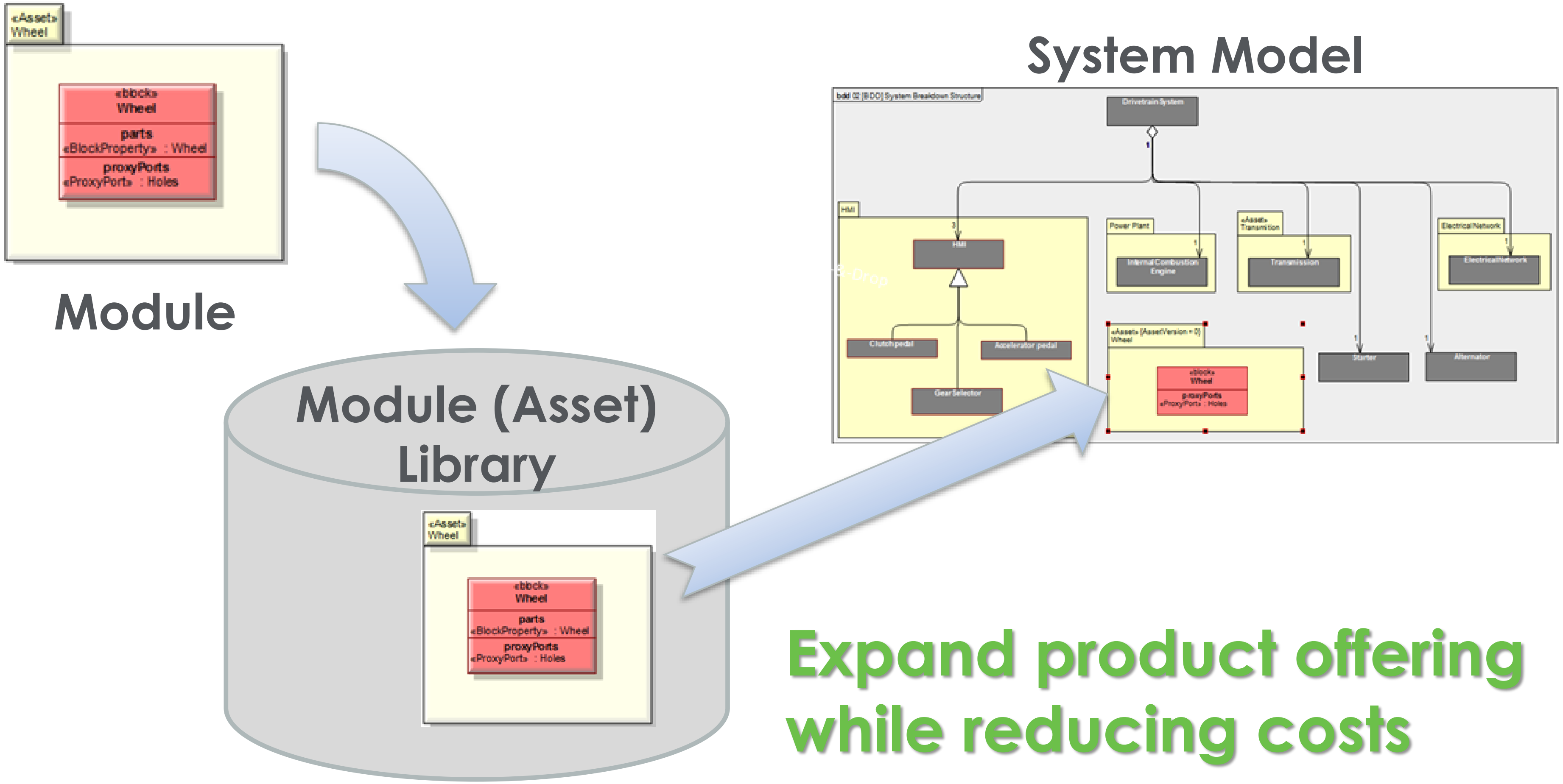
## Product Line System Model



**Up front design for variability**

# WHAT IS MODULAR DESIGN?

- **Modular Design** is an approach which segments the design of whole systems into linked, manageable and reusable sub-system designs



Module

Module (Asset) Library

System Model

**Expand product offering while reducing costs**

# SYSTEMS ENGINEERING

## Practice Groups

### *Systems Engineering*

### Requirements Engineering

- Collaborative requirements definition
- Accurate requirements analysis and approval
- Comprehensive requirements traceability
- Standardized requirements and test change management

### Test Management

- Comprehensive test traceability and coverage
- Collaborative test definition and authoring
- Managed test execution and results analysis
- Integrated test planning
- Standardized requirements and test change management

### System Design

- **Model Based System Engineering**
- Automated System Design Review
- System Level Simulation
- System Model Management
- System Model Publishing
- Domain Specific Languages Implementation

### Product Line Design

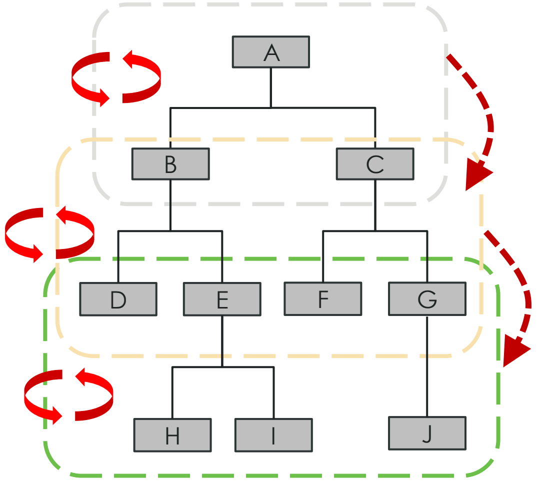
- Model Based Product Line Engineering
- Architected Modular Design
- Product Line Analysis and Validation

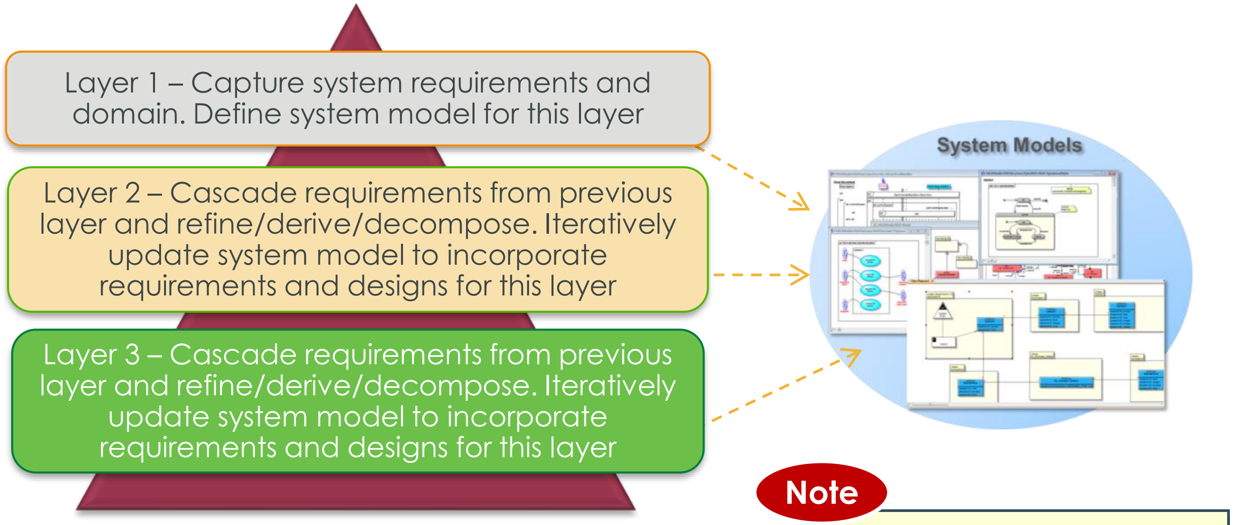
### System Engineering Process Governance

- Standardized System Engineering Best Practices
- Tailored System Engineering Practice Application

# MODEL BASED SYSTEM ENGINEERING APPROACH

- 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




Layer 1 – Capture system requirements and domain. Define system model for this layer

Layer 2 – Cascade requirements from previous layer and refine/derive/decompose. Iteratively update system model to incorporate requirements and designs for this layer

Layer 3 – Cascade requirements from previous layer and refine/derive/decompose. Iteratively update system model to incorporate requirements and designs for this layer

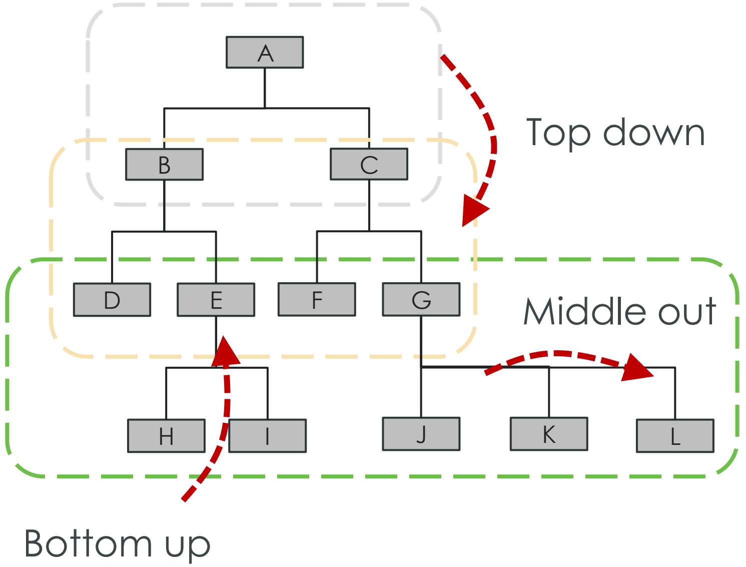
**Note**

**This is an example – the number of layers needed will depend on the system**

# MODEL BASED SYSTEM ENGINEERING APPROACH

## 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



# MODEL BASED SYSTEM ENGINEERING APPROACH

## 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

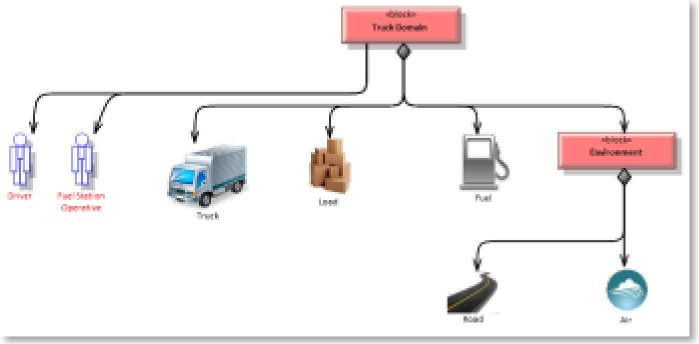
# AGENDA

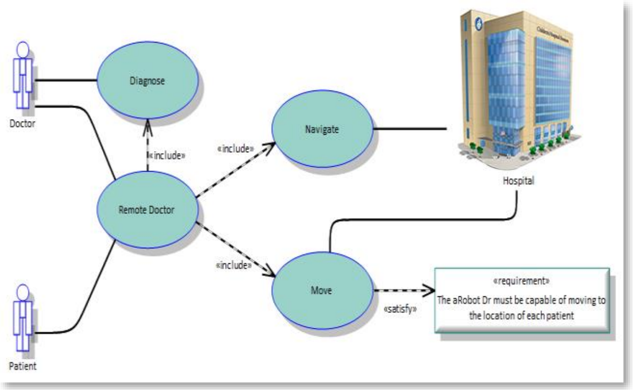
- Introduction
  - Key Concepts
  - Primary Challenges
- PTC Best Practice Storyboard
  - Model Based System Engineering

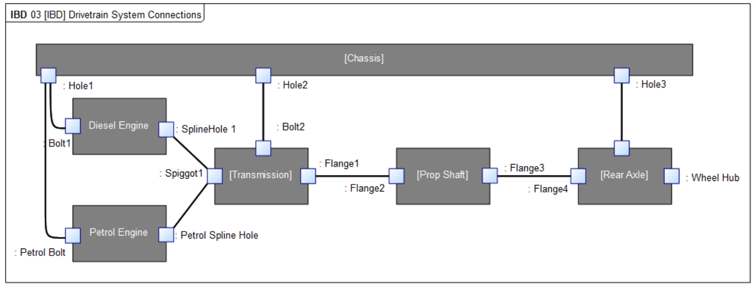
# MODEL BASED SYSTEM ENGINEERING BEST PRACTICE

*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



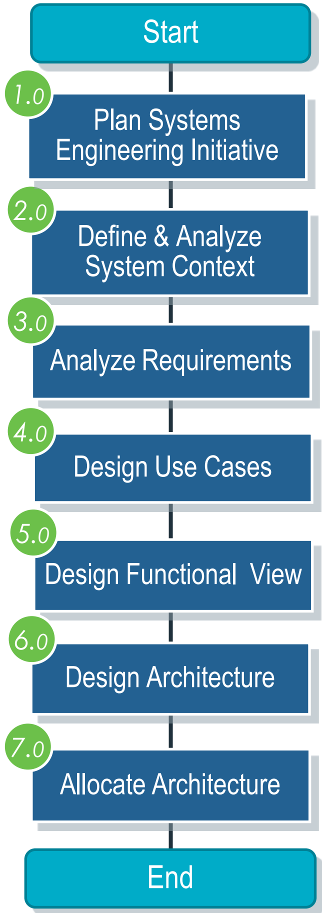




# MODEL BASED SYSTEM ENGINEERING BEST PRACTICE

System Engineer

System Engineering CFT



Start

1.0 Plan Systems Engineering Initiative

2.0 Define & Analyze System Context

3.0 Analyze Requirements

4.0 Design Use Cases

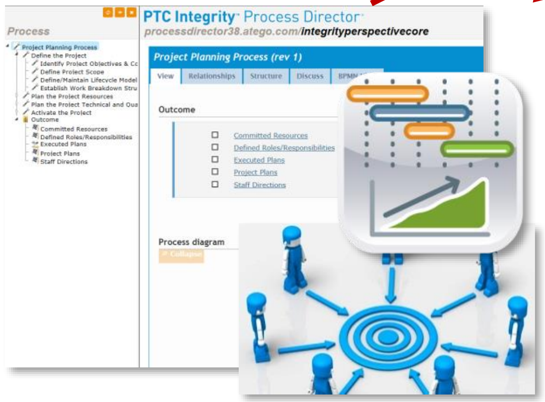
5.0 Design Functional View

6.0 Design Architecture

7.0 Allocate Architecture

End

**1.0 Plan Systems Engineering Initiative**



Process

- Project Planning Process
  - Define the Project
    - Identify Project Objectives & Cc
    - Define Project Scope
    - Define/Maintain Lifecycle Model
    - Establish Work Breakdown Stru
  - Plan the Project Resources
  - Plan the Project Technical and Qua
  - Activate the Project
  - Outcome
    - Committed Resources
    - Defined Roles/Responsibilities
    - Executed Plans
    - Project Plans
    - Staff Directions

PTC Integrity Process Director

processdirector38.atego.com/integrityperspectivecore

Project Planning Process (rev 1)

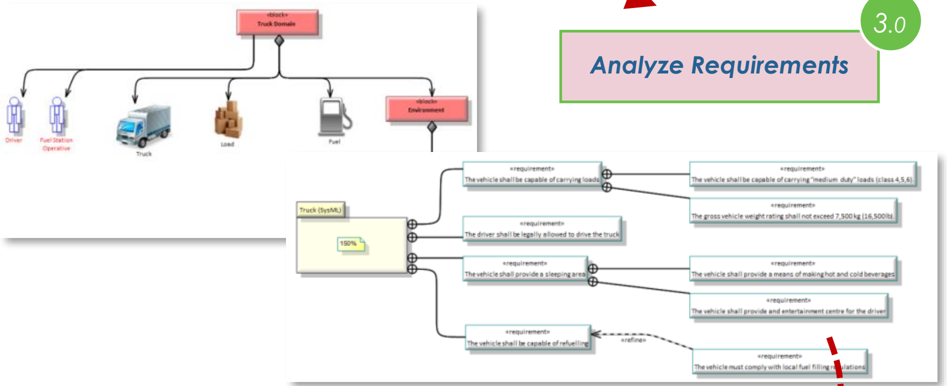
Outcome

- □ Committed Resources
- □ Defined Roles/Responsibilities
- □ Executed Plans
- □ Project Plans
- □ Staff Directions

Process diagram

**2.0 Define & Analyze System Context**

**3.0 Analyze Requirements**



«requirement» The vehicle shall be capable of carrying loads

«requirement» The vehicle shall be capable of carrying "medium duty" loads (class 4,5,6).

«requirement» The gross vehicle weight rating shall not exceed 7,500 kg (16,500lb).

«requirement» The driver shall be legally allowed to drive the truck

«requirement» The vehicle shall provide a sleeping area

«requirement» The vehicle shall provide a means of making hot and cold beverages

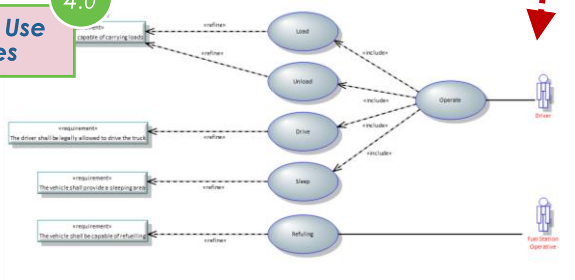
«requirement» The vehicle shall provide and entertainment centre for the driver

«requirement» The vehicle shall be capable of refuelling

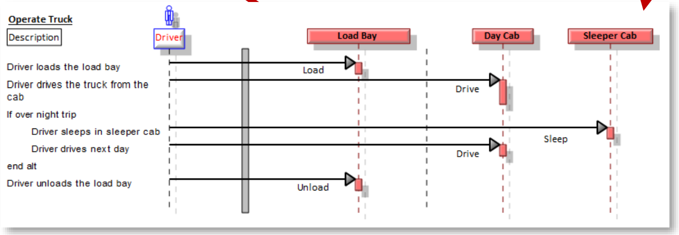
«refine»

«requirement» The vehicle must comply with local fuel filling regulations

**4.0 Design Use Cases**



**5.0 Design Functional View**



**Operate Truck**

Description

Driver loads the load bay

Driver drives the truck from the cab

If over night trip

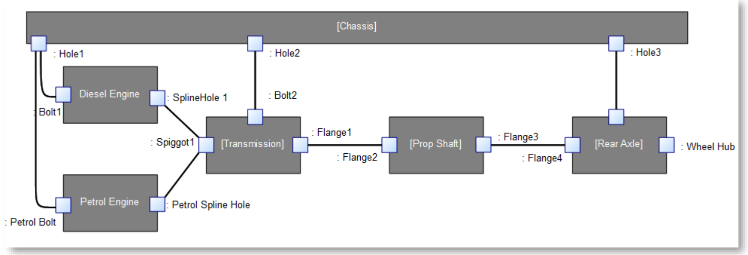
Driver sleeps in sleeper cab

Driver drives next day

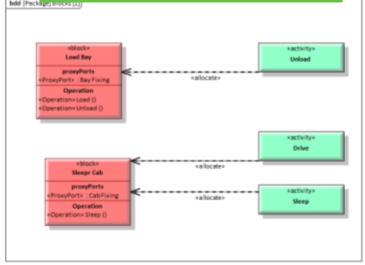
end alt

Driver unloads the load bay

**6.0 Design Architecture**

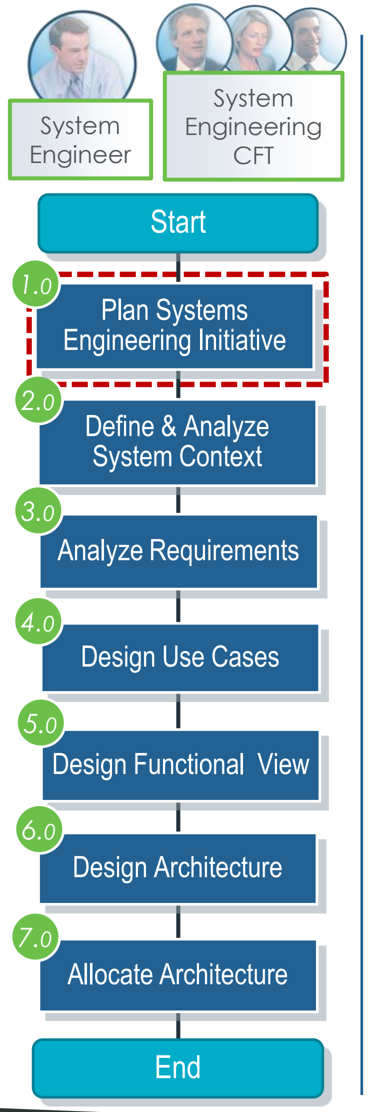


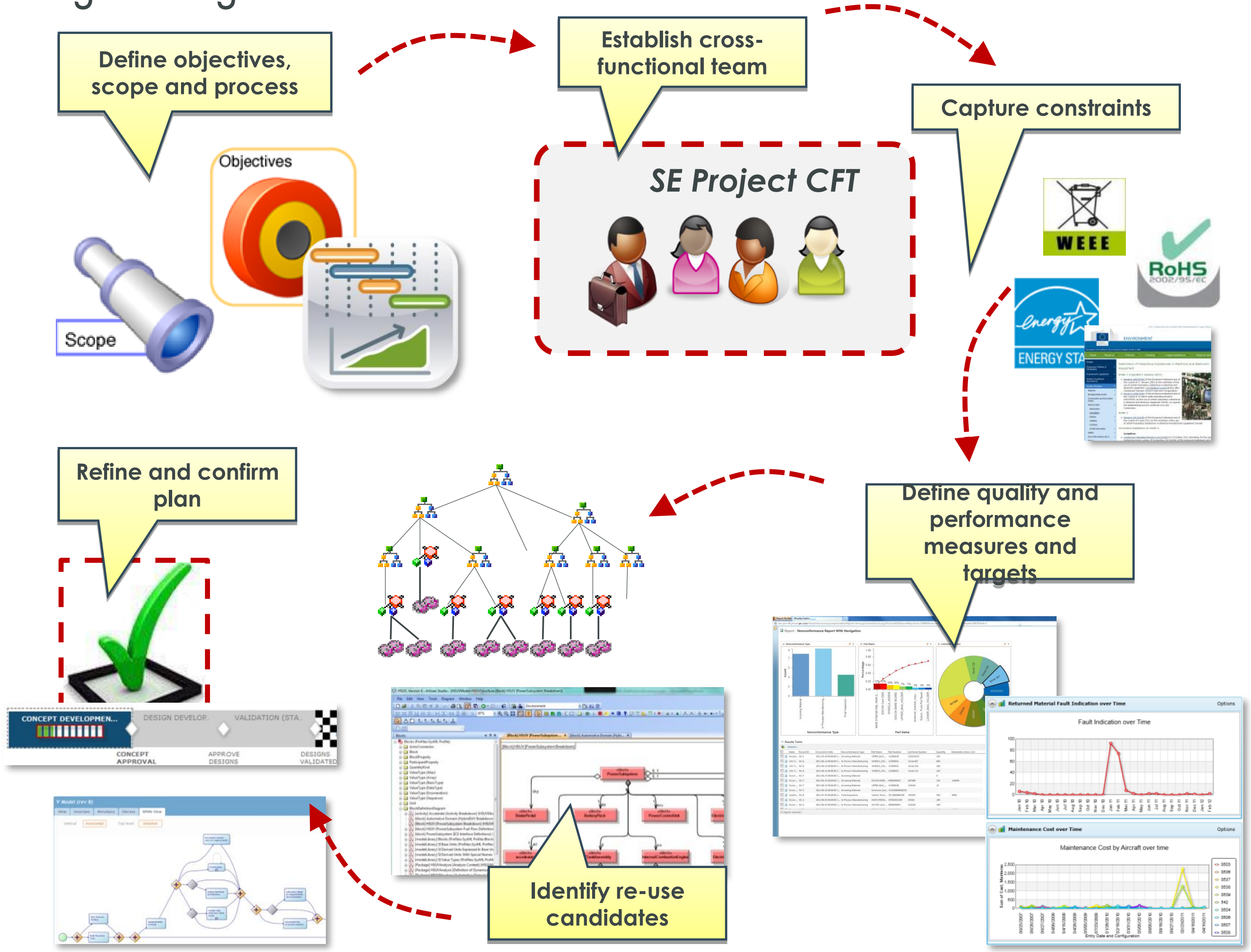
**7.0 Allocate Architecture**



# MODEL BASED SYSTEM ENGINEERING PROCEDURE OVERVIEW

► Plan Systems Engineering Initiative




Define objectives, scope and process

Establish cross-functional team

Capture constraints

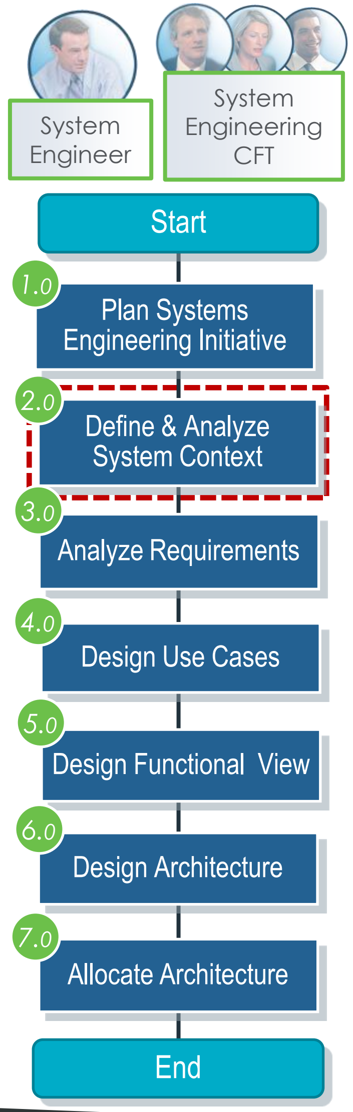
Define quality and performance measures and targets

Identify re-use candidates

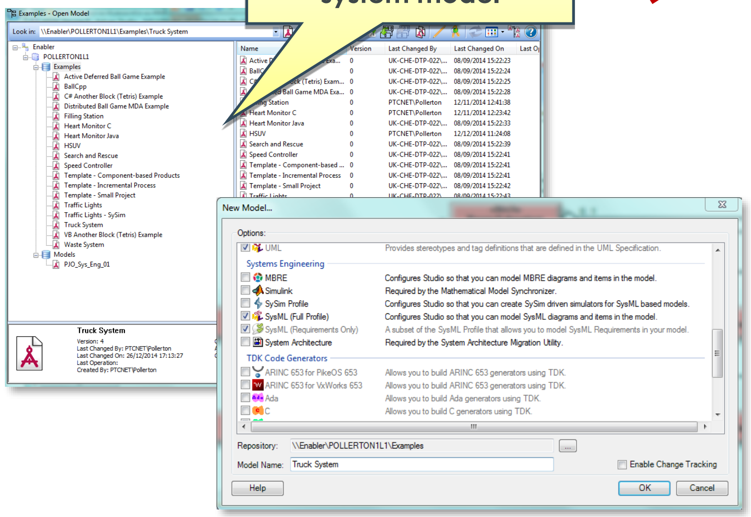
Refine and confirm plan

# MODEL BASED SYSTEM ENGINEERING PROCEDURE OVERVIEW

## ► Define and Analyze System Context



**Create and configure system model**



Examples - Open Model

Look in: \\Enabler\POLLERTON1L1\Examples\Truck System

- Enabler
  - POLLERTON1L1
    - Examples
      - Active Deferred Ball Game Example
      - BallCpp
      - C# Another Block (Tetris) Example
      - Distributed Ball Game MDA Example
      - Filling Station
      - Heart Monitor C
      - Heart Monitor Java
      - HSUV
      - Search and Rescue
      - Speed Controller
      - Template - Component-based Products
      - Template - Incremental Process
      - Template - Small Project
      - Traffic Lights
      - Traffic Lights - SySim
      - Truck System
      - VB Another Block (Tetris) Example
      - Waste System
    - Models
      - PJO_Sys_Eng_01

| Name | Version | Last Changed By | Last Changed On | Last O |
|---|---|---|---|---|
| Active D ... Exa... | 0 | UK-CHE-DTP-022\... | 08/09/2014 15:22:23 | |
| BallC | 0 | UK-CHE-DTP-022\... | 08/09/2014 15:22:24 | |
| C# ...ck (Tetris) Exam... | 0 | UK-CHE-DTP-022\... | 08/09/2014 15:22:25 | |
| ...ed Ball Game MDA Exa... | 0 | UK-CHE-DTP-022\... | 08/09/2014 15:22:28 | |
| ...ng Station | 0 | PTCNET\Pollerton | 12/11/2014 12:41:38 | |
| Heart Monitor C | 0 | PTCNET\Pollerton | 12/11/2014 12:23:42 | |
| Heart Monitor Java | 0 | UK-CHE-DTP-022\... | 08/09/2014 15:22:33 | |
| HSUV | 0 | PTCNET\Pollerton | 12/12/2014 11:24:08 | |
| Search and Rescue | 0 | UK-CHE-DTP-022\... | 08/09/2014 15:22:39 | |
| Speed Controller | 0 | UK-CHE-DTP-022\... | 08/09/2014 15:22:41 | |
| Template - Component-based ... | 0 | UK-CHE-DTP-022\... | 08/09/2014 15:22:41 | |
| Template - Incremental Process | 0 | UK-CHE-DTP-022\... | 08/09/2014 15:22:41 | |
| Template - Small Project | 0 | UK-CHE-DTP-022\... | 08/09/2014 15:22:42 | |
| Traffic Lights | 0 | UK-CHE-DTP-022\... | 08/09/2014 15:22:43 | |

Truck System
Version: 4
Last Changed By: PTCNET\Pollerton
Last Changed On: 26/12/2014 17:13:27
Last Operation:
Created By: PTCNET\Pollerton

New Model...

Options:

☑ UML — Provides stereotypes and tag definitions that are defined in the UML Specification.

Systems Engineering

- ☐ MBRE — Configures Studio so that you can model MBRE diagrams and items in the model.
- ☐ Simulink — Required by the Mathematical Model Synchronizer.
- ☐ SySim Profile — Configures Studio so that you can create SySim driven simulators for SysML based models.
- ☑ SysML (Full Profile) — Configures Studio so that you can model SysML diagrams and items in the model.
- ☑ SysML (Requirements Only) — A subset of the SysML Profile that allows you to model SysML Requirements in your model.
- ☐ System Architecture — Required by the System Architecture Migration Utility.

TDK Code Generators

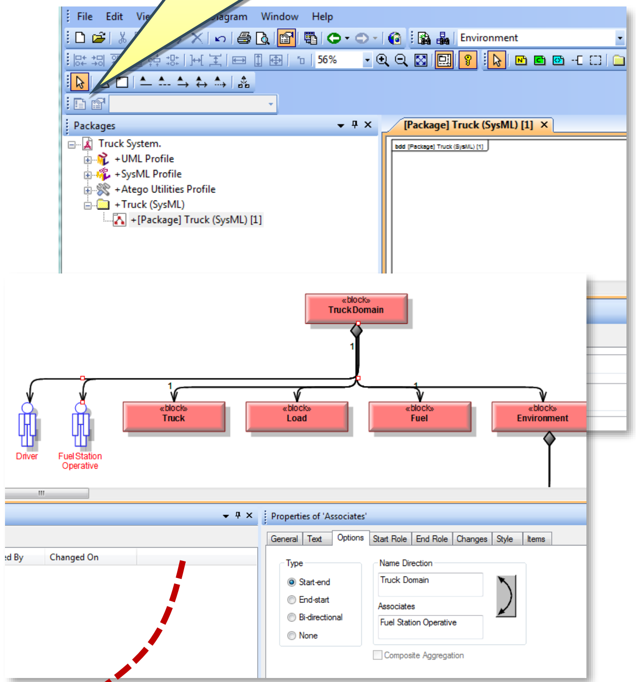
- ☐ ARINC 653 for PikeOS 653 — Allows you to build ARINC 653 generators using TDK.
- ☐ ARINC 653 for VxWorks 653 — Allows you to build ARINC 653 generators using TDK.
- ☐ Ada — Allows you to build Ada generators using TDK.
- ☐ C — Allows you to build C generators using TDK.

Repository: \\Enabler\POLLERTON1L1\Examples

Model Name: Truck System

☐ Enable Change Tracking

**Capture system domain information in consultation with CFT**



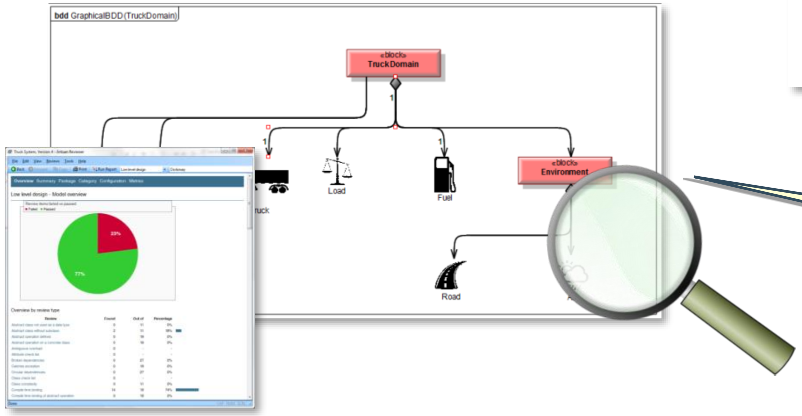
- Truck System.
  - +UML Profile
  - +SysML Profile
  - +Atego Utilities Profile
  - +Truck (SysML)
    - +[Package] Truck (SysML) [1]

Type: ● Start-end ○ End-start ○ Bi-directional ○ None

Name Direction: Truck Domain

Associates: Fuel Station Operative

☐ Composite Aggregation

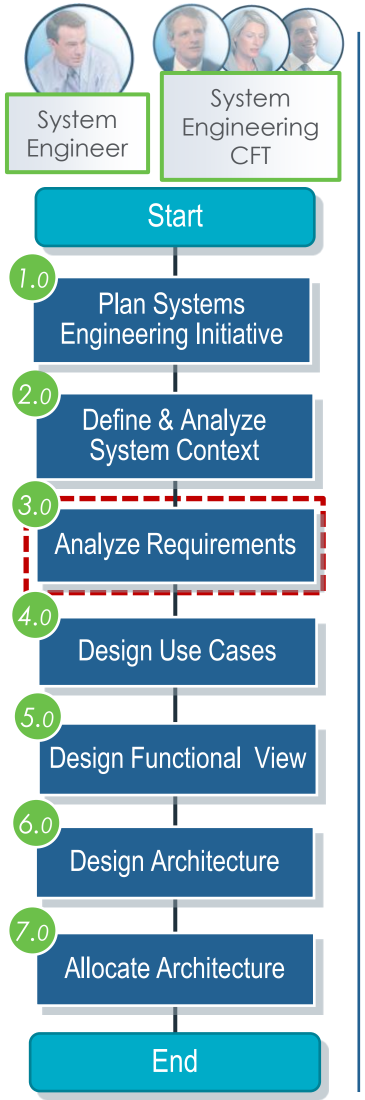


**Analyze, refine and update system domain information**

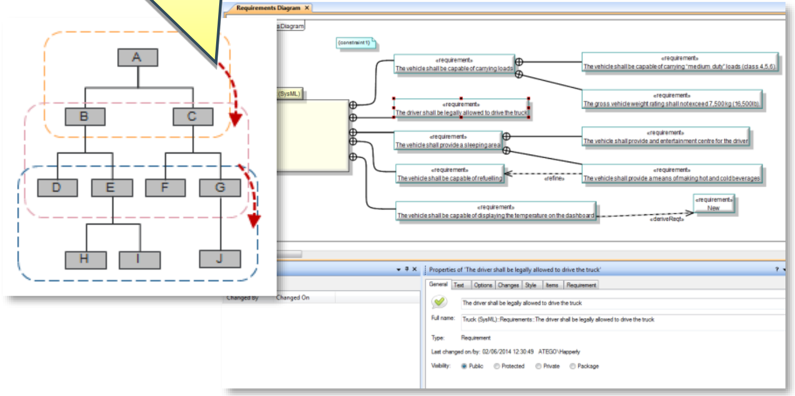
# MODEL BASED SYSTEM ENGINEERING PROCEDURE OVERVIEW

ptc

## ► Analyze Requirements



Capture system or sub-system requirements with CFT



A
B
C
D
E
F
G
H
I
J
Requirements Diagram
{constraint1}
(SysML)
«requirement» The vehicle shall be capable of carrying loads
«requirement» The vehicle shall be capable of carrying "medium duty" loads (class 4,5,6)
«requirement» The driver shall be legally allowed to drive the truck
«requirement» The gross vehicle weight rating shall not exceed 7,500kg (16,500lb).
«requirement» The vehicle shall provide a sleeping area
«requirement» The vehicle shall provide and entertainment centre for the driver
«requirement» The vehicle shall be capable of refuelling
«refine»
«requirement» The vehicle shall provide a means of making hot and cold beverages
«requirement» The vehicle shall be capable of displaying the temperature on the dashboard
«deriveReqt»
«requirement» New
Changed By
Changed On
General
Text
Options
Changes
Style
Items
Requirement

Properties of 'The driver shall be legally allowed to drive the truck'

The driver shall be legally allowed to drive the truck

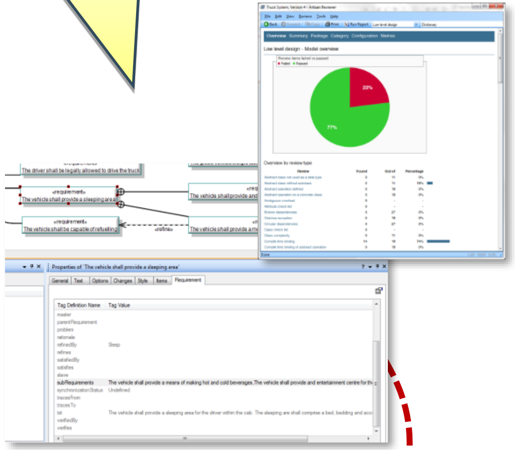
Full name: Truck (SysML)::Requirements::The driver shall be legally allowed to drive the truck

Type: Requirement

Last changed on/by: 02/06/2014 12:30:49 ATEGO\Happerly

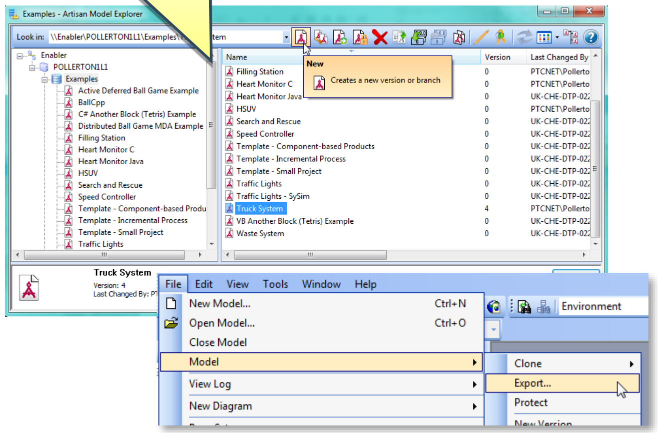
Visibility: ● Public ○ Protected ○ Private ○ Package

Analyze and refine requirements



Overview
The driver shall be legally allowed to drive the truck
«requirement»
The vehicle shall provide a sleeping area
The vehicle shall be capable of refuelling
«refine»
Properties of 'The vehicle shall provide a sleeping area'
General
Text
Options
Changes
Style
Items
Requirement
Tag Definition Name
Tag Value

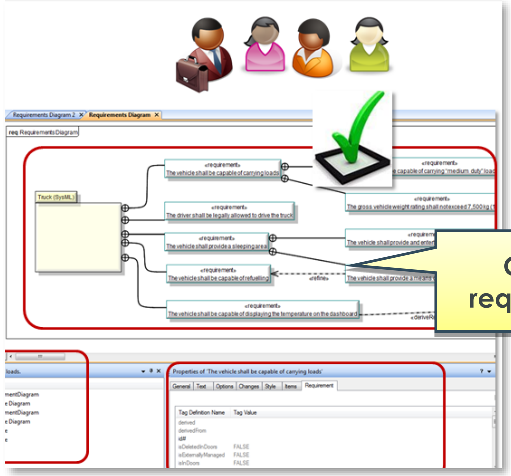
Update and baseline system model



- Enabler
  - POLLERTON1L1
    - Examples
      - Active Deferred Ball Game Example
      - BallCpp
      - C# Another Block (Tetris) Example
      - Distributed Ball Game MDA Example
      - Filling Station
      - Heart Monitor C
      - Heart Monitor Java
      - HSUV
      - Search and Rescue
      - Speed Controller
      - Template - Component-based Produ
      - Template - Incremental Process
      - Template - Small Project
      - Traffic Lights

| Name | Version | Last Changed By |
|---|---|---|
| Filling Station | 0 | PTCNET\Pollerto |
| Heart Monitor C | 0 | PTCNET\Pollerto |
| Heart Monitor Java | 0 | UK-CHE-DTP-02 |
| HSUV | 0 | PTCNET\Pollerto |
| Search and Rescue | 0 | UK-CHE-DTP-02 |
| Speed Controller | 0 | UK-CHE-DTP-02 |
| Template - Component-based Products | 0 | UK-CHE-DTP-02 |
| Template - Incremental Process | 0 | UK-CHE-DTP-02 |
| Template - Small Project | 0 | UK-CHE-DTP-02 |
| Traffic Lights | 0 | UK-CHE-DTP-02 |
| Traffic Lights - SySim | 0 | UK-CHE-DTP-02 |
| Truck System | 4 | PTCNET\Pollerto |
| VB Another Block (Tetris) Example | 0 | UK-CHE-DTP-02 |
| Waste System | 0 | UK-CHE-DTP-02 |

Truck System
Version: 4
Last Changed By: P

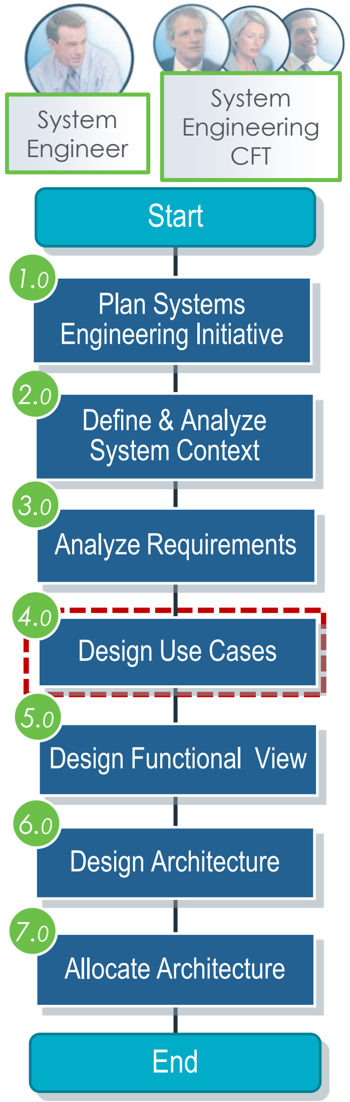


Requirements Diagram 2
Requirements Diagram
req Requirements Diagram
Truck (SysML)
«requirement»
The vehicle shall be capable of carrying loads
The driver shall be legally allowed to drive the truck
The vehicle shall provide a sleeping area
The vehicle shall be capable of refuelling
The vehicle shall be capable of displaying the temperature on the dashboard
The gross vehicle weight rating shall not exceed 7,500kg
«refine»
Properties of 'The vehicle shall be capable of carrying loads'
Tag Definition Name
Tag Value
FALSE

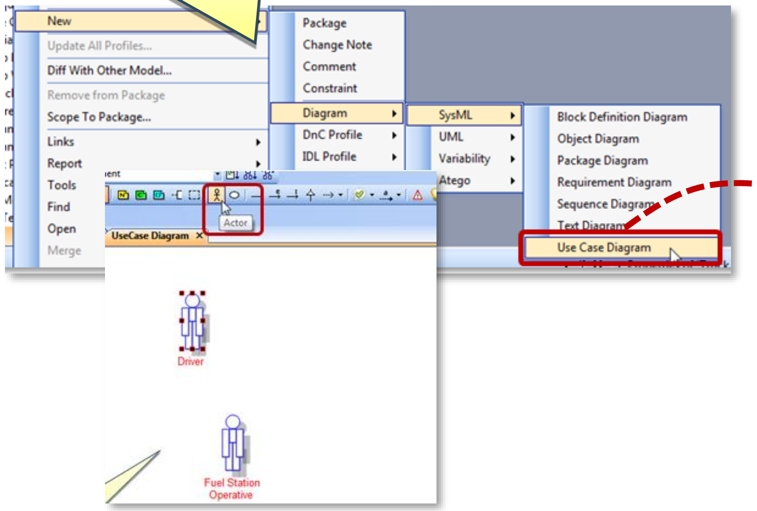
Confirm requirements

# MODEL BASED SYSTEM ENGINEERING PROCEDURE OVERVIEW

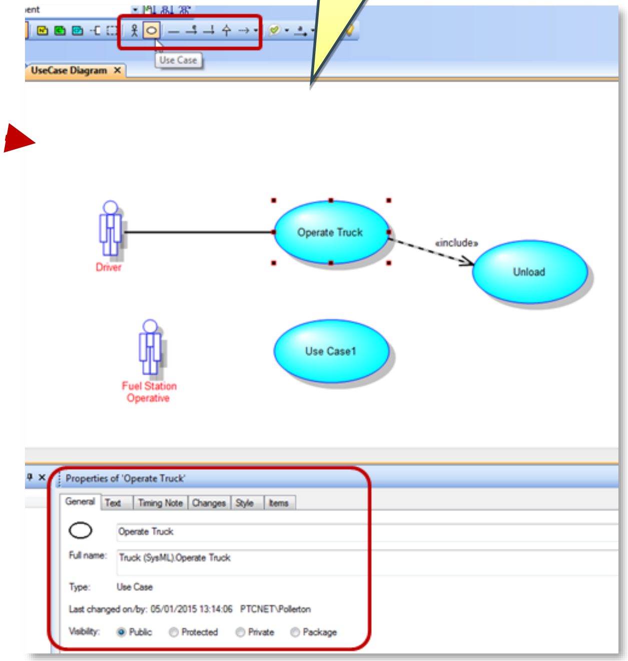
## ► Design Use Cases



Define system actors with CFT



Define use cases and relationships



Properties of 'Operate Truck'

General | Text | Timing Note | Changes | Style | Items

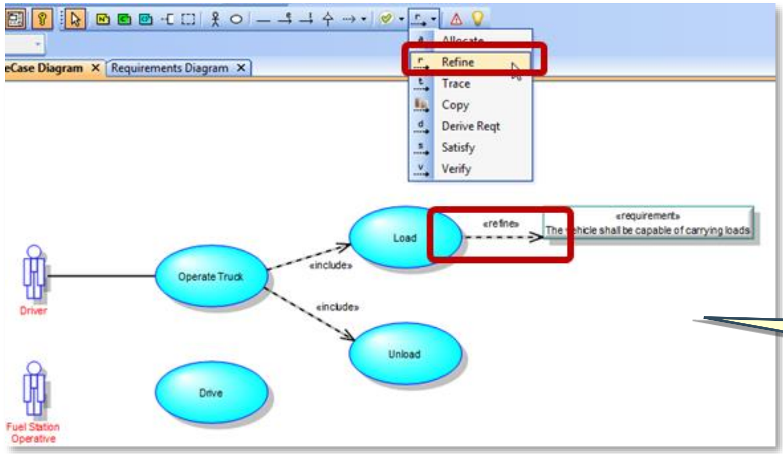
Operate Truck

Full name: Truck (SysML).Operate Truck

Type: Use Case

Last changed on/by: 05/01/2015 13:14:06 PTCNET\Pollerton

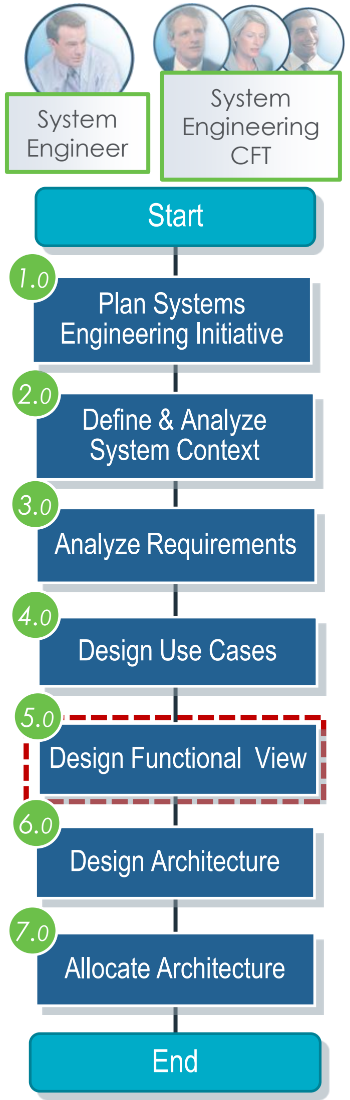
Visibility: ● Public ○ Protected ○ Private ○ Package

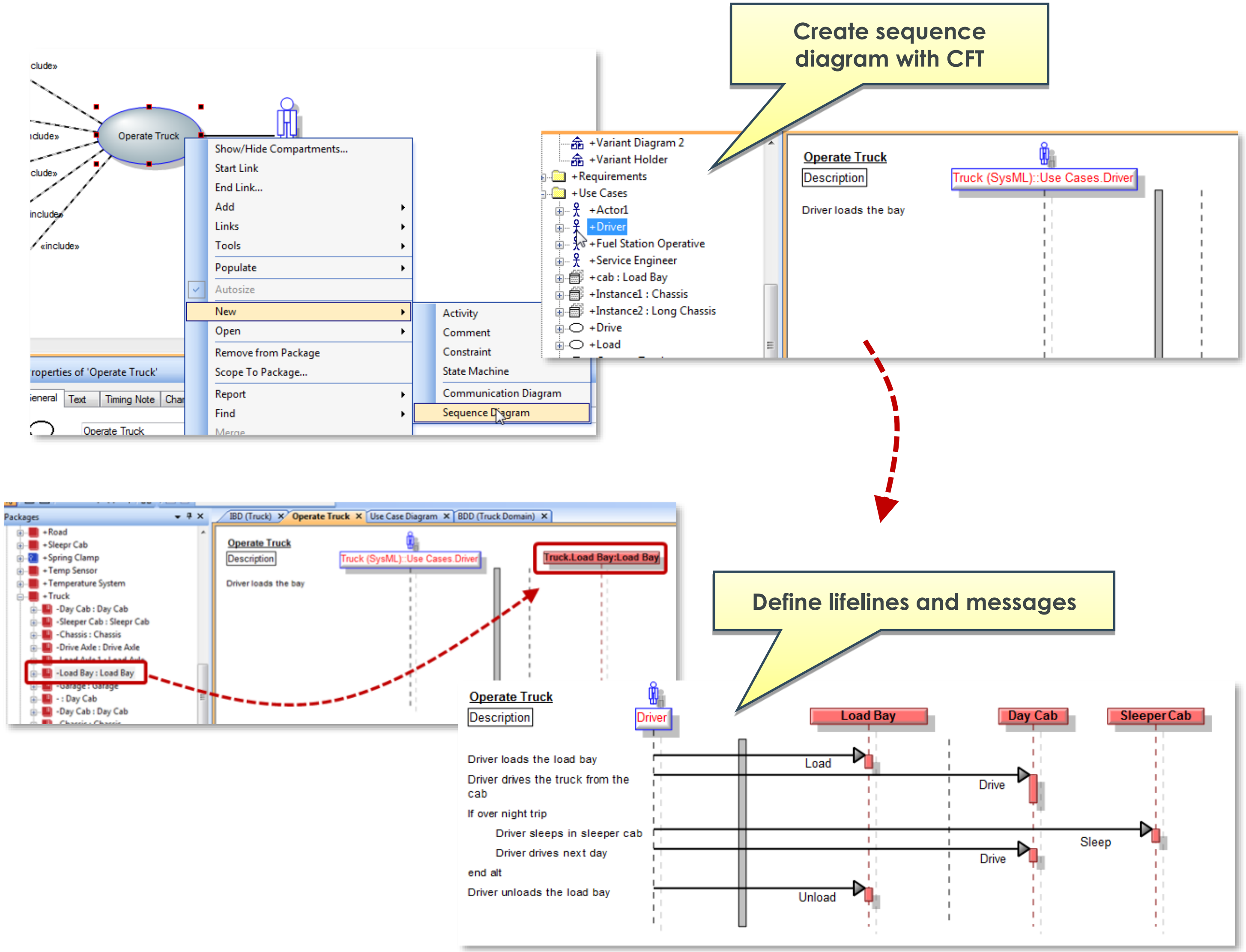


Associate use cases to requirements

# MODEL BASED SYSTEM ENGINEERING PROCEDURE OVERVIEW

## ► Design Functional View





Create sequence diagram with CFT

Define lifelines and messages

Driver loads the bay

Driver loads the load bay

Driver drives the truck from the cab

If over night trip

Driver sleeps in sleeper cab

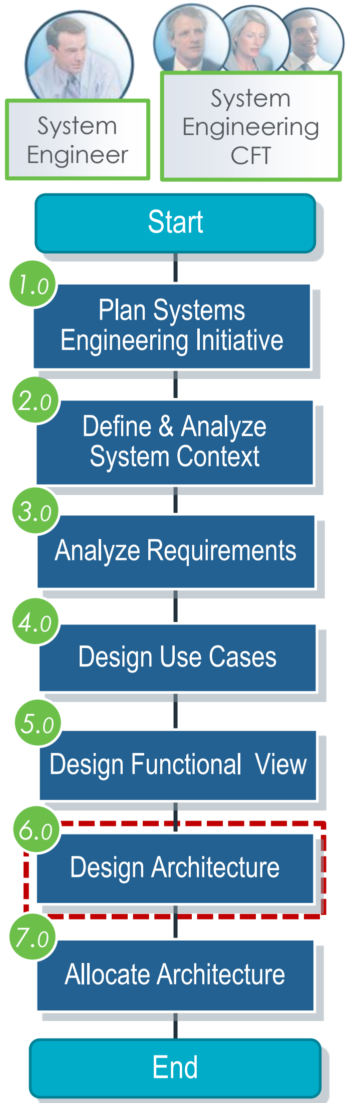
Driver drives next day

end alt

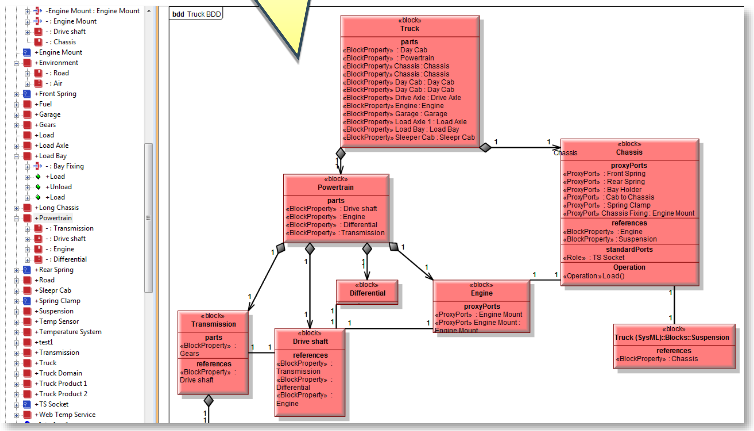
Driver unloads the load bay

# MODEL BASED SYSTEM ENGINEERING PROCEDURE OVERVIEW

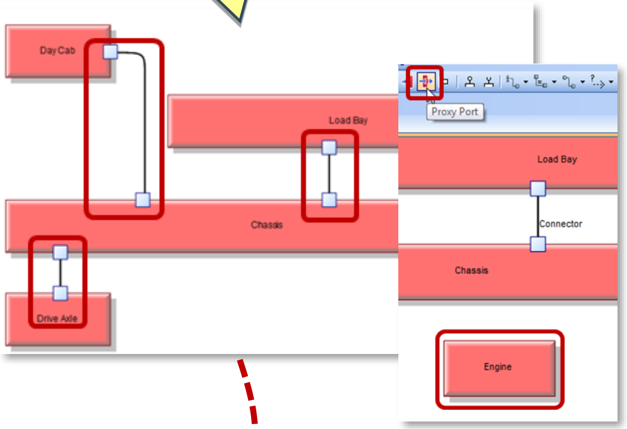
## ► Design Architecture


System Engineer
System Engineering CFT
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

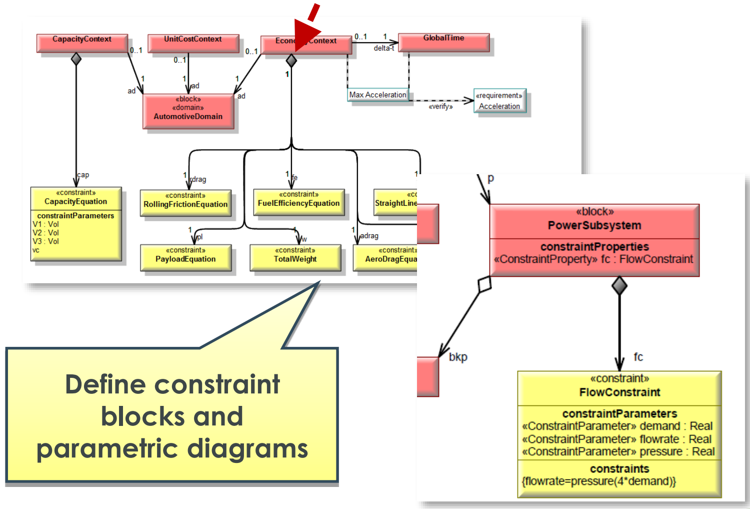
Define system blocks


bdd Truck BDD
«block» Truck
parts
«BlockProperty» : Day Cab
«BlockProperty» : Powertrain
«BlockProperty» Chassis : Chassis
«BlockProperty» Chassis : Chassis
«BlockProperty» Day Cab : Day Cab
«BlockProperty» Day Cab : Day Cab
«BlockProperty» Drive Axle : Drive Axle
«BlockProperty» Engine : Engine
«BlockProperty» Garage : Garage
«BlockProperty» Load Axle 1 : Load Axle
«BlockProperty» Load Bay : Load Bay
«BlockProperty» Sleeper Cab : Sleepr Cab
«block» Chassis
proxyPorts
«ProxyPort» : Front Spring
«ProxyPort» : Rear Spring
«ProxyPort» : Bay Holder
«ProxyPort» : Cab to Chassis
«ProxyPort» : Spring Clamp
«ProxyPort» Chassis Fixing : Engine Mount
references
«BlockProperty» : Engine
«BlockProperty» : Suspension
standardPorts
«Role» : TS Socket
Operation
«Operation» Load()
«block» Powertrain
parts
«BlockProperty» : Drive shaft
«BlockProperty» : Engine
«BlockProperty» : Differential
«BlockProperty» : Transmission
«block» Differential
«block» Engine
proxyPorts
«ProxyPort» : Engine Mount
«ProxyPort» Engine Mount : Engine Mount
«block» Transmission
parts
«BlockProperty» : Gears
references
«BlockProperty» : Drive shaft
«block» Drive shaft
references
«BlockProperty» : Transmission
«BlockProperty» : Differential
«BlockProperty» : Engine
«block» Truck (SysML)::Blocks::Suspension
references
«BlockProperty» : Chassis
Engine Mount : Engine Mount
Drive shaft
Chassis
Engine Mount
Environment
Road
Air
Front Spring
Fuel
Garage
Gears
Load
Load Axle
Load Bay
Bay Fixing
Unload
Long Chassis
Powertrain
Transmission
Engine
Differential
Rear Spring
Sleepr Cab
Spring Clamp
Suspension
Temp Sensor
Temperature System
test1
Truck
Truck Domain
Truck Product 1
Truck Product 2
TS Socket
Web Temp Service

Define internal blocks, ports and item flows



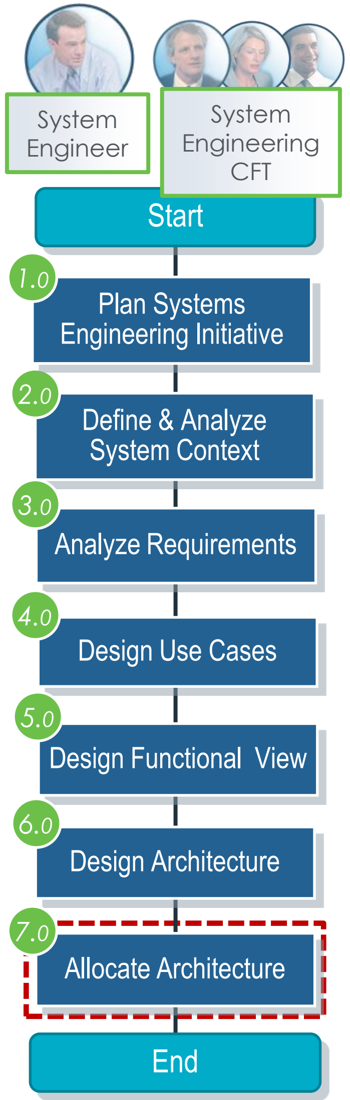
Review system architecture design

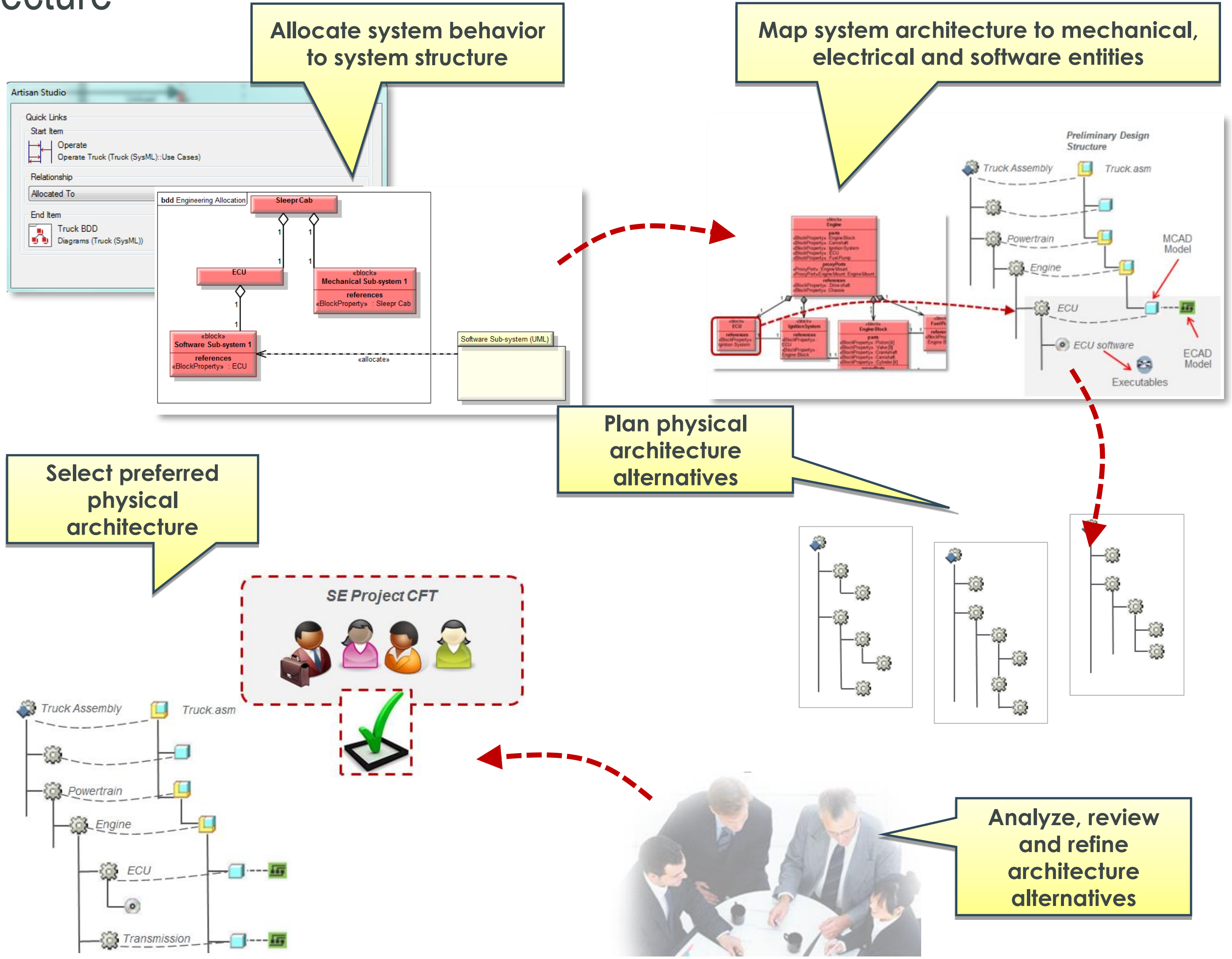


Define constraint blocks and parametric diagrams

# MODEL BASED SYSTEM ENGINEERING PROCEDURE OVERVIEW

## ► Allocate Architecture





Allocate system behavior to system structure

Artisan Studio

Quick Links

Start Item

Operate
Operate Truck (Truck (SysML)::Use Cases)

Relationship

Allocated To

End Item

Truck BDD
Diagrams (Truck (SysML))

Map system architecture to mechanical, electrical and software entities

Plan physical architecture alternatives

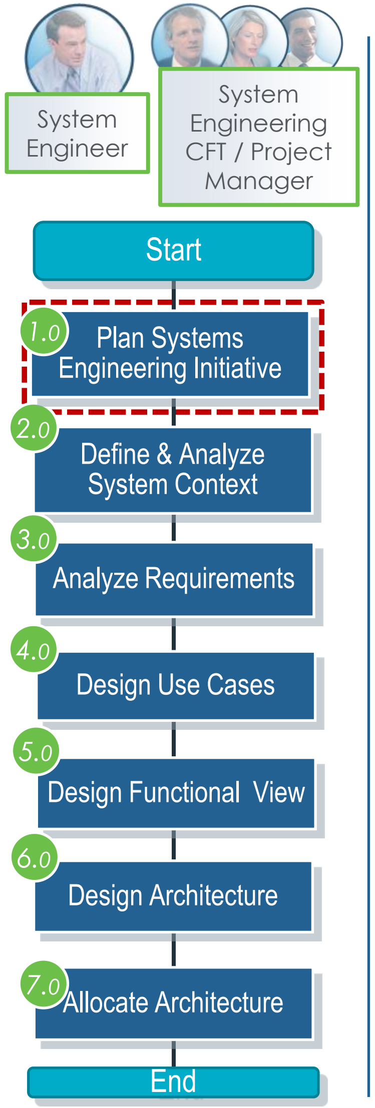
Select preferred physical architecture

Analyze, review and refine architecture alternatives

# PLAN
# SYSTEMS ENGINEERING
# INITIATIVE

# MODEL BASED SYSTEM ENGINEERING BEST PRACTICE PROCEDURE

## ► Plan Systems Engineering Initiative

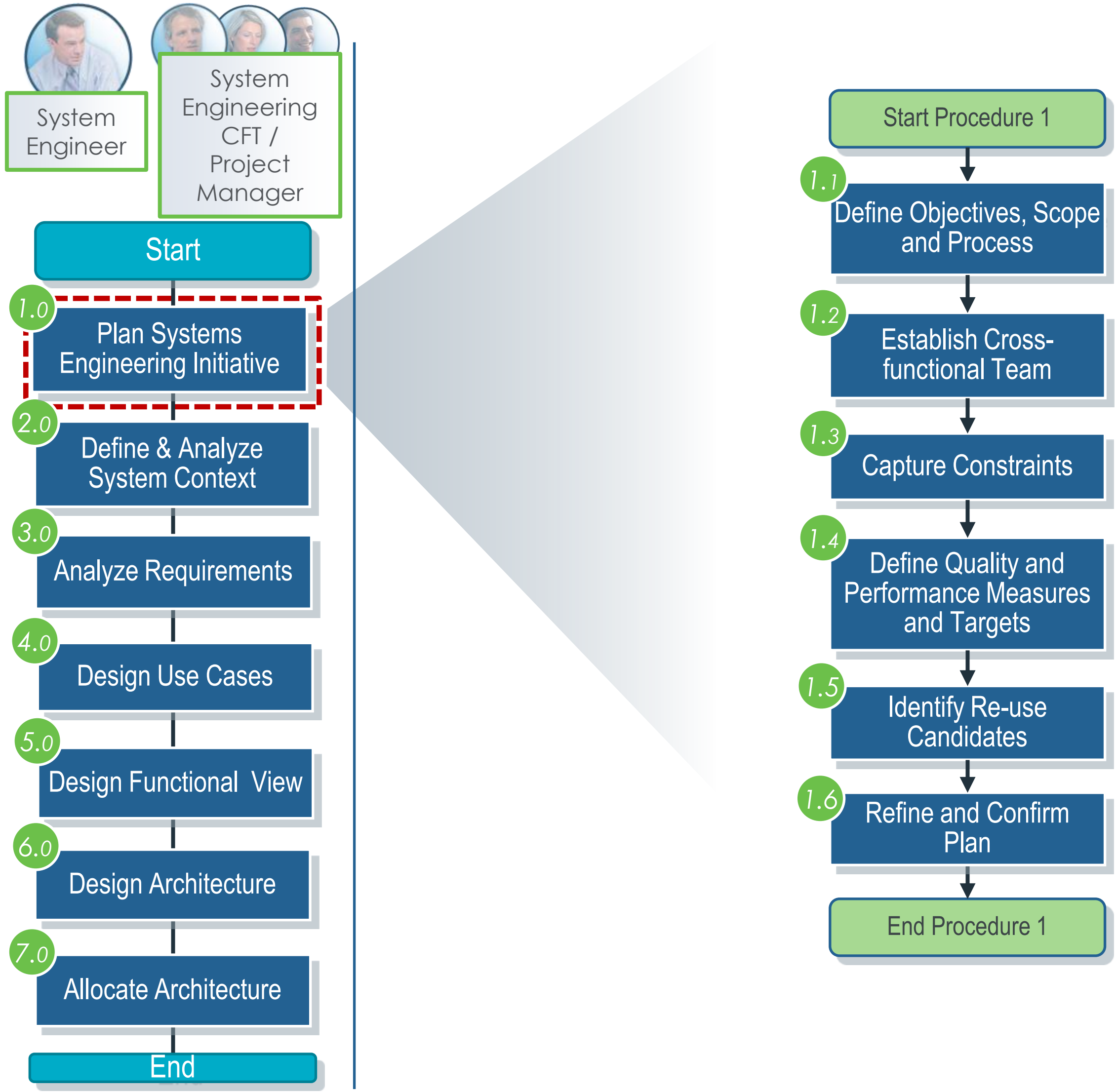


System Engineer
System Engineering CFT / Project Manager
Start
1.0 Plan Systems Engineering Initiative
2.0 Define & Analyze System Context
3.0 Analyze Requirements
4.0 Design Use Cases
5.0 Design Functional View
6.0 Design Architecture
7.0 Allocate Architecture
End

- 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

# MODEL BASED SYSTEM ENGINEERING

## ▶ Plan Systems Engineering Initiative



System Engineer

System Engineering CFT / Project Manager

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

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

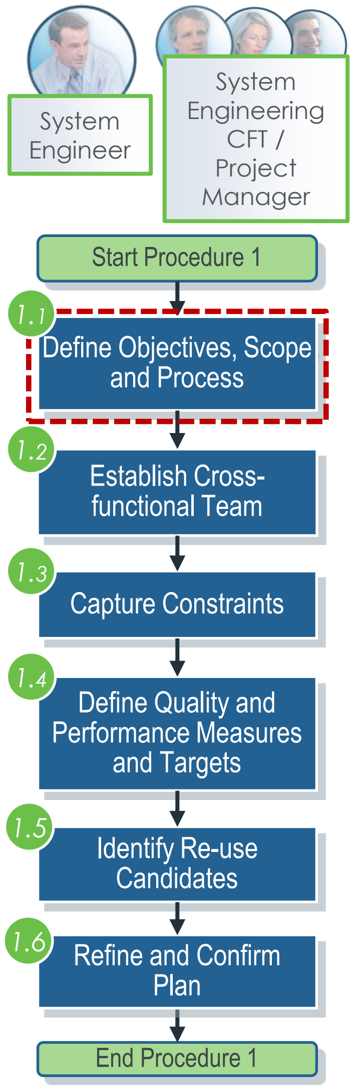
1.5 Identify Re-use Candidates

1.6 Refine and Confirm Plan

End Procedure 1

# MODEL BASED SYSTEM ENGINEERING

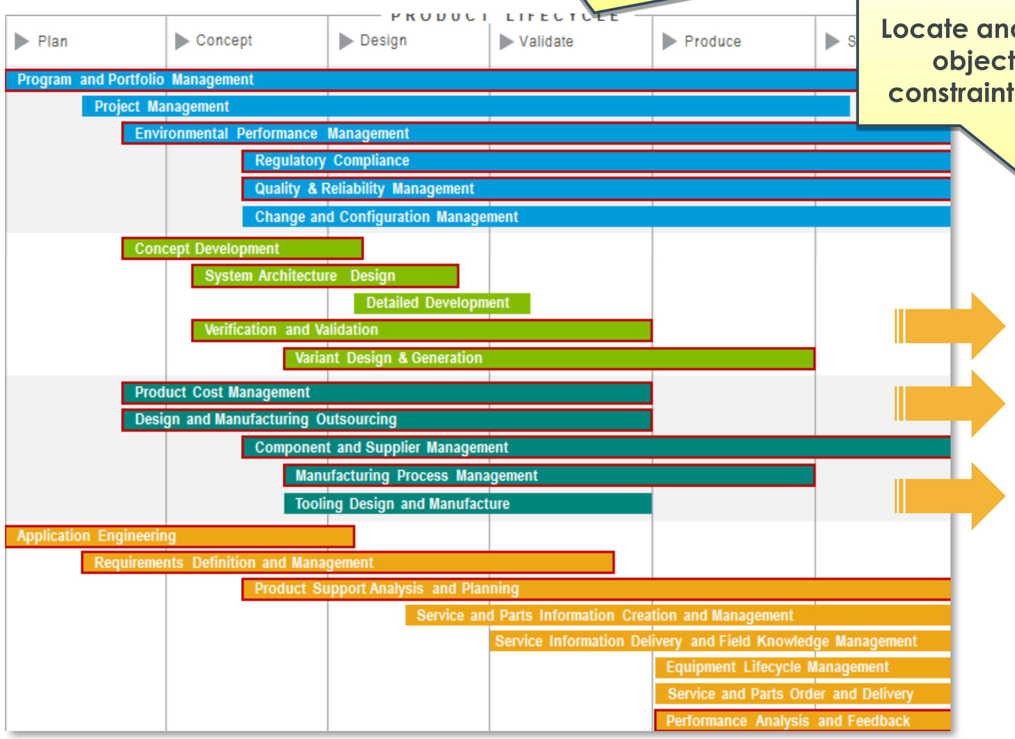
## ▶ Define Objectives and Scope


System Engineer
System Engineering CFT / Project Manager
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

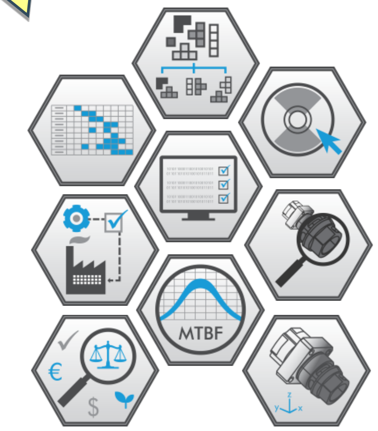
There are many different starting points for a Systems Engineering initiative:

- 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**

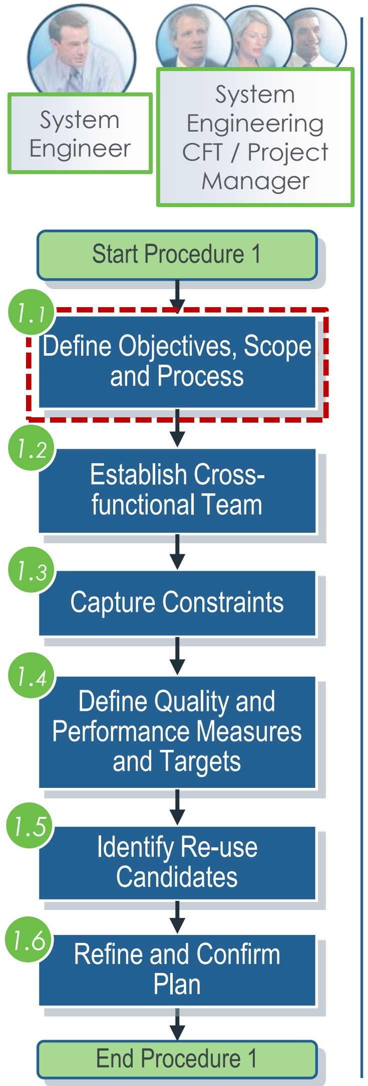


**Locate and verify any existing requirements, objectives, scope, domain, process, constraints, performance measures/targets**



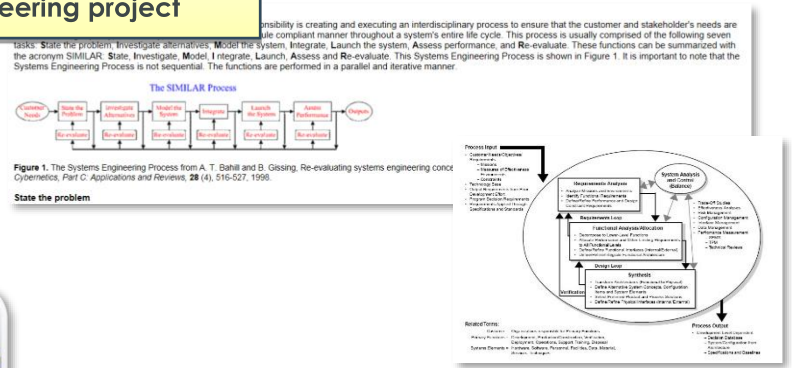
# MODEL BASED SYSTEM ENGINEERING

## ► Define Objectives and Scope


System Engineer
System Engineering CFT / Project Manager
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

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

*Source: INCOSE, Wikipedia*

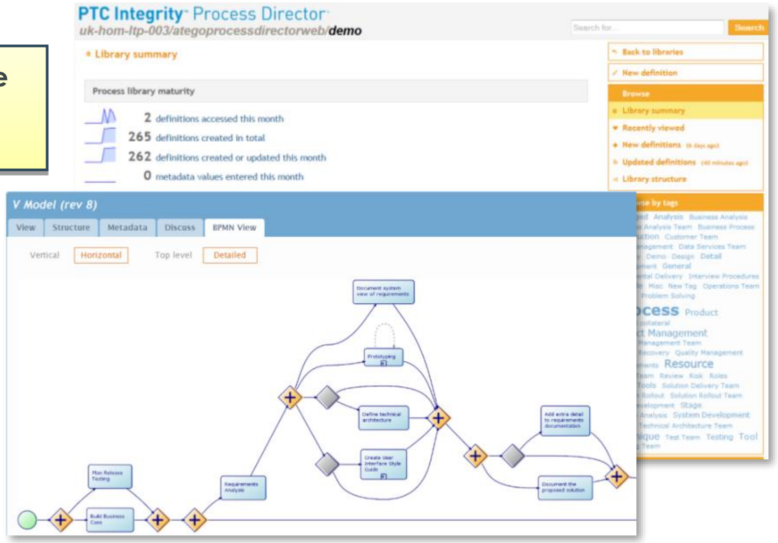


...nsibility is creating and executing an interdisciplinary process to ensure that the customer and stakeholder's needs are ...ule compliant manner throughout a system's entire life cycle. This process is usually comprised of the following seven tasks: **S**tate the problem, **I**nvestigate alternatives, **M**odel the system, **I**ntegrate, **L**aunch the system, **A**ssess performance, and **R**e-evaluate. These functions can be summarized with the acronym SIMILAR: **S**tate, **I**nvestigate, **M**odel, **I** ntegrate, **L**aunch, **A**ssess and **R**e-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. Gissing, Re-evaluating systems engineering conce... *Cybernetics, Part C: Applications and Reviews*, **28** (4), 516-527, 1998.

**State the problem**

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



PTC Integrity™ Process Director™

*uk-hom-ltp-003/ategoprocessdirectorweb/***demo**

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

PTC Integrity™ Process Perspective™

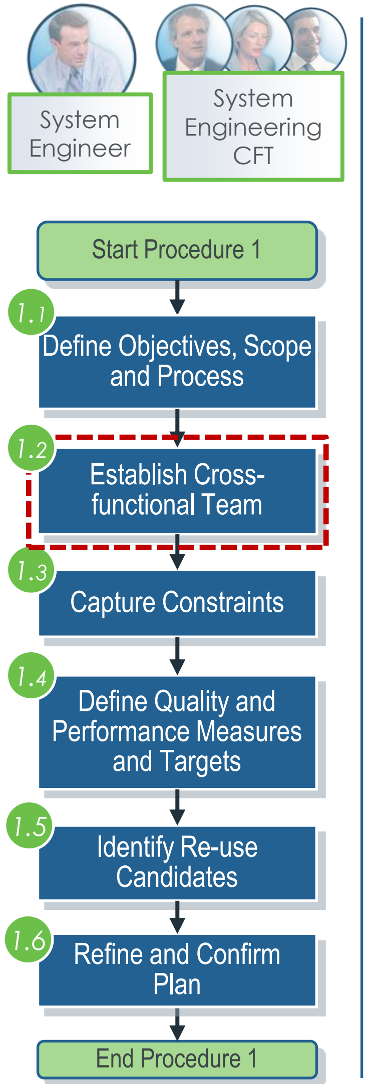
ISO

OMG OBJECT MANAGEMENT GROUP

INCOSE International Council on Systems Engineering

# MODEL BASED SYSTEM ENGINEERING

## ► Establish Cross-functional Team



System Engineer

System Engineering CFT

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

**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.**

*SE Project CFT*

Manageme nt

Marketing

Electrical

Mechanic al

Software (Device, Cloud, Mobile)

System

Analysis

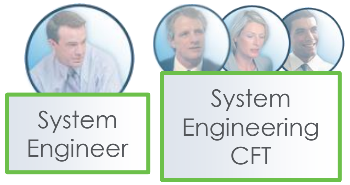
Service

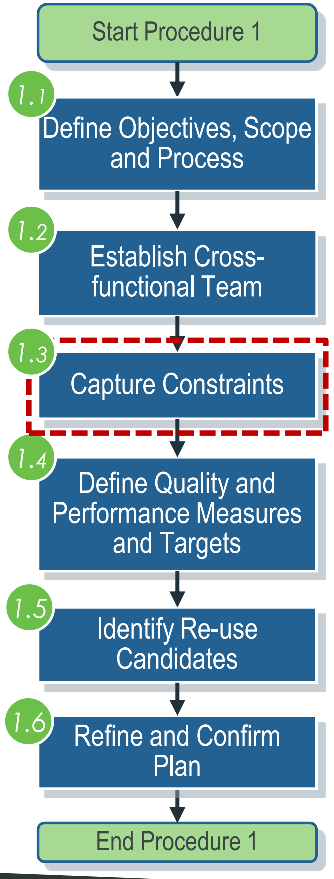
Manufacturing

# MODEL BASED SYSTEM ENGINEERING

ptc

## ► Capture Constraints

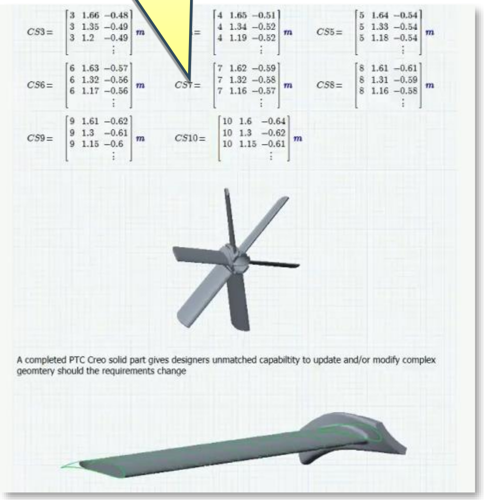




Start Procedure 1
1.1
Define Objectives, Scope and Process
1.2
Establish Cross-functional Team
1.3
Capture Constraints
1.4
Define Quality and Performance Measures and Targets
1.5
Identify Re-use Candidates
1.6
Refine and Confirm Plan
End Procedure 1

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

**Include any known engineering limitations or constraints**



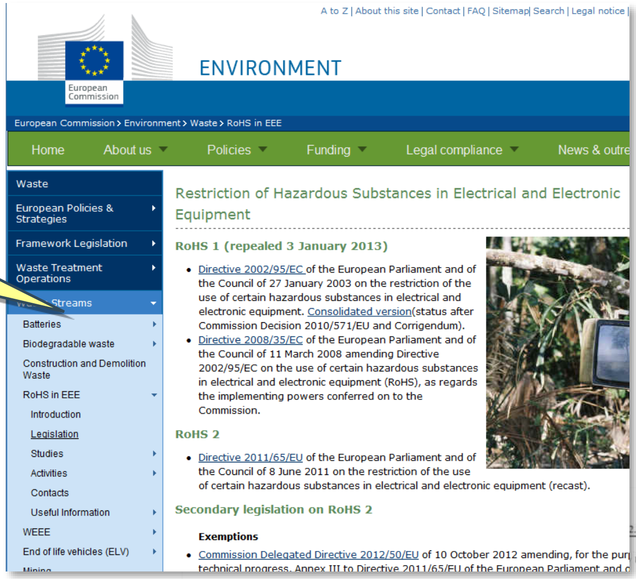
$$CS3=\begin{bmatrix}3 & 1.66 & -0.48\\ 3 & 1.35 & -0.49\\ 3 & 1.2 & -0.49\\ & & \vdots\end{bmatrix} m \quad =\begin{bmatrix}4 & 1.65 & -0.51\\ 4 & 1.34 & -0.52\\ 4 & 1.19 & -0.52\\ & & \vdots\end{bmatrix} m \quad CS5=\begin{bmatrix}5 & 1.64 & -0.54\\ 5 & 1.33 & -0.54\\ 5 & 1.18 & -0.54\\ & & \vdots\end{bmatrix} m$$

$$CS6=\begin{bmatrix}6 & 1.63 & -0.57\\ 6 & 1.32 & -0.56\\ 6 & 1.17 & -0.56\\ & & \vdots\end{bmatrix} m \quad CS7=\begin{bmatrix}7 & 1.62 & -0.59\\ 7 & 1.32 & -0.58\\ 7 & 1.16 & -0.57\\ & & \vdots\end{bmatrix} m \quad CS8=\begin{bmatrix}8 & 1.61 & -0.61\\ 8 & 1.31 & -0.59\\ 8 & 1.16 & -0.58\\ & & \vdots\end{bmatrix} m$$

$$CS9=\begin{bmatrix}9 & 1.61 & -0.62\\ 9 & 1.3 & -0.61\\ 9 & 1.15 & -0.6\\ & & \vdots\end{bmatrix} m \quad CS10=\begin{bmatrix}10 & 1.6 & -0.64\\ 10 & 1.3 & -0.62\\ 10 & 1.15 & -0.61\\ & & \vdots\end{bmatrix} m$$

A completed PTC Creo solid part gives designers unmatched capabiltity to update and/or modify complex geomtery should the requirements change

*PTC Mathcad*



A to Z | About this site | Contact | FAQ | Sitemap| Search | Legal notice |

European Commission

ENVIRONMENT

European Commission > Environment > Waste > RoHS in EEE

Restriction of Hazardous Substances in Electrical and Electronic Equipment

RoHS 1 (repealed 3 January 2013)

- Directive 2002/95/EC of the European Parliament and of the Council of 27 January 2003 on the restriction of the use of certain hazardous substances in electrical and electronic equipment. Consolidated version(status after Commission Decision 2010/571/EU and Corrigendum).
- Directive 2008/35/EC of the European Parliament and of the Council of 11 March 2008 amending Directive 2002/95/EC on the use of certain hazardous substances in electrical and electronic equipment (RoHS), as regards the implementing powers conferred on to the Commission.

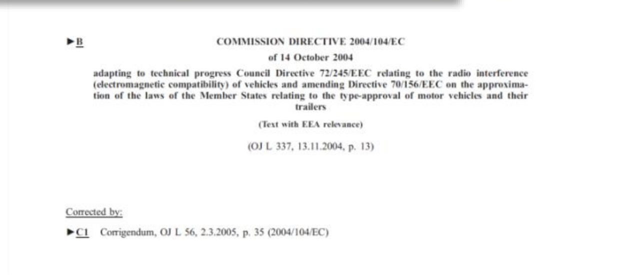
RoHS 2

- Directive 2011/65/EU of the European Parliament and of the Council of 8 June 2011 on the restriction of the use of certain hazardous substances in electrical and electronic equipment (recast).

Secondary legislation on RoHS 2

**Exemptions**

- Commission Delegated Directive 2012/50/EU of 10 October 2012 amending, for the pur technical progress, Annex III to Directive 2011/65/EU of the European Parliament and o



2.2004 — 000.001 — 1

liability for its contents

► B COMMISSION DIRECTIVE 2004/104/EC

of 14 October 2004

adapting to technical progress Council Directive 72/245/EEC relating to the radio interference (electromagnetic compatibility) of vehicles and amending Directive 70/156/EEC on the approximation of the laws of the Member States relating to the type-approval of motor vehicles and their trailers

(Text with EEA relevance)

(OJ L 337, 13.11.2004, p. 13)

Corrected by:

► C1 Corrigendum, OJ L 56, 2.3.2005, p. 35 (2004/104/EC)

WEEE

RoHS 2002/95/EC

ENERGY STAR

REACH The new EU chemicals legislation

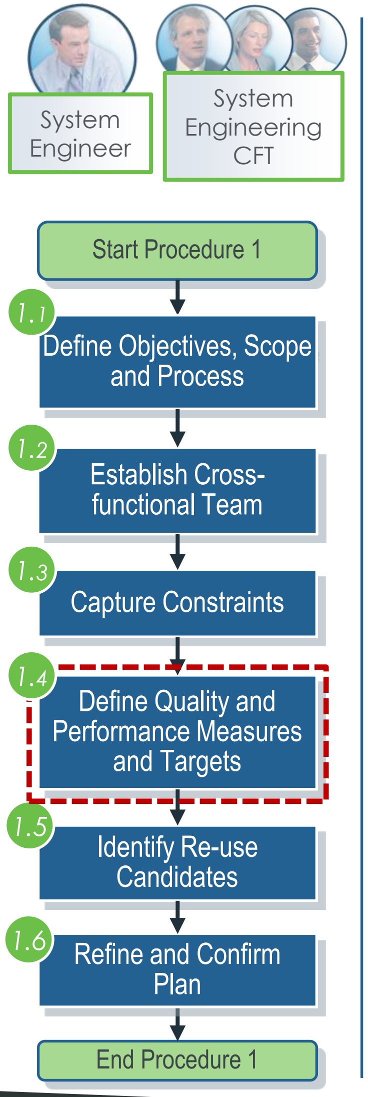
IPC

**Note**

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

# MODEL BASED SYSTEM ENGINEERING

## ► Define Quality and Performance Targets



System Engineer

System Engineering CFT

Start Procedure 1

1.1 Define Objectives, Scope and Process

1.2 Establish Cross-functional Team

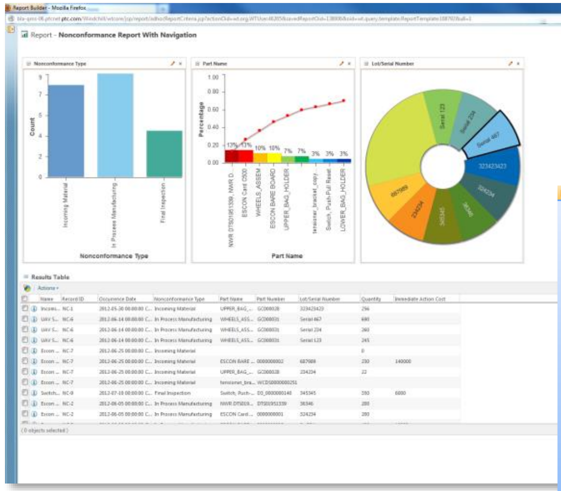
1.3 Capture Constraints

1.4 Define Quality and Performance Measures and Targets

1.5 Identify Re-use Candidates

1.6 Refine and Confirm Plan

End Procedure 1



Report Builder - Mozilla Firefox

Report - Nonconformance Report With Navigation

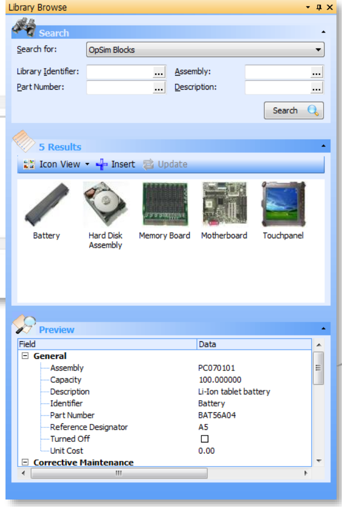
Results Table

Actions

| Name | Record ID | Occurrence Date | Nonconformance Type | Part Name | Part Number | Lot/Serial Number | Quantity | Immediate Action Cost |
|---|---|---|---|---|---|---|---|---|
| Incomi... | NC-1 | 2012-05-30 00:00:00 C... | Incoming Material | UPPER_BAG_... | GC000028 | 323423423 | 256 | |
| UAV S... | NC-6 | 2012-06-14 00:00:00 C... | In Process Manufacturing | WHEELS_ASS... | GC000031 | Serial 467 | 690 | |
| UAV S... | NC-6 | 2012-06-14 00:00:00 C... | In Process Manufacturing | WHEELS_ASS... | GC000031 | Serial 234 | 260 | |
| UAV S... | NC-6 | 2012-06-14 00:00:00 C... | In Process Manufacturing | WHEELS_ASS... | GC000031 | Serial 123 | 245 | |
| Escon ... | NC-7 | 2012-06-25 00:00:00 C... | Incoming Material | | | | 0 | |
| Escon ... | NC-7 | 2012-06-25 00:00:00 C... | Incoming Material | ESCON BARE ... | 0000000002 | 687989 | 230 | 140000 |
| Escon ... | NC-7 | 2012-06-25 00:00:00 C... | Incoming Material | UPPER_BAG_... | GC000028 | 234234 | 22 | |
| Escon ... | NC-7 | 2012-06-25 00:00:00 C... | Incoming Material | tensioner_bra... | WCDS0000000251 | | | |
| Switch... | NC-9 | 2012-07-18 00:00:00 C... | Final Inspection | Switch, Push-... | DS_0000000140 | 345345 | 350 | 6000 |
| Escon ... | NC-2 | 2012-06-05 00:00:00 C... | In Process Manufacturing | NWR DTS019... | DTS01951339 | 36346 | 200 | |
| Escon ... | NC-2 | 2012-06-05 00:00:00 C... | In Process Manufacturing | ESCON Card... | 0000000001 | 324234 | 200 | |

(0 objects selected)

Coordinate with Quality Mgmt processes and define quality and performance measures



Library Browse

Search

Search for: OpSim Blocks

Library Identifier:

Assembly:

Part Number:

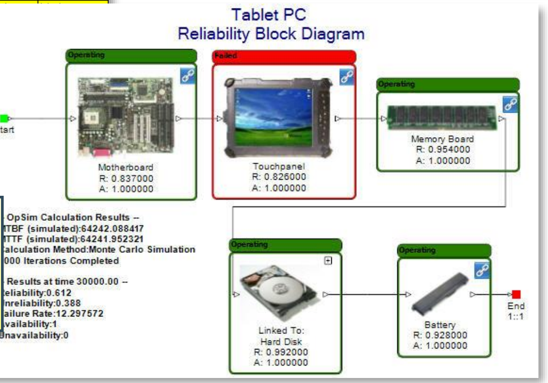
Description:

Preview

| Field | Data |
|---|---|
| ⊟ General | |
| Assembly | PC070101 |
| Capacity | 100.000000 |
| Description | Li-Ion tablet battery |
| Identifier | Battery |
| Part Number | BAT56A04 |
| Reference Designator | A5 |
| Turned Off | □ |
| Unit Cost | 0.00 |
| ⊟ Corrective Maintenance | |

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 | | |



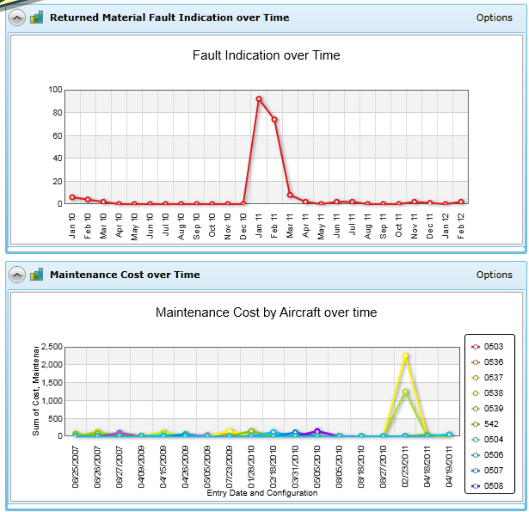
Tablet PC
Reliability Block Diagram

OpSim Calculation Results --
ITBF (simulated):64242.088417
ITTF (simulated):64241.952321
alculation Method:Monte Carlo Simulation
000 Iterations Completed

Results at time 30000.00 --
eliability:0.612
nreliability:0.388
ailure Rate:12.297572
vailability:1
Unavailability:0

**Note**

This is an advanced, optional technique

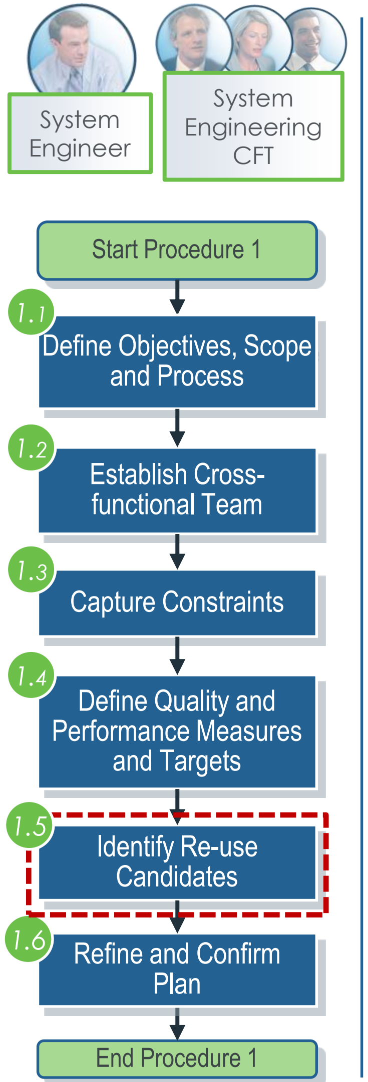


Returned Material Fault Indication over Time

Maintenance Cost over Time

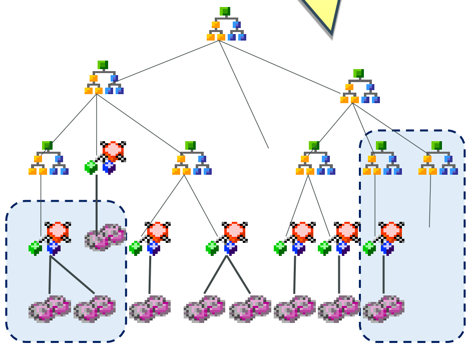
# MODEL BASED SYSTEM ENGINEERING

## ▶ Identify Re-use Candidates

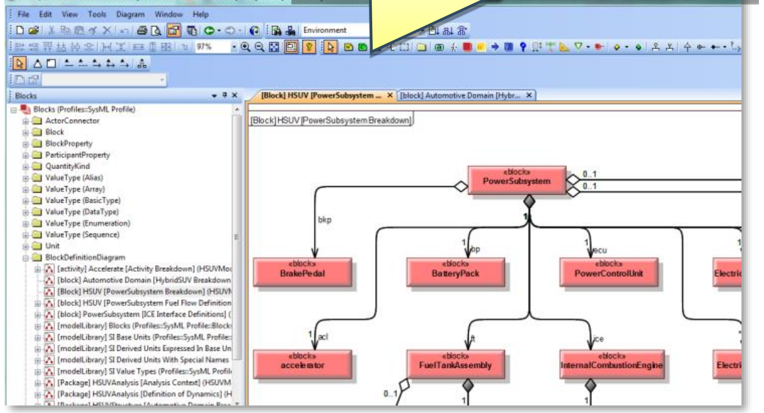


System Engineer
System Engineering CFT
Start Procedure 1
1.1
Define Objectives, Scope and Process
1.2
Establish Cross-functional Team
1.3
Capture Constraints
1.4
Define Quality and Performance Measures and Targets
1.5
Identify Re-use Candidates
1.6
Refine and Confirm Plan
End Procedure 1

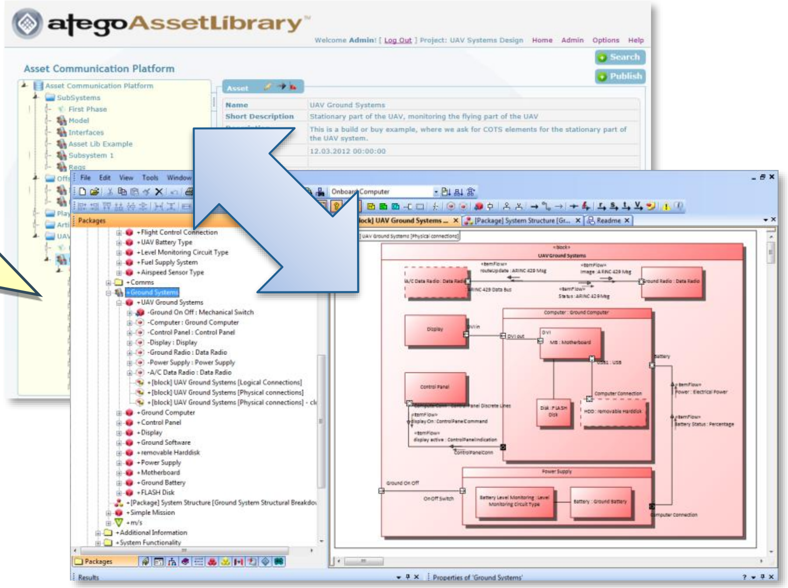
Identify re-use candidates from existing systems or designs and review with CFT



Consider existing SysML models for similar products



HSUV, Version 0 - Artisan Studio - [HSUVModel::HSUVStructure.[Block] HSUV [Po
File Edit View Tools Diagram Window Help
Blocks
[Block] HSUV [PowerSubsystem ...
[block] Automotive Domain [Hybr...
[Block] HSUV [PowerSubsystem Breakdown]
«block» PowerSubsystem
«block» BrakePedal
«block» BatteryPack
«block» PowerControlUnit
«block» accelerator
«block» FuelTankAssembly
«block» InternalCombustionEngine
bkp
bp
ecu
acl
ft
ice

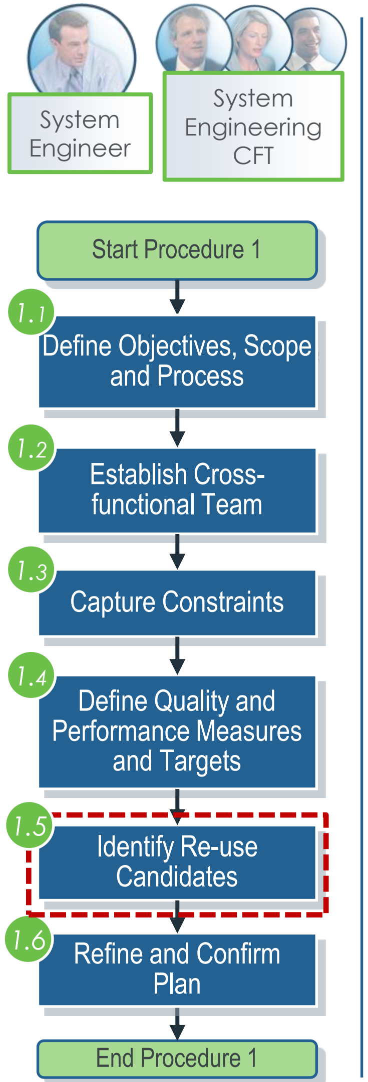


ategoAssetLibrary
Asset Communication Platform
Search
Publish
UAV Ground Systems
Stationary part of the UAV, monitoring the flying part of the UAV
This is a build or buy example, where we ask for COTS elements for the stationary part of the UAV system.
12.03.2012 00:00:00
Packages
UAV Ground Systems

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

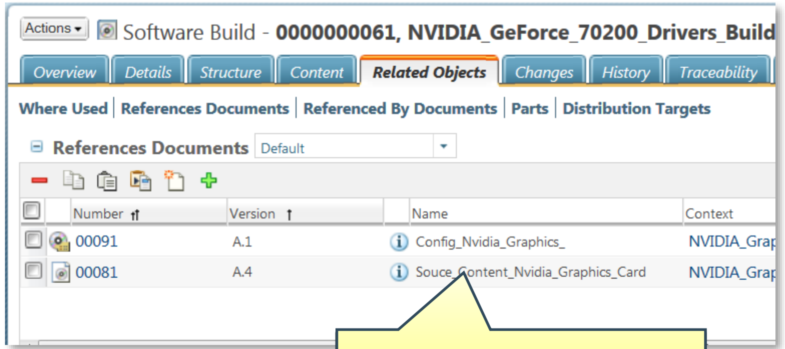
# MODEL BASED SYSTEM ENGINEERING

## ▶ Identify Re-use Candidates



System Engineer
System Engineering CFT
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

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

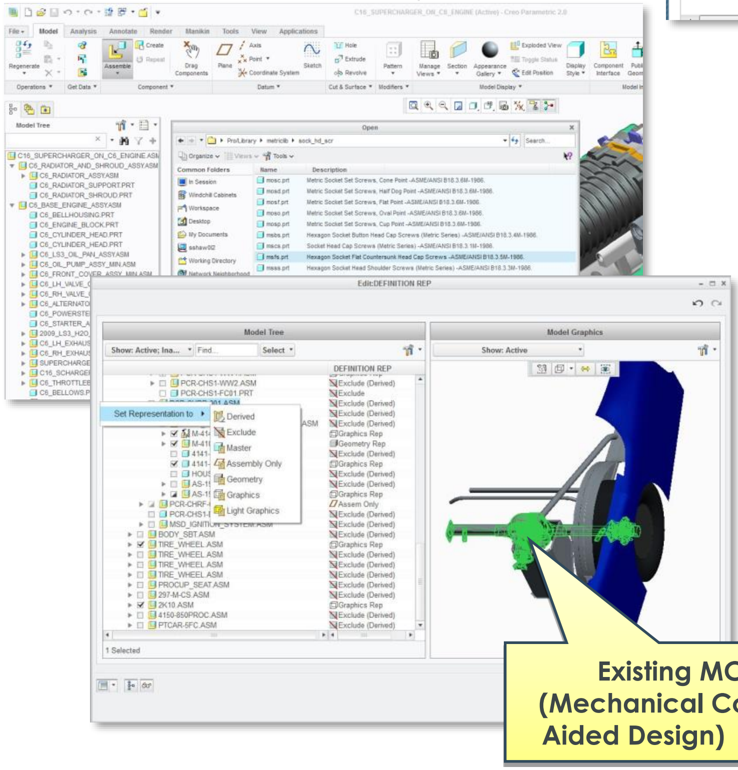


Software Build - 0000000061, NVIDIA_GeForce_70200_Drivers_Build

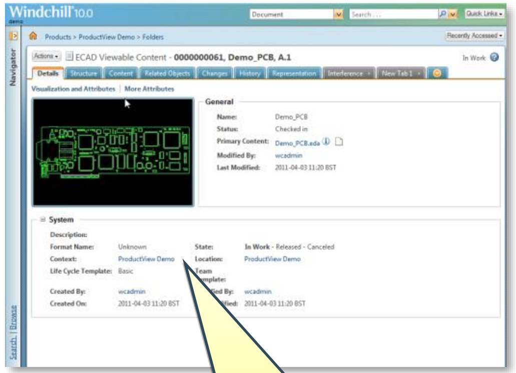
References Documents Default

| Number | Version | Name | Context |
|---|---|---|---|
| 00091 | A.1 | Config_Nvidia_Graphics_ | NVIDIA_Grap |
| 00081 | A.4 | Souce_Content_Nvidia_Graphics_Card | NVIDIA_Grap |

Existing software designs



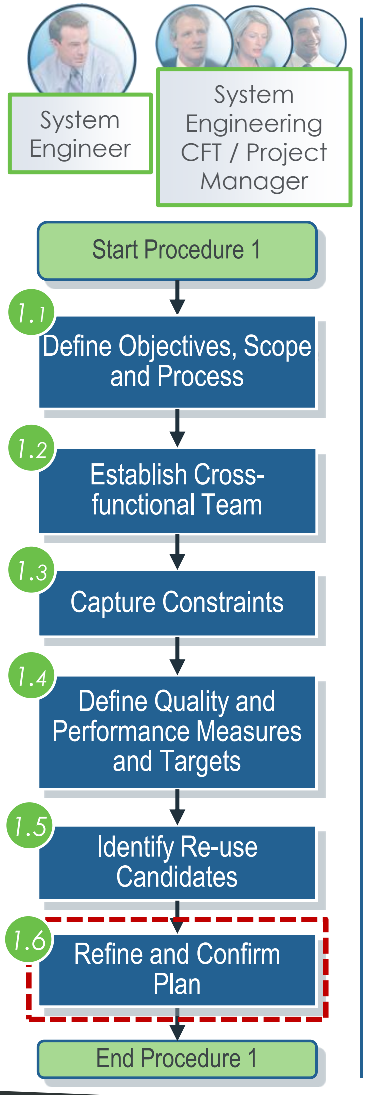
Existing MCAD (Mechanical Computer Aided Design) designs



Existing ECAD (Electrical Computer Aided Design) designs

# MODEL BASED SYSTEM ENGINEERING

## ► Refine and Confirm Plan



System Engineer

System Engineering CFT / Project Manager

Start Procedure 1

1.1 Define Objectives, Scope and Process

1.2 Establish Cross-functional Team

1.3 Capture Constraints

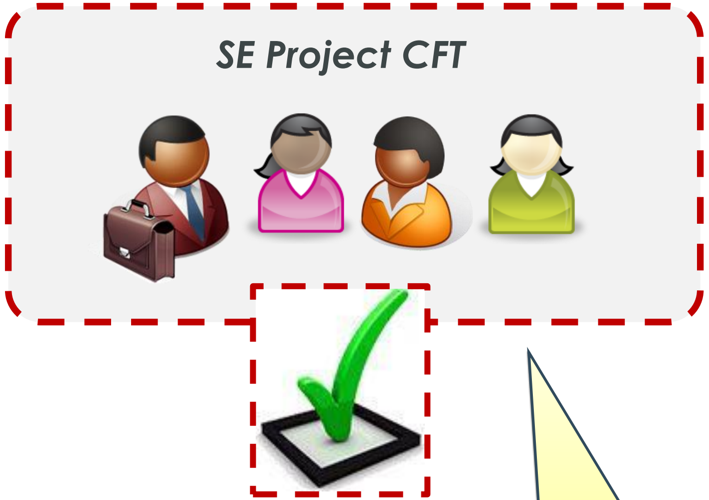
1.4 Define Quality and Performance Measures and Targets

1.5 Identify Re-use Candidates

1.6 Refine and Confirm Plan

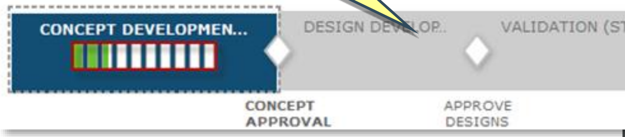
End Procedure 1

**If needed update plan objectives, scope, process, constraints, CFT, quality and performance targets and re-use candidates**



**Review and confirm plan with CFT**

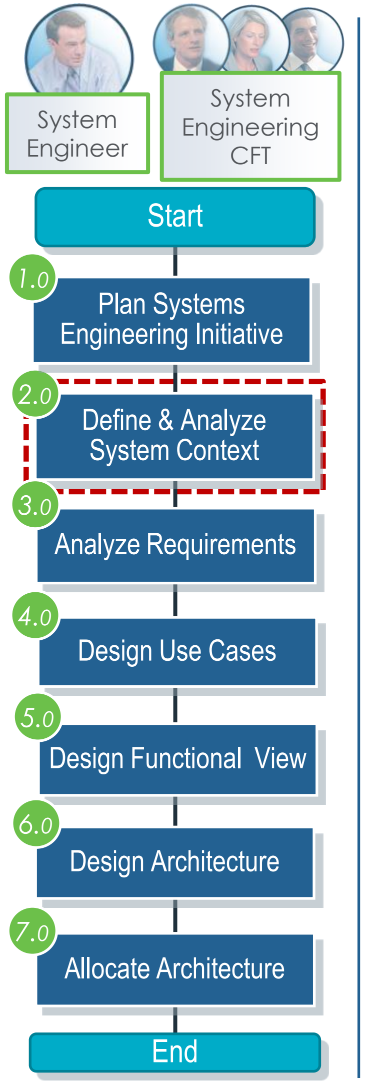
**Kick off and manage project**



# DEFINE AND ANALYZE SYSTEM CONTEXT

# MODEL BASED SYSTEM ENGINEERING BEST PRACTICE PROCEDURE

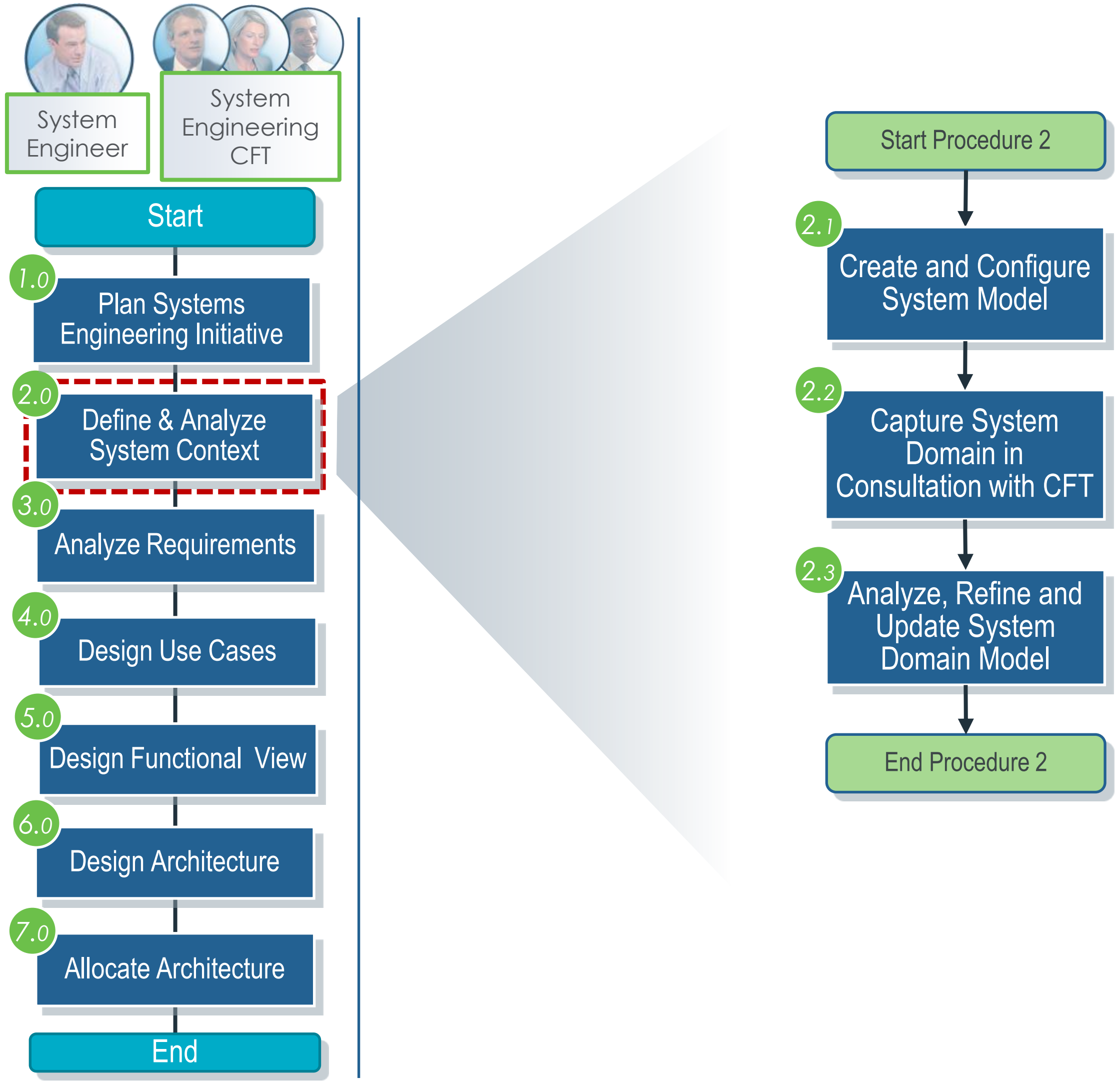
## ► Define and Analyze System Context



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

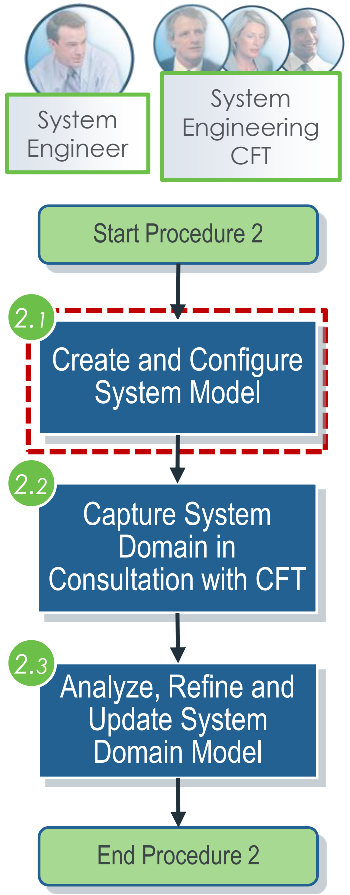
# MODEL BASED SYSTEM ENGINEERING

- Define and Analyze System Context

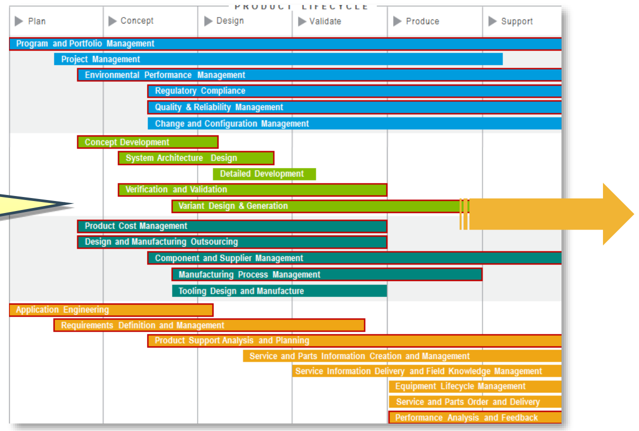


# MODEL BASED SYSTEM ENGINEERING

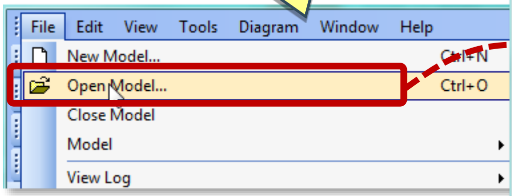
## ▶ Create and Configure System Model


System Engineer
System Engineering CFT
Start Procedure 2
2.1
Create and Configure System Model
2.2
Capture System Domain in Consultation with CFT
2.3
Analyze, Refine and Update System Domain Model
End Procedure 2

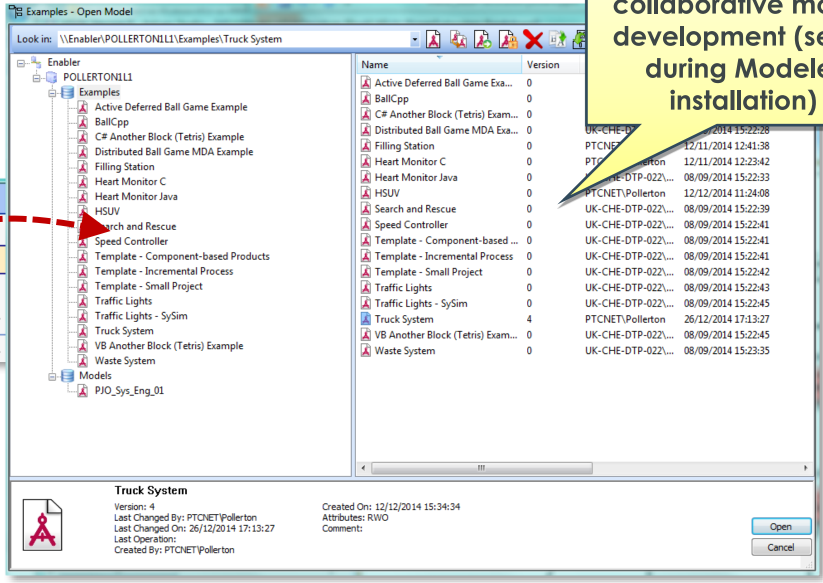
Coordinate with related processes such as Requirements Management, Quality Mgmt and identify any relevant existing information



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



Enabler supports collaborative model development (setup during Modeler installation)



Look in: \\Enabler\POLLERTON1L1\Examples\Truck System

- Enabler
  - POLLERTON1L1
    - Examples
      - Active Deferred Ball Game Example
      - BallCpp
      - C# Another Block (Tetris) Example
      - Distributed Ball Game MDA Example
      - Filling Station
      - Heart Monitor C
      - Heart Monitor Java
      - HSUV
      - arch and Rescue
      - Speed Controller
      - Template - Component-based Products
      - Template - Incremental Process
      - Template - Small Project
      - Traffic Lights
      - Traffic Lights - SySim
      - Truck System
      - VB Another Block (Tetris) Example
      - Waste System
    - Models
      - PJO_Sys_Eng_01

| Name | Version | | |
|---|---|---|---|
| Active Deferred Ball Game Exa... | 0 | | |
| BallCpp | 0 | | |
| C# Another Block (Tetris) Exam... | 0 | | |
| Distributed Ball Game MDA Exa... | 0 | UK-CHE-D | /2014 15:22:28 |
| Filling Station | 0 | PTCNE | 12/11/2014 12:41:38 |
| Heart Monitor C | 0 | PTC erton | 12/11/2014 12:23:42 |
| Heart Monitor Java | 0 | E-DTP-022\... | 08/09/2014 15:22:33 |
| HSUV | 0 | PTCNET\Pollerton | 12/12/2014 11:24:08 |
| Search and Rescue | 0 | UK-CHE-DTP-022\... | 08/09/2014 15:22:39 |
| Speed Controller | 0 | UK-CHE-DTP-022\... | 08/09/2014 15:22:41 |
| Template - Component-based ... | 0 | UK-CHE-DTP-022\... | 08/09/2014 15:22:41 |
| Template - Incremental Process | 0 | UK-CHE-DTP-022\... | 08/09/2014 15:22:41 |
| Template - Small Project | 0 | UK-CHE-DTP-022\... | 08/09/2014 15:22:42 |
| Traffic Lights | 0 | UK-CHE-DTP-022\... | 08/09/2014 15:22:43 |
| 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 |
| Waste System | 0 | UK-CHE-DTP-022\... | 08/09/2014 15:23:35 |

**Truck System**

Version: 4
Last Changed By: PTCNET\Pollerton
Last Changed On: 26/12/2014 17:13:27
Last Operation:
Created By: PTCNET\Pollerton

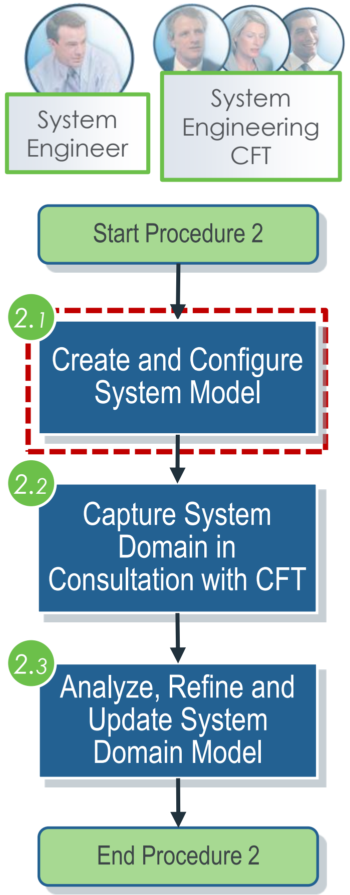
Created On: 12/12/2014 15:34:34
Attributes: RWO
Comment:

**Note**

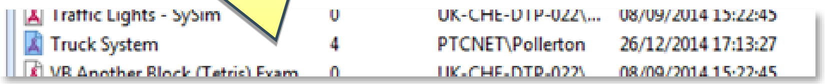
If using Asset Library, this may be a collection of linked models for systems and sub-systems

# MODEL BASED SYSTEM ENGINEERING

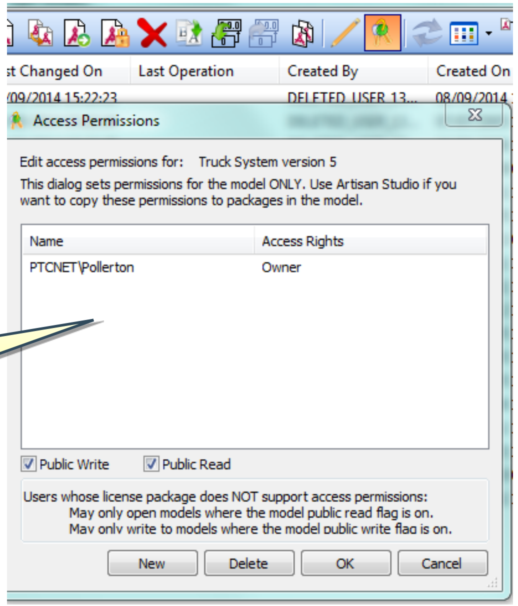
## ▶ Create and Configure System Model


System Engineer
System Engineering CFT
Start Procedure 2
2.1
Create and Configure System Model
2.2
Capture System Domain in Consultation with CFT
2.3
Analyze, Refine and Update System Domain Model
End Procedure 2

**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 |



| st Changed On | Last Operation | Created By | Created On |
|---|---|---|---|
| /09/2014 15:22:23 | | DELETED USER 13... | 08/09/2014 |

Access Permissions

Edit access permissions for: Truck System version 5

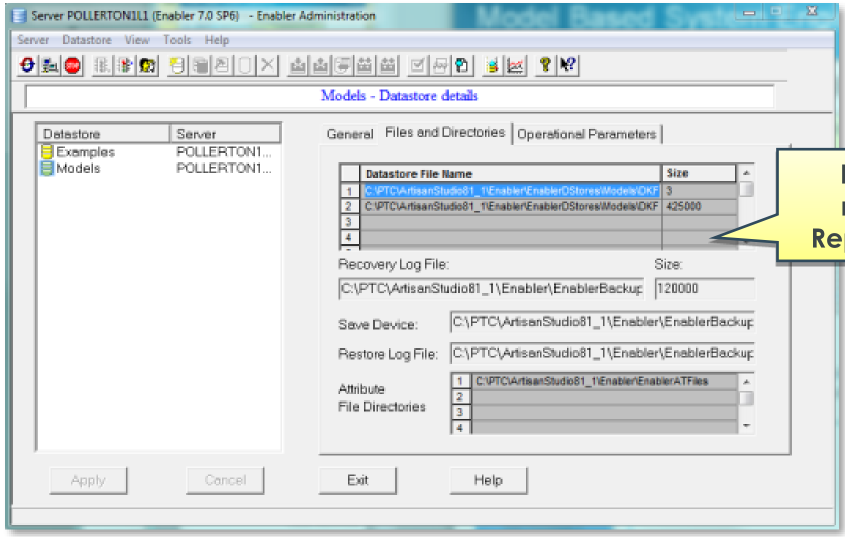
This dialog sets permissions for the model ONLY. Use Artisan Studio if you want to copy these permissions to packages in the model.

| Name | Access Rights |
|---|---|
| PTCNET\Pollerton | Owner |

☑ Public Write ☑ Public Read

Users whose license package does NOT support access permissions:
May only open models where the model public read flag is on.
May only write to models where the model public write flag is on.

**Access permissions on models can be defined**



Server POLLERTON1L1 (Enabler 7.0 SP6) - Enabler Administration

Models - Datastore details

| Datastore | Server |
|---|---|
| Examples | POLLERTON1... |
| Models | POLLERTON1... |

| | Datastore File Name | Size |
|---|---|---|
| 1 | C:\PTC\ArtisanStudio81_1\Enabler\EnablerDStores\Models\DKF | 3 |
| 2 | C:\PTC\ArtisanStudio81_1\Enabler\EnablerDStores\Models\DKF | 425000 |
| 3 | | |
| 4 | | |

Recovery Log File: C:\PTC\ArtisanStudio81_1\Enabler\EnablerBackup Size: 120000

Save Device: C:\PTC\ArtisanStudio81_1\Enabler\EnablerBackup

Restore Log File: C:\PTC\ArtisanStudio81_1\Enabler\EnablerBackup

Attribute File Directories:

| | |
|---|---|
| 1 | C:\PTC\ArtisanStudio81_1\Enabler\EnablerATFiles |
| 2 | |
| 3 | |
| 4 | |

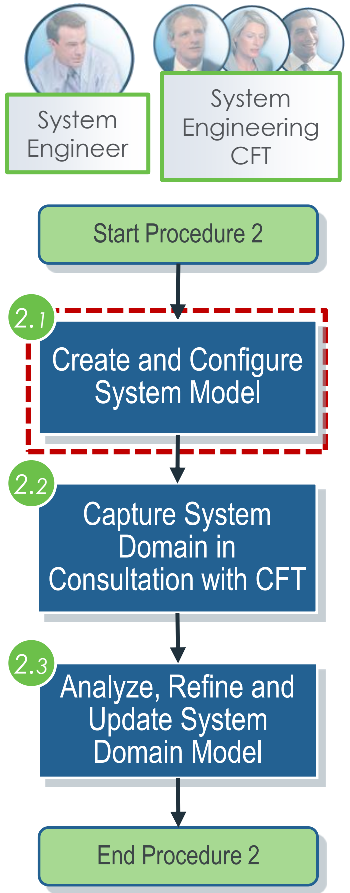
**Repositories can be managed using the Repository Administrator**

**Note**

**Refer to System Model Management Best Practice for more information**

# MODEL BASED SYSTEM ENGINEERING

## ► Create and Configure System Model



System Engineer

System Engineering CFT

Start Procedure 2

2.1 Create and Configure System Model

2.2 Capture System Domain in Consultation with CFT

2.3 Analyze, Refine and Update System Domain Model

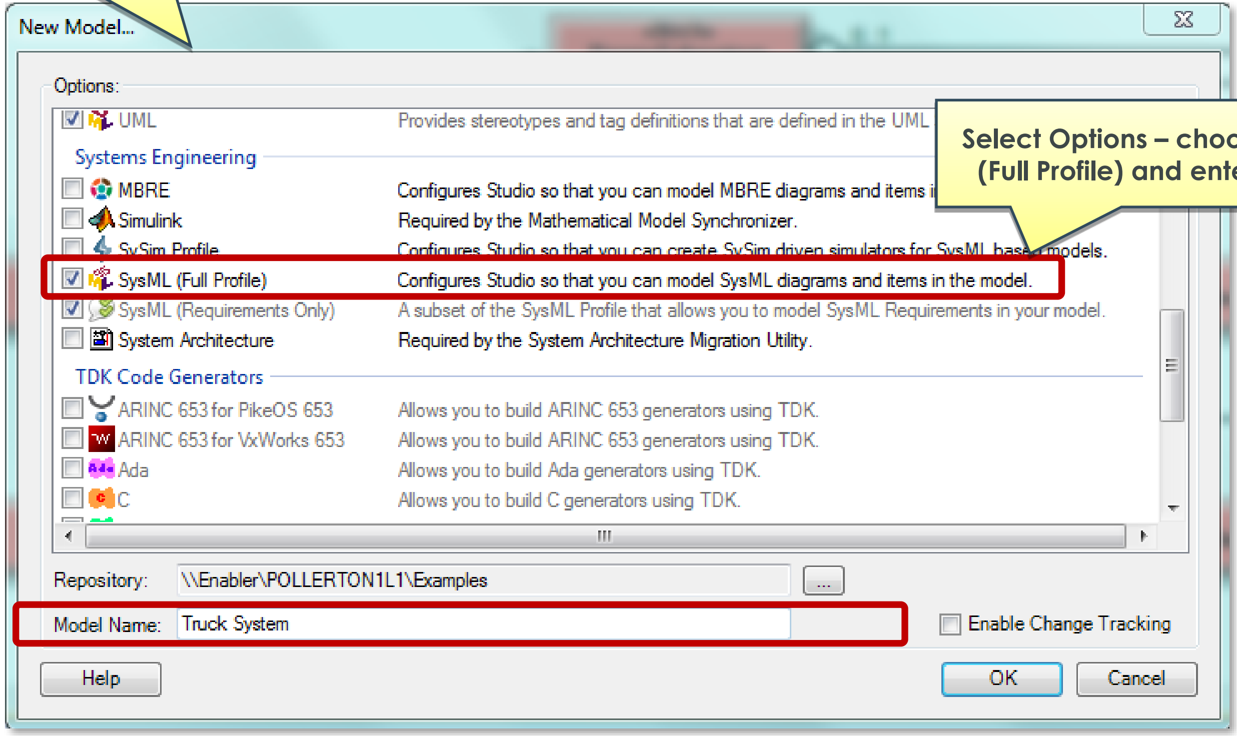
End Procedure 2

**Note**

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

**Select File > New Model to create new system model**

**Select Options – choose SysML (Full Profile) and enter name**



New Model...

Options:

☑ UML — Provides stereotypes and tag definitions that are defined in the UML

Systems Engineering

☐ MBRE — Configures Studio so that you can model MBRE diagrams and items i

☐ Simulink — Required by the Mathematical Model Synchronizer.

☐ SySim Profile — Configures Studio so that you can create SySim driven simulators for SysML base models.

☑ SysML (Full Profile) — Configures Studio so that you can model SysML diagrams and items in the model.

☑ SysML (Requirements Only) — A subset of the SysML Profile that allows you to model SysML Requirements in your model.

☐ System Architecture — Required by the System Architecture Migration Utility.

TDK Code Generators

☐ ARINC 653 for PikeOS 653 — Allows you to build ARINC 653 generators using TDK.

☐ ARINC 653 for VxWorks 653 — Allows you to build ARINC 653 generators using TDK.

☐ Ada — Allows you to build Ada generators using TDK.

☐ C — Allows you to build C generators using TDK.

Repository: \\Enabler\POLLERTON1L1\Examples

Model Name: Truck System

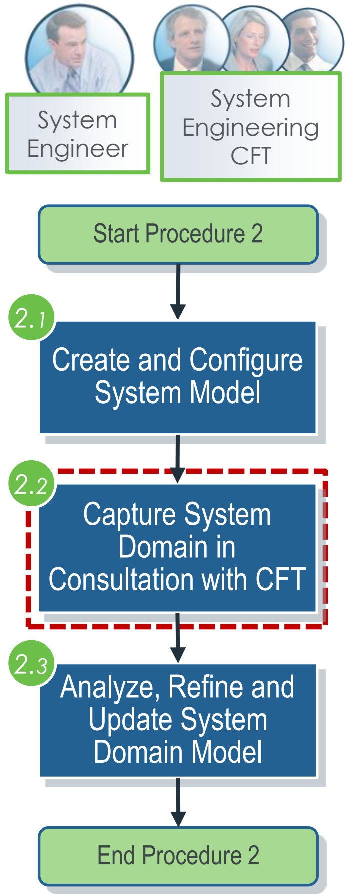
☐ Enable Change Tracking

**Note**

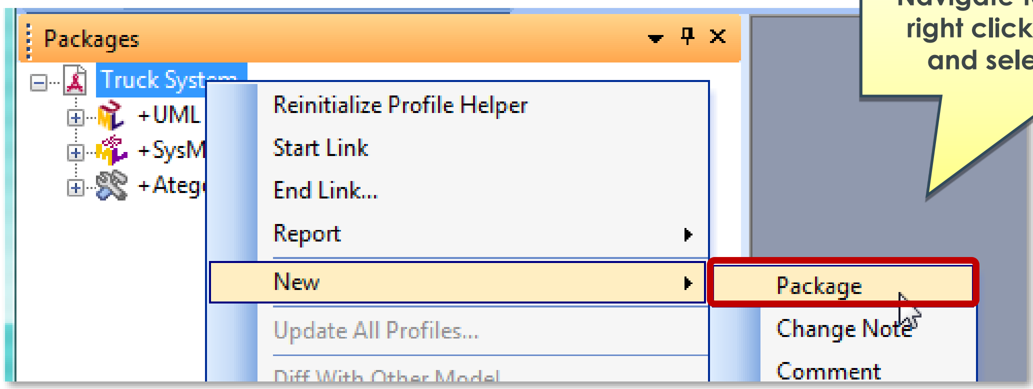
**Profile extensions can be used to support a customer specific data model**

# MODEL BASED SYSTEM ENGINEERING

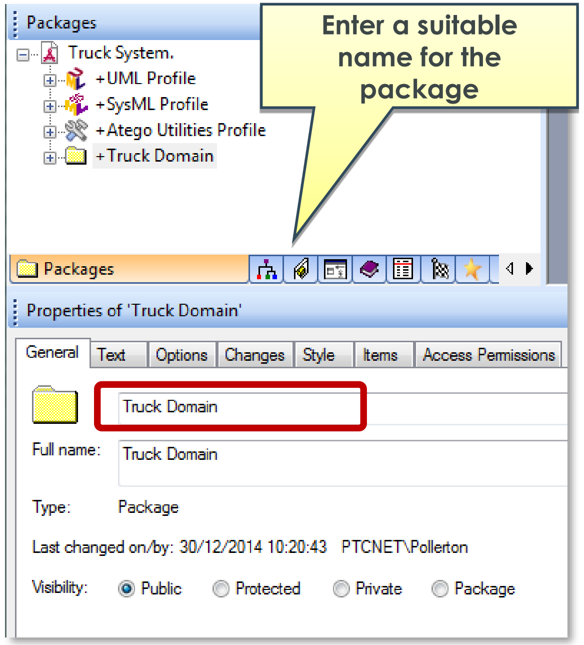
## ▶ Capture System Domain in Consultation with CFT



System Engineer
System Engineering CFT
Start Procedure 2
2.1
Create and Configure System Model
2.2
Capture System Domain in Consultation with CFT
2.3
Analyze, Refine and Update System Domain Model
End Procedure 2



Navigate to the Packages view, right click on the Truck System and select New > Package



Truck Domain

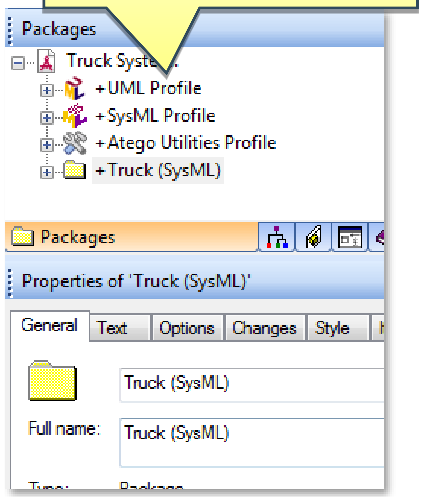
Full name: Truck Domain

Type: Package

Last changed on/by: 30/12/2014 10:20:43 PTCNET\Pollerton

Visibility: ● Public ○ Protected ○ Private ○ Package

Enter a suitable name for the package

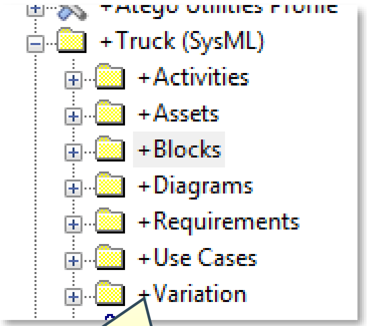


Truck (SysML)

Full name: Truck (SysML)

Type: Package

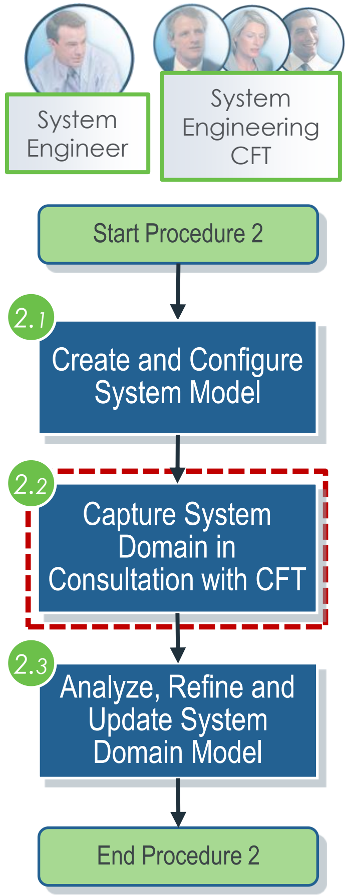
Can be renamed easily by selecting the item and changing the Name field value



Additional packages can be created to structure and organize model elements

# MODEL BASED SYSTEM ENGINEERING

## ▶ Capture System Domain in Consultation with CFT



System Engineer

System Engineering CFT

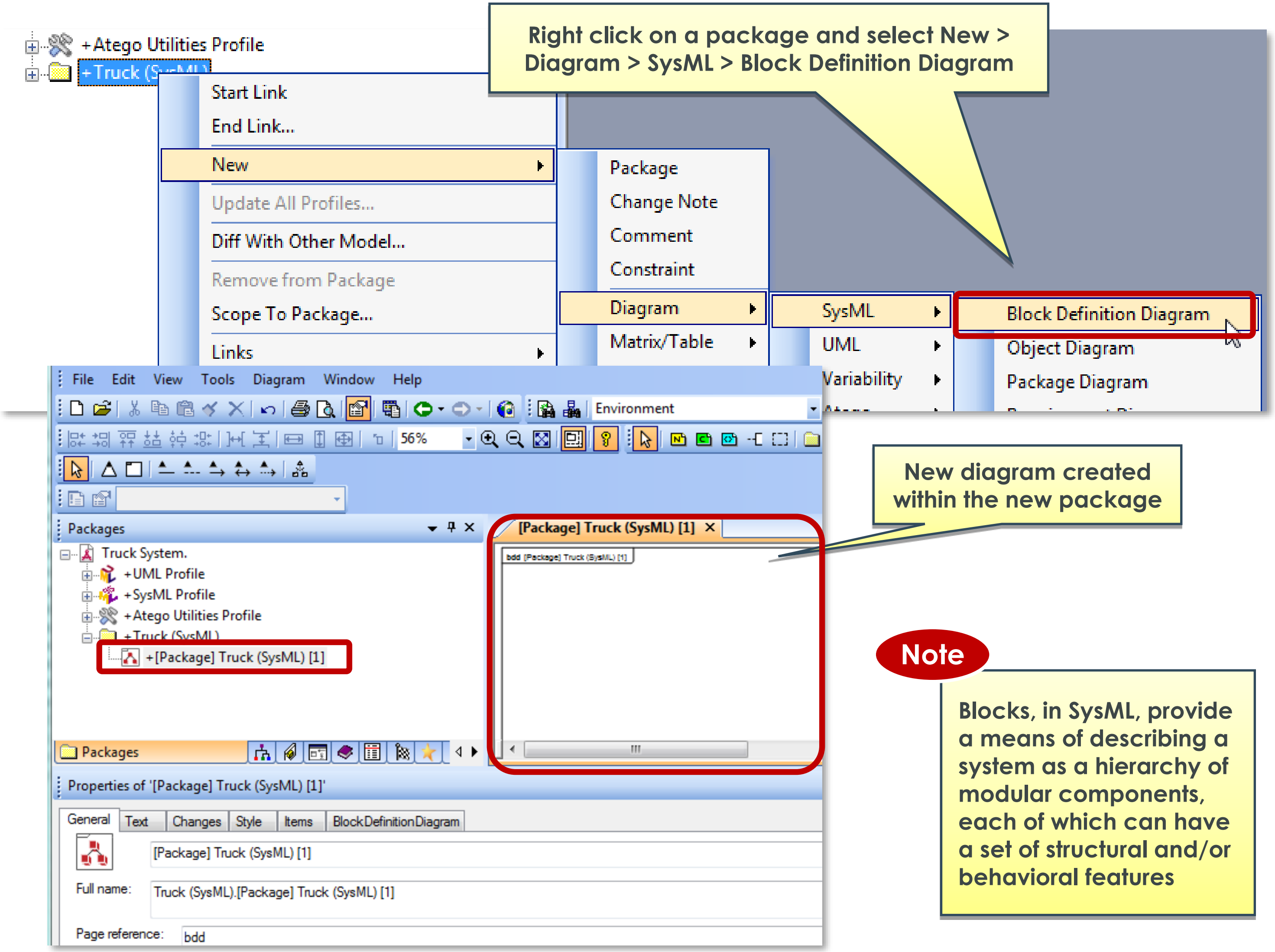
Start Procedure 2

2.1 Create and Configure System Model

2.2 Capture System Domain in Consultation with CFT

2.3 Analyze, Refine and Update System Domain Model

End Procedure 2



Properties of '[Package] Truck (SysML) [1]'

[Package] Truck (SysML) [1]

Full name: Truck (SysML).[Package] Truck (SysML) [1]

Page reference: bdd

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

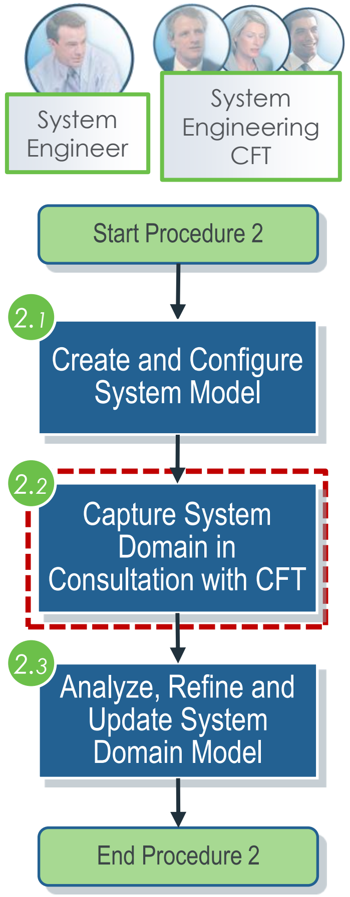
New diagram created within the new package

**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

# MODEL BASED SYSTEM ENGINEERING

## ► Capture System Domain in Consultation with CFT



System Engineer

System Engineering CFT

Start Procedure 2

2.1 Create and Configure System Model

2.2 Capture System Domain in Consultation with CFT

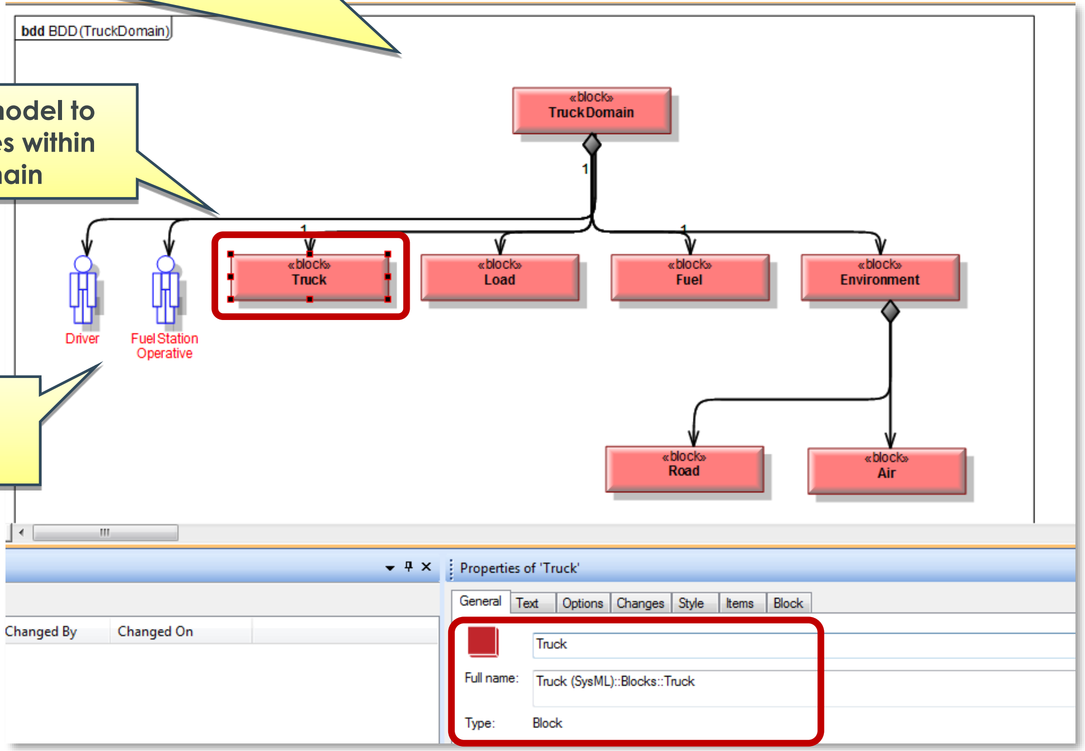
2.3 Analyze, Refine and Update System Domain Model

End Procedure 2

Capture all physical entities involved in the system such as users, operators as well as non-human entities such as equipment, facilities and technology for the user interface

Add Blocks to the model to represent the entities within the system domain

Define Actors that interact with the system

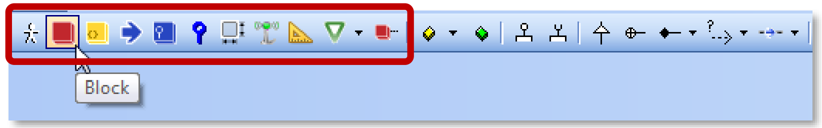


Changed By | Changed On

Truck

Full name: Truck (SysML)::Blocks::Truck

Type: Block

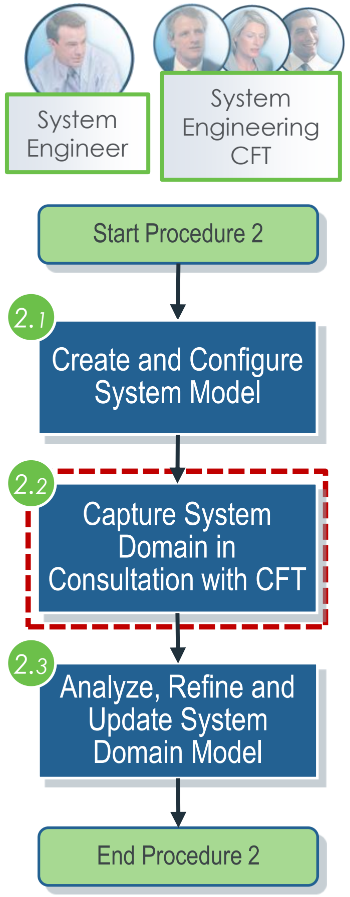


**Note**

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

# MODEL BASED SYSTEM ENGINEERING

## ▶ Capture System Domain in Consultation with CFT



System Engineer

System Engineering CFT

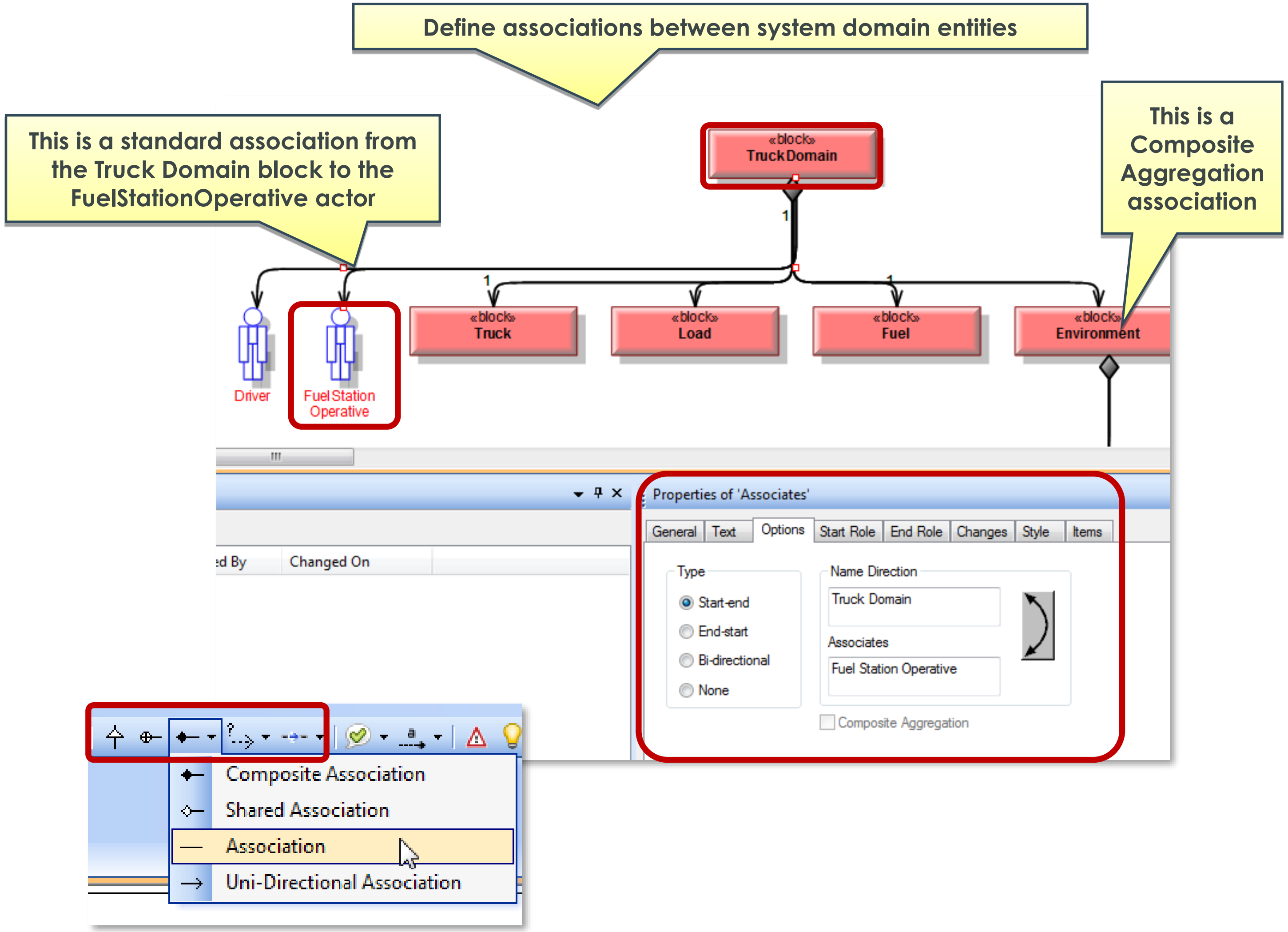
Start Procedure 2

2.1 Create and Configure System Model

2.2 Capture System Domain in Consultation with CFT

2.3 Analyze, Refine and Update System Domain Model

End Procedure 2



Define associations between system domain entities

This is a standard association from the Truck Domain block to the FuelStationOperative actor

This is a Composite Aggregation association

Properties of 'Associates'

General | Text | Options | Start Role | End Role | Changes | Style | Items

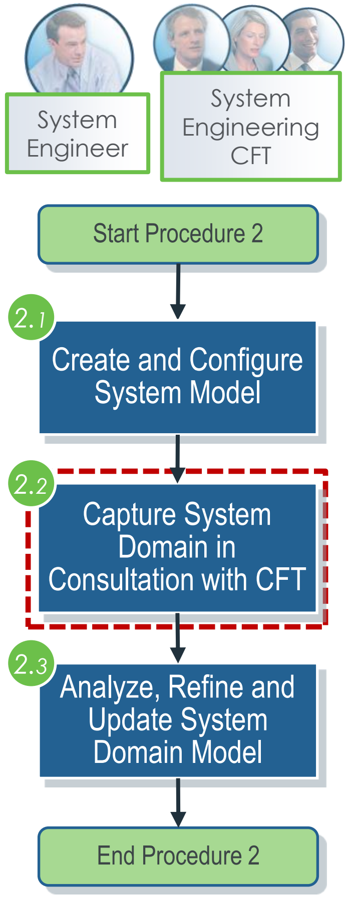
Type
● Start-end
○ End-start
○ Bi-directional
○ None

Name Direction
Truck Domain
Associates
Fuel Station Operative

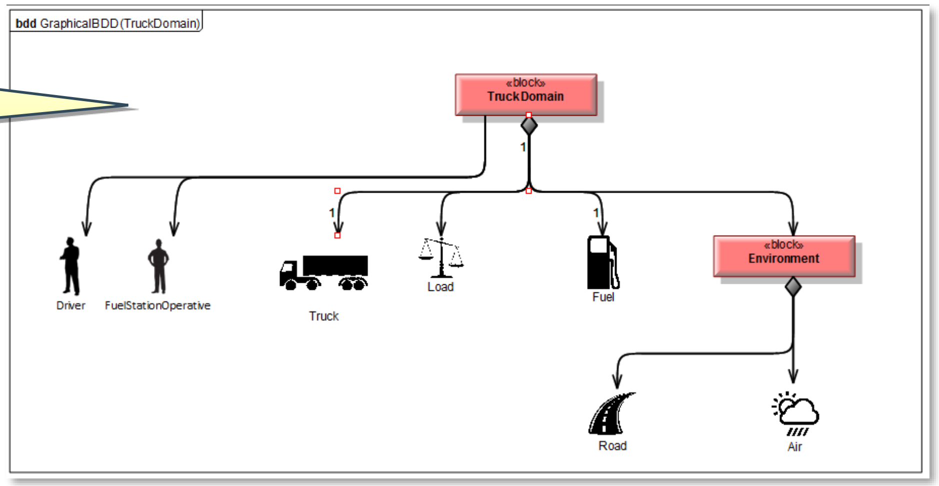
□ Composite Aggregation

# MODEL BASED SYSTEM ENGINEERING

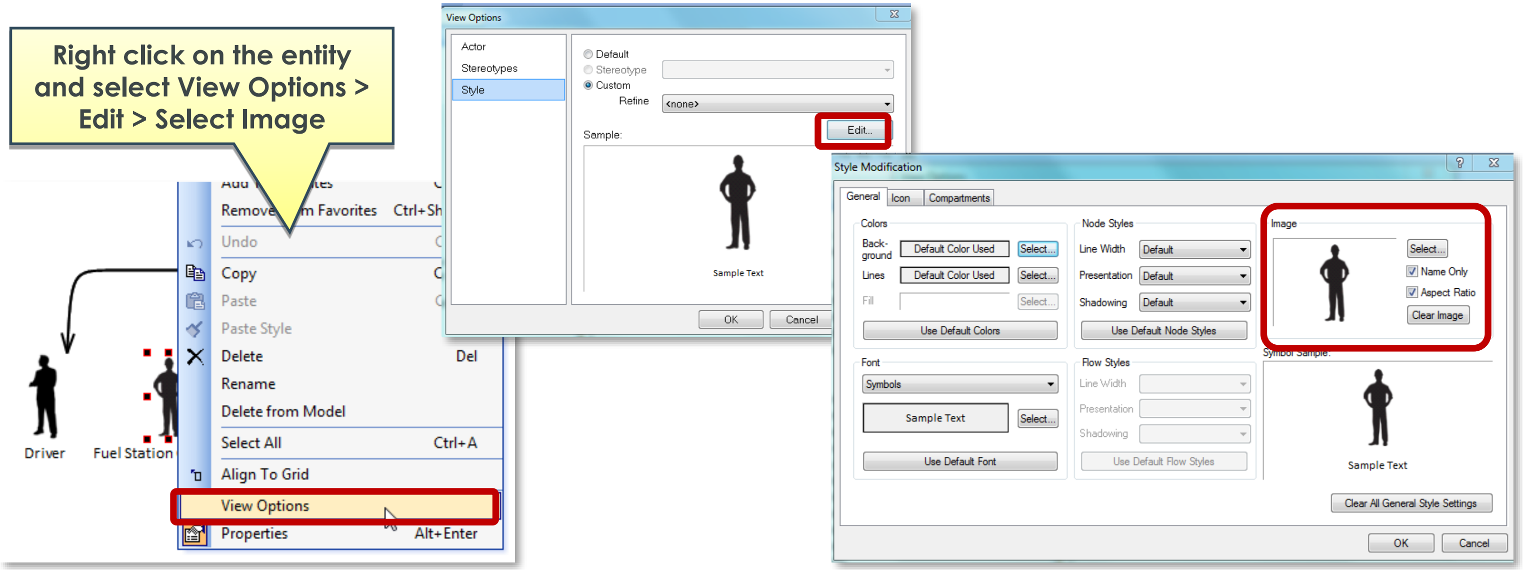
## ► Capture System Domain in Consultation with CFT


System Engineer
System Engineering CFT
Start Procedure 2
2.1
Create and Configure System Model
2.2
Capture System Domain in Consultation with CFT
2.3
Analyze, Refine and Update System Domain Model
End Procedure 2

**Representative images can be used for diagram entities**



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



○ Default ○ Stereotype ● Custom; Refine: <none>

Colors: Back-ground: Default Color Used; Lines: Default Color Used; Fill

Node Styles: Line Width: Default; Presentation: Default; Shadowing: Default

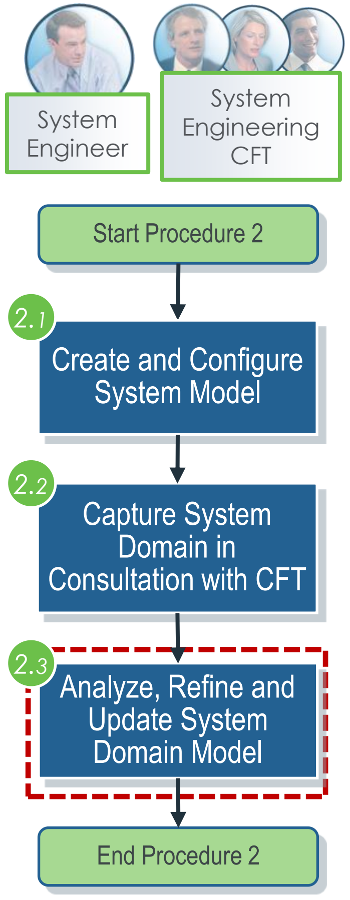
Font: Symbols; Sample Text

Flow Styles: Line Width; Presentation; Shadowing

Image: ☑ Name Only ☑ Aspect Ratio

# MODEL BASED SYSTEM ENGINEERING

## ► Analyze, Refine and Update System Model


System Engineer

System Engineering CFT

Start Procedure 2

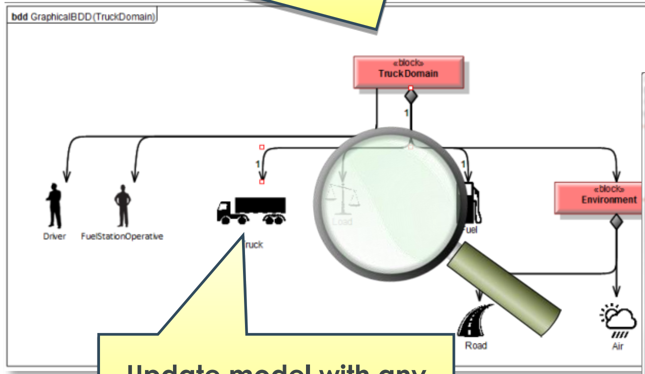
2.1 Create and Configure System Model

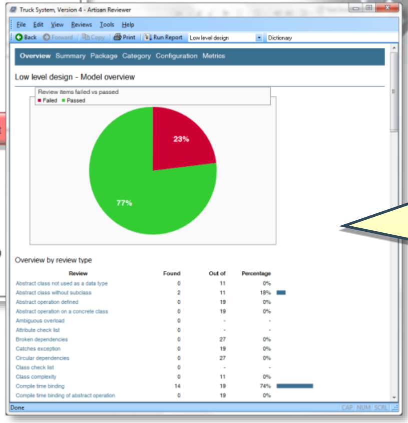
2.2 Capture System Domain in Consultation with CFT

2.3 Analyze, Refine and Update System Domain Model

End Procedure 2

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




Truck System, Version 4 - Artisan Reviewer

File Edit View Reviews Tools Help
Back Forward Copy Print Run Report
Low level design
Dictionary
Overview Summary Package Category Configuration Metrics

Low level design - Model overview

Review items failed vs passed
Failed
Passed
23%
77%

Overview by review type

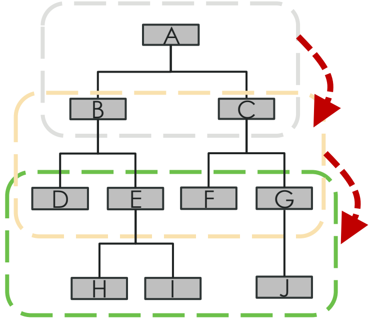
| Review | Found | Out of | Percentage |
|---|---|---|---|
| Abstract class not used as a data type | 0 | 11 | 0% |
| Abstract class without subclass | 2 | 11 | 18% |
| Abstract operation defined | 0 | 19 | 0% |
| Abstract operation on a concrete class | 0 | 19 | 0% |
| Ambiguous overload | 0 | - | - |
| Attribute check list | 0 | - | - |
| Broken dependencies | 0 | 27 | 0% |
| Catches exception | 0 | 19 | 0% |
| Circular dependencies | 0 | 27 | 0% |
| Class check list | 0 | - | - |
| Class complexity | 0 | 11 | 0% |
| Compile time binding | 14 | 19 | 74% |
| Compile time binding of abstract operation | 0 | 19 | 0% |

Update model with any changes as needed

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

**Note**

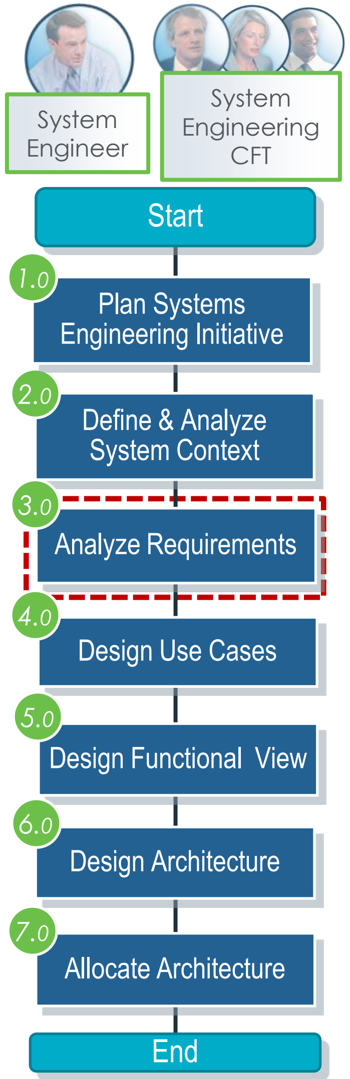
There may be incomplete information available at this point so the creation of block and internal block diagrams should occur iteratively and recursively as more system and sub-system requirements are identified. Further iterations occur in the following procedures



# ANALYZE REQUIREMENTS

# MODEL BASED SYSTEM ENGINEERING BEST PRACTICE PROCEDURE

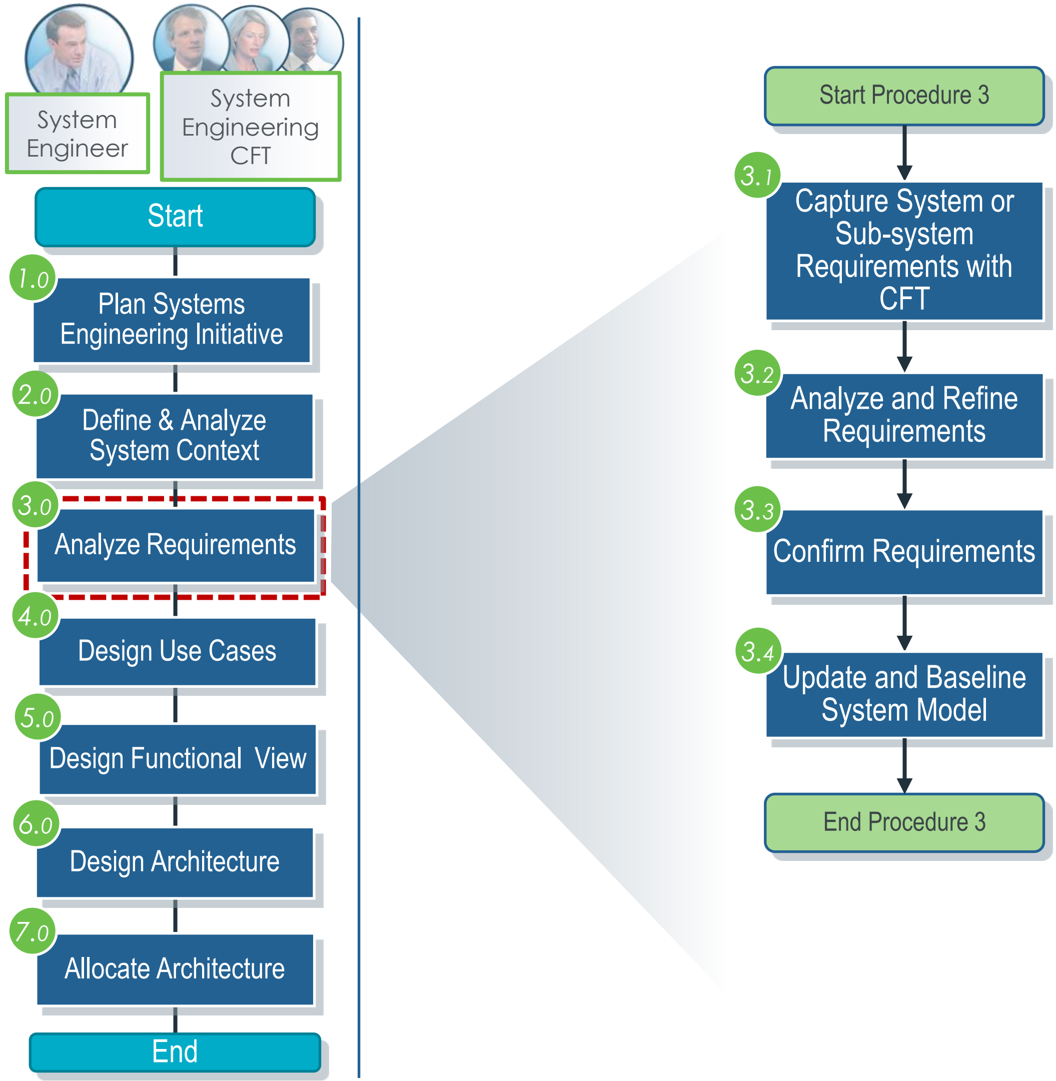
## ▶ Analyze Requirements



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

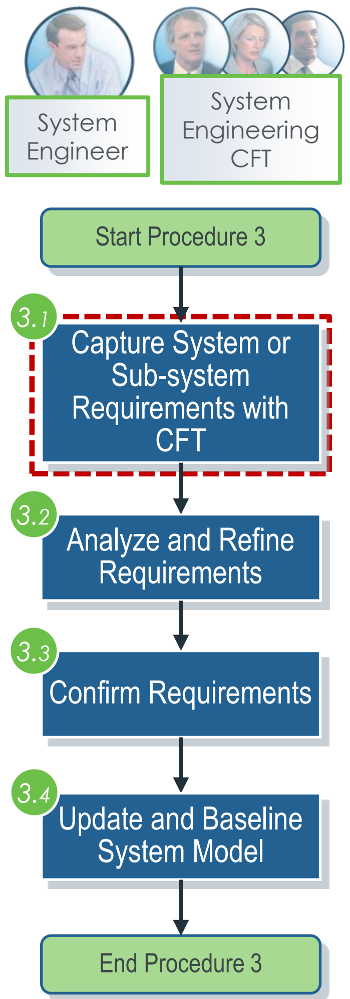
# MODEL BASED SYSTEM ENGINEERING

## ▶ Analyze Requirements

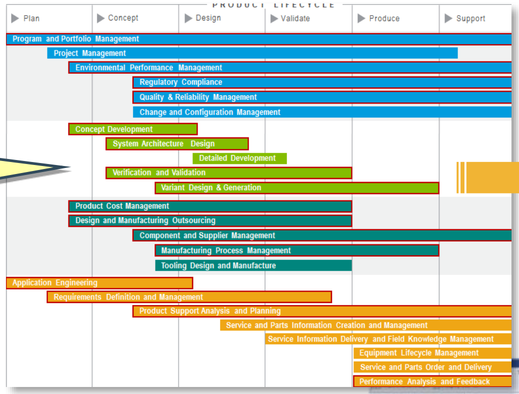


# MODEL BASED SYSTEM ENGINEERING

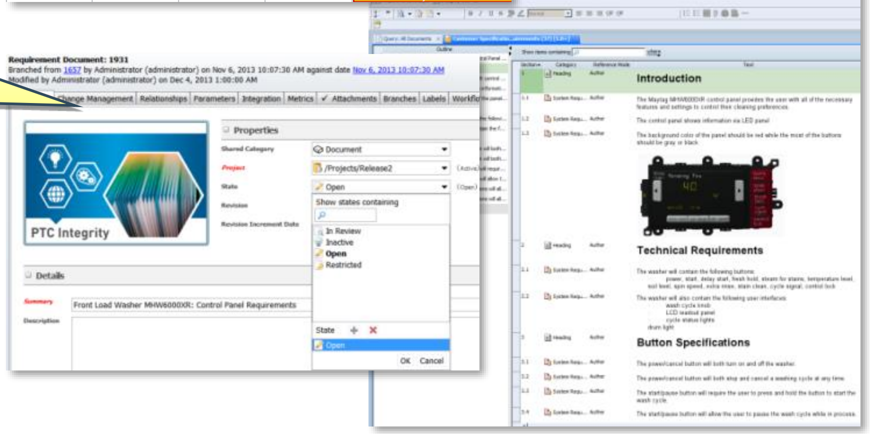
- ▶ Capture System or Sub-system Requirements with CFT


System Engineer
System Engineering 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

**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**



Requirement Document: 1931
Branched from 1657 by Administrator (administrator) on Nov 6, 2013 10:07:30 AM against date Nov 6, 2013 10:07:30 AM
Modified by Administrator (administrator) on Dec 4, 2013 1:00:00 AM

Properties

Shared Category: Document
Project: /Projects/Release2 (Active)
State: Open (Open)
Revision:
Revision Increment Date:

Show states containing
In Review
Inactive
Open
Restricted
State: Open

Details

Summary: Front Load Washer MHW6000XR: Control Panel Requirements
Description:

Introduction

The Maytag MHW6000XR control panel provides the user with all of the necessary features and settings to control their cleaning preferences.

The control panel shows information via LED panel.

The background color of the panel should be red while the most of the buttons should be gray or black.

Technical Requirements

The washer will contain the following buttons: power, start, delay start, fresh hold, steam for stains, temperature level, soil level, spin speed, extra rinse, stain clean, cycle signal, control lock

The washer will also contain the following user interfaces: wash cycle knob, LCD readout panel, cycle status lights, drum light

Button Specifications

The power/cancel button will both turn on and off the washer.

The power/cancel button will both stop and cancel a washing cycle at any time.

The start/pause button will require the user to press and hold the button to start the wash cycle.

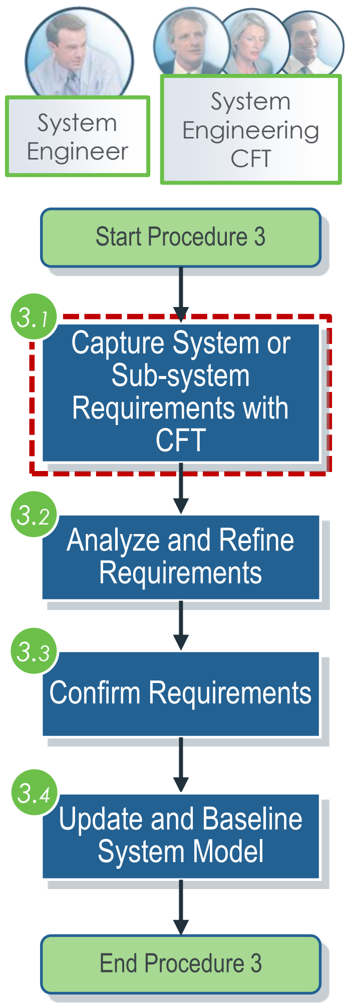
The start/pause button will allow the user to pause the wash cycle while in process.

**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**

# MODEL BASED SYSTEM ENGINEERING

- Capture System or Sub-system Requirements with CFT



System Engineer

System Engineering CFT

Start Procedure 3

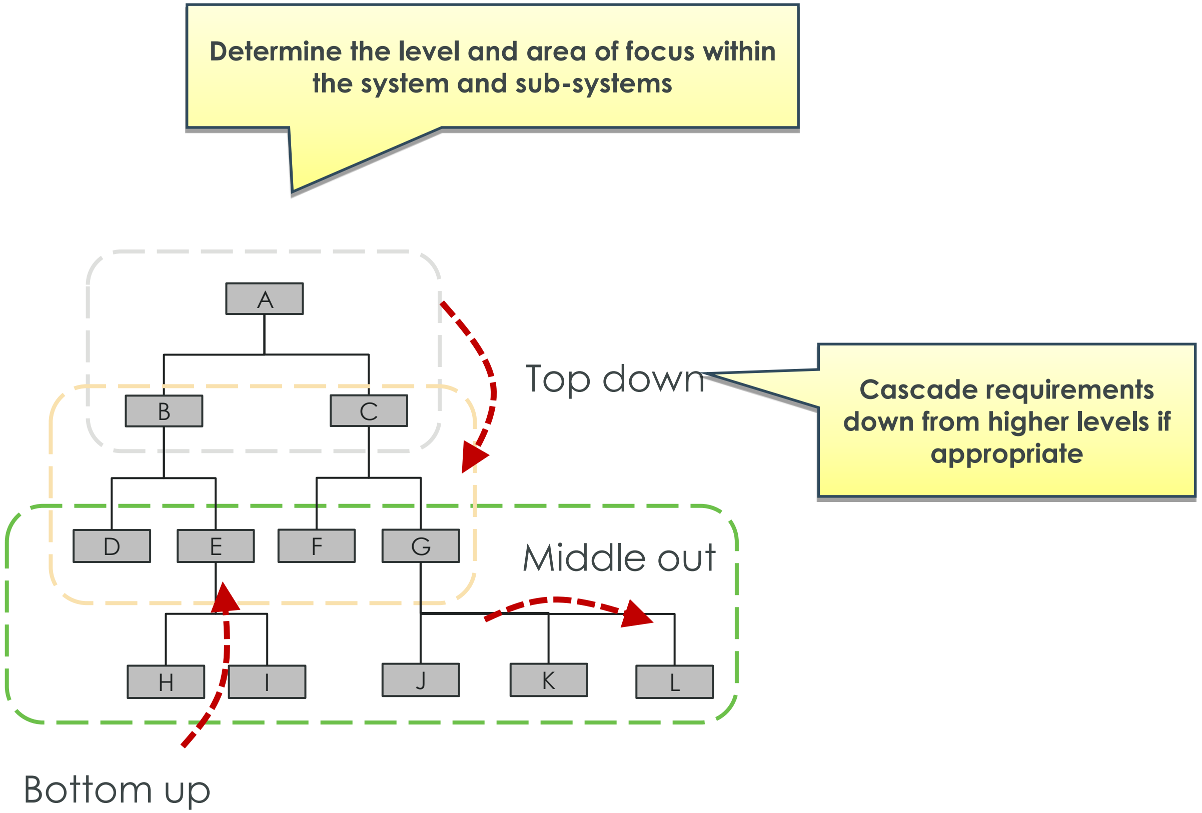
3.1 Capture System or Sub-system Requirements with CFT

3.2 Analyze and Refine Requirements

3.3 Confirm Requirements

3.4 Update and Baseline System Model

End Procedure 3

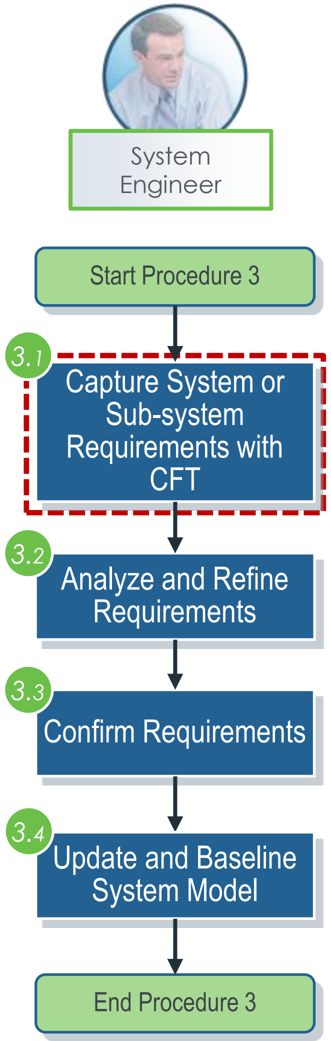


Determine the level and area of focus within the system and sub-systems

Cascade requirements down from higher levels if appropriate

# MODEL BASED SYSTEM ENGINEERING

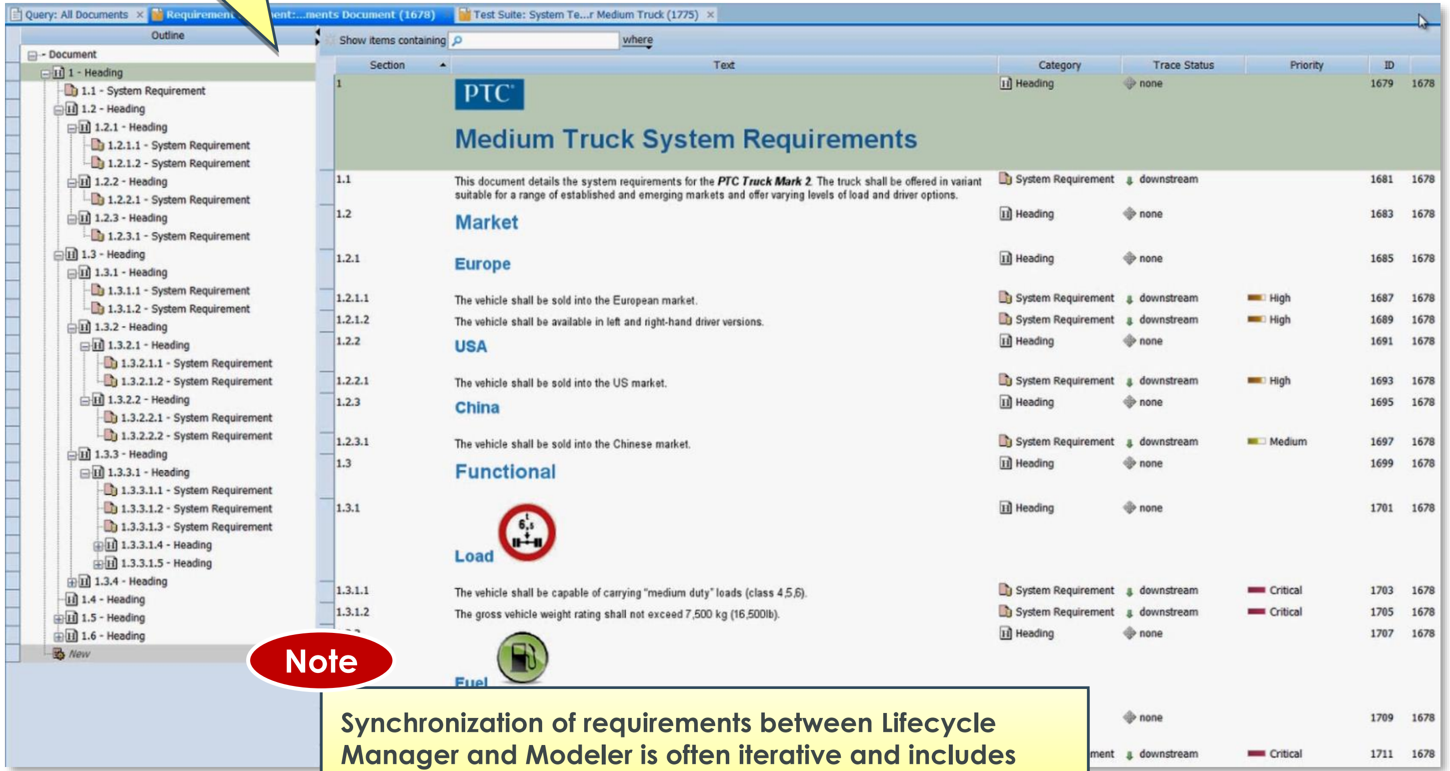
## ► Capture System or Sub-system Requirements with CFT


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*


Query: All Documents
Requirement...ent:...ments Document (1678)
Test Suite: System Te...r Medium Truck (1775)
Outline
Show items containing
where

- Document
  - 1 - Heading
    - 1.1 - System Requirement
    - 1.2 - Heading
      - 1.2.1 - Heading
        - 1.2.1.1 - System Requirement
        - 1.2.1.2 - System Requirement
      - 1.2.2 - Heading
        - 1.2.2.1 - System Requirement
      - 1.2.3 - Heading
        - 1.2.3.1 - System Requirement
    - 1.3 - Heading
      - 1.3.1 - Heading
        - 1.3.1.1 - System Requirement
        - 1.3.1.2 - System Requirement
      - 1.3.2 - Heading
        - 1.3.2.1 - Heading
          - 1.3.2.1.1 - System Requirement
          - 1.3.2.1.2 - System Requirement
        - 1.3.2.2 - Heading
          - 1.3.2.2.1 - System Requirement
          - 1.3.2.2.2 - System Requirement
      - 1.3.3 - Heading
        - 1.3.3.1 - Heading
          - 1.3.3.1.1 - System Requirement
          - 1.3.3.1.2 - System Requirement
          - 1.3.3.1.3 - System Requirement
          - 1.3.3.1.4 - Heading
          - 1.3.3.1.5 - Heading
      - 1.3.4 - Heading
    - 1.4 - Heading
    - 1.5 - Heading
    - 1.6 - Heading
  - *New*

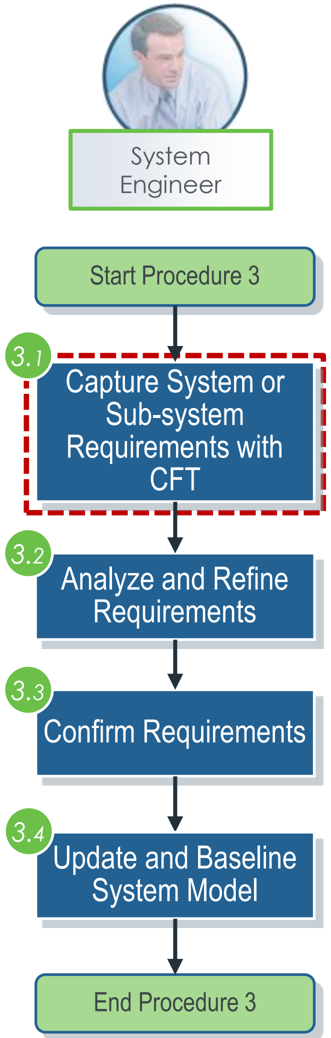
| Section | Text | Category | Trace Status | Priority | ID | |
|---|---|---|---|---|---|---|
| 1 | PTC<br>Medium Truck System Requirements | Heading | none | | 1679 | 1678 |
| 1.1 | This document details the system requirements for the ***PTC Truck Mark 2***. The truck shall be offered in variant suitable 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 |
| | Fuel | Heading | none | | 1707 | 1678 |
| | | | none | | 1709 | 1678 |
| | | ment | downstream | Critical | 1711 | 1678 |

**Note**

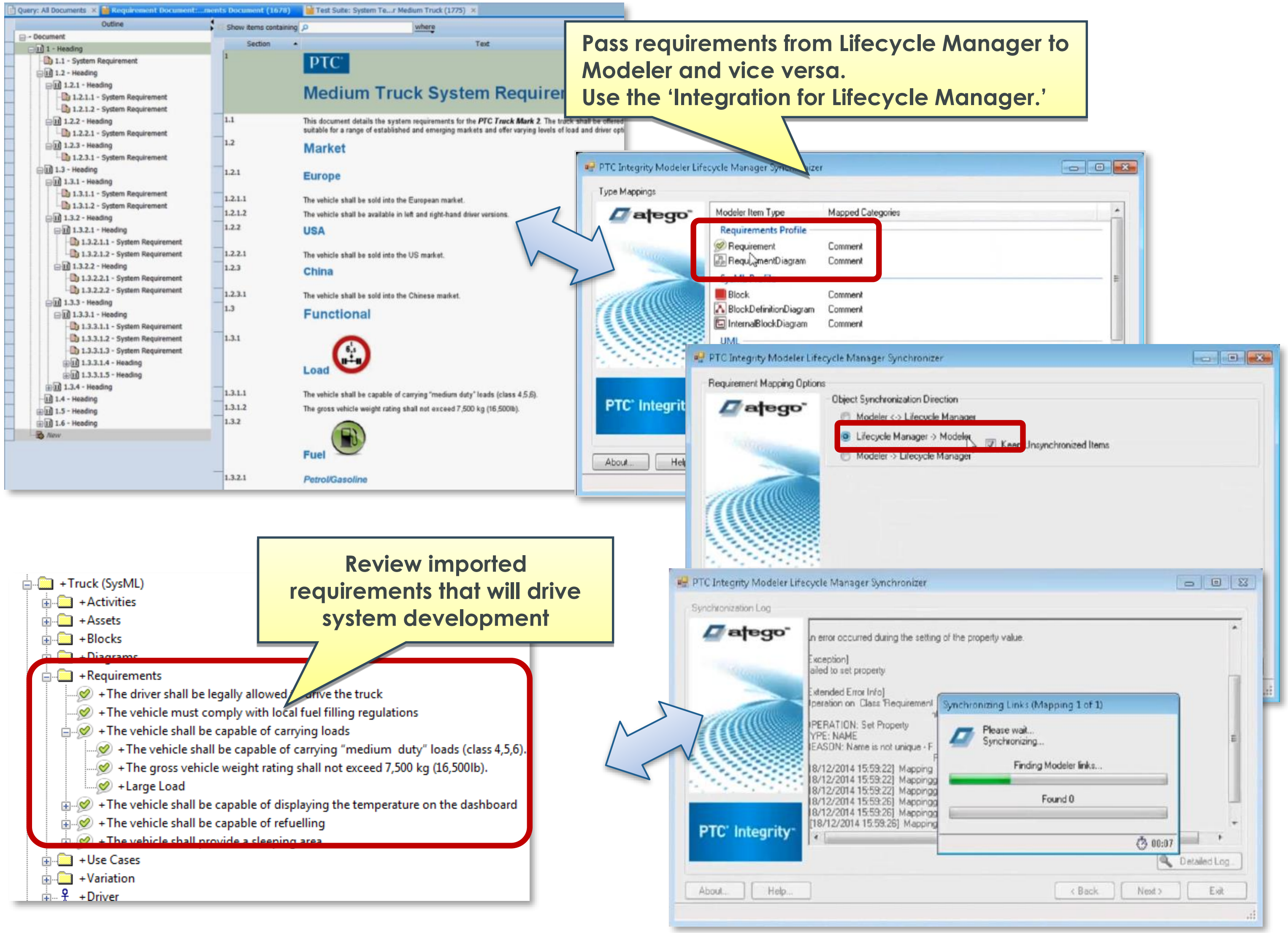
Synchronization of requirements between Lifecycle Manager and Modeler is often iterative and includes additions, deletions and modifications

# MODEL BASED SYSTEM ENGINEERING

▶ 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

Query: All Documents
Requirement Document:...ments Document (1678)
Test Suite: System Te...r Medium Truck (1775)
Outline
Show items containing
where
Section
Text
PTC

| Section | Text |
|---|---|
| 1 | Medium Truck System Requirer |
| 1.1 | This document details the system requirements for the *PTC Truck Mark 2*. The truck shall be offered suitable for a range of established and emerging markets and offer varying levels of load and driver opt |
| 1.2 | Market |
| 1.2.1 | Europe |
| 1.2.1.1 | The vehicle shall be sold into the European market. |
| 1.2.1.2 | The vehicle shall be available in left and right-hand driver versions. |
| 1.2.2 | USA |
| 1.2.2.1 | The vehicle shall be sold into the US market. |
| 1.2.3 | China |
| 1.2.3.1 | The vehicle shall be sold into the Chinese market. |
| 1.3 | Functional |
| 1.3.1 | Load |
| 1.3.1.1 | The vehicle shall be capable of carrying "medium duty" loads (class 4,5,6). |
| 1.3.1.2 | The gross vehicle weight rating shall not exceed 7,500 kg (16,500lb). |
| 1.3.2 | Fuel |
| 1.3.2.1 | *Petrol/Gasoline* |

PTC Integrity Modeler Lifecycle Manager Synchronizer

Type Mappings

| Modeler Item Type | Mapped Categories |
|---|---|
| Requirements Profile | |
| Requirement | Comment |
| RequirementDiagram | Comment |
| Block | Comment |
| BlockDefinitionDiagram | Comment |
| InternalBlockDiagram | Comment |

PTC Integrity Modeler Lifecycle Manager Synchronizer

Requirement Mapping Options

Object Synchronization Direction

○ Modeler <-> Lifecycle Manager
● Lifecycle Manager -> Modeler ☑ Keep Unsynchronized Items
○ Modeler -> Lifecycle Manager

PTC Integrity Modeler Lifecycle Manager Synchronizer

Synchronization Log

```
n error occurred during the setting of the property value.

xception]
ailed to set property

xtended Error Info]
peration on  Class 'Requirement
PERATION: Set Property
YPE: NAME
EASON: Name is not unique - F
8/12/2014 15:59:22]  Mapping
8/12/2014 15:59:22]  Mappingg
8/12/2014 15:59:22]  Mappingg
8/12/2014 15:59:26]  Mappingg
8/12/2014 15:59:26]  Mappingg
[18/12/2014 15:59:26]  Mapping
```

Synchronizing Links (Mapping 1 of 1)

Please wait...
Synchronizing...

Finding Modeler links...

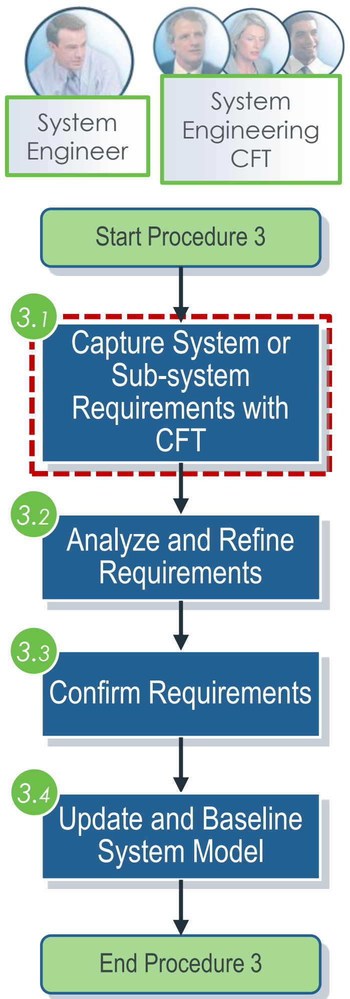
Found 0

00:07

- +Truck (SysML)
  - +Activities
  - +Assets
  - +Blocks
  - +Diagrams
  - +Requirements
    - +The driver shall be legally allowed to drive the truck
    - +The vehicle must comply with local fuel filling regulations
    - +The vehicle shall be capable of carrying loads
      - +The vehicle shall be capable of carrying "medium duty" loads (class 4,5,6).
      - +The gross vehicle weight rating shall not exceed 7,500 kg (16,500lb).
      - +Large Load
    - +The vehicle shall be capable of displaying the temperature on the dashboard
    - +The vehicle shall be capable of refuelling
    - +The vehicle shall provide a sleeping area
  - +Use Cases
  - +Variation
  - +Driver

# MODEL BASED SYSTEM ENGINEERING

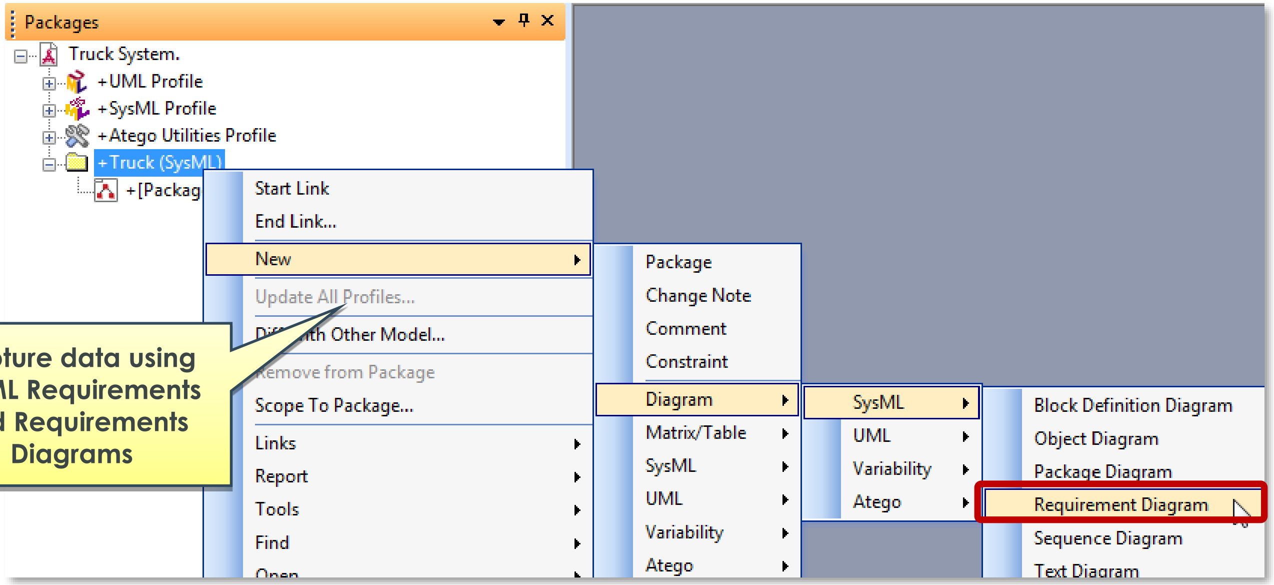
## ► Capture System or Sub-system Requirements with CFT


System Engineer
System Engineering 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

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



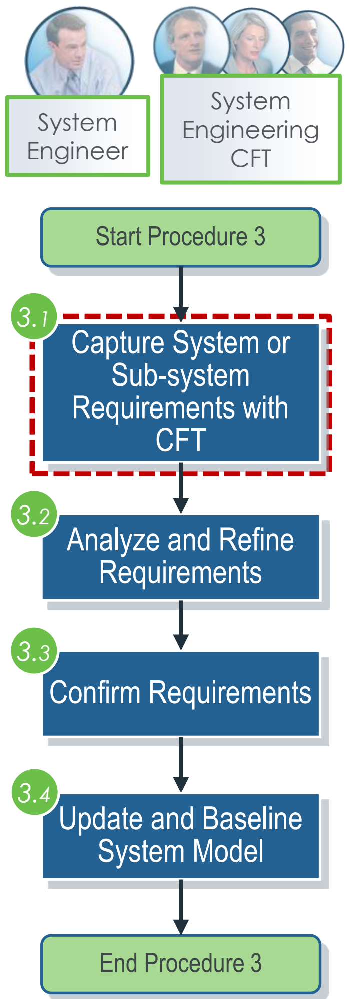
Capture data using SysML Requirements and Requirements Diagrams

**Note**

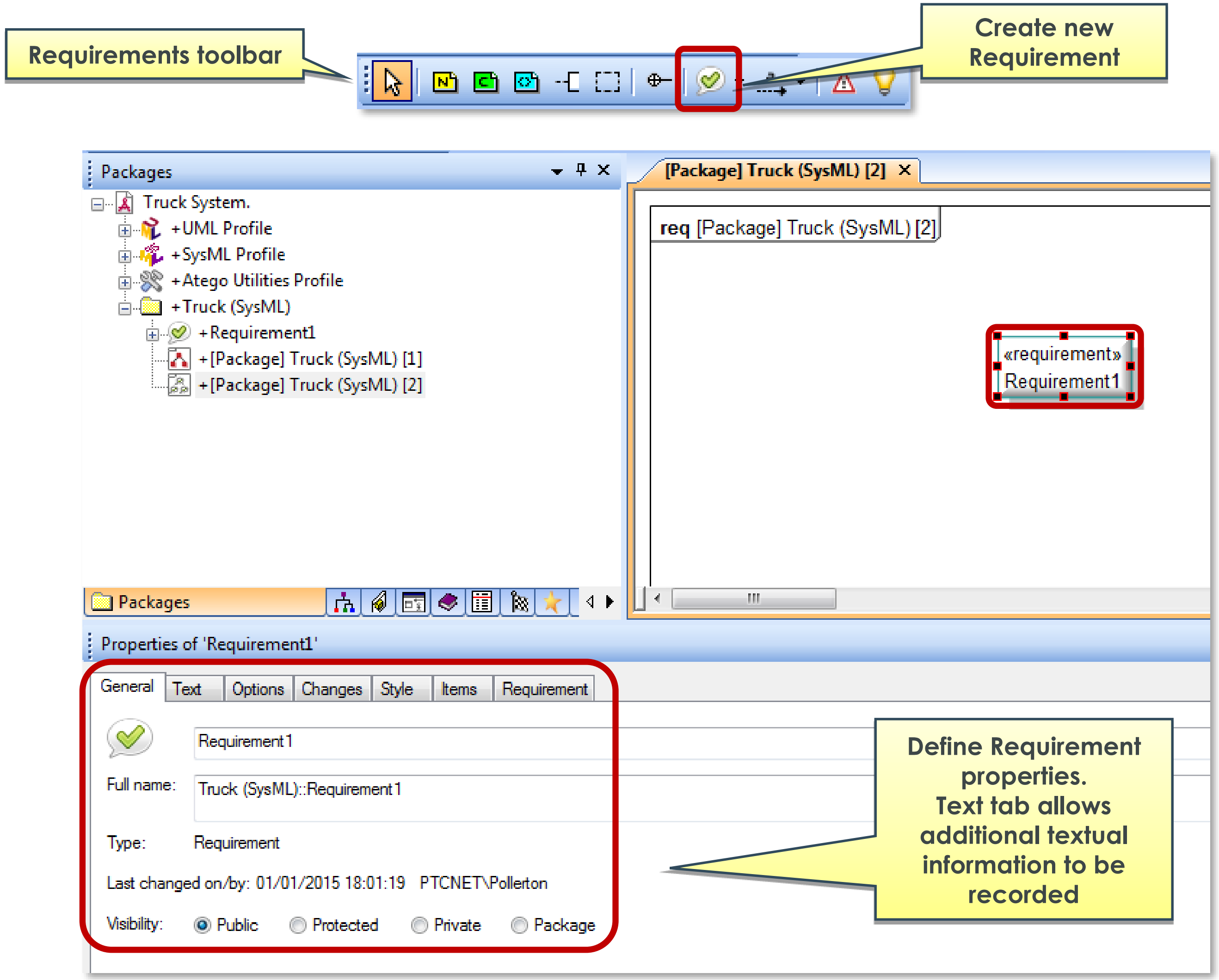
A requirement specifies a capability or condition that must (or should) be satisfied. A requirement may specify a function that a system must perform or a performance condition a system must achieve

# MODEL BASED SYSTEM ENGINEERING

- Capture System or Sub-system Requirements with CFT



System Engineer
System Engineering 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


Requirements toolbar

Create new Requirement

Packages

Truck System.
- +UML Profile
- +SysML Profile
- +Atego Utilities Profile
- +Truck (SysML)
  - +Requirement1
  - +[Package] Truck (SysML) [1]
  - +[Package] Truck (SysML) [2]

[Package] Truck (SysML) [2]

req [Package] Truck (SysML) [2]

«requirement»
Requirement1

Properties of 'Requirement1'

General | Text | Options | Changes | Style | Items | Requirement

Requirement1

Full name: Truck (SysML)::Requirement1

Type: Requirement

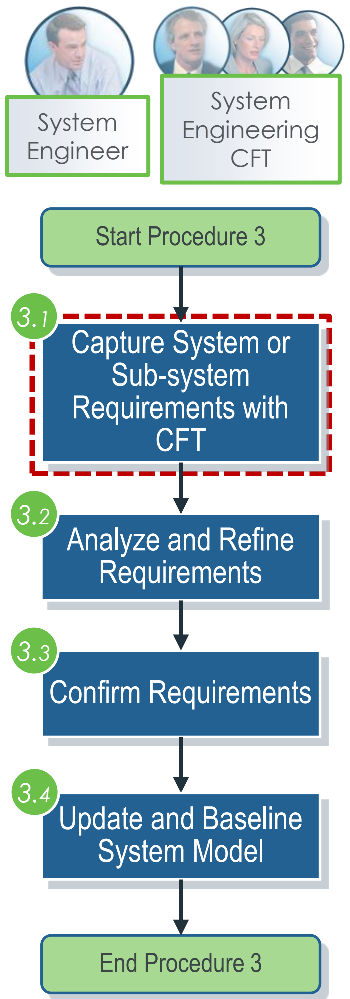
Last changed on/by: 01/01/2015 18:01:19 PTCNET\Pollerton

Visibility: ● Public ○ Protected ○ Private ○ Package

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

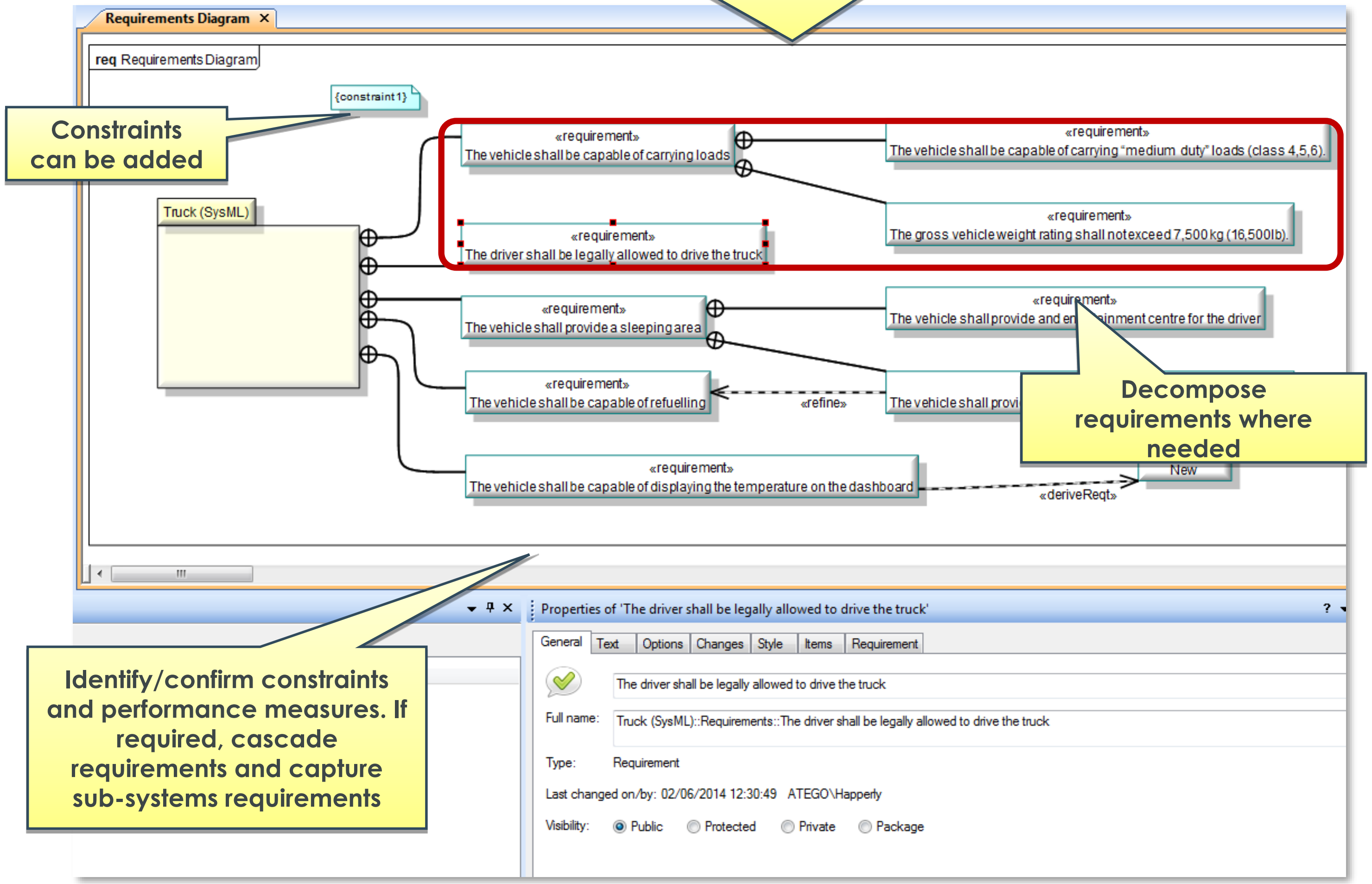
# MODEL BASED SYSTEM ENGINEERING

► Capture System or Sub-system Requirements with CFT



System Engineer
System Engineering 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



Requirements Diagram ✕

req Requirements Diagram

{constraint1}

Truck (SysML)

«requirement» The vehicle shall be capable of carrying loads

«requirement» The vehicle shall be capable of carrying "medium duty" loads (class 4,5,6).

«requirement» The gross vehicle weight rating shall not exceed 7,500 kg (16,500lb).

«requirement» The driver shall be legally allowed to drive the truck

«requirement» The vehicle shall provide a sleeping area

«requirement» The vehicle shall provide and en…inment centre for the driver

«requirement» The vehicle shall be capable of refuelling

«refine»

The vehicle shall provi…

«requirement» The vehicle shall be capable of displaying the temperature on the dashboard

«deriveReqt»

New

Properties of 'The driver shall be legally allowed to drive the truck'

General | Text | Options | Changes | Style | Items | Requirement

The driver shall be legally allowed to drive the truck

Full name: Truck (SysML)::Requirements::The driver shall be legally allowed to drive the truck

Type: Requirement

Last changed on/by: 02/06/2014 12:30:49 ATEGO\Happerly

Visibility: ● Public ○ Protected ○ Private ○ Package

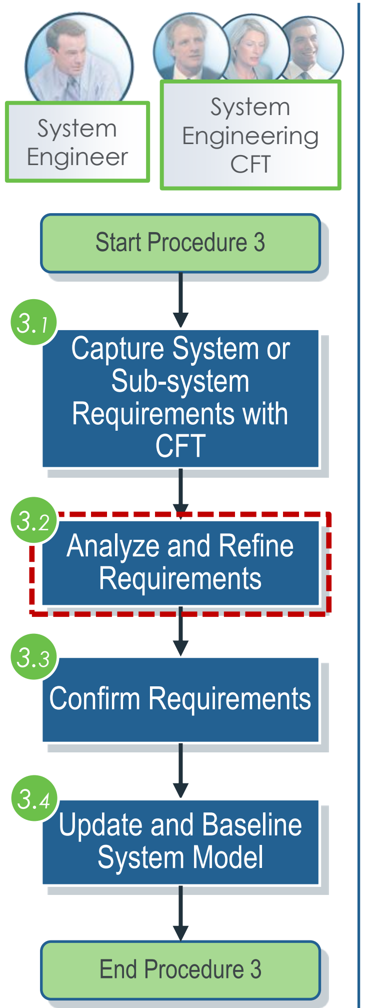
Constraints can be added

Decompose requirements where needed

Identify/confirm constraints and performance measures. If required, cascade requirements and capture sub-systems requirements

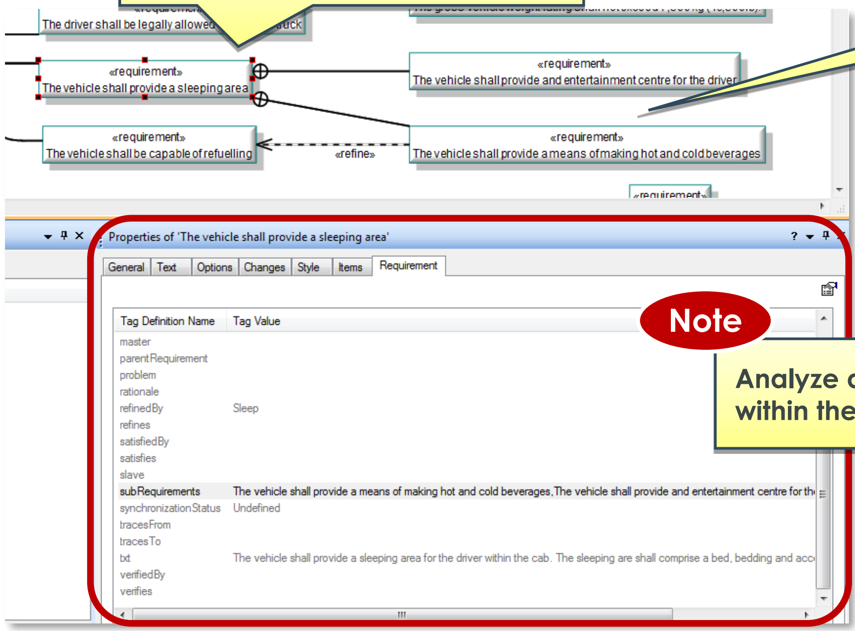
# MODEL BASED SYSTEM ENGINEERING

## ▶ Analyze and Refine Requirements



System Engineer
System Engineering 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

Review, analyze and refine and requirements with the CFT



The driver shall be legally allowed ... truck
«requirement»
The vehicle shall provide a sleeping area
«requirement»
The vehicle shall provide and entertainment centre for the driver
«requirement»
The vehicle shall be capable of refuelling
«refine»
«requirement»
The vehicle shall provide a means of making hot and cold beverages
«requirement»
General
Text
Options
Changes
Style
Items
Requirement

Properties of 'The vehicle shall provide a sleeping area'

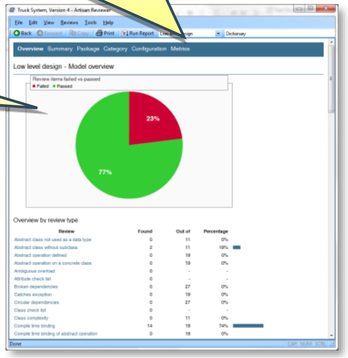
| Tag Definition Name | Tag Value |
|---|---|
| master | |
| parentRequirement | |
| problem | |
| rationale | |
| refinedBy | Sleep |
| refines | |
| satisfiedBy | |
| satisfies | |
| slave | |
| subRequirements | The vehicle shall provide a means of making hot and cold beverages,The vehicle shall provide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

Review requirements definitions and associations to other requirements

**Note**

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

Validate correctness and eliminate overlap



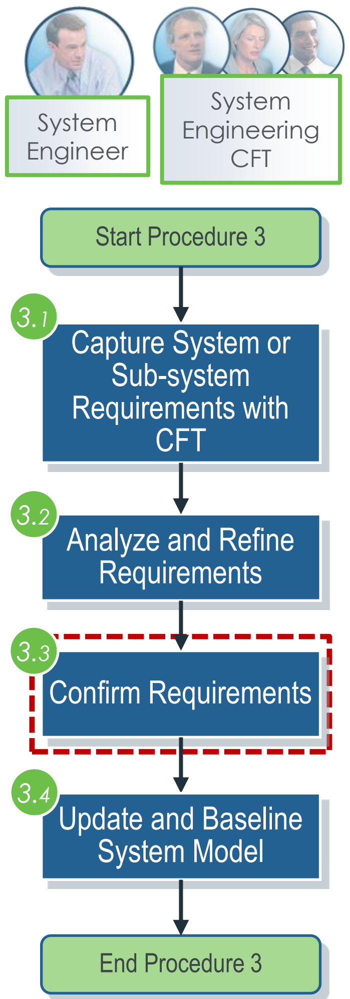
Perform analysis to confirm model is complete and correct (Refer to Automated System Design Review Best Practice)

**Note**

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

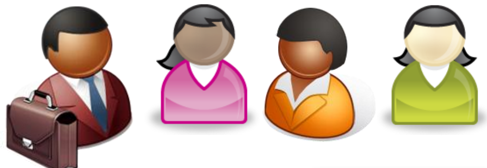
# MODEL BASED SYSTEM ENGINEERING

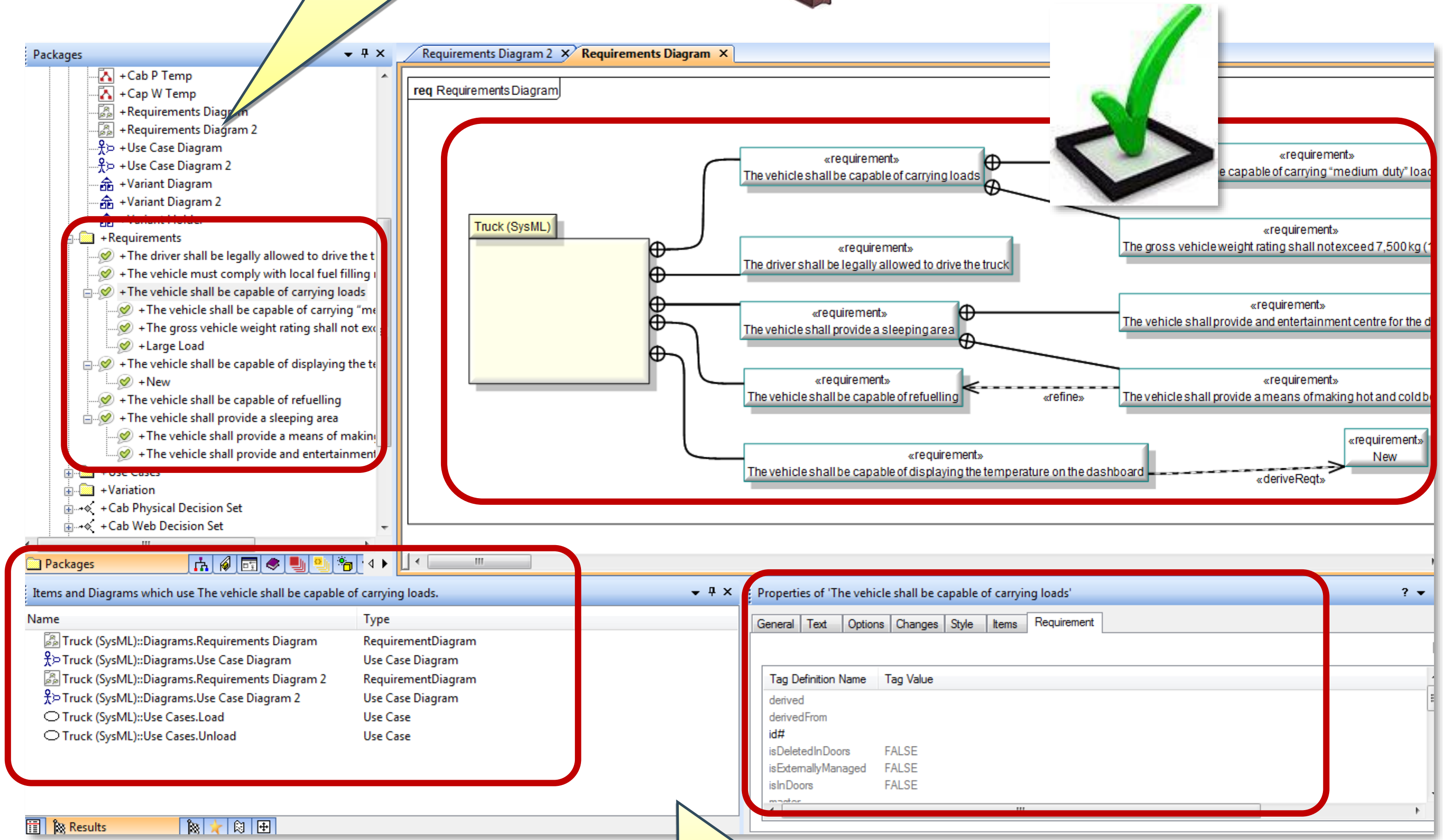
## ► Confirm Requirements



System Engineer
System Engineering 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

**Confirm and approve all requirements with the CFT**





Packages

- +Cab P Temp
- +Cap W Temp
- +Requirements Diagram
- +Requirements Diagram 2
- +Use Case Diagram
- +Use Case Diagram 2
- +Variant Diagram
- +Variant Diagram 2
- +Variant Holder
- +Requirements
  - +The driver shall be legally allowed to drive the t
  - +The vehicle must comply with local fuel filling
  - +The vehicle shall be capable of carrying loads
    - +The vehicle shall be capable of carrying "me
    - +The gross vehicle weight rating shall not exc
    - +Large Load
  - +The vehicle shall be capable of displaying the te
    - +New
  - +The vehicle shall be capable of refuelling
  - +The vehicle shall provide a sleeping area
    - +The vehicle shall provide a means of makin
    - +The vehicle shall provide and entertainment
- +Use Cases
- +Variation
- +Cab Physical Decision Set
- +Cab Web Decision Set

Requirements Diagram 2 | Requirements Diagram

req Requirements Diagram

Truck (SysML)
«requirement» The vehicle shall be capable of carrying loads
«requirement» e capable of carrying "medium duty" loa
«requirement» The gross vehicle weight rating shall not exceed 7,500 kg (
«requirement» The driver shall be legally allowed to drive the truck
«requirement» The vehicle shall provide a sleeping area
«requirement» The vehicle shall provide and entertainment centre for the d
«requirement» The vehicle shall be capable of refuelling
«refine»
«requirement» The vehicle shall provide a means of making hot and cold b
«requirement» New
«requirement» The vehicle shall be capable of displaying the temperature on the dashboard
«deriveReqt»
Packages
Results

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

| Name | Type |
|---|---|
| Truck (SysML)::Diagrams.Requirements Diagram | RequirementDiagram |
| Truck (SysML)::Diagrams.Use Case Diagram | Use Case Diagram |
| Truck (SysML)::Diagrams.Requirements Diagram 2 | RequirementDiagram |
| Truck (SysML)::Diagrams.Use Case Diagram 2 | Use Case Diagram |
| Truck (SysML)::Use Cases.Load | Use Case |
| Truck (SysML)::Use Cases.Unload | Use Case |

Properties of 'The vehicle shall be capable of carrying loads'

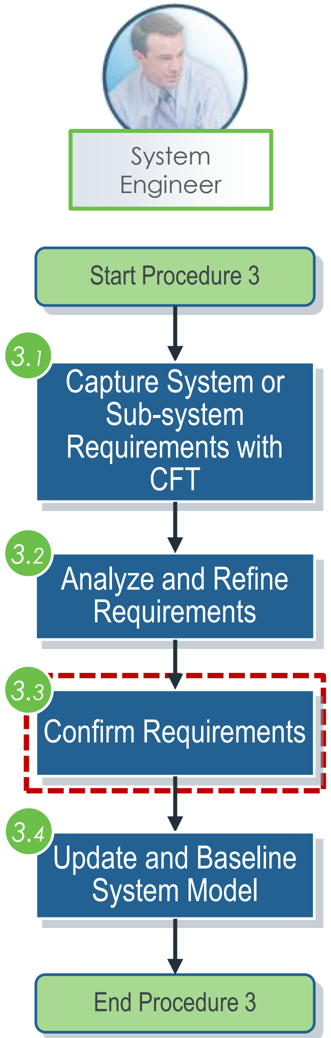
General | Text | Options | Changes | Style | Items | Requirement

| Tag Definition Name | Tag Value |
|---|---|
| derived | |
| derivedFrom | |
| id# | |
| isDeletedInDoors | FALSE |
| isExternallyManaged | FALSE |
| isInDoors | FALSE |
| master | |

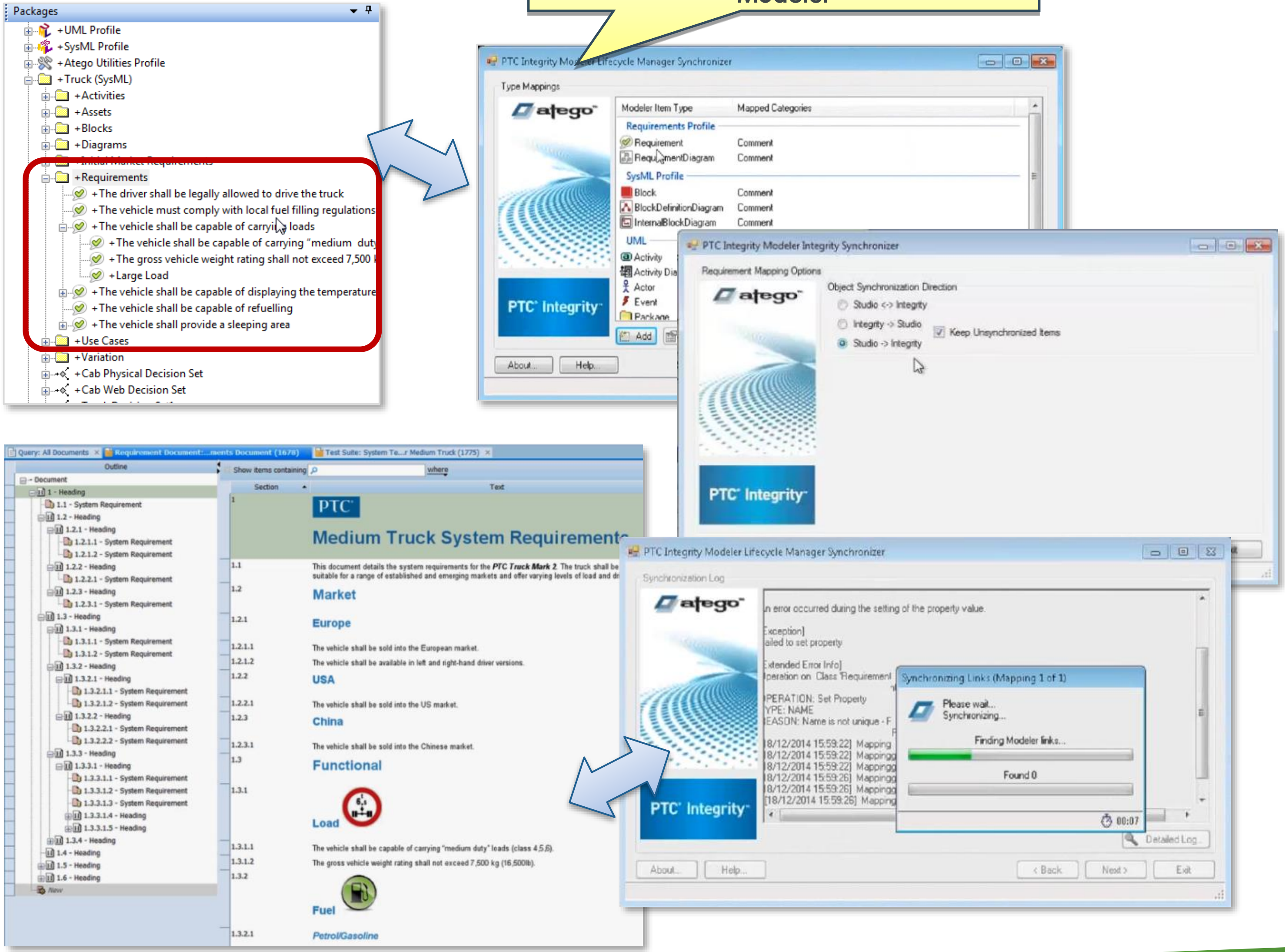
**Update requirements model if needed**

# MODEL BASED PRODUCT LINE ENGINEERING BEST PRACTICE

## ▶ 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**



- +The driver shall be legally allowed to drive the truck
- +The vehicle must comply with local fuel filling regulations
- +The vehicle shall be capable of carrying loads
  - +The vehicle shall be capable of carrying "medium duty
  - +The gross vehicle weight rating shall not exceed 7,500 k
  - +Large Load
- +The vehicle shall be capable of displaying the temperature
- +The vehicle shall be capable of refuelling
- +The vehicle shall provide a sleeping area

| Modeler Item Type | Mapped Categories |
|---|---|
| Requirements Profile | |
| Requirement | Comment |
| RequirementDiagram | Comment |
| SysML Profile | |
| Block | Comment |
| BlockDefinitionDiagram | Comment |
| InternalBlockDiagram | Comment |
| UML | |
| Activity | |
| Activity Dia | |
| Actor | |
| Event | |
| Package | |

Object Synchronization Direction

○ Studio <-> Integrity

○ Integrity -> Studio

● Studio -> Integrity

☑ Keep Unsynchronized Items

- Document
  - 1 - Heading
    - 1.1 - System Requirement
    - 1.2 - Heading
      - 1.2.1 - Heading
        - 1.2.1.1 - System Requirement
        - 1.2.1.2 - System Requirement
      - 1.2.2 - Heading
        - 1.2.2.1 - System Requirement
      - 1.2.3 - Heading
        - 1.2.3.1 - System Requirement
    - 1.3 - Heading
      - 1.3.1 - Heading
        - 1.3.1.1 - System Requirement
        - 1.3.1.2 - System Requirement
      - 1.3.2 - Heading
        - 1.3.2.1 - Heading
          - 1.3.2.1.1 - System Requirement
          - 1.3.2.1.2 - System Requirement
        - 1.3.2.2 - Heading
          - 1.3.2.2.1 - System Requirement
          - 1.3.2.2.2 - System Requirement
      - 1.3.3 - Heading
        - 1.3.3.1 - Heading
          - 1.3.3.1.1 - System Requirement
          - 1.3.3.1.2 - System Requirement
          - 1.3.3.1.3 - System Requirement
          - 1.3.3.1.4 - Heading
          - 1.3.3.1.5 - Heading
      - 1.3.4 - Heading
    - 1.4 - Heading
    - 1.5 - Heading
    - 1.6 - Heading
  - New

| Section | Text |
|---|---|
| 1 | PTC<br>Medium Truck System Requirements |
| 1.1 | This document details the system requirements for the *PTC Truck Mark 2*. The truck shall be suitable for a range of established and emerging markets and offer varying levels of load and dr |
| 1.2 | Market |
| 1.2.1 | Europe |
| 1.2.1.1 | The vehicle shall be sold into the European market. |
| 1.2.1.2 | The vehicle shall be available in left and right-hand driver versions. |
| 1.2.2 | USA |
| 1.2.2.1 | The vehicle shall be sold into the US market. |
| 1.2.3 | China |
| 1.2.3.1 | The vehicle shall be sold into the Chinese market. |
| 1.3 | Functional |
| 1.3.1 | Load |
| 1.3.1.1 | The vehicle shall be capable of carrying "medium duty" loads (class 4,5,6). |
| 1.3.1.2 | The gross vehicle weight rating shall not exceed 7,500 kg (16,500lb). |
| 1.3.2 | Fuel |
| 1.3.2.1 | Petrol/Gasoline |

```
n error occurred during the setting of the property value.

xception]
ailed to set property

xtended Error Info]
peration on Class 'Requirement

PERATION: Set Property
YPE: NAME
EASON: Name is not unique - F

8/12/2014 15:59:22] Mapping
8/12/2014 15:59:22] Mappingg
8/12/2014 15:59:22] Mappingg
8/12/2014 15:59:26] Mappingg
8/12/2014 15:59:26] Mappingg
[18/12/2014 15:59:26] Mapping
```

Please wait...
Synchronizing...

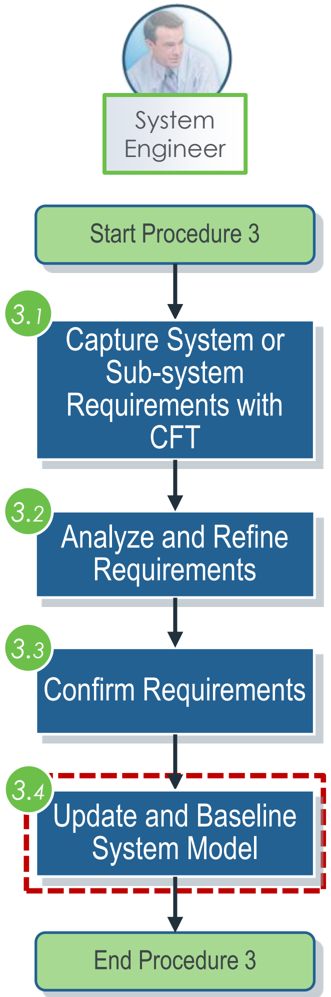
Finding Modeler links...

Found 0

00:07

# MODEL BASED SYSTEM ENGINEERING

## ▶ Update System Model



System Engineer

Start Procedure 3

3.1 Capture System or Sub-system Requirements with CFT

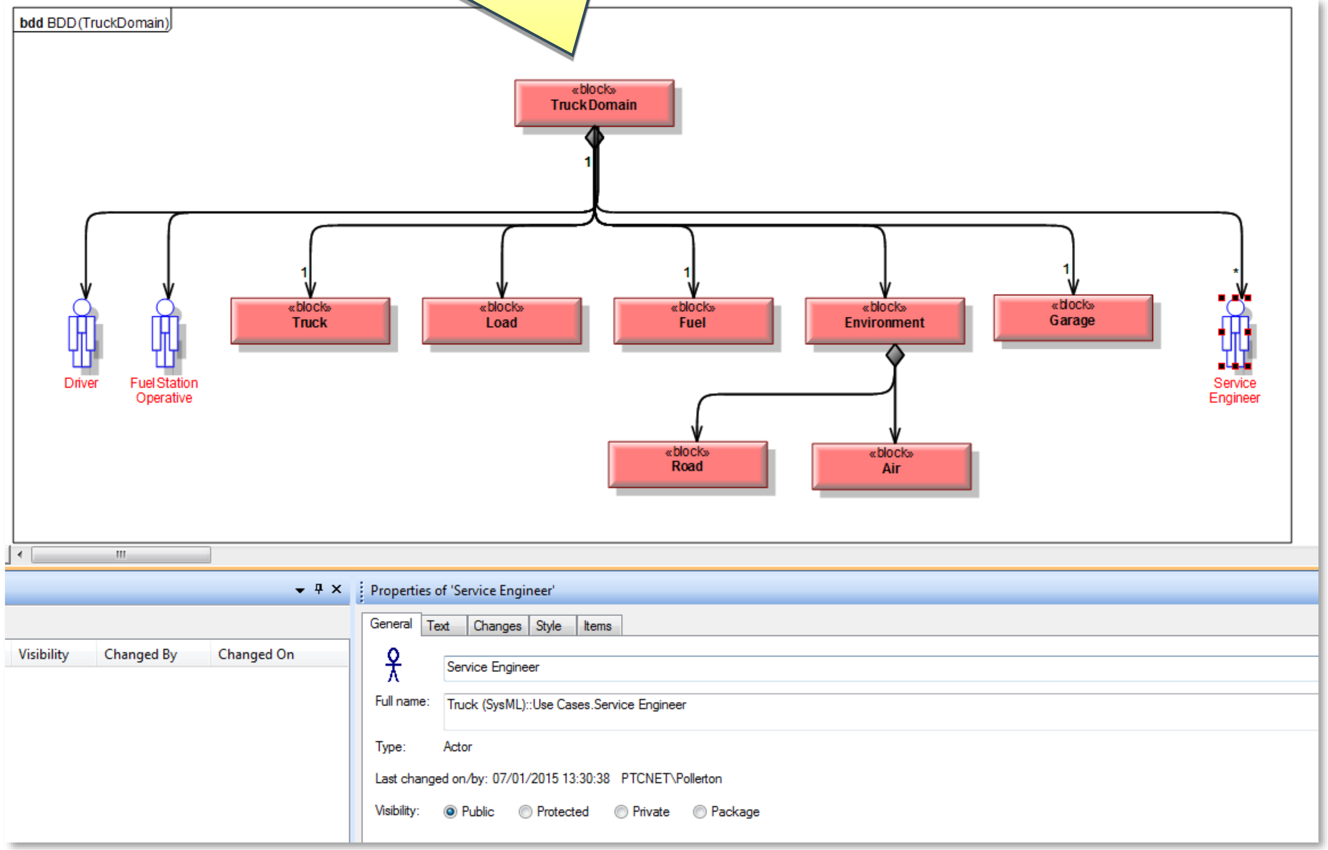
3.2 Analyze and Refine Requirements

3.3 Confirm Requirements

3.4 Update and Baseline System Model

End Procedure 3

**Update system domain model if needed (refine system structures in system domain model)**



Properties of 'Service Engineer'

Service Engineer

Full name: Truck (SysML)::Use Cases.Service Engineer

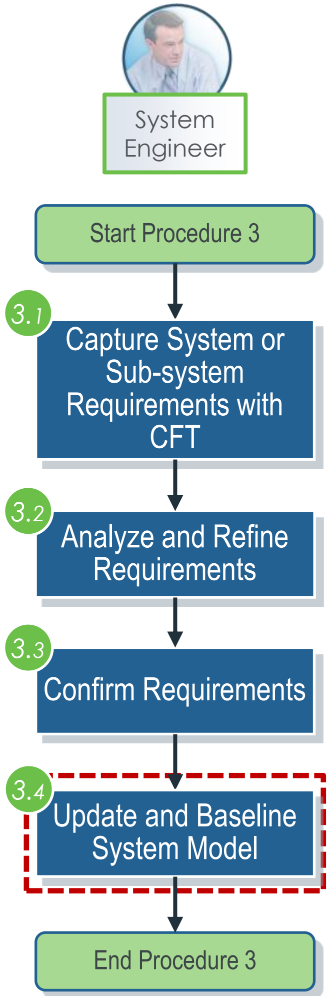
Type: Actor

Last changed on/by: 07/01/2015 13:30:38 PTCNET\Pollerton

Visibility: ● Public ○ Protected ○ Private ○ Package

# MODEL BASED SYSTEM ENGINEERING

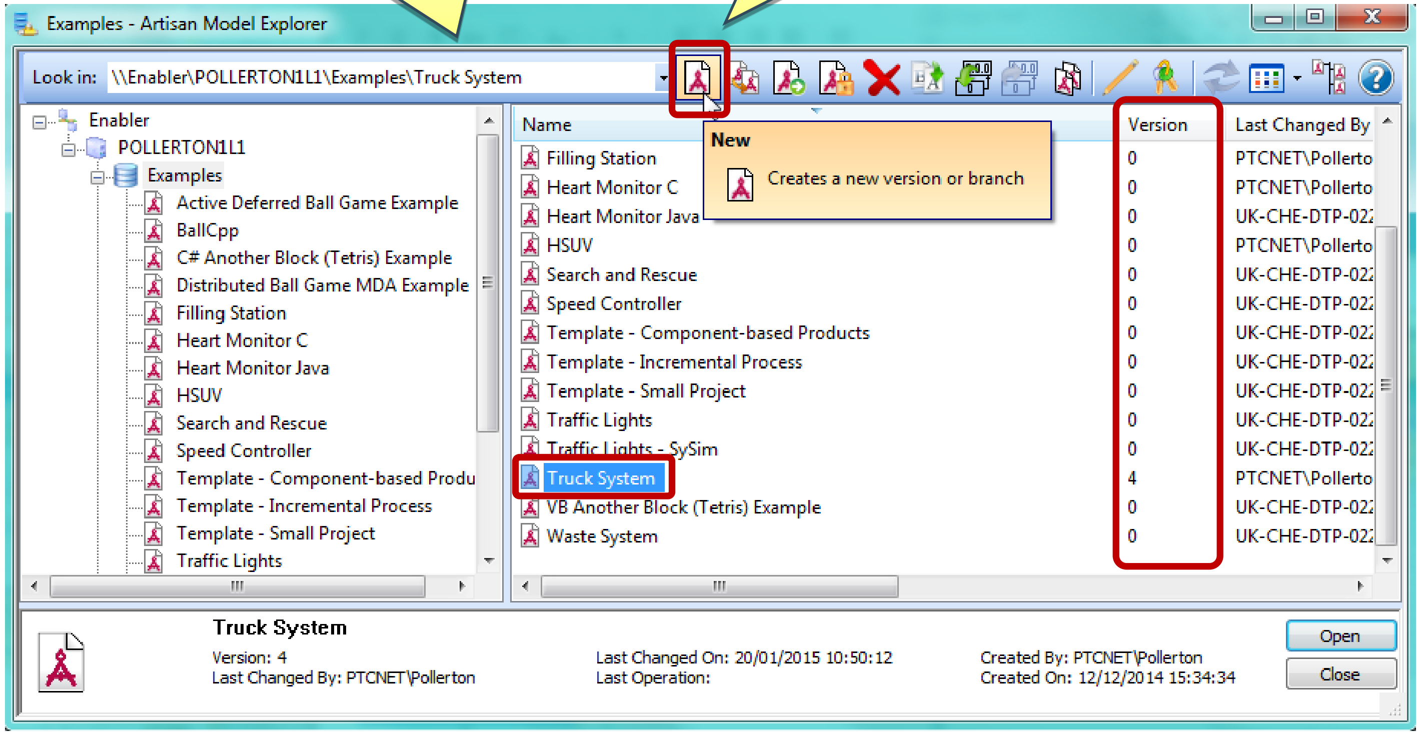
## ► Update System Model



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



Examples - Artisan Model Explorer
Look in:
New
Creates a new version or branch
Open
Close

Look in: \\Enabler\POLLERTON1L1\Examples\Truck System

- Enabler
  - POLLERTON1L1
    - Examples
      - Active Deferred Ball Game Example
      - BallCpp
      - C# Another Block (Tetris) Example
      - Distributed Ball Game MDA Example
      - Filling Station
      - Heart Monitor C
      - Heart Monitor Java
      - HSUV
      - Search and Rescue
      - Speed Controller
      - Template - Component-based Produ
      - Template - Incremental Process
      - Template - Small Project
      - Traffic Lights

| 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 |
| Truck System | 4 | PTCNET\Pollerto |
| VB Another Block (Tetris) Example | 0 | UK-CHE-DTP-022 |
| Waste System | 0 | UK-CHE-DTP-022 |

**Truck System**

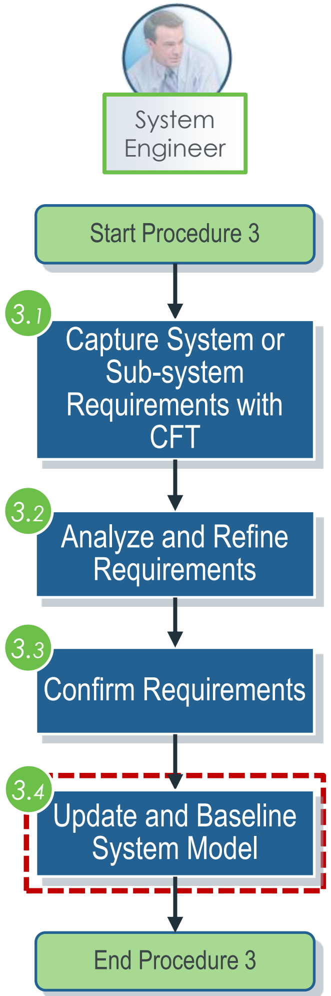
Version: 4
Last Changed By: PTCNET\Pollerton

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

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

# MODEL BASED SYSTEM ENGINEERING

## ► Update System Model


System Engineer

Start Procedure 3

3.1 Capture System or Sub-system Requirements with CFT

3.2 Analyze and Refine Requirements

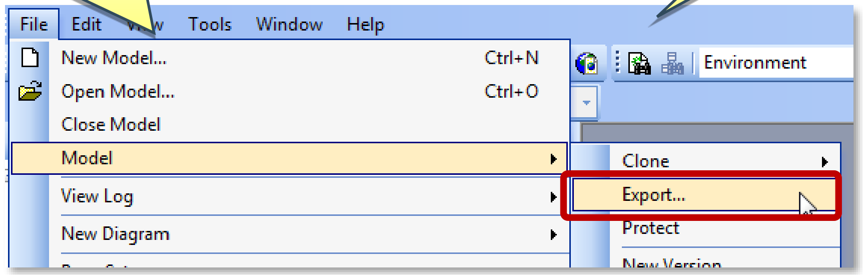
3.3 Confirm Requirements

3.4 Update and Baseline System Model

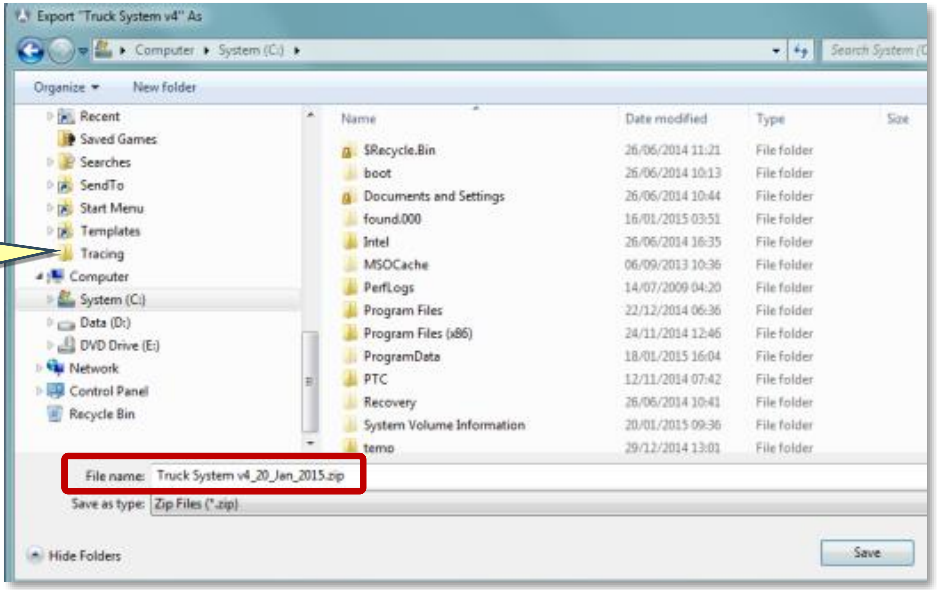
End Procedure 3

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)



| Name | Date modified | Type | Size |
|---|---|---|---|
| $Recycle.Bin | 26/06/2014 11:21 | File folder | |
| boot | 26/06/2014 10:13 | File folder | |
| Documents and Settings | 26/06/2014 10:44 | File folder | |
| found.000 | 16/01/2015 03:51 | File folder | |
| Intel | 26/06/2014 16:35 | File folder | |
| MSOCache | 06/09/2013 10:36 | File folder | |
| PerfLogs | 14/07/2009 04:20 | File folder | |
| Program Files | 22/12/2014 06:36 | File folder | |
| Program Files (x86) | 24/11/2014 12:46 | File folder | |
| ProgramData | 18/01/2015 16:04 | File folder | |
| PTC | 12/11/2014 07:42 | File folder | |
| Recovery | 26/06/2014 10:41 | File folder | |
| System Volume Information | 20/01/2015 09:36 | File folder | |
| temp | 29/12/2014 13:01 | File folder | |

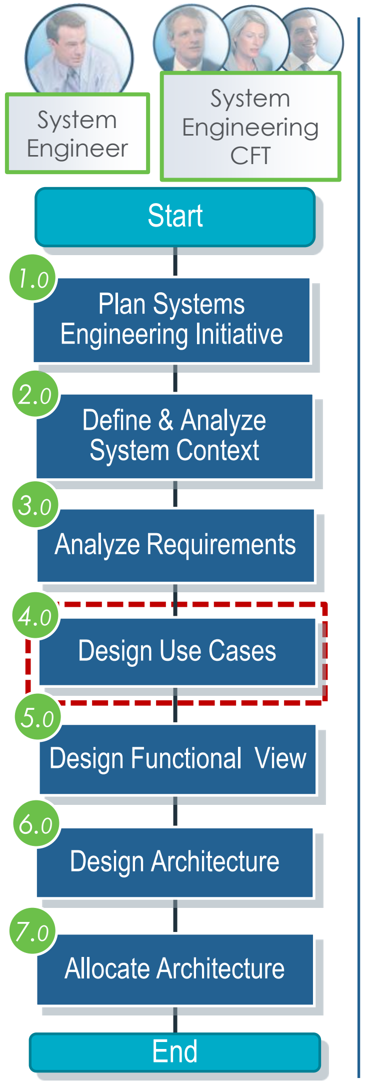
File name: Truck System v4_20_Jan_2015.zip

Save as type: Zip Files (*.zip)

# DESIGN USE CASES

# MODEL BASED SYSTEM ENGINEERING BEST PRACTICE PROCEDURE

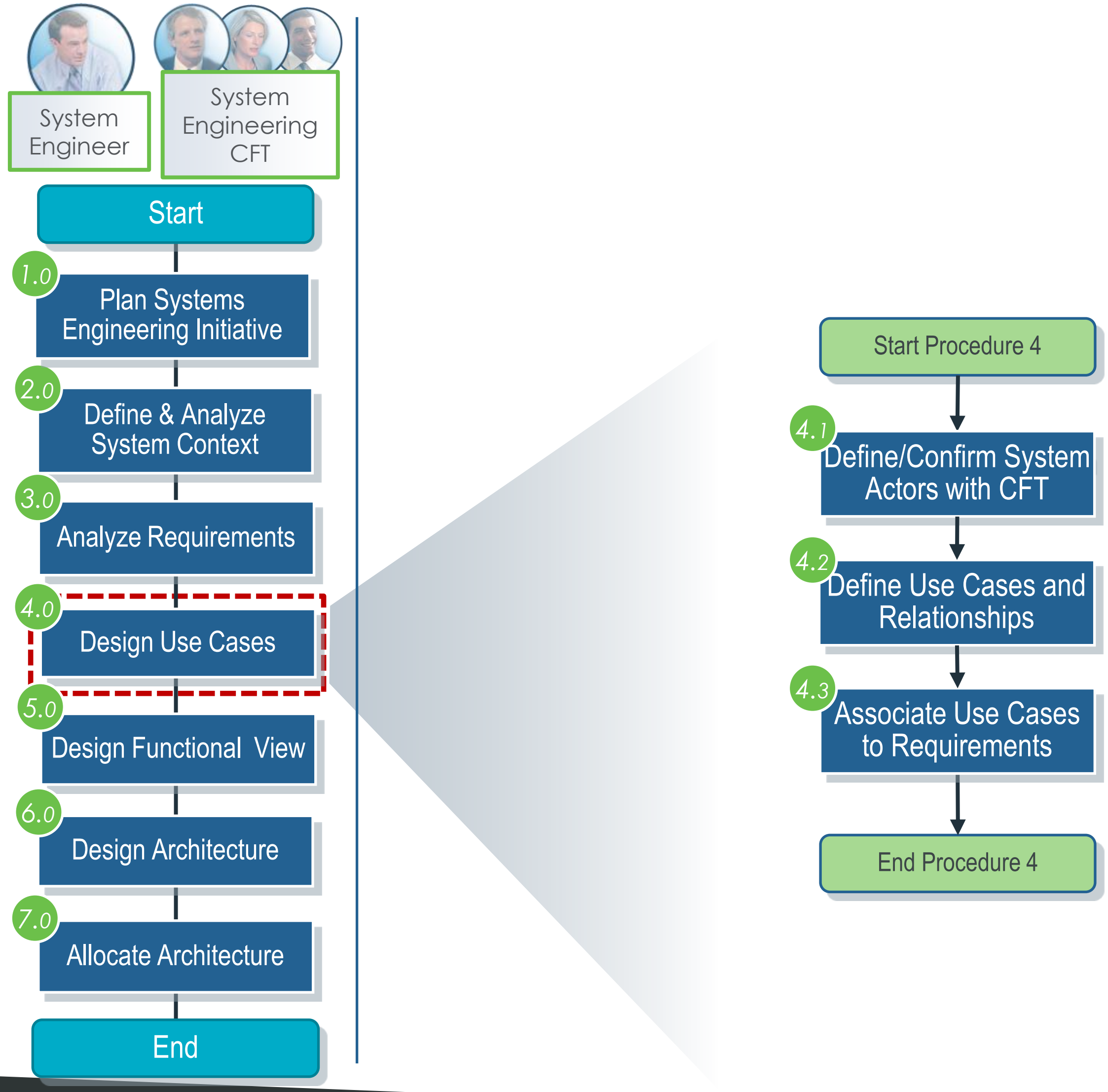
## ▶ Design Use Cases



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

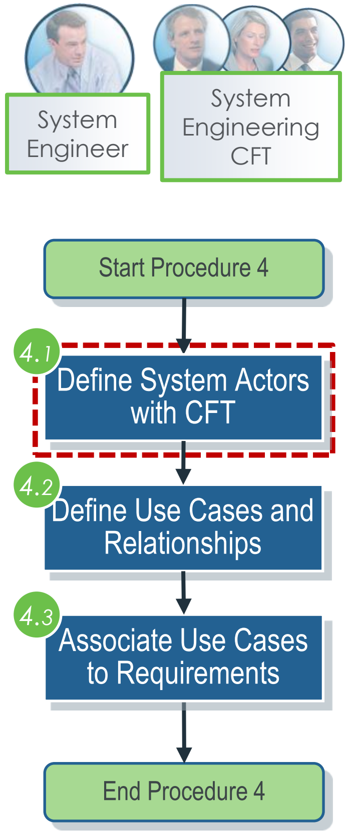
# MODEL BASED SYSTEM ENGINEERING

- Design Use Cases



# MODEL BASED SYSTEM ENGINEERING

## ► Design System Actors with CFT

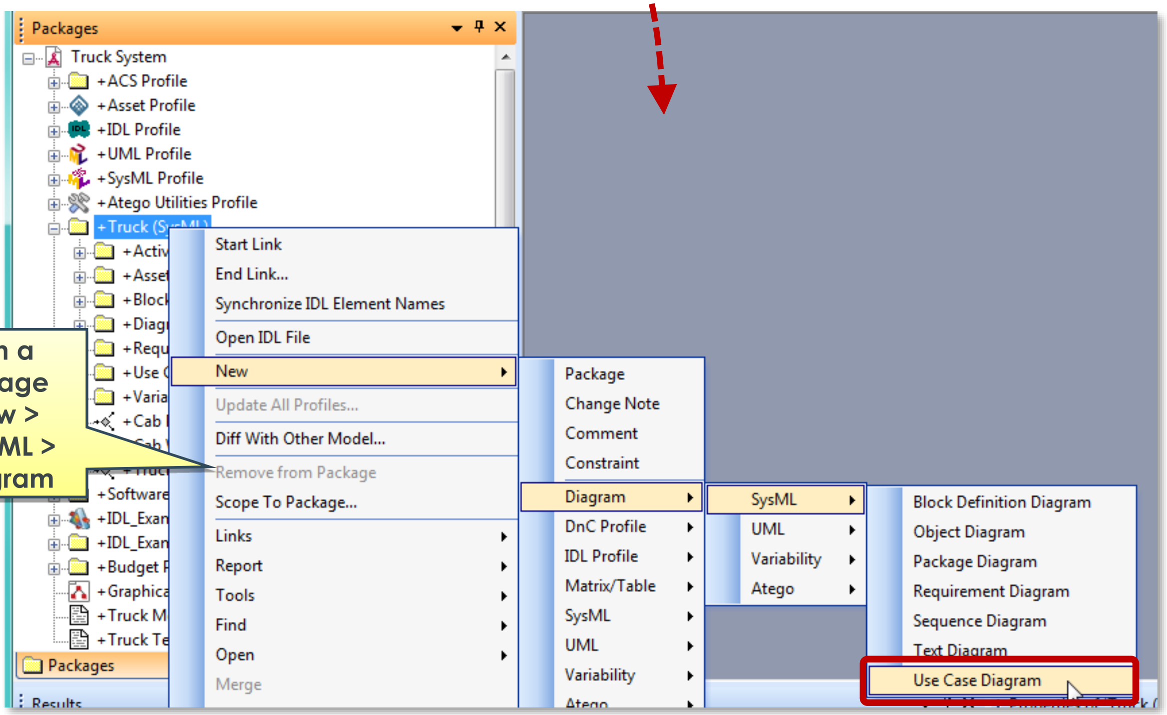

System Engineer
System Engineering CFT
Start Procedure 4
4.1
Define System Actors with CFT
4.2
Define Use Cases and Relationships
4.3
Associate Use Cases to Requirements
End Procedure 4

**Work with the CFT to identify and document use cases**

**Note**

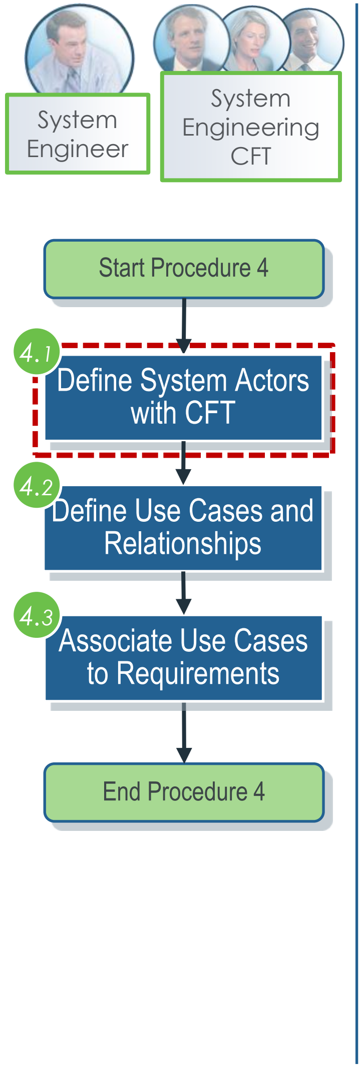
**Use case diagrams focus on system behavior. They describe the usage of a system (subject) by its actors (environment) to achieve a goal, which is realized by the subject providing a set of services to selected actors**

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



# MODEL BASED SYSTEM ENGINEERING

## ► Design System Actors with CFT



System Engineer

System Engineering CFT

Start Procedure 4

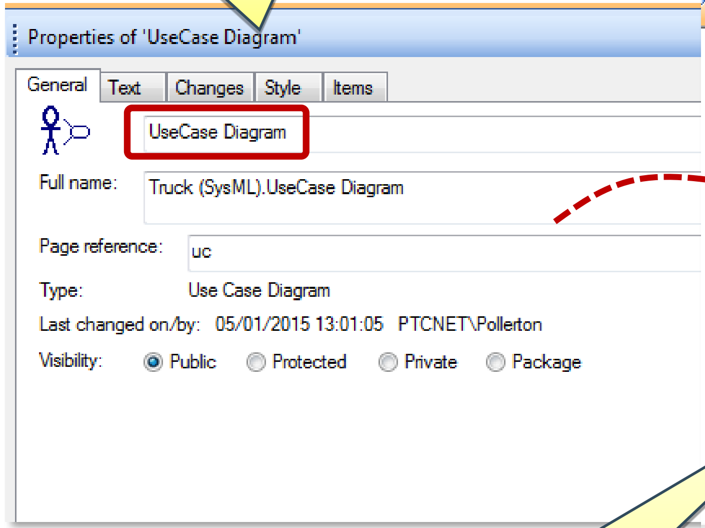
4.1 Define System Actors with CFT

4.2 Define Use Cases and Relationships

4.3 Associate Use Cases to Requirements

End Procedure 4

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



Properties of 'UseCase Diagram'

UseCase Diagram

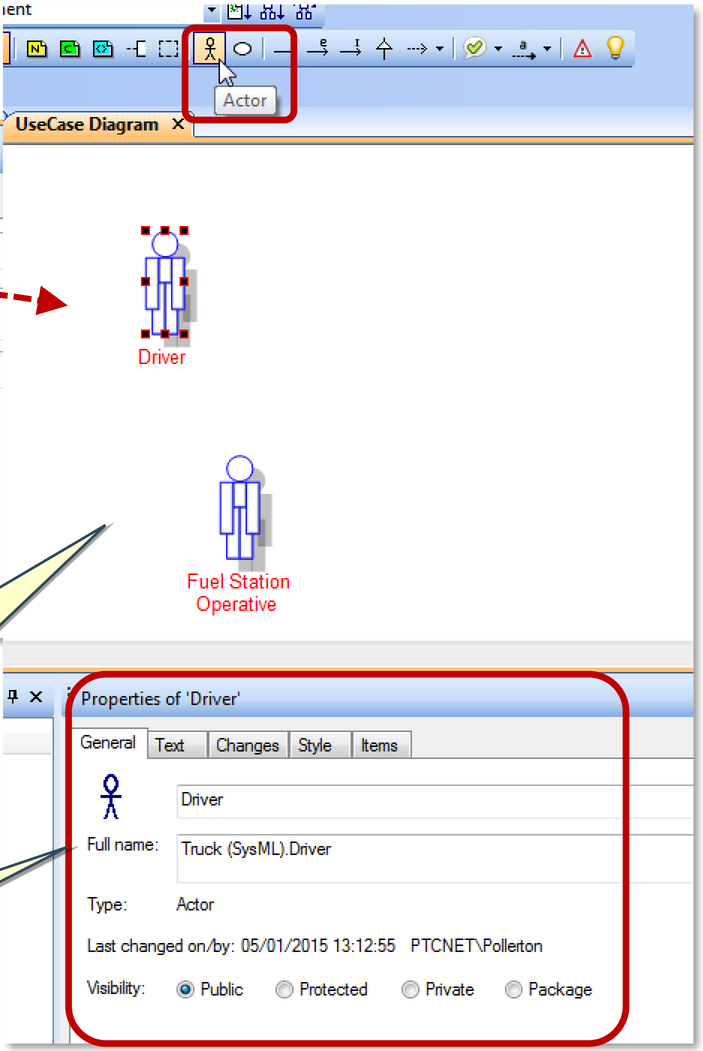
Full name: Truck (SysML).UseCase Diagram

Page reference: uc

Type: Use Case Diagram

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

Visibility: ● Public ○ Protected ○ Private ○ Package



Properties of 'Driver'

Driver

Full name: Truck (SysML).Driver

Type: Actor

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

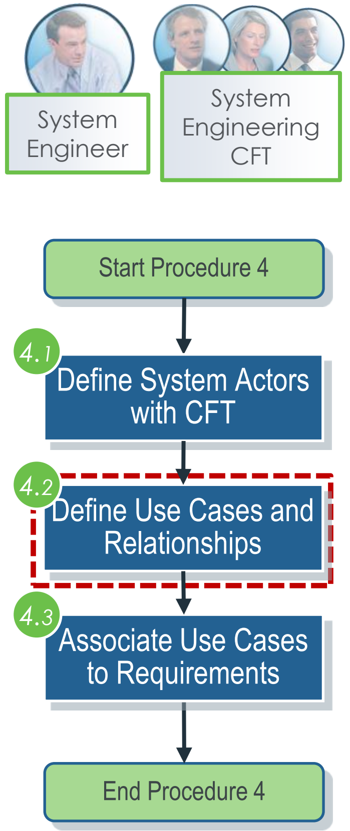
Visibility: ● Public ○ Protected ○ Private ○ Package

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

**Define properties for Actor**

# MODEL BASED SYSTEM ENGINEERING

## ► Define Use Cases and Relationships



System Engineer

System Engineering CFT

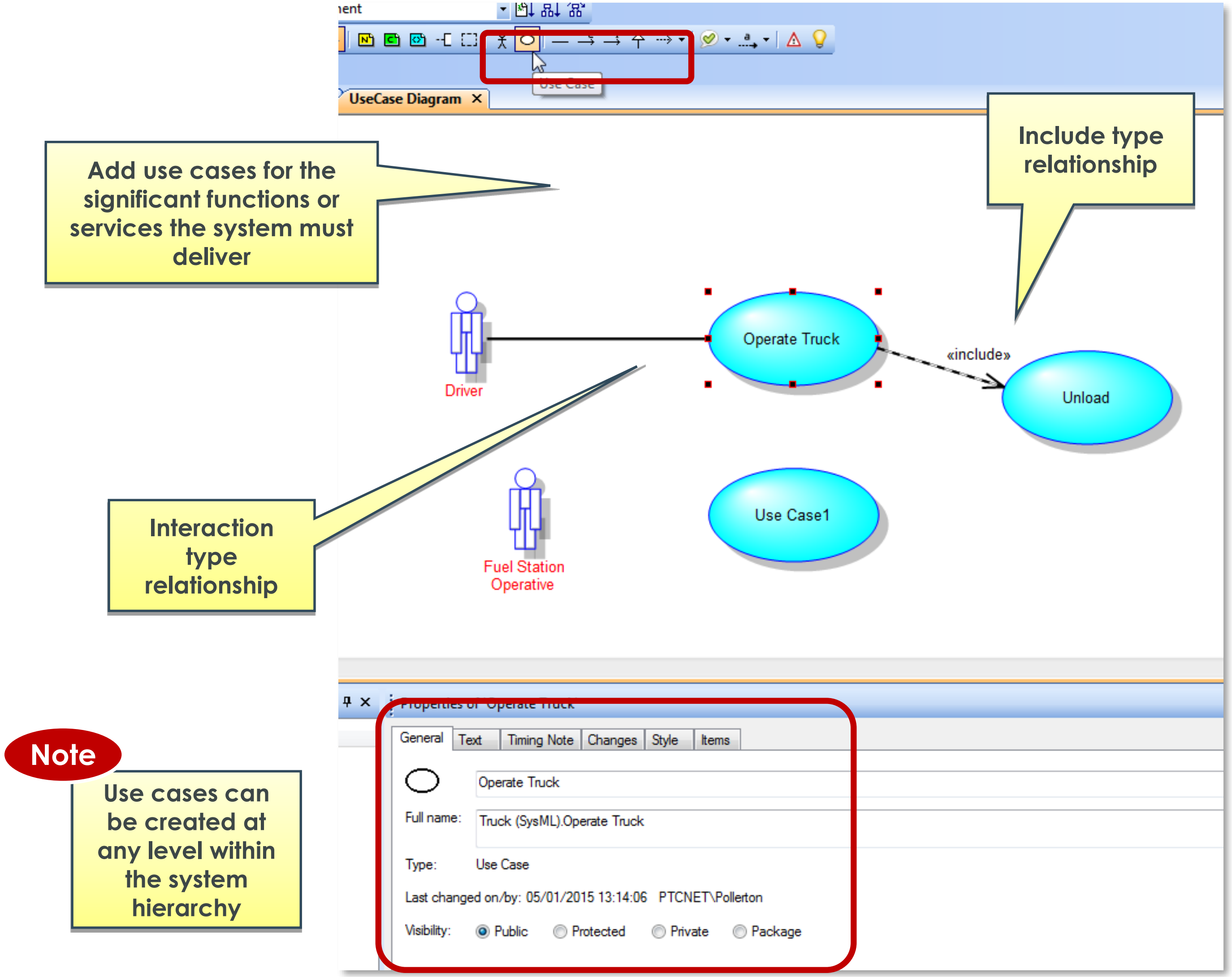
Start Procedure 4

4.1 Define System Actors with CFT

4.2 Define Use Cases and Relationships

4.3 Associate Use Cases to Requirements

End Procedure 4



Add use cases for the significant functions or services the system must deliver

Include type relationship

Interaction type relationship

**Note** Use cases can be created at any level within the system hierarchy

Operate Truck

Full name: Truck (SysML).Operate Truck

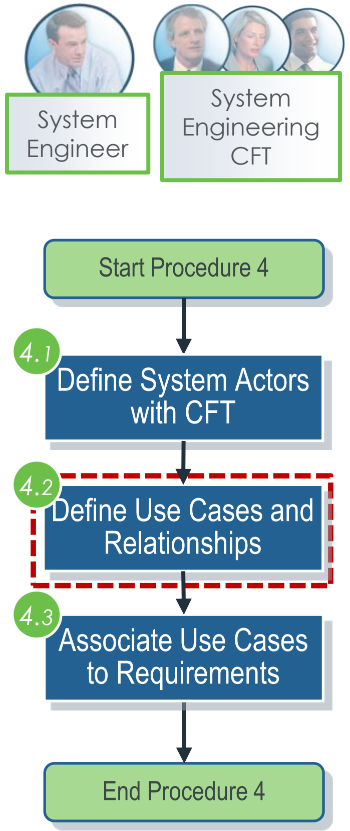
Type: Use Case

Last changed on/by: 05/01/2015 13:14:06 PTCNET\Pollerton

Visibility: ● Public ○ Protected ○ Private ○ Package

# MODEL BASED SYSTEM ENGINEERING

## ► Define Use Cases and Relationships


System Engineer

System Engineering CFT

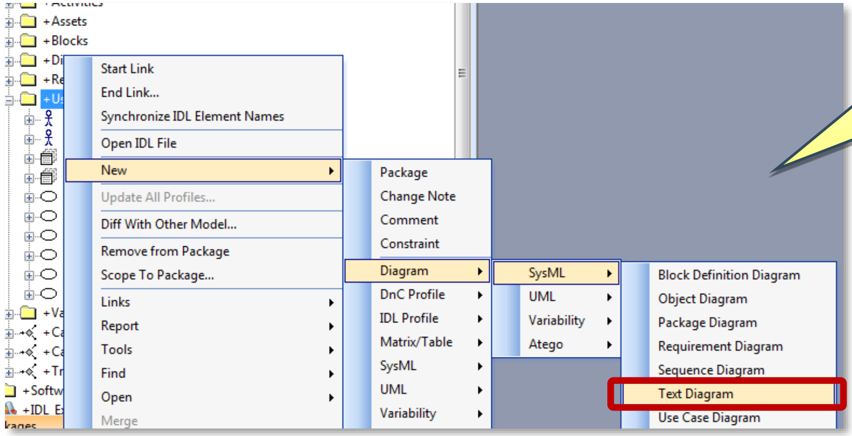
Start Procedure 4

4.1 Define System Actors with CFT

4.2 Define Use Cases and Relationships

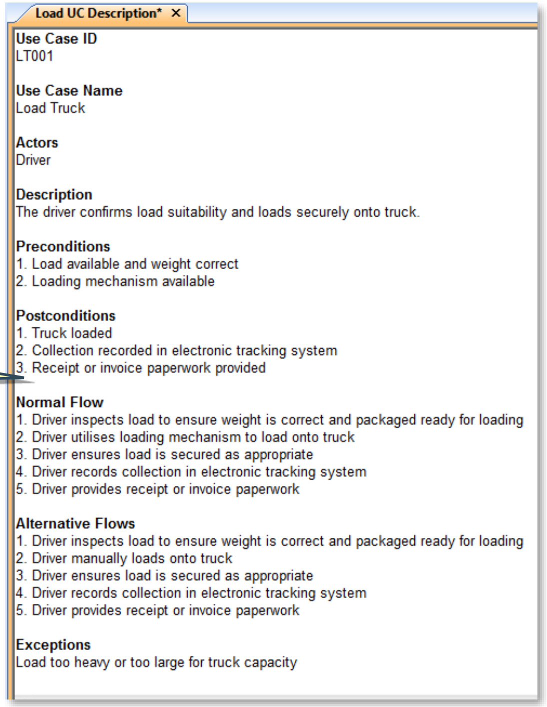
4.3 Associate Use Cases to Requirements

End Procedure 4



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* ×

**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

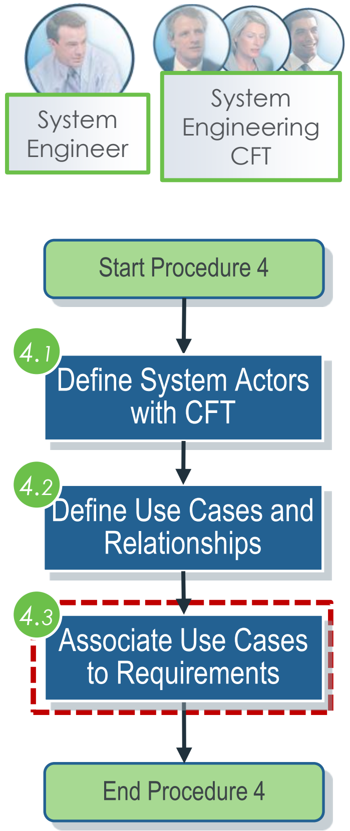
**Normal Flow**
1. Driver inspects load to ensure weight is correct and packaged ready for loading
2. Driver utilises loading mechanism to load onto truck
3. Driver ensures load is secured as appropriate
4. Driver records collection in electronic tracking system
5. Driver provides receipt or invoice 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

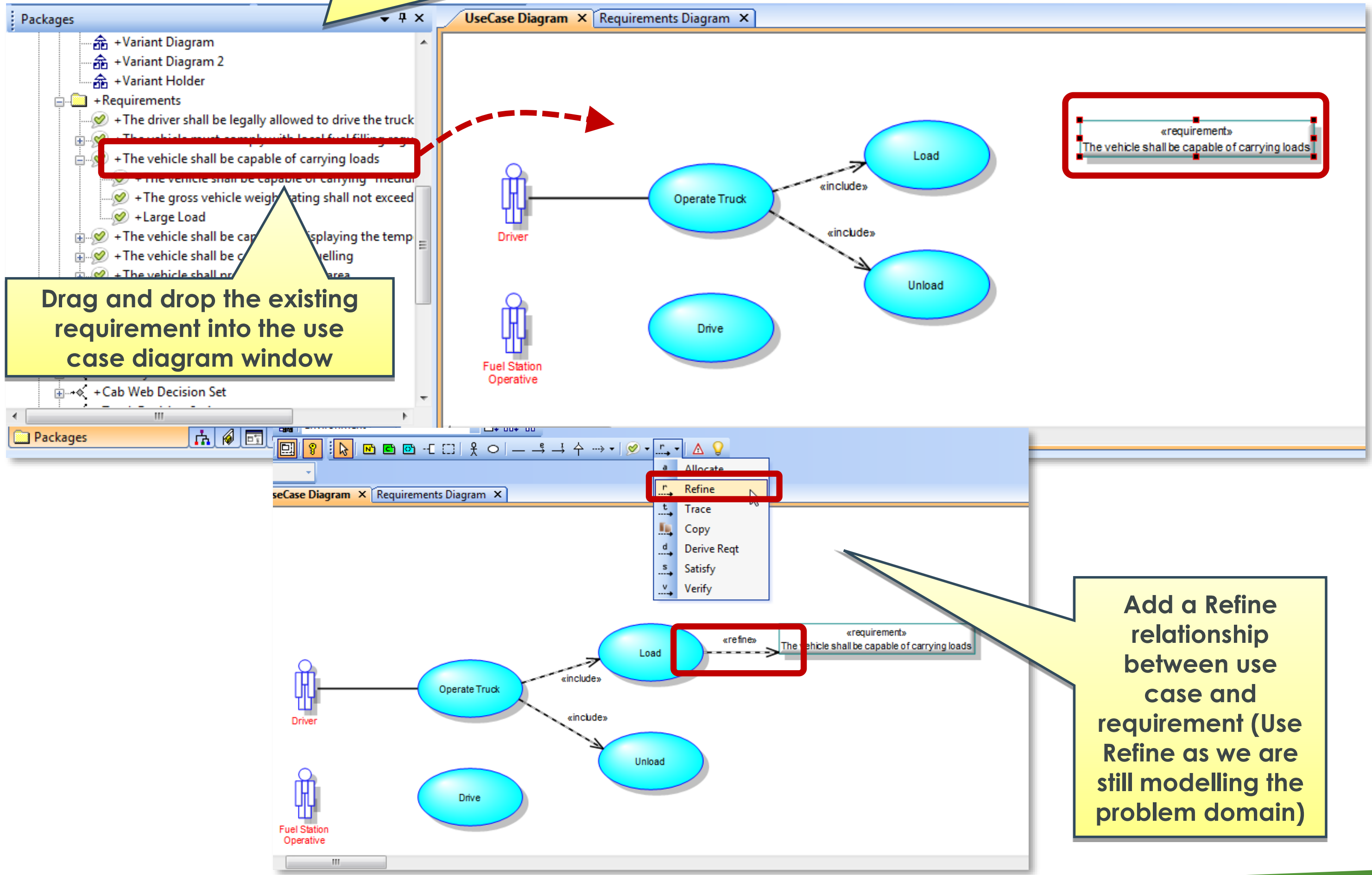
**Exceptions**
Load too heavy or too large for truck capacity

# MODEL BASED SYSTEM ENGINEERING

## ▶ Associate Use Cases to Requirements


System Engineer
System Engineering CFT
Start Procedure 4
4.1
Define System Actors with CFT
4.2
Define Use Cases and Relationships
4.3
Associate Use Cases to Requirements
End Procedure 4

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

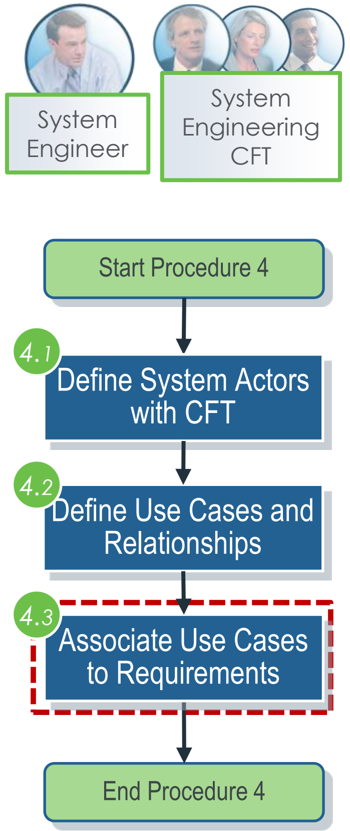


Drag and drop the existing requirement into the use case diagram window

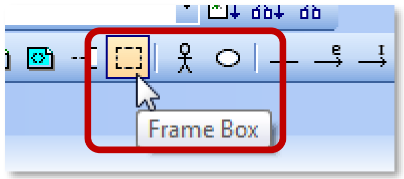
Add a Refine relationship between use case and requirement (Use Refine as we are still modelling the problem domain)

# MODEL BASED SYSTEM ENGINEERING

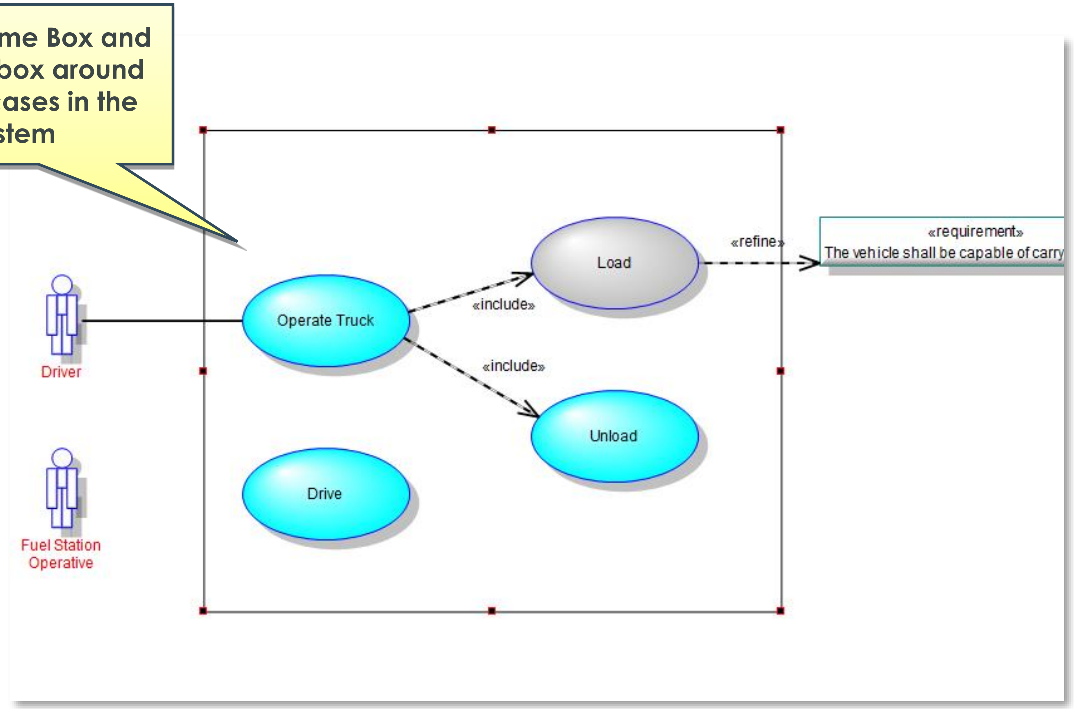
## ► Associate Use Cases to Requirements


System Engineer
System Engineering CFT
Start Procedure 4
4.1
Define System Actors with CFT
4.2
Define Use Cases and Relationships
4.3
Associate Use Cases to Requirements
End Procedure 4

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

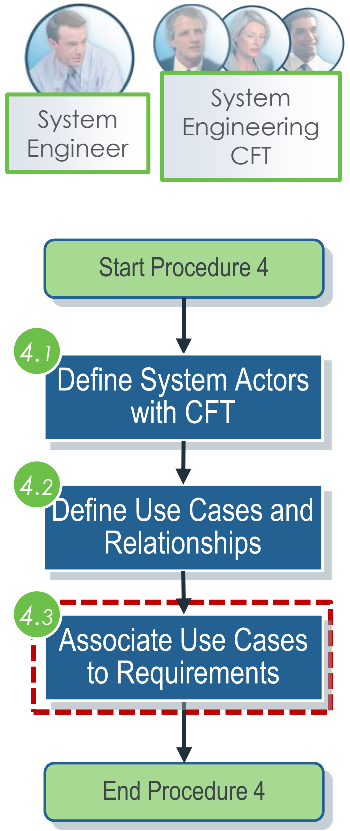


Select Frame Box and drag the box around the use cases in the system



# MODEL BASED SYSTEM ENGINEERING

## ▶ Associate Use Cases to Requirements


System Engineer

System Engineering CFT

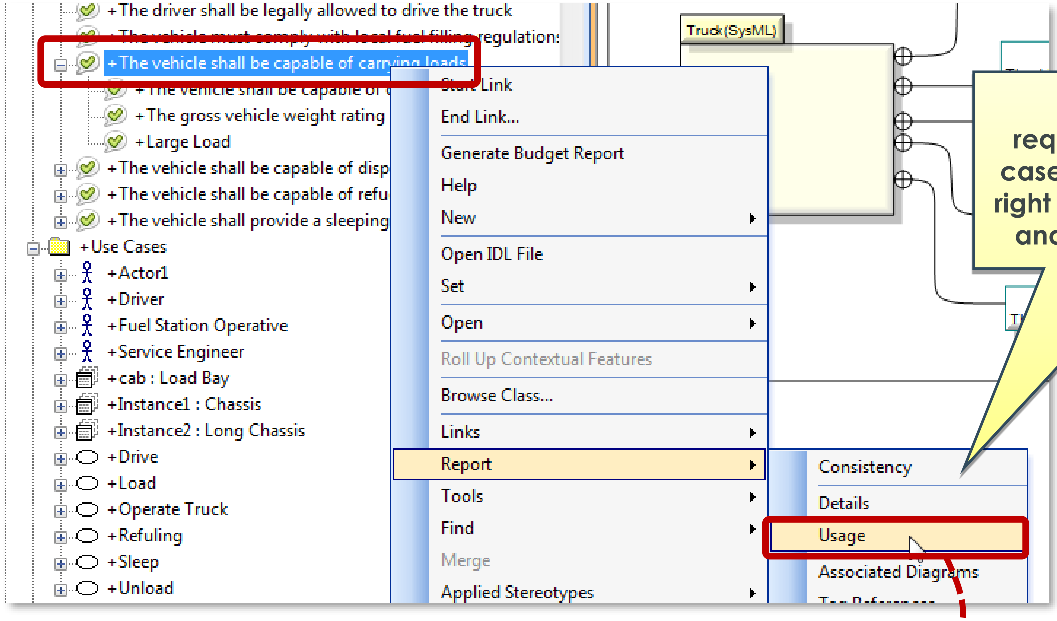
Start Procedure 4

4.1 Define System Actors with CFT

4.2 Define Use Cases and Relationships

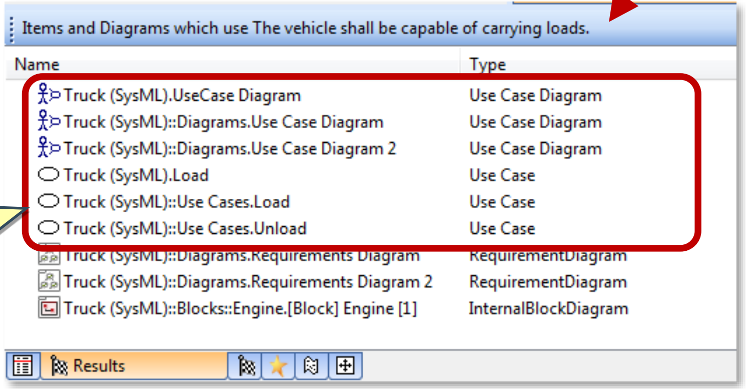
4.3 Associate Use Cases to Requirements

End Procedure 4



+The driver shall be legally allowed to drive the truck
+The vehicle must comply with local fuel filling regulations
+The vehicle shall be capable of carrying loads
+The vehicle shall be capable of
+The gross vehicle weight rating
+Large Load
+The vehicle shall be capable of disp
+The vehicle shall be capable of refu
+The vehicle shall provide a sleeping
+Use Cases
+Actor1
+Driver
+Fuel Station Operative
+Service Engineer
+cab : Load Bay
+Instance1 : Chassis
+Instance2 : Long Chassis
+Drive
+Load
+Operate Truck
+Refuling
+Sleep
+Unload
Start Link
End Link...
Generate Budget Report
Help
New
Open IDL File
Set
Open
Roll Up Contextual Features
Browse Class...
Links
Report
Tools
Find
Merge
Applied Stereotypes
Consistency
Details
Usage
Associated Diagrams
Tag References
Truck (SysML)

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


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

| Name | Type |
|---|---|
| Truck (SysML).UseCase 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 |

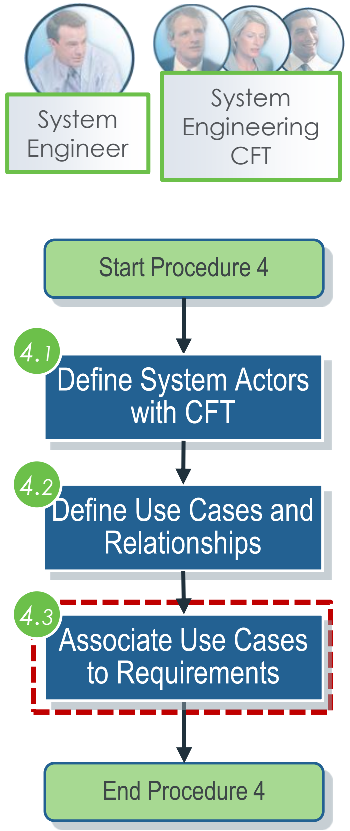
Related use cases and use case diagrams are displayed in the Results window. Related diagrams can be opened by double clicking

**Note**

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

# MODEL BASED SYSTEM ENGINEERING

## ► Associate Use Cases to Requirements



System Engineer

System Engineering CFT

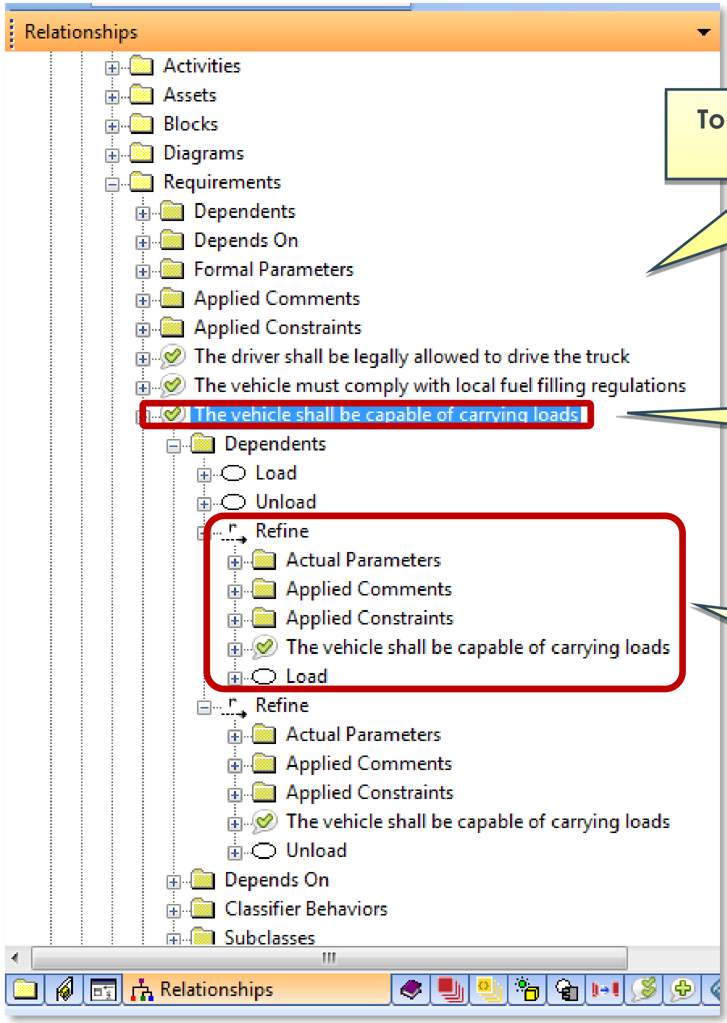
Start Procedure 4

4.1 Define System Actors with CFT

4.2 Define Use Cases and Relationships

4.3 Associate Use Cases to Requirements

End Procedure 4



Relationships

- Activities
- Assets
- Blocks
- Diagrams
- Requirements
  - Dependents
  - Depends On
  - Formal Parameters
  - Applied Comments
  - Applied Constraints
  - The driver shall be legally allowed to drive the truck
  - The vehicle must comply with local fuel filling regulations
  - The vehicle shall be capable of carrying loads
    - Dependents
      - Load
      - Unload
      - Refine
        - Actual Parameters
        - Applied Comments
        - Applied Constraints
        - The vehicle shall be capable of carrying loads
        - Load
      - Refine
        - Actual Parameters
        - Applied Comments
        - Applied Constraints
        - The vehicle shall be capable of carrying loads
        - Unload
    - Depends On
    - Classifier Behaviors
    - Subclasses

**To display all relationships between model items use the Relationship browser**

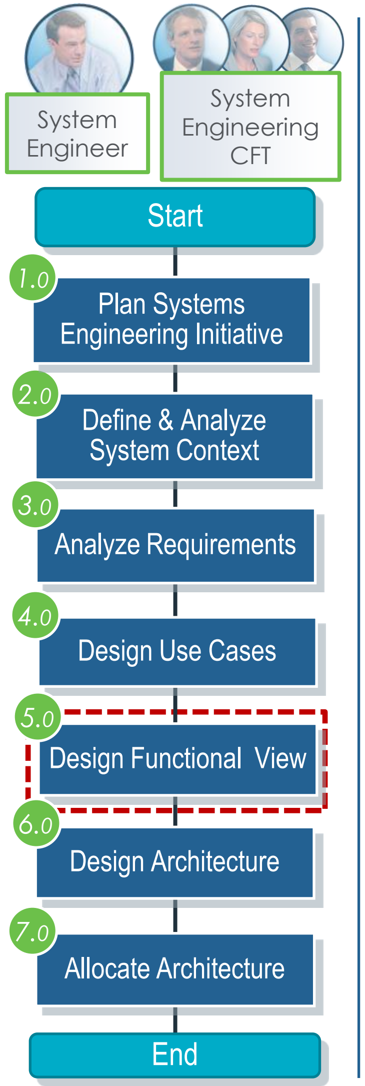
**Locate the requirement and expand the Dependents folder**

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

# DESIGN FUNCTIONAL VIEW

# MODEL BASED SYSTEM ENGINEERING BEST PRACTICE PROCEDURE

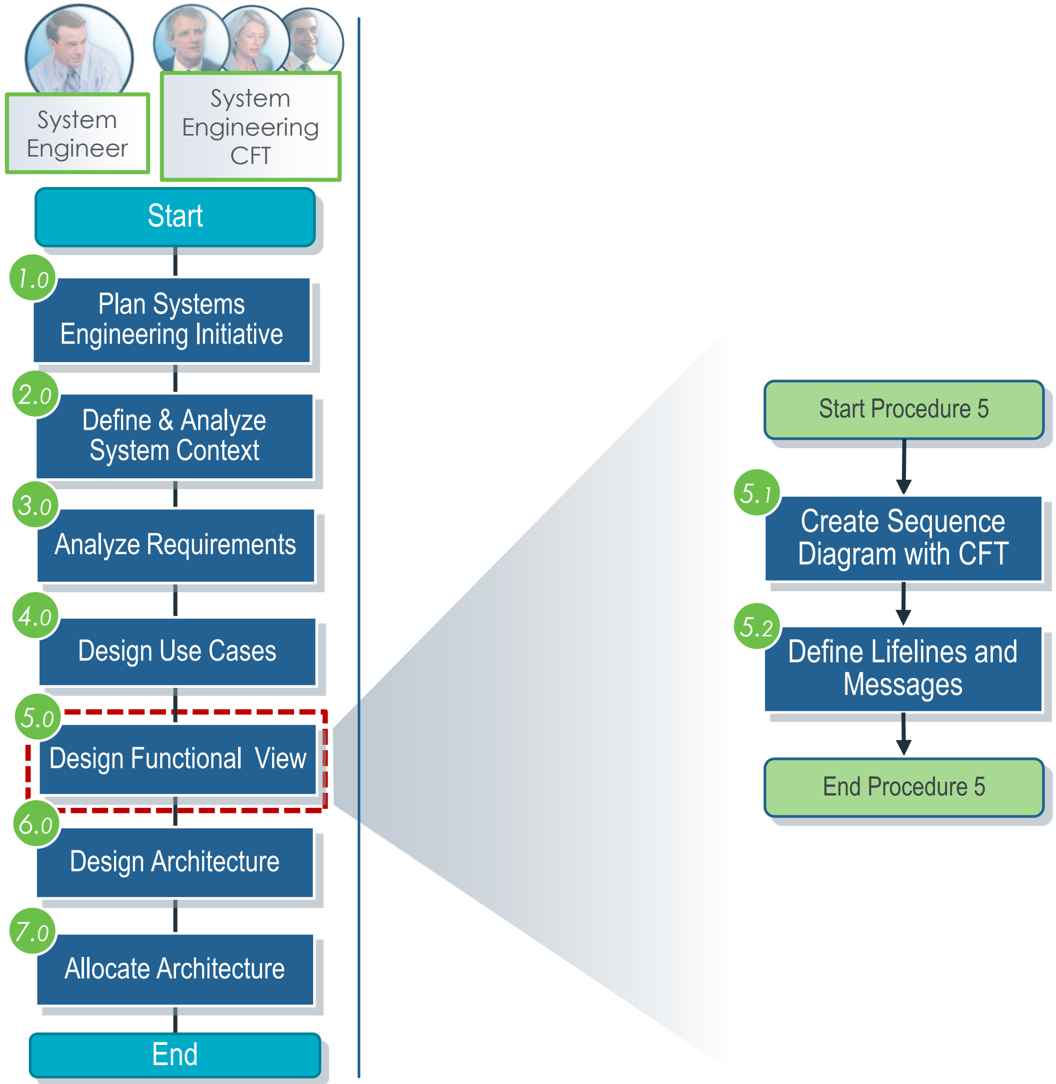
## ▶ Design Functional View



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

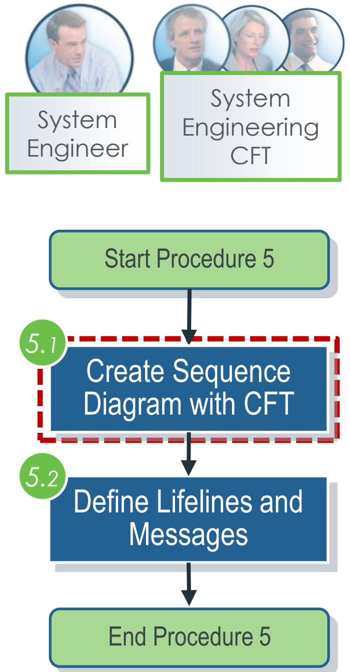
# MODEL BASED SYSTEM ENGINEERING

## ▶ Design Functional View

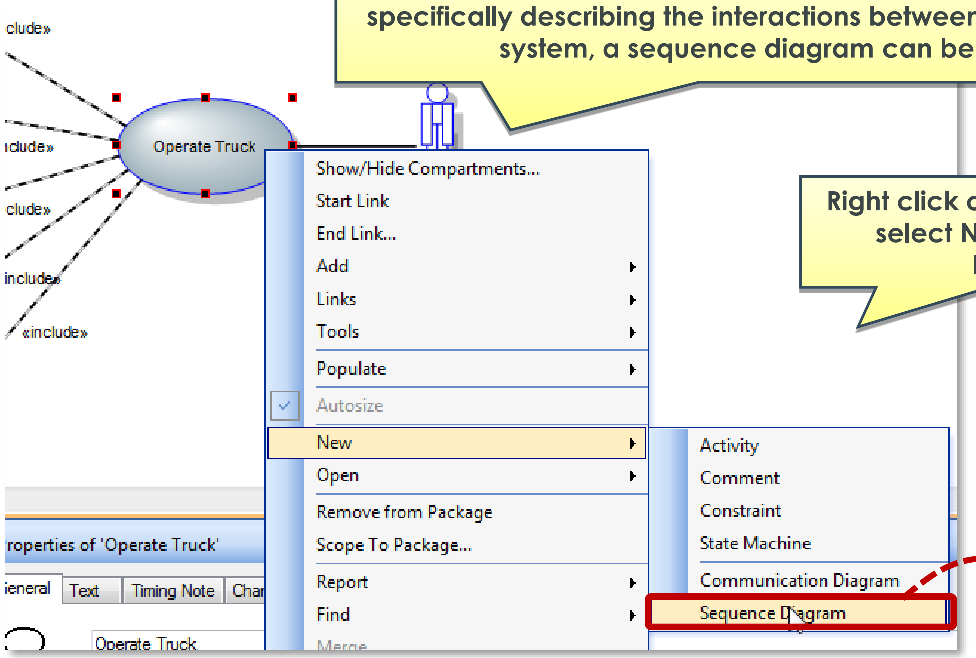


# MODEL BASED SYSTEM ENGINEERING

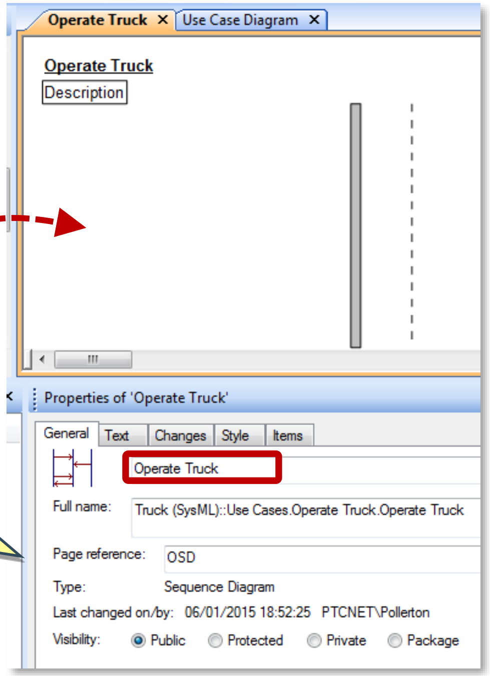
## ► Create Sequence Diagram with CFT



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



Operate Truck

Full name: Truck (SysML)::Use Cases.Operate Truck.Operate Truck

Page reference: OSD

Type: Sequence Diagram

Last changed on/by: 06/01/2015 18:52:25 PTCNET\Polleton

Visibility: ● Public ○ Protected ○ Private ○ Package

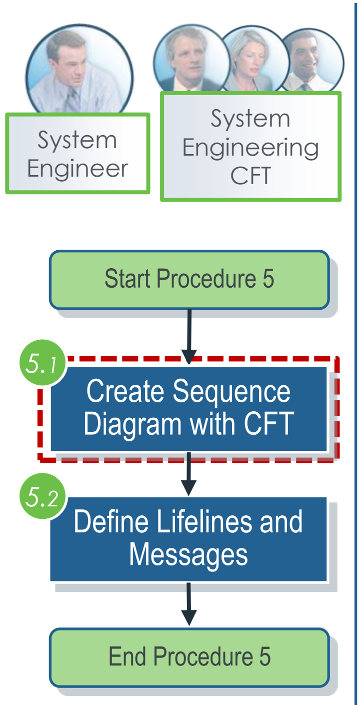
Enter name for diagram and select options

**Note**

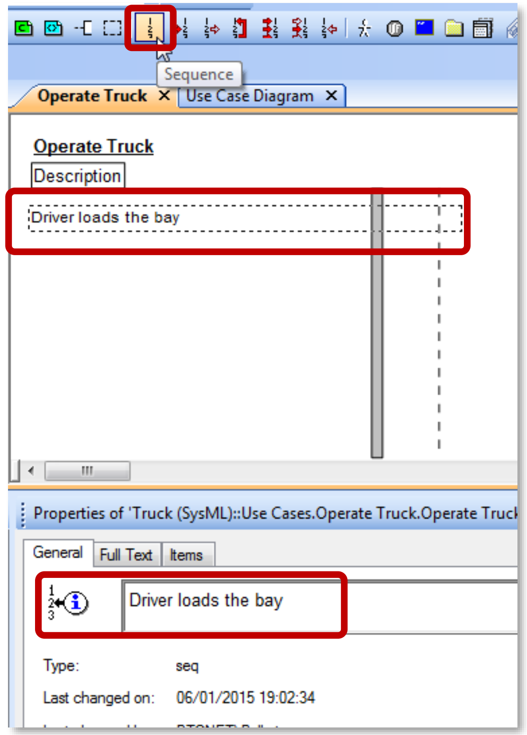
The Sequence diagram describes the flow of control between actors and systems (blocks) or between parts of a system. This diagram represents the sending and receiving of messages between the interacting entities called lifelines, where time is represented along the vertical axis

# MODEL BASED SYSTEM ENGINEERING

## ► Create Sequence Diagram with CFT

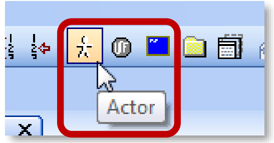


Click on Sequence and then click in diagram window below the text Description. Add a sequence to the diagram and enter a name

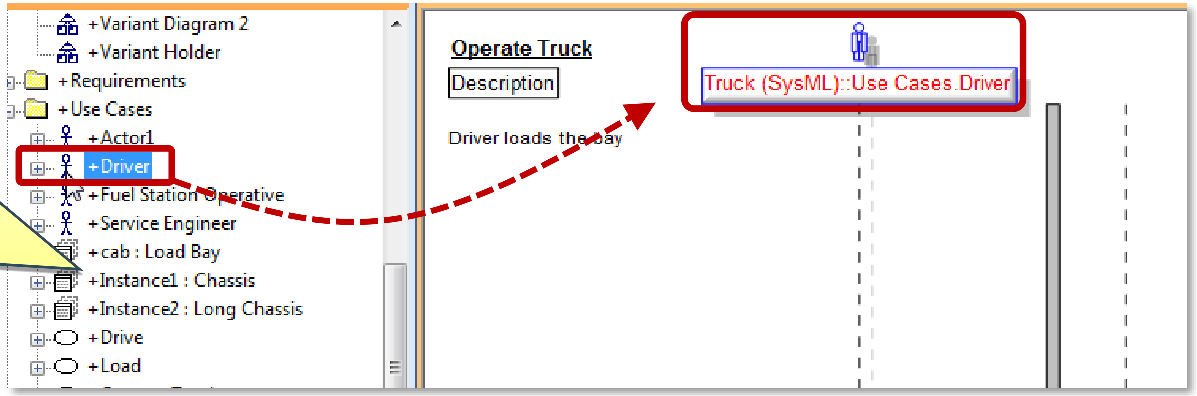


Type: seq

Last changed on: 06/01/2015 19:02:34

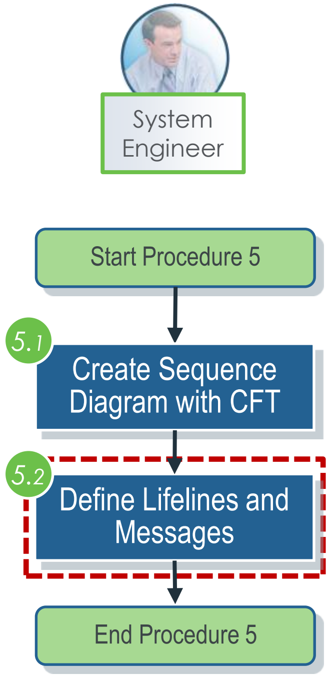


A new actor can be added, but if the correct actor already exists in the model, drag and drop into the diagram as shown below



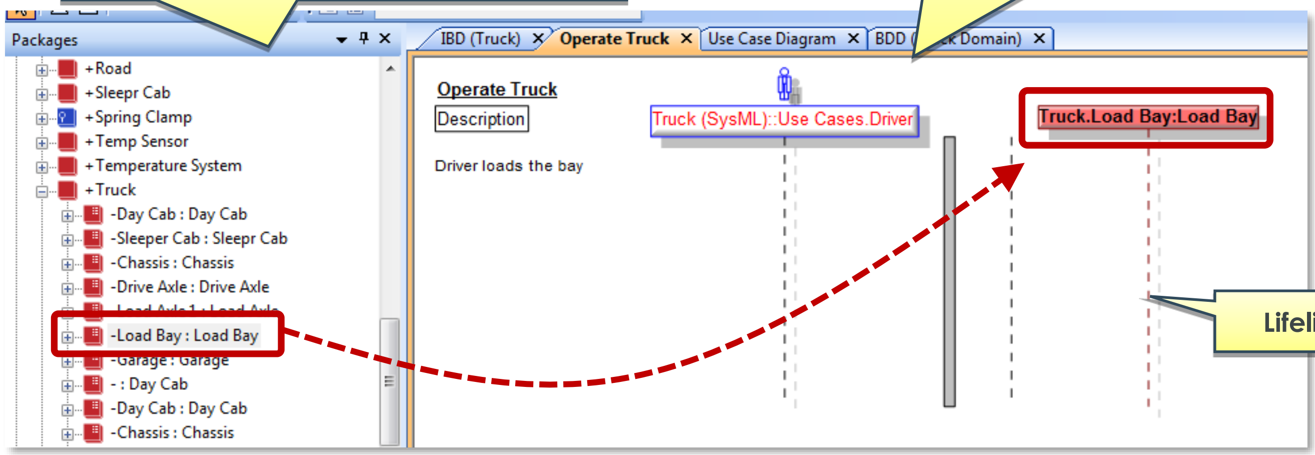
# MODEL BASED SYSTEM ENGINEERING

## ▶ Define Lifelines and Messages


System Engineer
Start Procedure 5
5.1
Create Sequence Diagram with CFT
5.2
Define Lifelines and Messages
End Procedure 5

Add BlockProperty items to the diagram to represent system components

Drag and drop existing BlockProperty items (created earlier and displayed on Internal Block Diagrams)



Packages:

- +Road
- +Sleepr Cab
- +Spring Clamp
- +Temp Sensor
- +Temperature System
- +Truck
  - -Day Cab : Day Cab
  - -Sleeper Cab : Sleepr Cab
  - -Chassis : Chassis
  - -Drive Axle : Drive Axle
  - -Load Axle 1 : Load Axle
  - -Load Bay : Load Bay
  - -Garage : Garage
  - - : Day Cab
  - -Day Cab : Day Cab
  - -Chassis : Chassis

Operate Truck

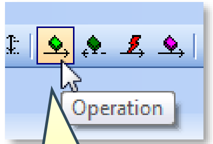
Description

Driver loads the bay

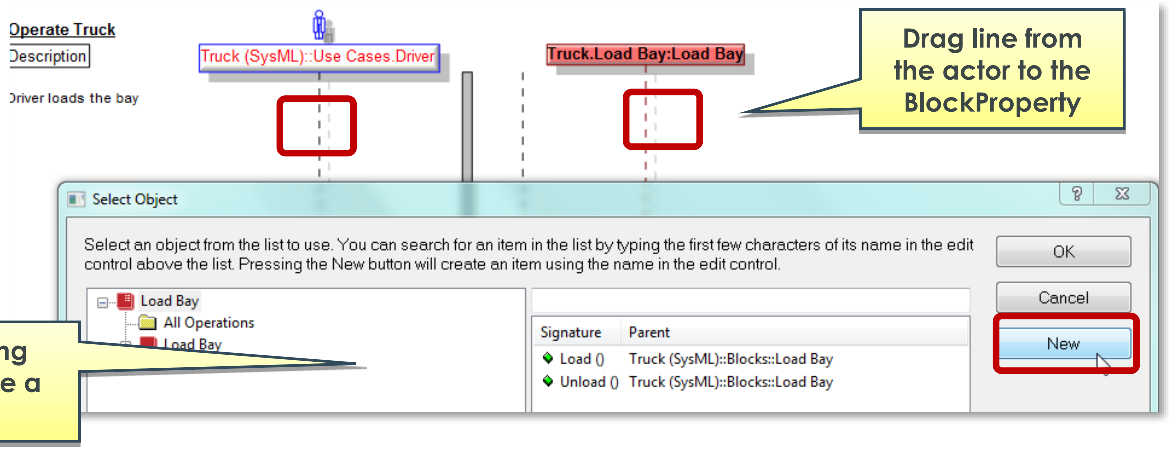
Truck (SysML)::Use Cases.Driver

Truck.Load Bay:Load Bay

Lifeline



Select Operation to add a message



Drag line from the actor to the BlockProperty

Select an object from the list to use. You can search for an item in the list by typing the first few characters of its name in the edit control above the list. Pressing the New button will create an item using the name in the edit control.

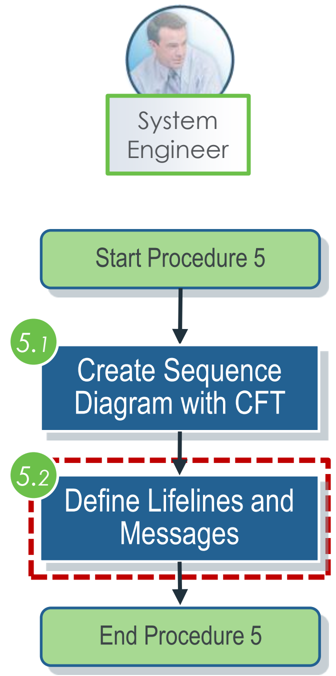
- Load Bay
  - All Operations
  - Load Bay

| Signature | Parent |
|---|---|
| Load () | Truck (SysML)::Blocks::Load Bay |
| Unload () | Truck (SysML)::Blocks::Load Bay |

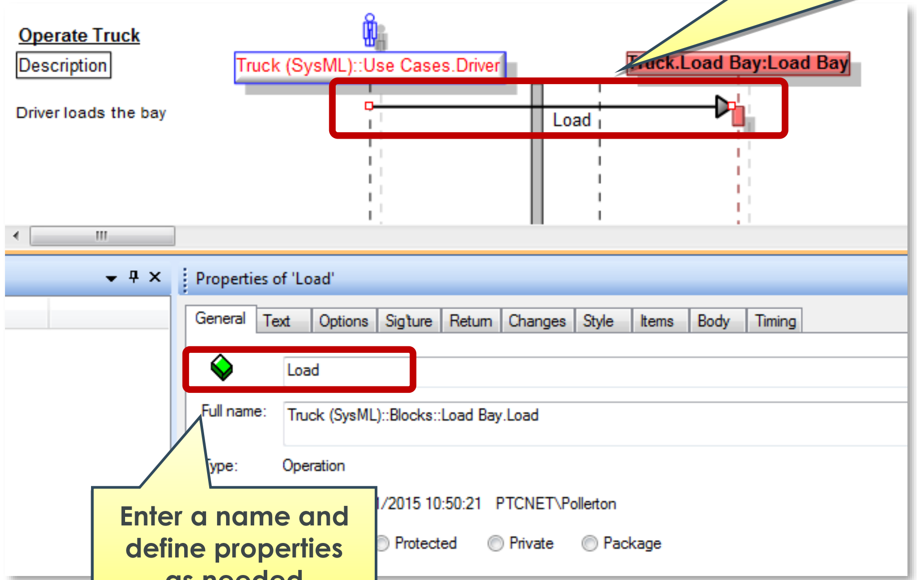
Can re-use existing operation or create a new operation

# MODEL BASED SYSTEM ENGINEERING

## Define Lifelines and Messages


System Engineer
Start Procedure 5
5.1
Create Sequence Diagram with CFT
5.2
Define Lifelines and Messages
End Procedure 5

The operation will be added to diagram



Driver loads the bay

Load

Full name: Truck (SysML)::Blocks::Load Bay.Load

Type: Operation

/2015 10:50:21 PTCNET\Pollerton

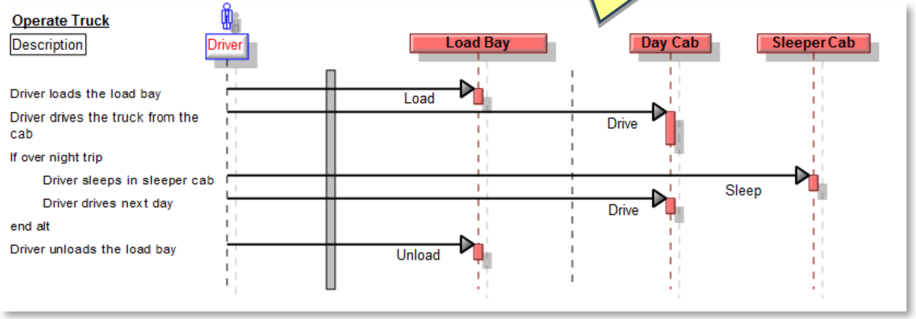
○ Protected ○ Private ○ Package

Enter a name and define properties as needed

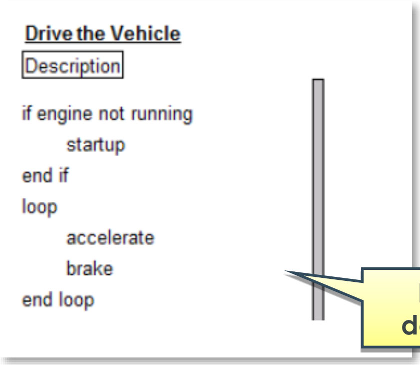
**Note**

Operations describe functions that should be provided by the system

Continue to add operations to define the sequence of interactions



Driver loads the load bay
Driver drives the truck from the cab
If over night trip
    Driver sleeps in sleeper cab
    Driver drives next day
end alt
Driver unloads the load bay

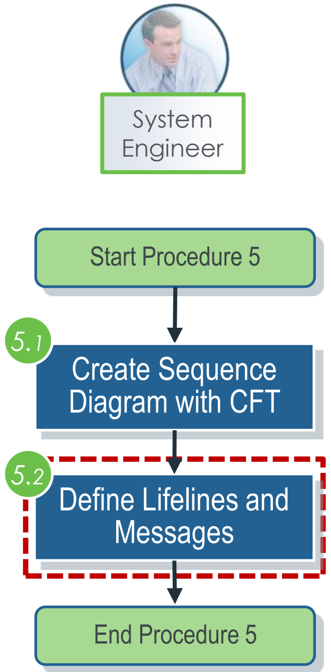


```
if engine not running
    startup
end if
loop
    accelerate
    brake
end loop
```

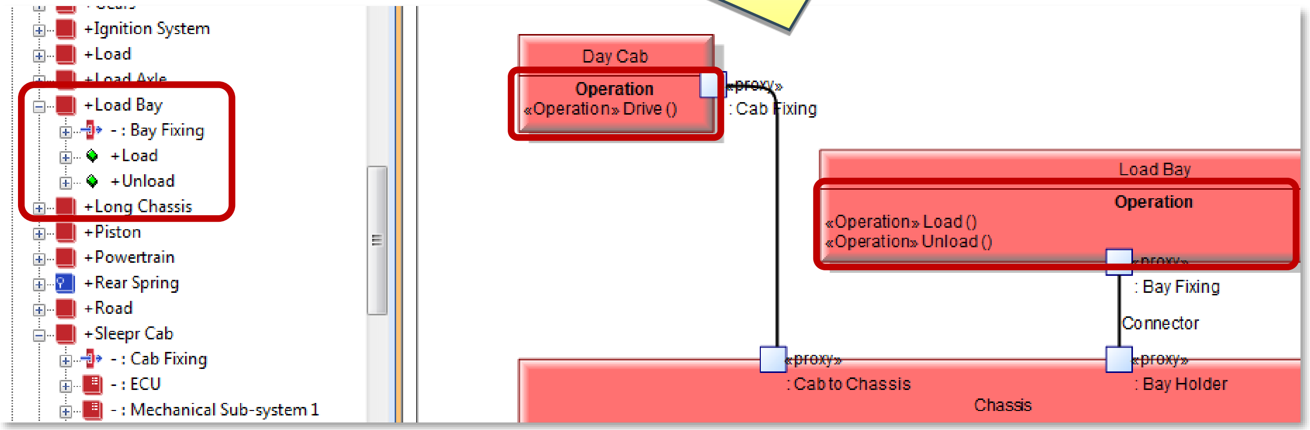
Logical constructs can be defined within the sequence

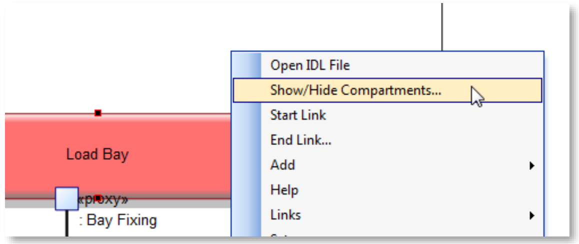
# MODEL BASED SYSTEM ENGINEERING

## ► Define Lifelines and Messages

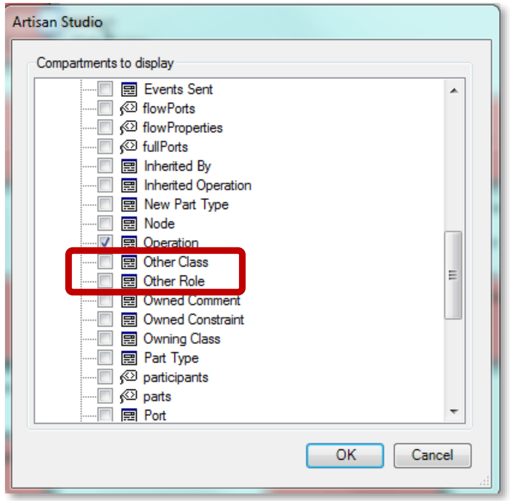

System Engineer
Start Procedure 5
5.1
Create Sequence Diagram with CFT
5.2
Define Lifelines and Messages
End Procedure 5

Operations become part of the target block and are displayed on diagrams (if compartment is visible)





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


Artisan Studio

Compartments to display

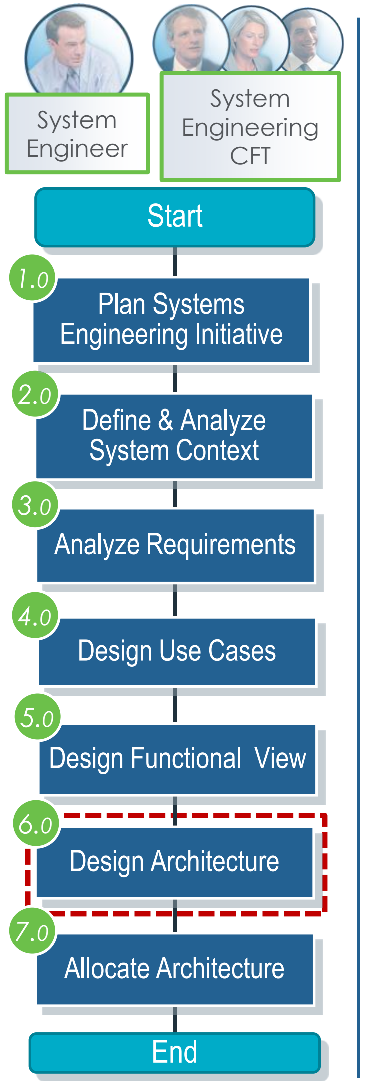
- ☐ Events Sent
- ☐ flowPorts
- ☐ flowProperties
- ☐ fullPorts
- ☐ Inherited By
- ☐ Inherited Operation
- ☐ New Part Type
- ☐ Node
- ☑ Operation
- ☐ Other Class
- ☐ Other Role
- ☐ Owned Comment
- ☐ Owned Constraint
- ☐ Owning Class
- ☐ Part Type
- ☐ participants
- ☐ parts
- ☐ Port

OK Cancel

# DESIGN ARCHITECTURE

# MODEL BASED SYSTEM ENGINEERING BEST PRACTICE PROCEDURE

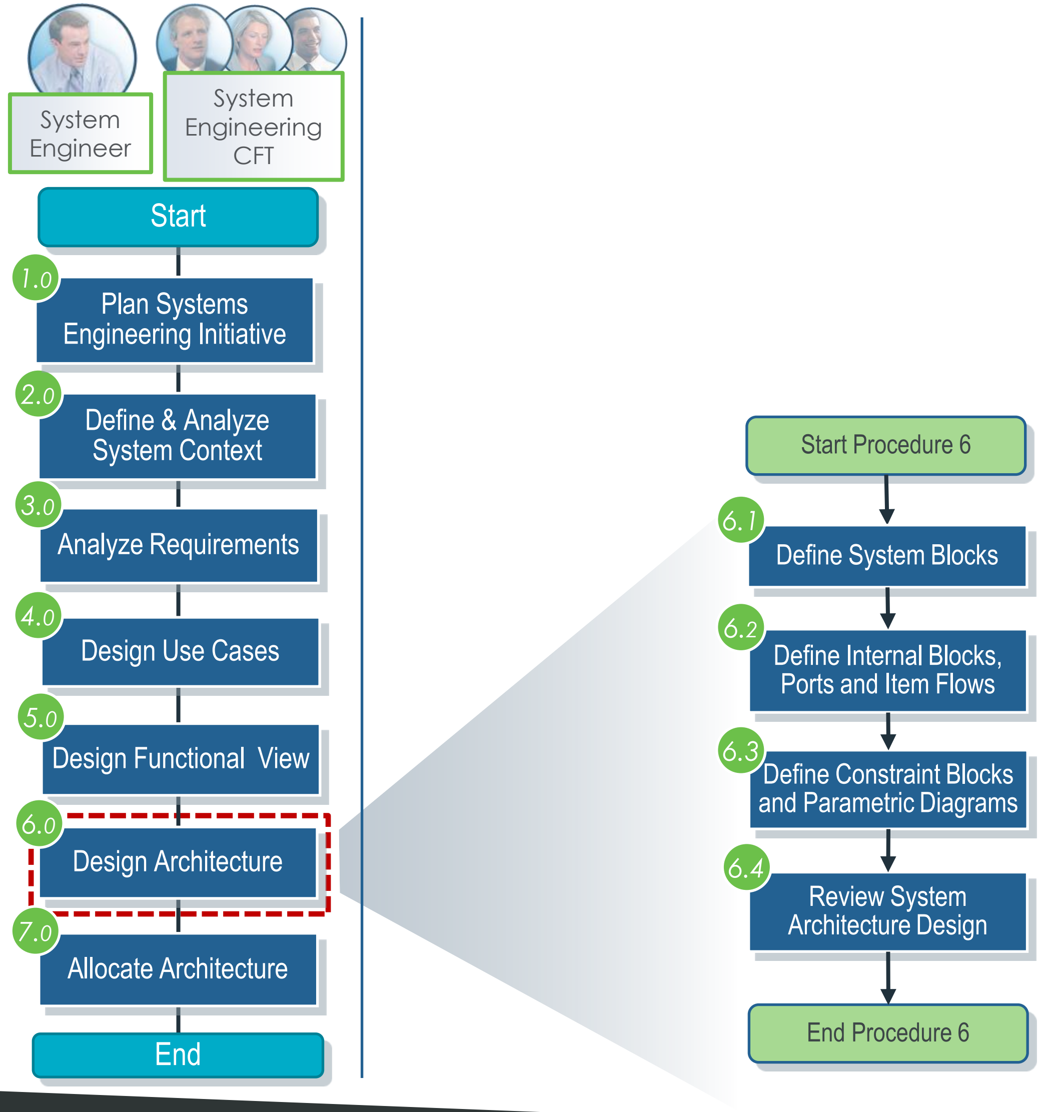
## ► Design Architecture



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

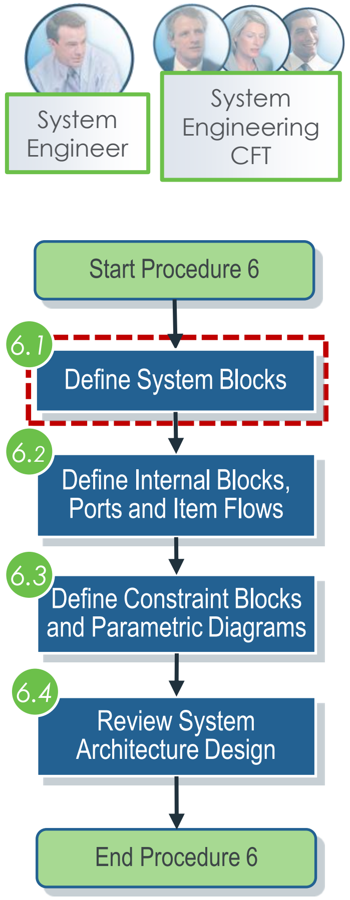
# MODEL BASED SYSTEM ENGINEERING

## ▶ Design Architecture



# MODEL BASED SYSTEM ENGINEERING

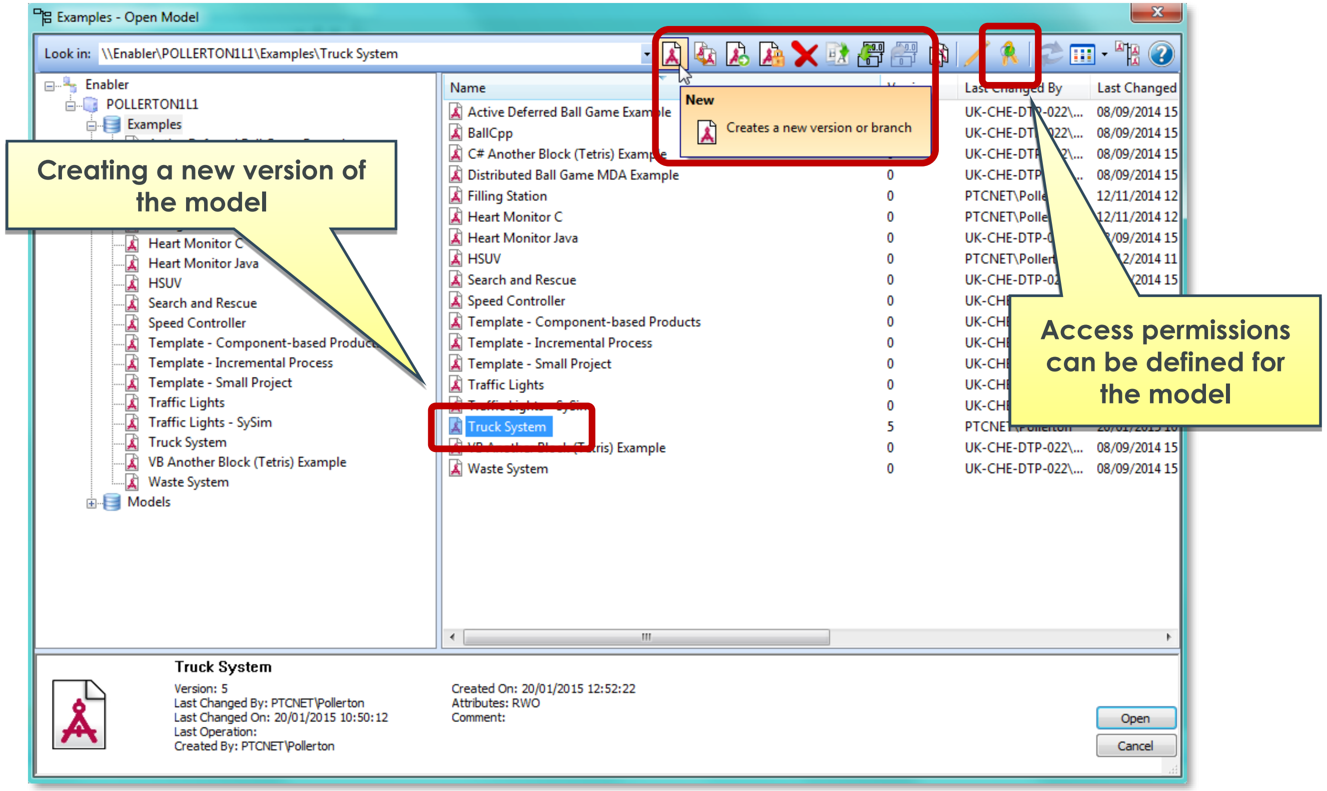
## ► Update System Model



System Engineer
System Engineering CFT
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**

**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**



| Name | Version | Last Changed By | Last Changed |
|---|---|---|---|
| Active Deferred Ball Game Example | | UK-CHE-DTP-022\... | 08/09/2014 15 |
| BallCpp | | UK-CHE-DT 022\... | 08/09/2014 15 |
| C# Another Block (Tetris) Example | | UK-CHE-DTP 2\... | 08/09/2014 15 |
| Distributed Ball Game MDA Example | 0 | UK-CHE-DTP ... | 08/09/2014 15 |
| Filling Station | 0 | PTCNET\Polle | 12/11/2014 12 |
| Heart Monitor C | 0 | PTCNET\Polle | 12/11/2014 12 |
| Heart Monitor Java | 0 | UK-CHE-DTP-0 | /09/2014 15 |
| HSUV | 0 | PTCNET\Pollert | 2/2014 11 |
| Search and Rescue | 0 | UK-CHE-DTP-02 | 2014 15 |
| Speed Controller | 0 | UK-CHE | |
| Template - Component-based Products | 0 | UK-CHE | |
| Template - Incremental Process | 0 | UK-CHE | |
| Template - Small Project | 0 | UK-CHE | |
| Traffic Lights | 0 | UK-CHE | |
| Traffic Lights - SySim | 0 | UK-CHE | |
| Truck System | 5 | PTCNET\Pollerton | |
| VB Another Block (Tetris) Example | 0 | UK-CHE-DTP-022\... | 08/09/2014 15 |
| Waste System | 0 | UK-CHE-DTP-022\... | 08/09/2014 15 |

**Truck System**

Version: 5
Last Changed By: PTCNET\Pollerton
Last Changed On: 20/01/2015 10:50:12
Last Operation:
Created By: PTCNET\Pollerton

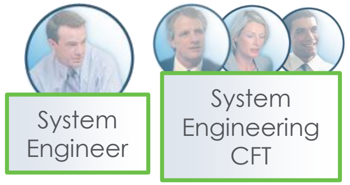
Created On: 20/01/2015 12:52:22
Attributes: RWO
Comment:

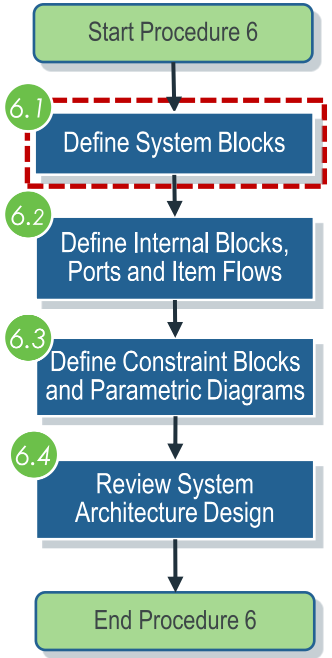
**Creating a new version of the model**

**Access permissions can be defined for the model**

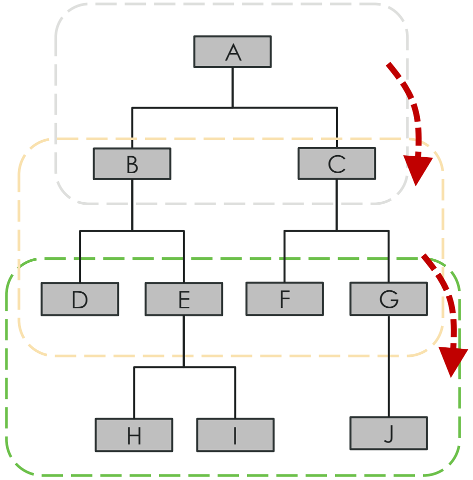
# MODEL BASED SYSTEM ENGINEERING

## ► Define System Blocks

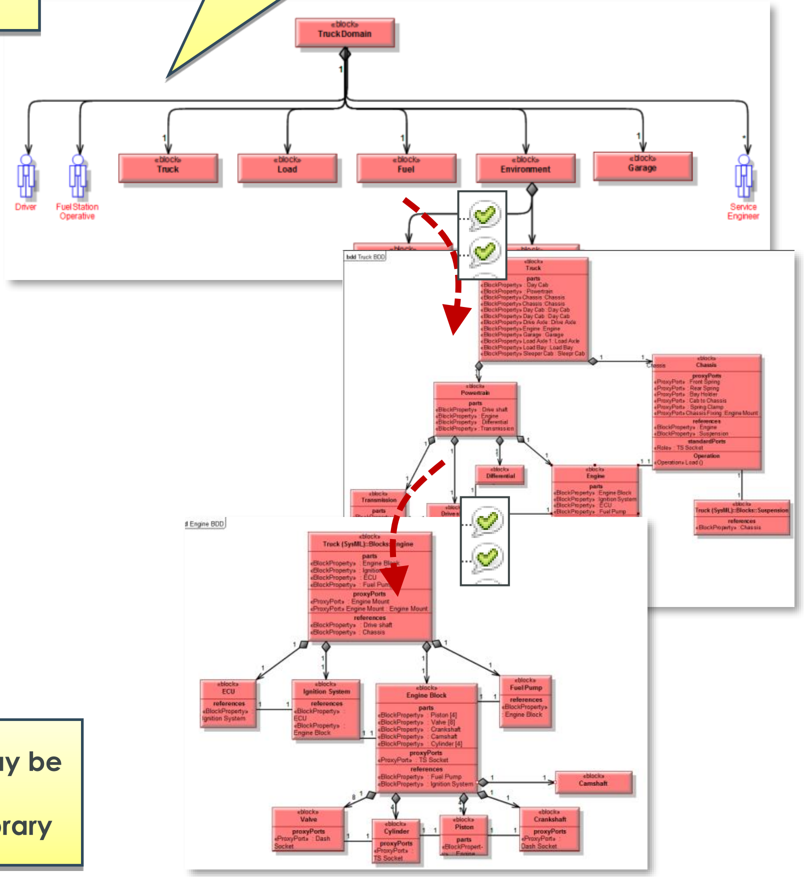



Start Procedure 6
6.1
Define System Blocks
6.2
Define Internal Blocks, Ports and Item Flows
6.3
Define Constraint Blocks and Parametric Diagrams
6.4
Review System Architecture Design
End Procedure 6

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



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

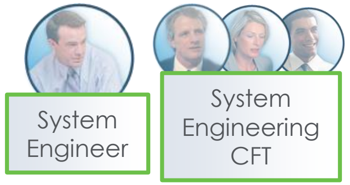


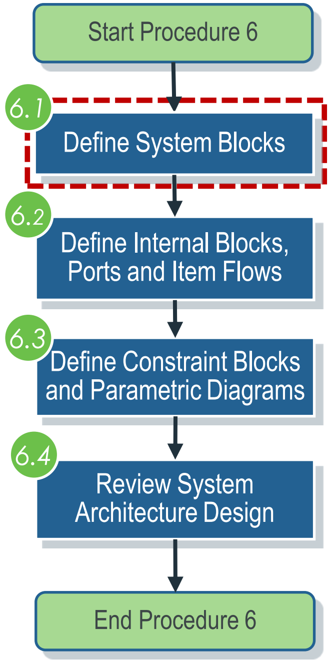
**Note**

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

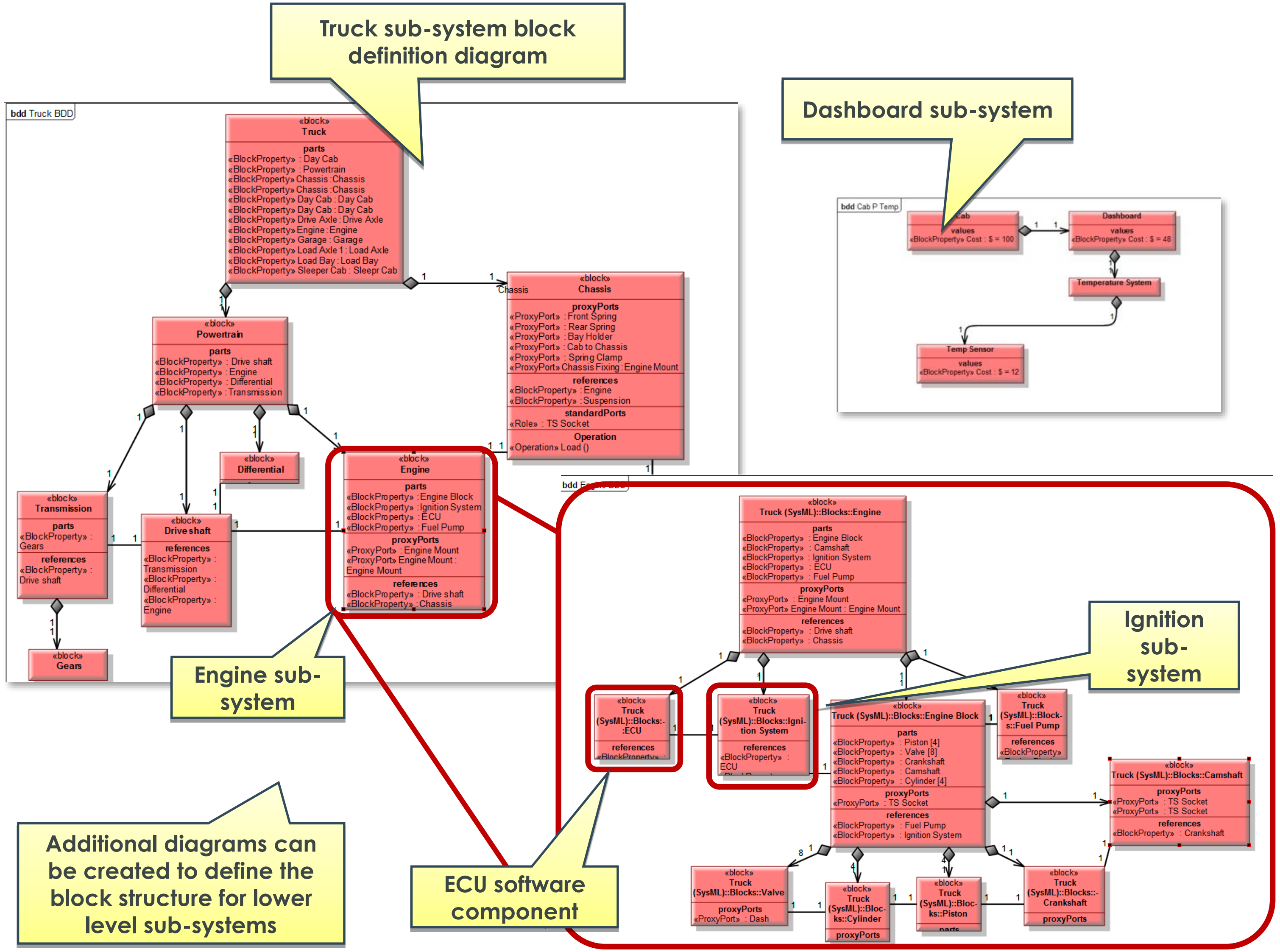
# MODEL BASED SYSTEM ENGINEERING

## ► 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



bdd Truck BDD
«block»
Truck
parts
«BlockProperty» : Day Cab
«BlockProperty» : Powertrain
«BlockProperty» Chassis : Chassis
«BlockProperty» Chassis : Chassis
«BlockProperty» Day Cab : Day Cab
«BlockProperty» Day Cab : Day Cab
«BlockProperty» Drive Axle : Drive Axle
«BlockProperty» Engine : Engine
«BlockProperty» Garage : Garage
«BlockProperty» Load Axle 1 : Load Axle
«BlockProperty» Load Bay : Load Bay
«BlockProperty» Sleeper Cab : Sleepr Cab
Chassis
«block»
Chassis
proxyPorts
«ProxyPort» : Front Spring
«ProxyPort» : Rear Spring
«ProxyPort» : Bay Holder
«ProxyPort» : Cab to Chassis
«ProxyPort» : Spring Clamp
«ProxyPort» Chassis Fixing : Engine Mount
references
«BlockProperty» : Engine
«BlockProperty» : Suspension
standardPorts
«Role» : TS Socket
Operation
«Operation» Load ()
«block»
Powertrain
parts
«BlockProperty» : Drive shaft
«BlockProperty» : Engine
«BlockProperty» : Differential
«BlockProperty» : Transmission
«block»
Differential
«block»
Transmission
parts
«BlockProperty» : Gears
references
«BlockProperty» : Drive shaft
«block»
Drive shaft
references
«BlockProperty» : Transmission
«BlockProperty» : Differential
«BlockProperty» : Engine
«block»
Engine
parts
«BlockProperty» : Engine Block
«BlockProperty» : Ignition System
«BlockProperty» : ECU
«BlockProperty» : Fuel Pump
proxyPorts
«ProxyPort» : Engine Mount
«ProxyPort» Engine Mount : Engine Mount
references
«BlockProperty» : Drive shaft
«BlockProperty» : Chassis
«block»
Gears
bdd Cab P Temp
Cab
values
«BlockProperty» Cost : $ = 100
Dashboard
values
«BlockProperty» Cost : $ = 48
Temperature System
Temp Sensor
values
«BlockProperty» Cost : $ = 12
bdd
«block»
Truck (SysML)::Blocks::Engine
parts
«BlockProperty» : Engine Block
«BlockProperty» : Camshaft
«BlockProperty» : Ignition System
«BlockProperty» : ECU
«BlockProperty» : Fuel Pump
proxyPorts
«ProxyPort» : Engine Mount
«ProxyPort» Engine Mount : Engine Mount
references
«BlockProperty» : Drive shaft
«BlockProperty» : Chassis
«block»
Truck (SysML)::Blocks::-:ECU
references
«BlockProperty» :
«block»
Truck (SysML)::Blocks::Igni-tion System
references
«BlockProperty» : ECU
«block»
Truck (SysML)::Blocks::Engine Block
parts
«BlockProperty» : Piston [4]
«BlockProperty» : Valve [8]
«BlockProperty» : Crankshaft
«BlockProperty» : Camshaft
«BlockProperty» : Cylinder [4]
proxyPorts
«ProxyPort» : TS Socket
references
«BlockProperty» : Fuel Pump
«BlockProperty» : Ignition System
«block»
Truck (SysML)::Block-s::Fuel Pump
references
«BlockProperty»
«block»
Truck (SysML)::Blocks::Camshaft
proxyPorts
«ProxyPort» : TS Socket
«ProxyPort» : TS Socket
references
«BlockProperty» : Crankshaft
«block»
Truck (SysML)::Blocks::Valve
proxyPorts
«ProxyPort» : Dash
«block»
Truck (SysML)::Bloc-ks::Cylinder
proxyPorts
«block»
Truck (SysML)::Bloc-ks::Piston
parts
«block»
Truck (SysML)::Blocks::-Crankshaft
proxyPorts
1
8
4

Truck sub-system block definition diagram

Dashboard sub-system

Engine sub-system

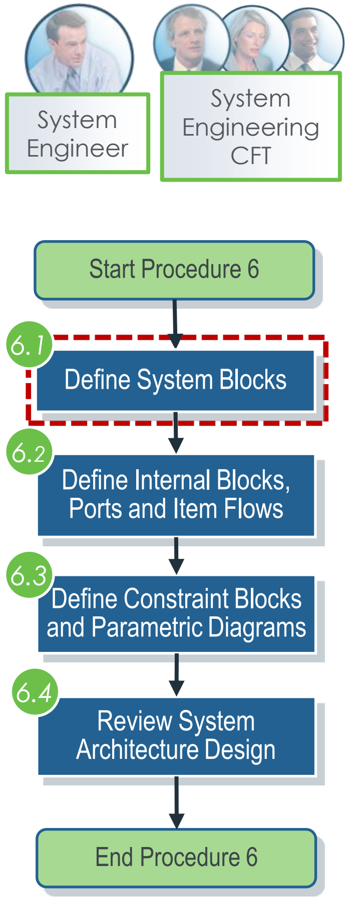
Ignition sub-system

ECU software component

Additional diagrams can be created to define the block structure for lower level sub-systems

# MODEL BASED SYSTEM ENGINEERING

## ▶ Define System Blocks


System Engineer

System Engineering CFT

Start Procedure 6

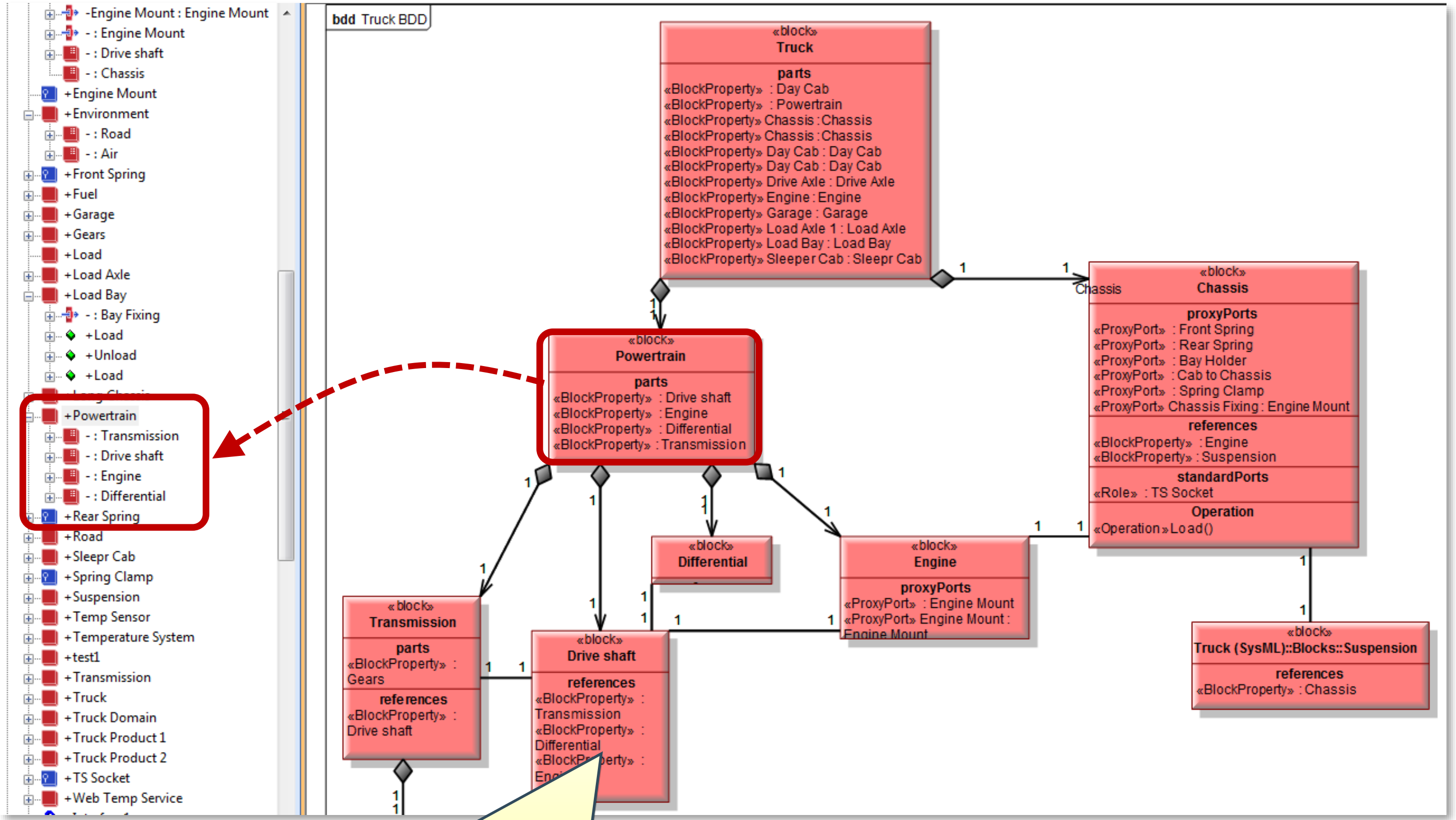
6.1 Define System Blocks

6.2 Define Internal Blocks, Ports and Item Flows

6.3 Define Constraint Blocks and Parametric Diagrams

6.4 Review System Architecture Design

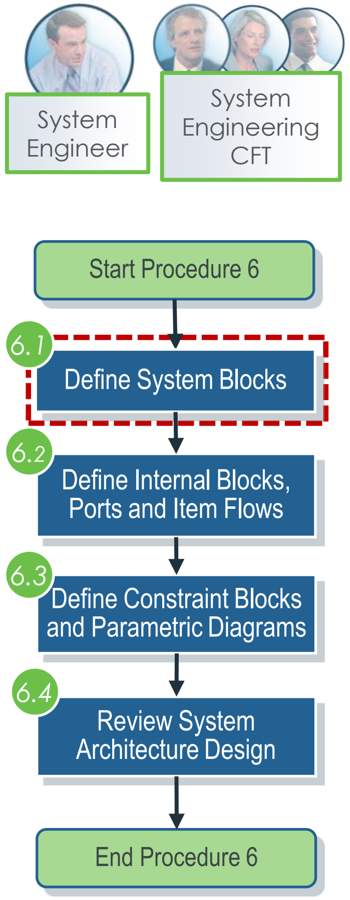
End Procedure 6


-Engine Mount : Engine Mount
- : Engine Mount
- : Drive shaft
- : Chassis
+Engine Mount
+Environment
- : Road
- : Air
+Front Spring
+Fuel
+Garage
+Gears
+Load
+Load Axle
+Load Bay
- : Bay Fixing
+Load
+Unload
+Load
+Powertrain
- : Transmission
- : Drive shaft
- : Engine
- : Differential
+Rear Spring
+Road
+Sleepr Cab
+Spring Clamp
+Suspension
+Temp Sensor
+Temperature System
+test1
+Transmission
+Truck
+Truck Domain
+Truck Product 1
+Truck Product 2
+TS Socket
+Web Temp Service
bdd Truck BDD
«block» Truck
parts
«BlockProperty» : Day Cab
«BlockProperty» : Powertrain
«BlockProperty» Chassis : Chassis
«BlockProperty» Chassis : Chassis
«BlockProperty» Day Cab : Day Cab
«BlockProperty» Day Cab : Day Cab
«BlockProperty» Drive Axle : Drive Axle
«BlockProperty» Engine : Engine
«BlockProperty» Garage : Garage
«BlockProperty» Load Axle 1 : Load Axle
«BlockProperty» Load Bay : Load Bay
«BlockProperty» Sleeper Cab : Sleeper Cab
«block» Chassis
Chassis
proxyPorts
«ProxyPort» : Front Spring
«ProxyPort» : Rear Spring
«ProxyPort» : Bay Holder
«ProxyPort» : Cab to Chassis
«ProxyPort» : Spring Clamp
«ProxyPort» Chassis Fixing : Engine Mount
references
«BlockProperty» : Engine
«BlockProperty» : Suspension
standardPorts
«Role» : TS Socket
Operation
«Operation» Load()
«block» Powertrain
parts
«BlockProperty» : Drive shaft
«BlockProperty» : Engine
«BlockProperty» : Differential
«BlockProperty» : Transmission
«block» Differential
«block» Engine
proxyPorts
«ProxyPort» : Engine Mount
«ProxyPort» Engine Mount : Engine Mount
«block» Transmission
parts
«BlockProperty» : Gears
references
«BlockProperty» : Drive shaft
«block» Drive shaft
references
«BlockProperty» : Transmission
«BlockProperty» : Differential
«BlockProperty» : Engine
«block» Truck (SysML)::Blocks::Suspension
references
«BlockProperty» : Chassis
1
1
1
1
1
1
1

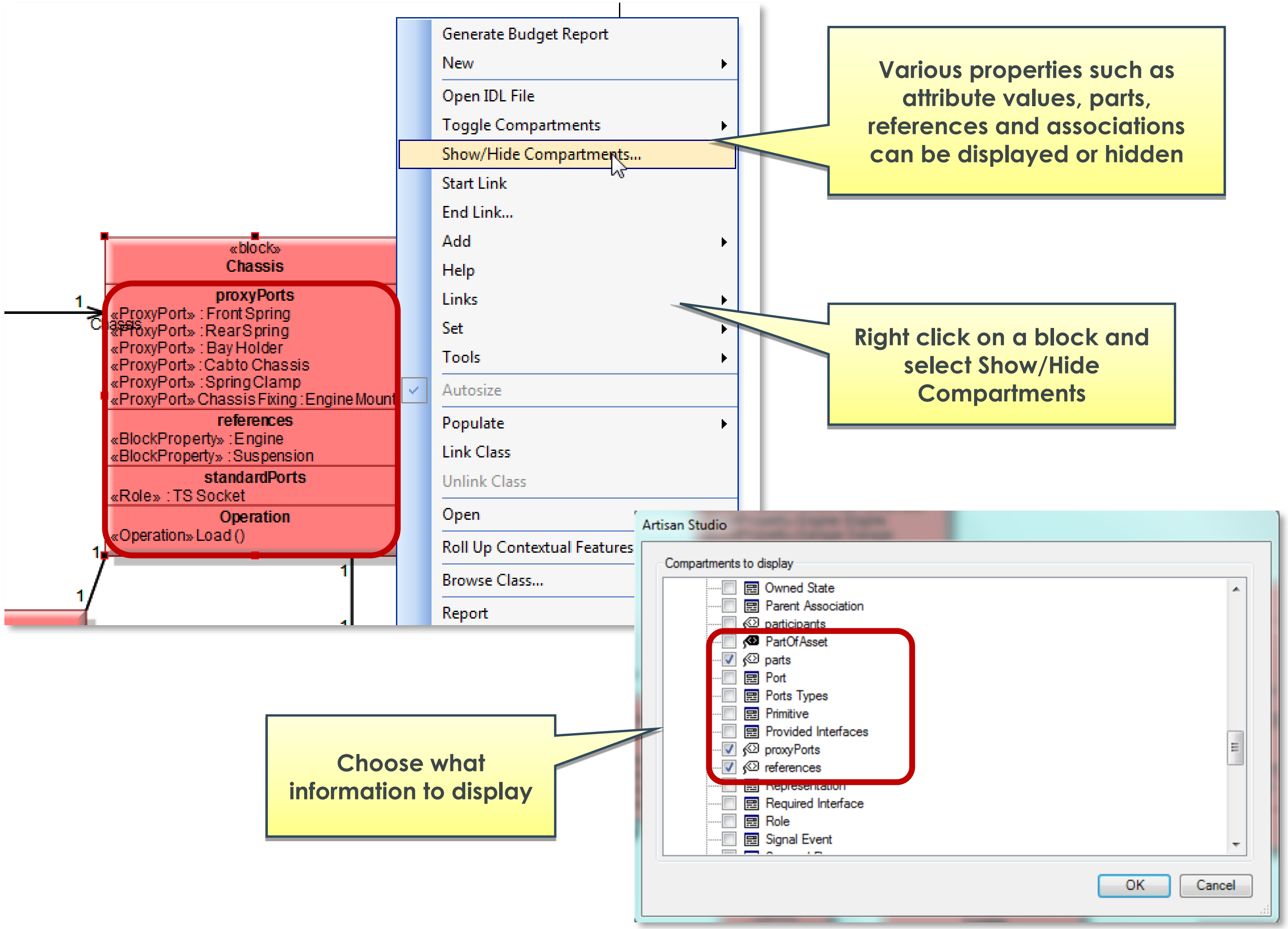
Note that BlockProperty entities are created where blocks are associated. (Composite aggregations are defined as parts on the parent block). This is also displayed in the Package view

# MODEL BASED SYSTEM ENGINEERING

## ► Define System Blocks



System Engineer
System Engineering CFT
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



«block»
**Chassis**

**proxyPorts**
«ProxyPort» : Front Spring
«ProxyPort» : Rear Spring
«ProxyPort» : Bay Holder
«ProxyPort» : Cabto Chassis
«ProxyPort» : Spring Clamp
«ProxyPort» Chassis Fixing : Engine Mount

**references**
«BlockProperty» : Engine
«BlockProperty» : Suspension

**standardPorts**
«Role» : TS Socket

**Operation**
«Operation» Load ()

Artisan Studio

Compartments to display

- □ Owned State
- □ Parent Association
- □ participants
- □ PartOfAsset
- ☑ parts
- □ Port
- □ Ports Types
- □ Primitive
- □ Provided Interfaces
- ☑ proxyPorts
- ☑ references
- □ Representation
- □ Required Interface
- □ Role
- □ Signal Event

OK Cancel

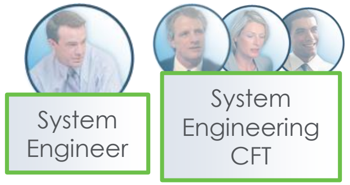
Various properties such as attribute values, parts, references and associations can be displayed or hidden

Right click on a block and select Show/Hide Compartments

Choose what information to display

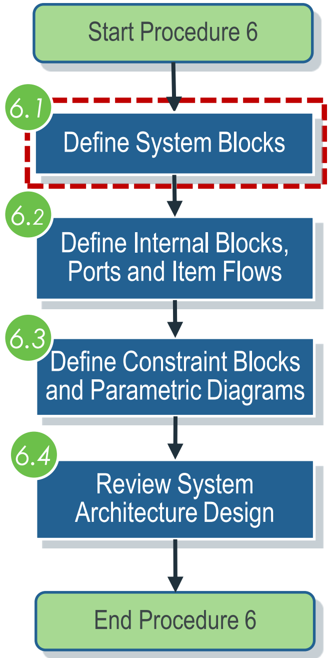
# MODEL BASED SYSTEM ENGINEERING

## ► Define System Blocks



System Engineer

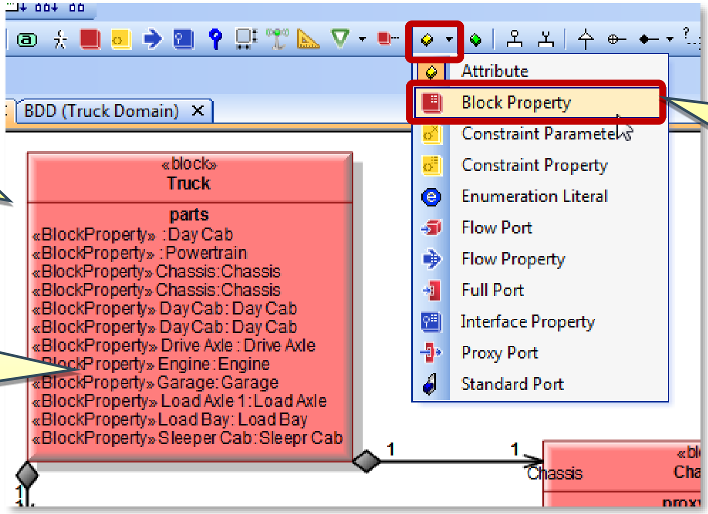
System Engineering CFT



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

Block Properties can be added to blocks to capture attribute values

Click on the Block that the attribute will be added to

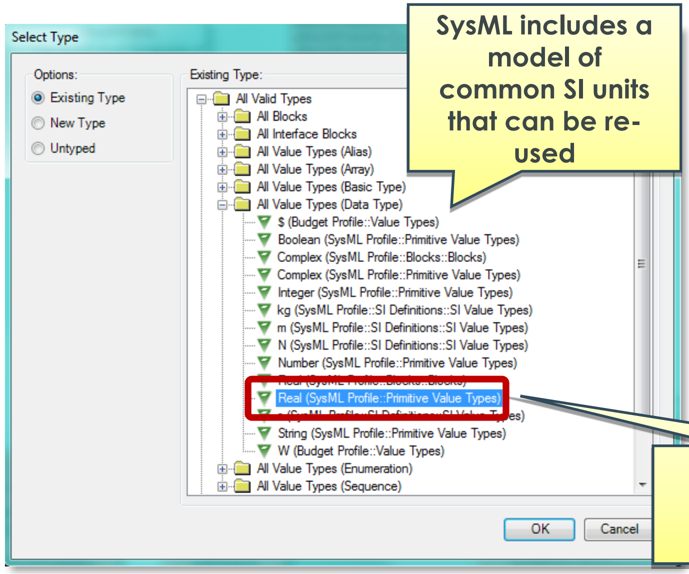


«block»
Truck

parts
«BlockProperty» :Day Cab
«BlockProperty» :Powertrain
«BlockProperty» Chassis:Chassis
«BlockProperty» Chassis:Chassis
«BlockProperty» Day Cab: Day Cab
«BlockProperty» Day Cab: Day Cab
«BlockProperty» Drive Axle : Drive Axle
«BlockProperty» Engine:Engine
«BlockProperty» Garage:Garage
«BlockProperty» Load Axle 1:Load Axle
«BlockProperty» Load Bay: Load Bay
«BlockProperty» Sleeper Cab:Sleepr Cab

Select the Attribute menu and choose Block Property

Name the property. It is displayed in the package hierarchy and diagrams



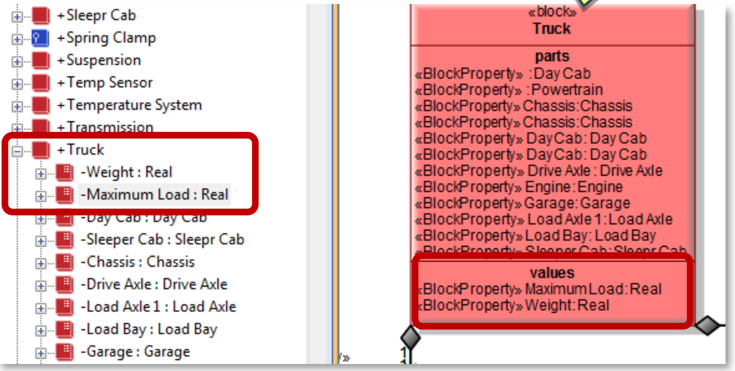
Select Type

Options:
● Existing Type
○ New Type
○ Untyped

Existing Type:
- All Valid Types
  - All Blocks
  - All Interface Blocks
  - All Value Types (Alias)
  - All Value Types (Array)
  - All Value Types (Basic Type)
  - All Value Types (Data Type)
    - $ (Budget Profile::Value Types)
    - Boolean (SysML Profile::Primitive Value Types)
    - Complex (SysML Profile::Blocks::Blocks)
    - Complex (SysML Profile::Primitive Value Types)
    - Integer (SysML Profile::Primitive Value Types)
    - kg (SysML Profile::SI Definitions::SI Value Types)
    - m (SysML Profile::SI Definitions::SI Value Types)
    - N (SysML Profile::SI Definitions::SI Value Types)
    - Number (SysML Profile::Primitive Value Types)
    - Real (SysML Profile::Blocks::Blocks)
    - Real (SysML Profile::Primitive Value Types)
    - s (SysML Profile::SI Definitions::SI Value Types)
    - String (SysML Profile::Primitive Value Types)
    - W (Budget Profile::Value Types)
  - All Value Types (Enumeration)
  - All Value Types (Sequence)

OK | Cancel

SysML includes a model of common SI units that can be re-used



- +Sleepr Cab
- +Spring Clamp
- +Suspension
- +Temp Sensor
- +Temperature System
- +Transmission
- +Truck
  - -Weight : Real
  - -Maximum Load : Real
  - -Day Cab : Day Cab
  - -Sleeper Cab : Sleepr Cab
  - -Chassis : Chassis
  - -Drive Axle : Drive Axle
  - -Load Axle 1 : Load Axle
  - -Load Bay : Load Bay
  - -Garage : Garage

«block»
Truck

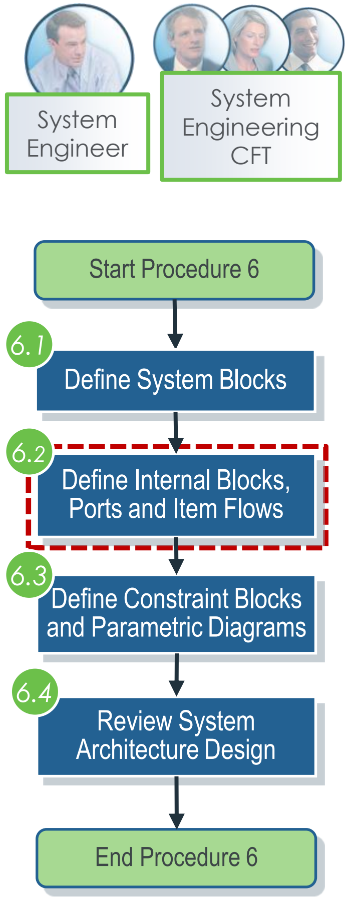
parts
«BlockProperty» :Day Cab
«BlockProperty» :Powertrain
«BlockProperty» Chassis:Chassis
«BlockProperty» Chassis:Chassis
«BlockProperty» Day Cab: Day Cab
«BlockProperty» Day Cab: Day Cab
«BlockProperty» Drive Axle : Drive Axle
«BlockProperty» Engine:Engine
«BlockProperty» Garage:Garage
«BlockProperty» Load Axle 1:Load Axle
«BlockProperty» Load Bay: Load Bay
«BlockProperty» Sleeper Cab:Sleepr Cab

values
«BlockProperty» Maximum Load:Real
«BlockProperty» Weight:Real

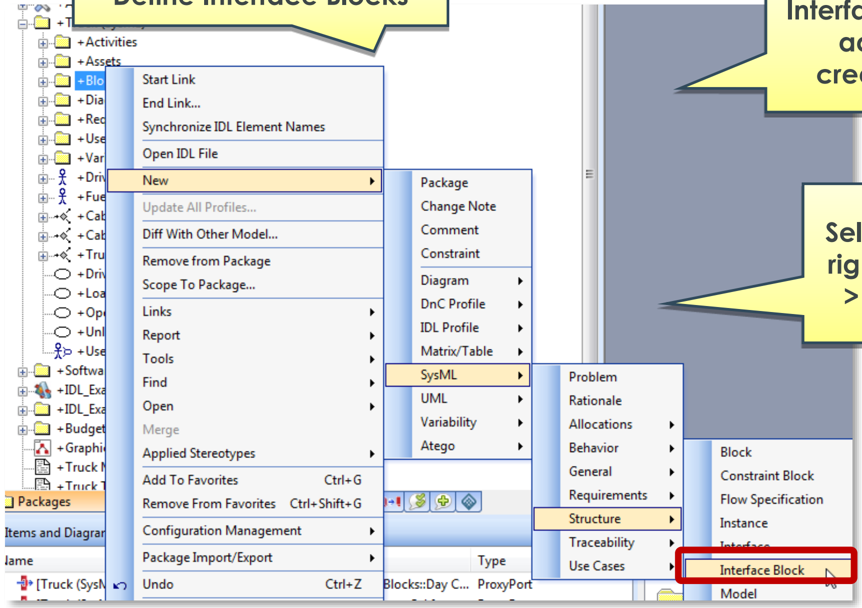
Select a data type for the attribute (Real primitive type selected in image)

# MODEL BASED SYSTEM ENGINEERING

## ▶ Define Internal Blocks, Ports and Item Flows


System Engineer
System Engineering CFT
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

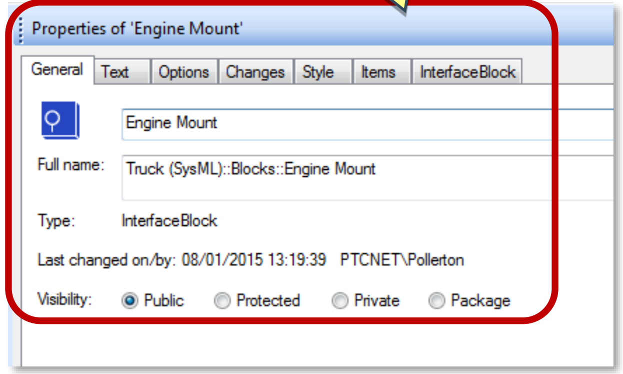
**Define Interface Blocks**



**Interface blocks do not need to be added to diagrams, so are created directly in a package**

**Select a suitable package and right click. Select New > SysML > Structure > Interface Block**

**Enter a name and specify options**



Properties of 'Engine Mount'

General | Text | Options | Changes | Style | Items | InterfaceBlock

Engine Mount

Full name: Truck (SysML)::Blocks::Engine Mount

Type: InterfaceBlock

Last changed on/by: 08/01/2015 13:19:39 PTCNET\Pollerton

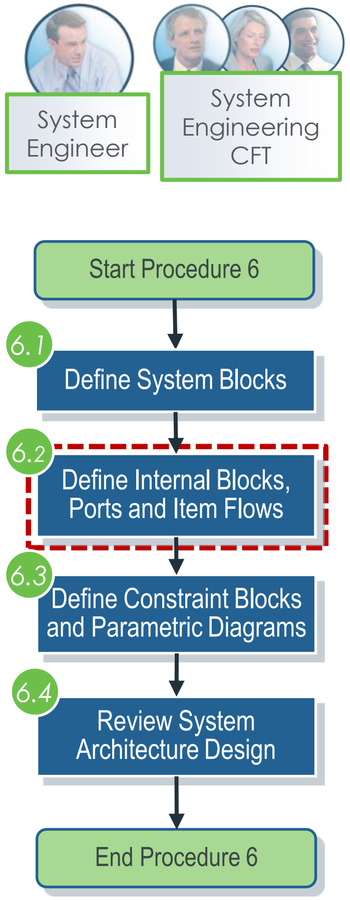
Visibility: ● Public ○ Protected ○ Private ○ Package

**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)**

# MODEL BASED SYSTEM ENGINEERING

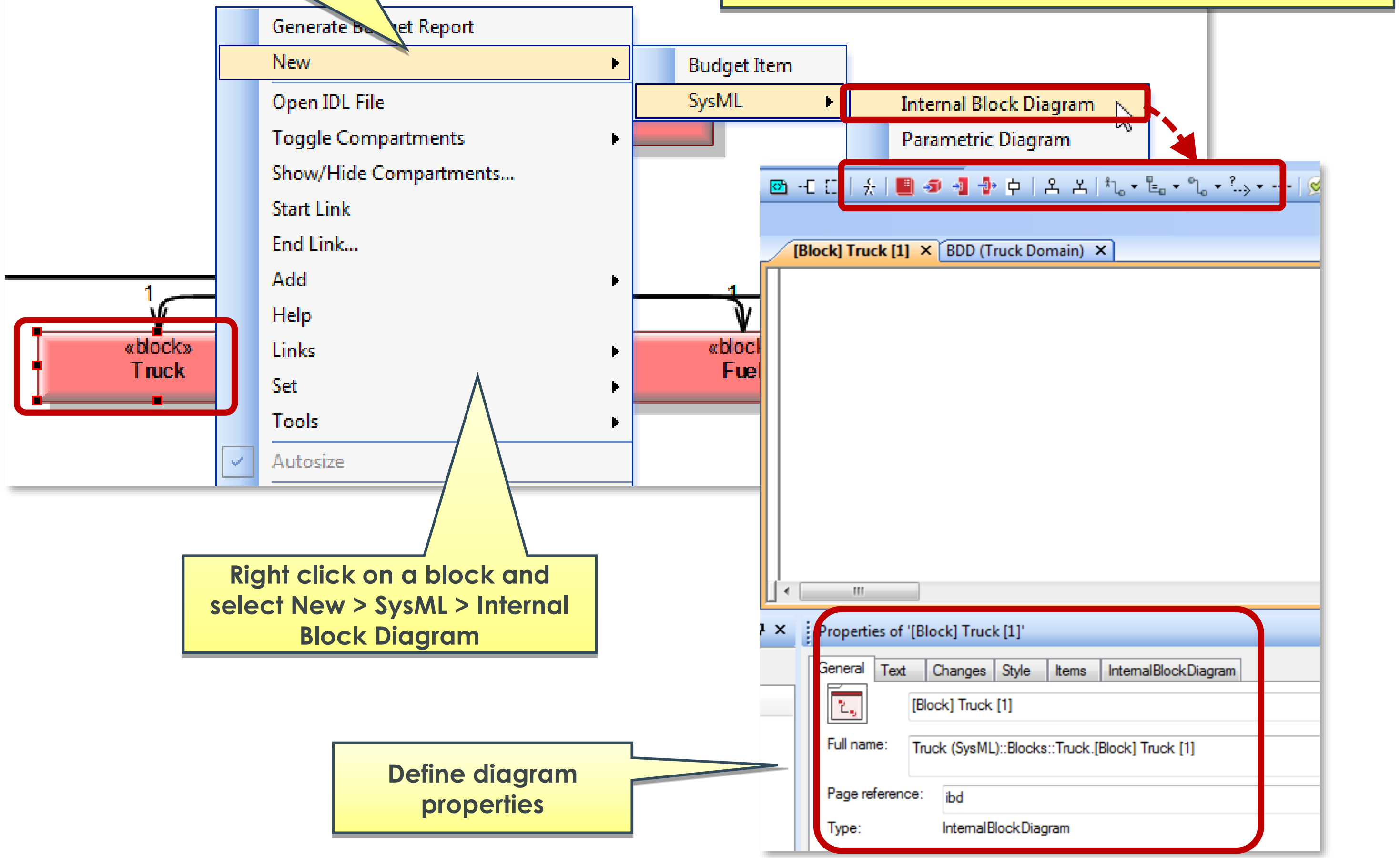
## ▶ Define Internal Blocks, Ports and Item Flows


System Engineer
System Engineering CFT
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

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



Properties of '[Block] Truck [1]'

General | Text | Changes | Style | Items | InternalBlockDiagram

[Block] Truck [1]

Full name: Truck (SysML)::Blocks::Truck.[Block] Truck [1]

Page reference: ibd

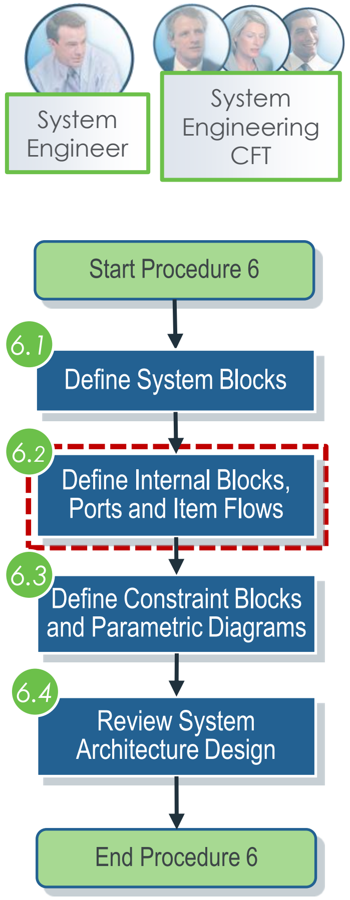
Type: InternalBlockDiagram

Right click on a block and select New > SysML > Internal Block Diagram

Define diagram properties

# MODEL BASED SYSTEM ENGINEERING

## ▶ Define Internal Blocks, Ports and Item Flows



System Engineer

System Engineering CFT

Start Procedure 6

6.1 Define System Blocks

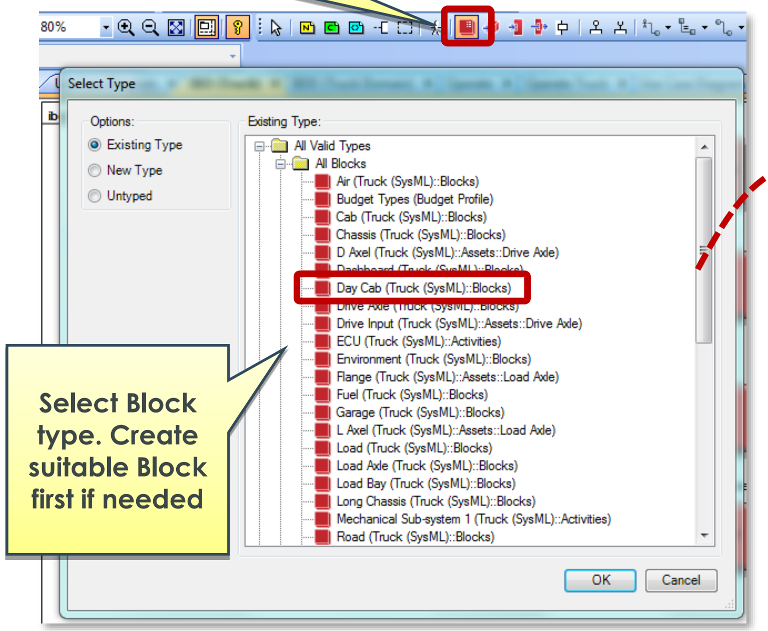
6.2 Define Internal Blocks, Ports and Item Flows

6.3 Define Constraint Blocks and Parametric Diagrams

6.4 Review System Architecture Design

End Procedure 6

**Add a Block Property**



Select Type

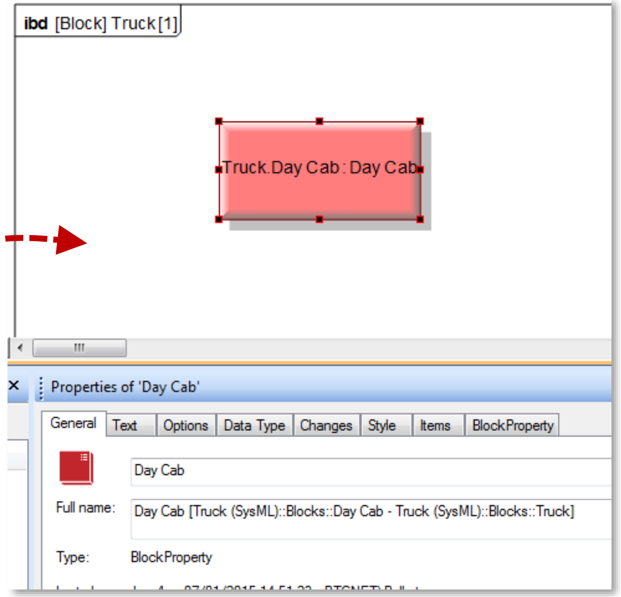
Options:

- ● Existing Type
- ○ New Type
- ○ Untyped

Existing Type:

- All Valid Types
  - All Blocks
    - Air (Truck (SysML)::Blocks)
    - Budget Types (Budget Profile)
    - Cab (Truck (SysML)::Blocks)
    - Chassis (Truck (SysML)::Blocks)
    - D Axel (Truck (SysML)::Assets::Drive Axle)
    - Dashboard (Truck (SysML)::Blocks)
    - Day Cab (Truck (SysML)::Blocks)
    - Drive Axle (Truck (SysML)::Blocks)
    - Drive Input (Truck (SysML)::Assets::Drive Axle)
    - ECU (Truck (SysML)::Activities)
    - Environment (Truck (SysML)::Blocks)
    - Flange (Truck (SysML)::Assets::Load Axle)
    - Fuel (Truck (SysML)::Blocks)
    - Garage (Truck (SysML)::Blocks)
    - L Axel (Truck (SysML)::Assets::Load Axle)
    - Load (Truck (SysML)::Blocks)
    - Load Axle (Truck (SysML)::Blocks)
    - Load Bay (Truck (SysML)::Blocks)
    - Long Chassis (Truck (SysML)::Blocks)
    - Mechanical Sub-system 1 (Truck (SysML)::Activities)
    - Road (Truck (SysML)::Blocks)

**Select Block type. Create suitable Block first if needed**



ibd [Block] Truck[1]

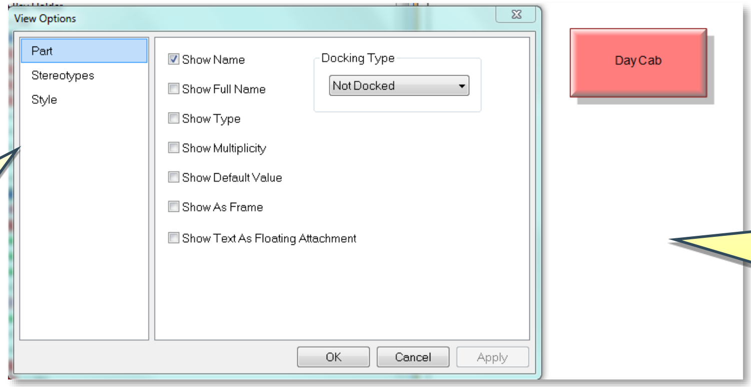
Truck.Day Cab : Day Cab

Properties of 'Day Cab'

Day Cab

Full name: Day Cab [Truck (SysML)::Blocks::Day Cab - Truck (SysML)::Blocks::Truck]

Type: BlockProperty



View Options

- ☑ Show Name
- □ Show Full Name
- □ Show Type
- □ Show Multiplicity
- □ Show Default Value
- □ Show As Frame
- □ Show Text As Floating Attachment

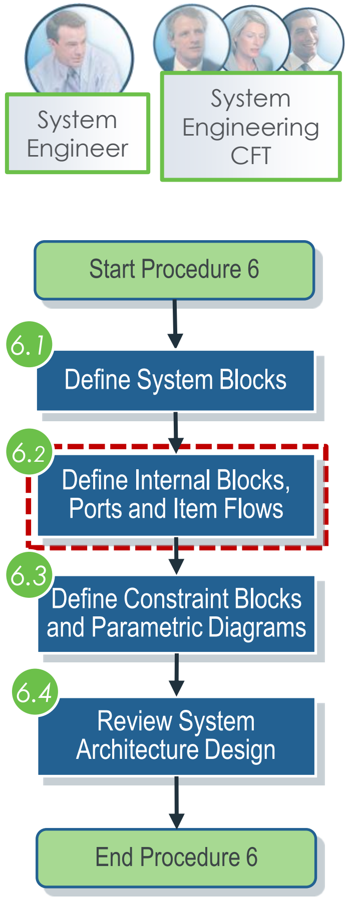
Docking Type: Not Docked

**Right click on the Block Property and select View Options**

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

# MODEL BASED SYSTEM ENGINEERING

## ▶ Define Internal Blocks, Ports and Item Flows



System Engineer

System Engineering CFT

Start Procedure 6

6.1 Define System Blocks

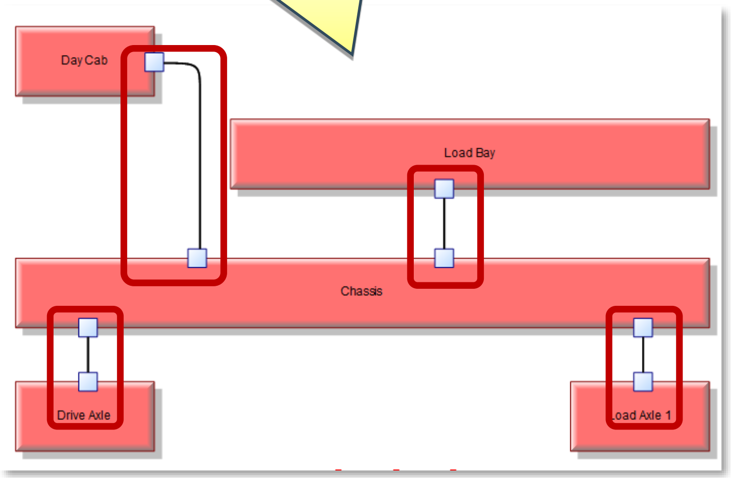
6.2 Define Internal Blocks, Ports and Item Flows

6.3 Define Constraint Blocks and Parametric Diagrams

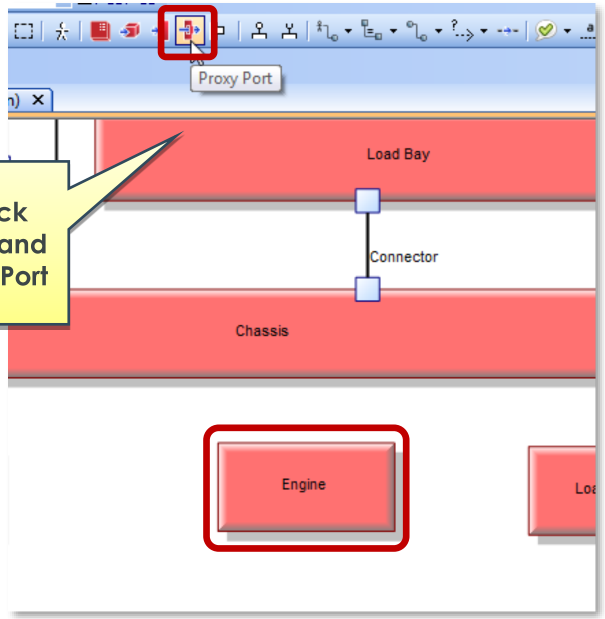
6.4 Review System Architecture Design

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



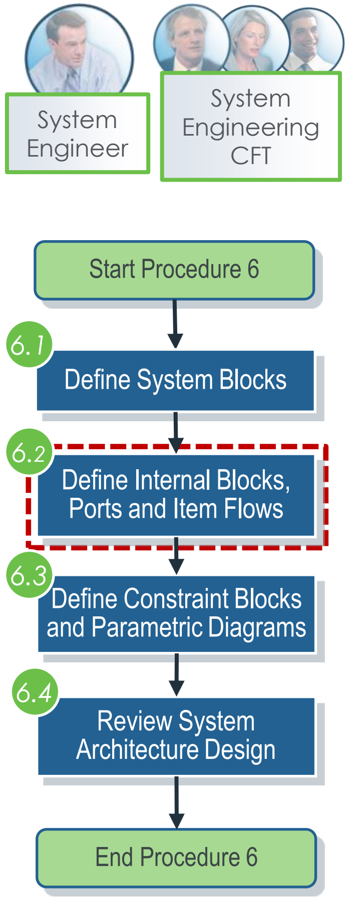
Add new Block called Engine and click on Proxy Port

**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

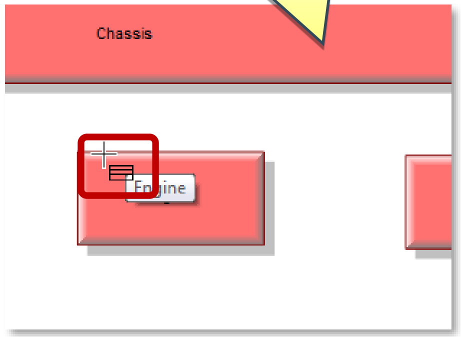
# MODEL BASED SYSTEM ENGINEERING

## ▶ Define Internal Blocks, Ports and Item Flows

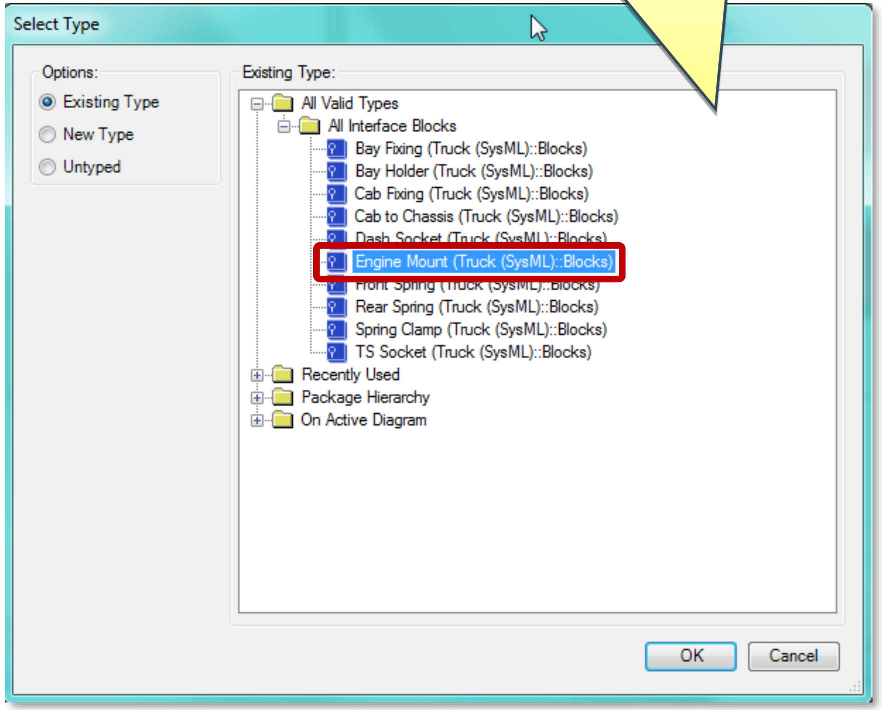


System Engineer
System Engineering CFT
Start Procedure 6
6.1
Define System Blocks
6.2
Define Internal Blocks, Ports and Item Flows
6.3
Define Constraint Blocks and Parametric Diagrams
6.4
Review System Architecture Design
End Procedure 6

**Click on the top edge of the Engine block**



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



Select Type

Options:

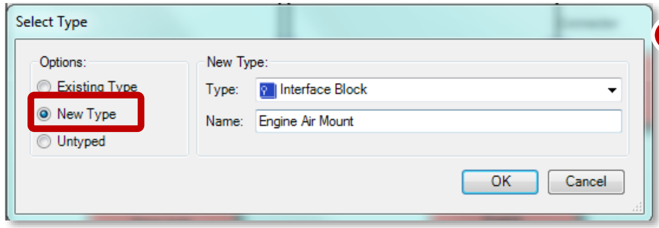
- ● Existing Type
- ○ New Type
- ○ Untyped

Existing Type:

- All Valid Types
  - All Interface Blocks
    - Bay Fixing (Truck (SysML)::Blocks)
    - Bay Holder (Truck (SysML)::Blocks)
    - Cab Fixing (Truck (SysML)::Blocks)
    - Cab to Chassis (Truck (SysML)::Blocks)
    - Dash Socket (Truck (SysML)::Blocks)
    - Engine Mount (Truck (SysML)::Blocks)
    - Front Spring (Truck (SysML)::Blocks)
    - Rear Spring (Truck (SysML)::Blocks)
    - Spring Clamp (Truck (SysML)::Blocks)
    - TS Socket (Truck (SysML)::Blocks)
  - Recently Used
  - Package Hierarchy
  - On Active Diagram

**Note**

**Can also create a new Interface Block type**



Select Type

Options:

- ○ Existing Type
- ● New Type
- ○ Untyped

New Type:

Type: Interface Block

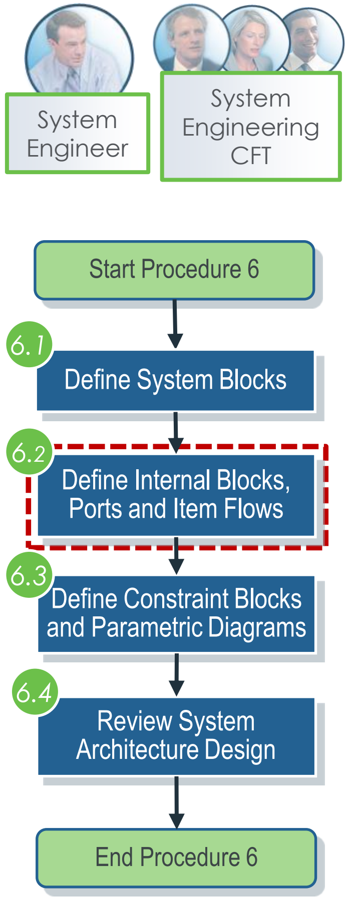
Name: Engine Air Mount

**Note**

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

# MODEL BASED SYSTEM ENGINEERING

## ▶ Define Internal Blocks, Ports and Item Flows


System Engineer

System Engineering CFT

Start Procedure 6

6.1 Define System Blocks

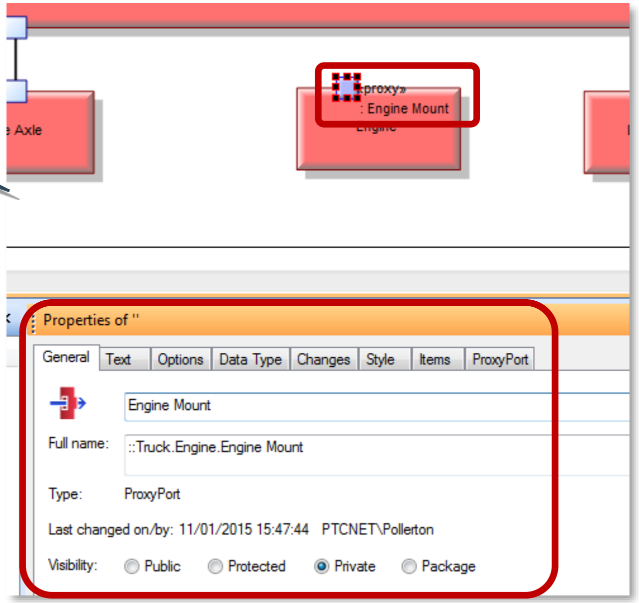
6.2 Define Internal Blocks, Ports and Item Flows

6.3 Define Constraint Blocks and Parametric Diagrams

6.4 Review System Architecture Design

End Procedure 6

**Name the port and choose options**



Properties of ''

General | Text | Options | Data Type | Changes | Style | Items | ProxyPort

Engine Mount

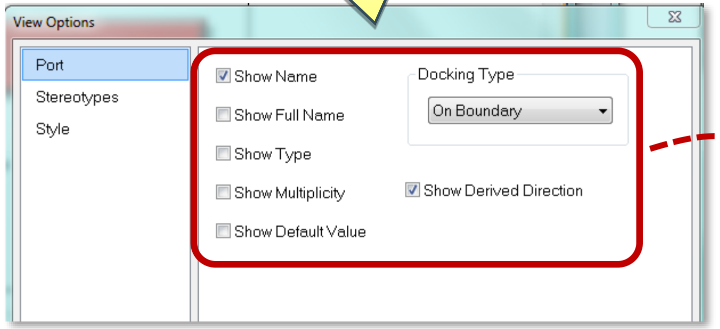
Full name: ::Truck.Engine.Engine Mount

Type: ProxyPort

Last changed on/by: 11/01/2015 15:47:44 PTCNET\Polleton

Visibility: ○ Public ○ Protected ● Private ○ Package

**Right click on the port and select View Options**


View Options

☑ Show Name

□ Show Full Name

□ Show Type

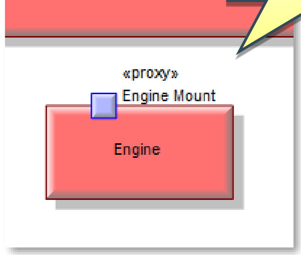
□ Show Multiplicity

□ Show Default Value

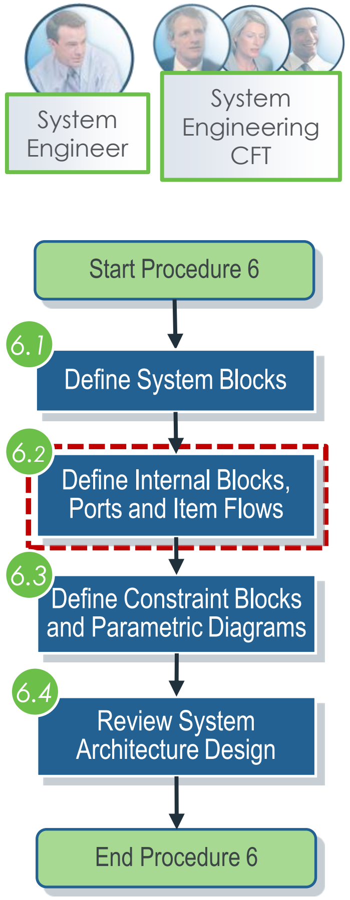
Docking Type: On Boundary

☑ Show Derived Direction

**Text can be repositioned**



# Define Internal Blocks, Ports and Item Flows



System Engineer

System Engineering CFT

Start Procedure 6

6.1 Define System Blocks

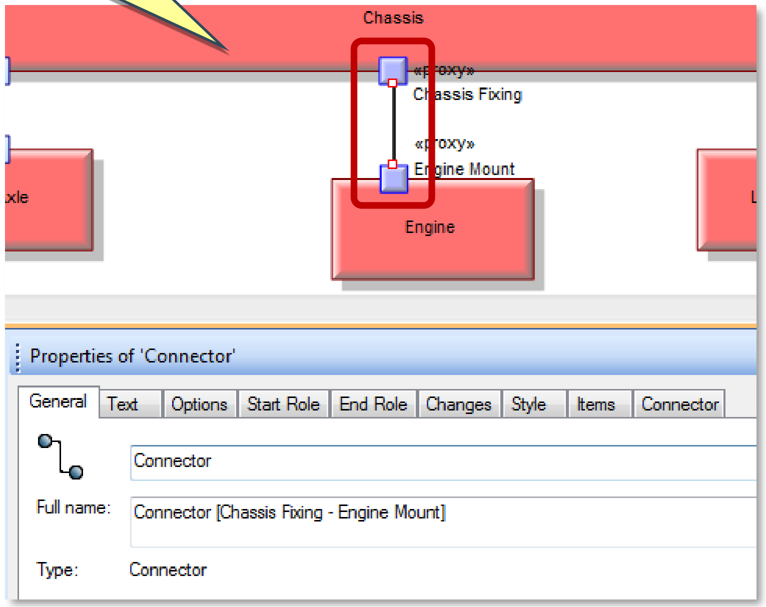
6.2 Define Internal Blocks, Ports and Item Flows

6.3 Define Constraint Blocks and Parametric Diagrams

6.4 Review System Architecture Design

End Procedure 6

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



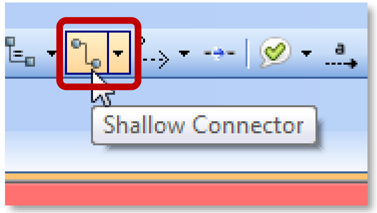
Properties of 'Connector'

General | Text | Options | Start Role | End Role | Changes | Style | Items | Connector

Connector

Full name: Connector [Chassis Fixing - Engine Mount]

Type: Connector

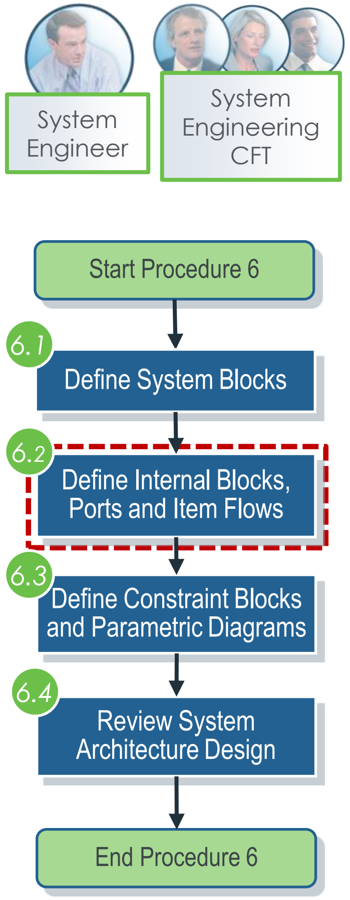


**Note**

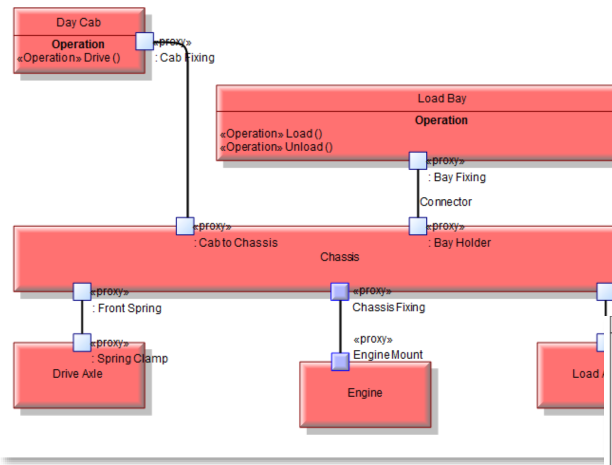
**Ports and connectors represent an interface between system entities. This concept is explained further in a later procedure.**

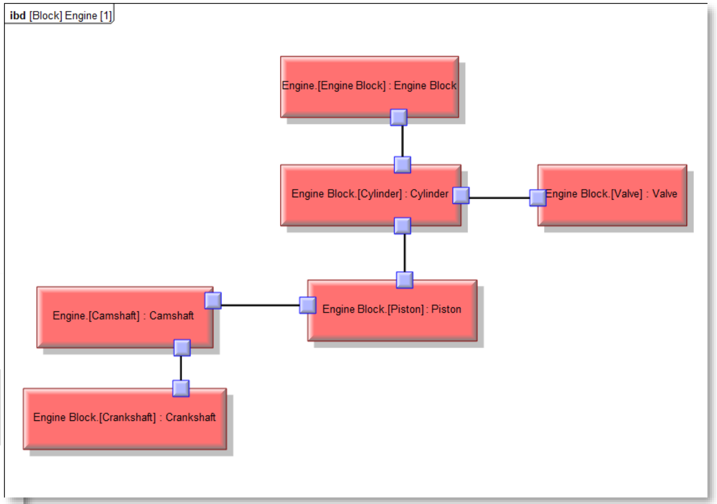
# MODEL BASED SYSTEM ENGINEERING

## ► Define Internal Blocks, Ports and Item Flows

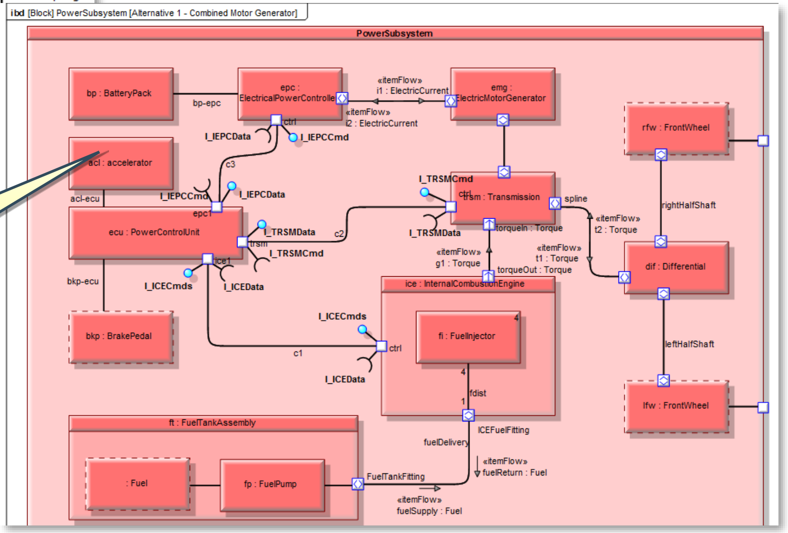

System Engineer
System Engineering CFT
Start Procedure 6
6.1
Define System Blocks
6.2
Define Internal Blocks, Ports and Item Flows
6.3
Define Constraint Blocks and Parametric Diagrams
6.4
Review System Architecture Design
End Procedure 6

Complete definition of internal blocks, ports and item flows

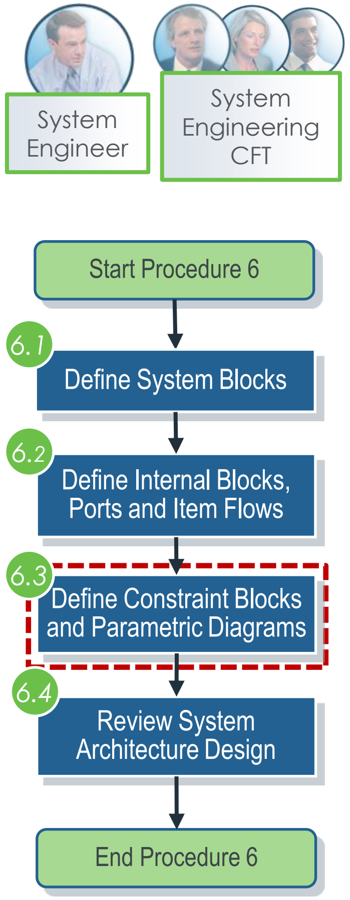



ibd [Block] Engine [1]
Engine.[Engine Block] : Engine Block
Engine Block.[Cylinder] : Cylinder
Engine Block.[Valve] : Valve
Engine.[Camshaft] : Camshaft
Engine Block.[Piston] : Piston
Engine Block.[Crankshaft] : Crankshaft

Define the data/information/item flows between the system blocks



## Define Constraint Blocks and Parametric Diagrams



System Engineer

System Engineering CFT

Start Procedure 6

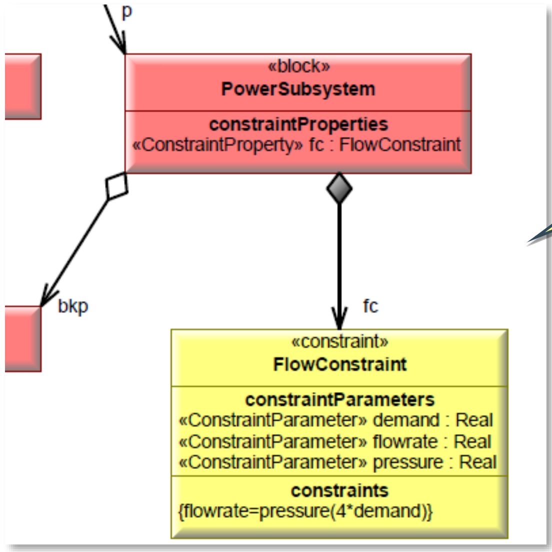
6.1 Define System Blocks

6.2 Define Internal Blocks, Ports and Item Flows

6.3 Define Constraint Blocks and Parametric Diagrams

6.4 Review System Architecture Design

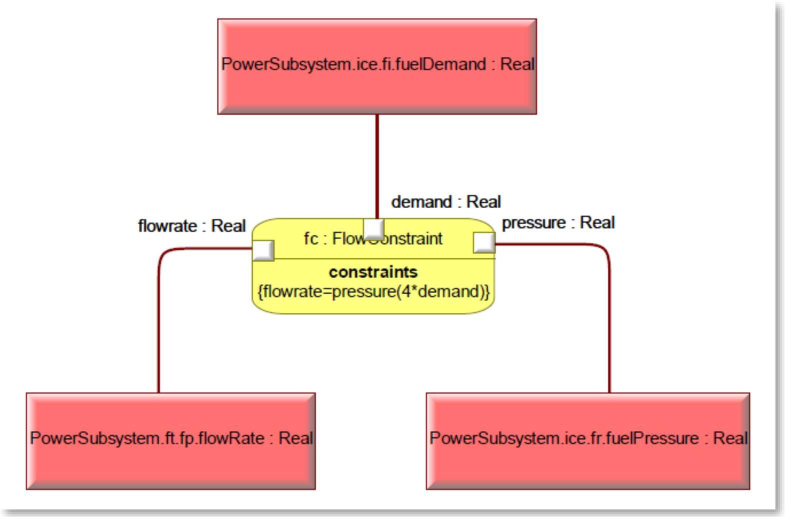
End Procedure 6



p
«block»
PowerSubsystem
constraintProperties
«ConstraintProperty» fc : FlowConstraint
bkp
fc
«constraint»
FlowConstraint
constraintParameters
«ConstraintParameter» demand : Real
«ConstraintParameter» flowrate : Real
«ConstraintParameter» pressure : Real
constraints
{flowrate=pressure(4*demand)}

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



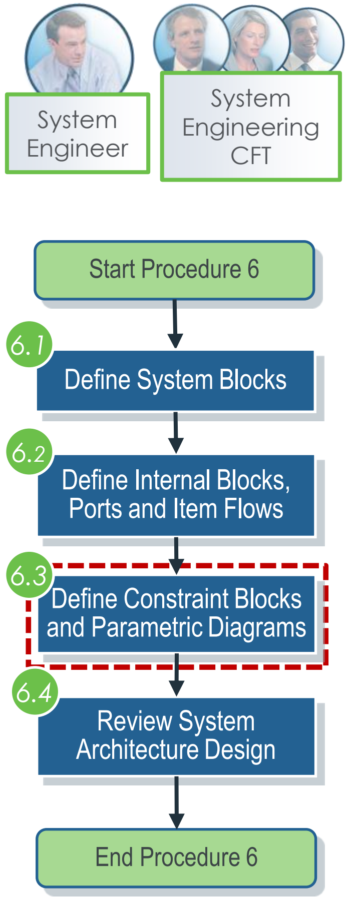
PowerSubsystem.ice.fi.fuelDemand : Real
demand : Real
flowrate : Real
pressure : Real
fc : FlowConstraint
constraints
{flowrate=pressure(4*demand)}
PowerSubsystem.ft.fp.flowRate : Real
PowerSubsystem.ice.fr.fuelPressure : Real

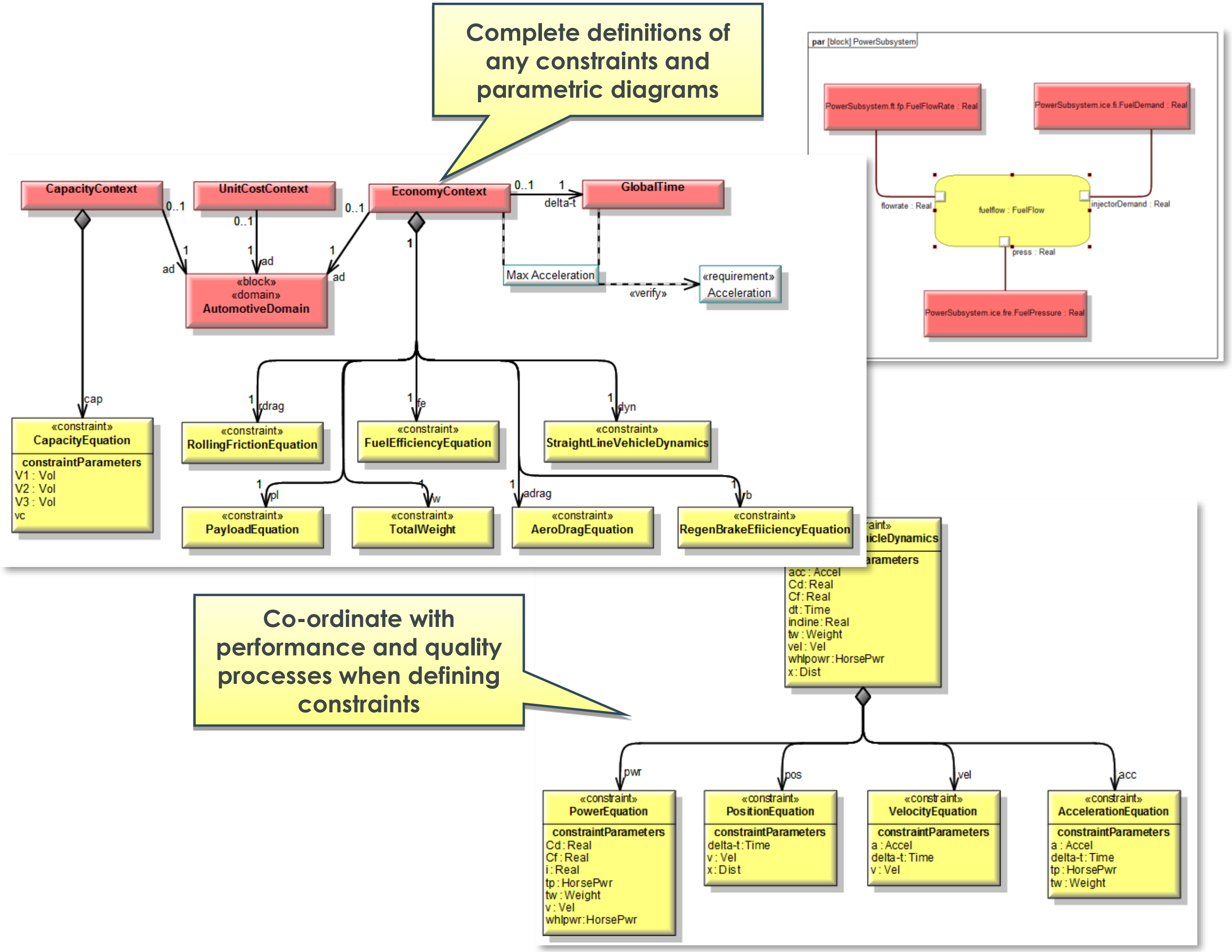
**Note**

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

# MODEL BASED SYSTEM ENGINEERING

► Define Constraint Blocks and Parametric Diagrams


System Engineer
System Engineering CFT
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

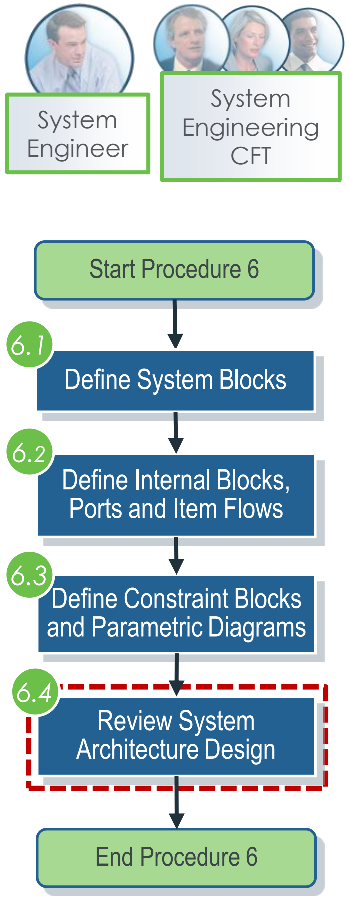


**Complete definitions of any constraints and parametric diagrams**

**Co-ordinate with performance and quality processes when defining constraints**

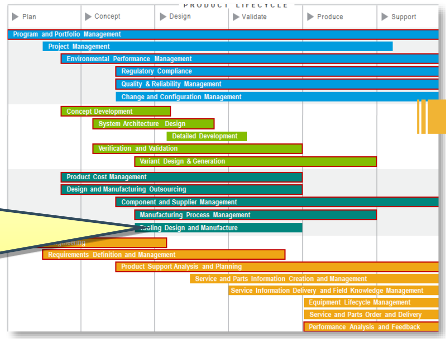
# MODEL BASED SYSTEM ENGINEERING

## ▶ Review System Architecture Design

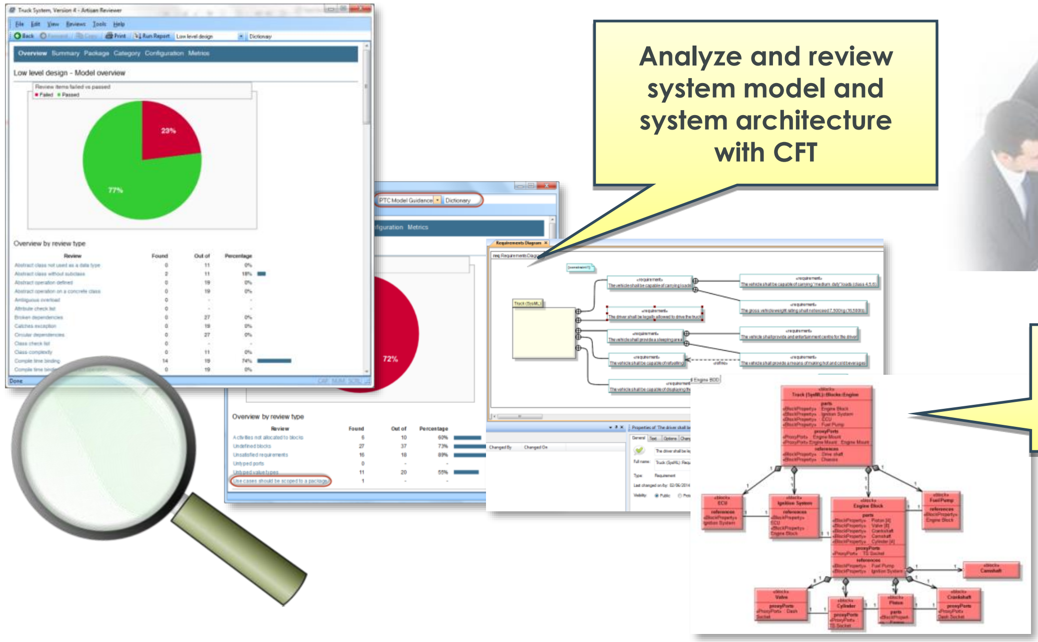


System Engineer
System Engineering CFT
Start Procedure 6
6.1
Define System Blocks
6.2
Define Internal Blocks, Ports and Item Flows
6.3
Define Constraint Blocks and Parametric Diagrams
6.4
Review System Architecture Design
End Procedure 6

**Coordinate with related processes and identify any relevant existing information: Automated System Design Review, Verification and Validation, Test Management**



**Analyze and review system model and system architecture with CFT**



Low level design - Model overview

Overview by review type

| Review | Found | Out of | Percentage |
|---|---|---|---|
| Abstract class not used as a data type | 0 | 11 | 0% |
| Abstract class without subclass | 2 | 11 | 18% |
| Abstract operation defined | 0 | 19 | 0% |
| Abstract operation on a concrete class | 0 | 19 | 0% |
| Ambiguous overload | 0 | - | - |
| Attribute check list | 0 | - | - |
| Broken dependencies | 0 | 27 | 0% |
| Catches exception | 0 | 19 | 0% |
| Circular dependencies | 0 | 27 | 0% |
| Class check list | 0 | - | - |
| Class complexity | 0 | 11 | 0% |
| Compile time binding | 14 | 19 | 74% |
| Compile time bin[illegible] operation | 0 | 19 | 0% |

Overview by review type

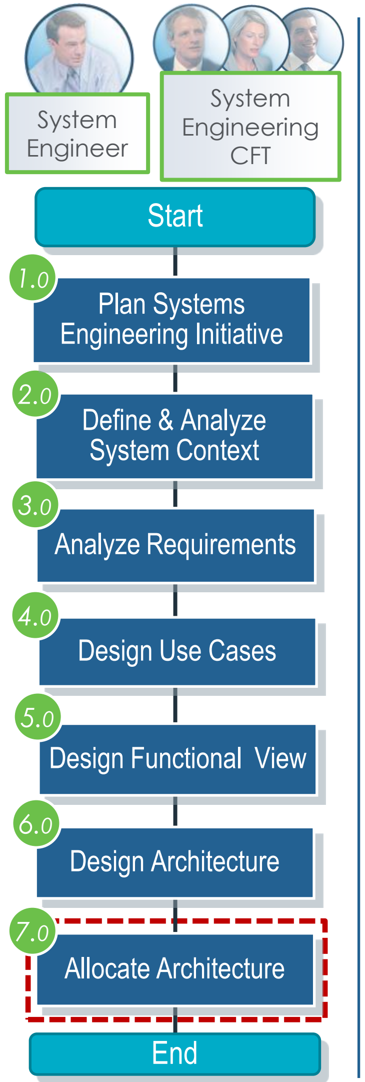
| Review | Found | Out of | Percentage |
|---|---|---|---|
| Activities not allocated to blocks | 6 | 10 | 60% |
| Undefined blocks | 27 | 37 | 73% |
| Unsatisfied requirements | 16 | 18 | 89% |
| Untyped ports | 0 | - | - |
| Untyped value types | 11 | 20 | 55% |
| Use cases should be scoped to a package | 1 | - | - |

**Update system model if needed**

# ALLOCATE ARCHITECTURE

# MODEL BASED SYSTEM ENGINEERING BEST PRACTICE PROCEDURE

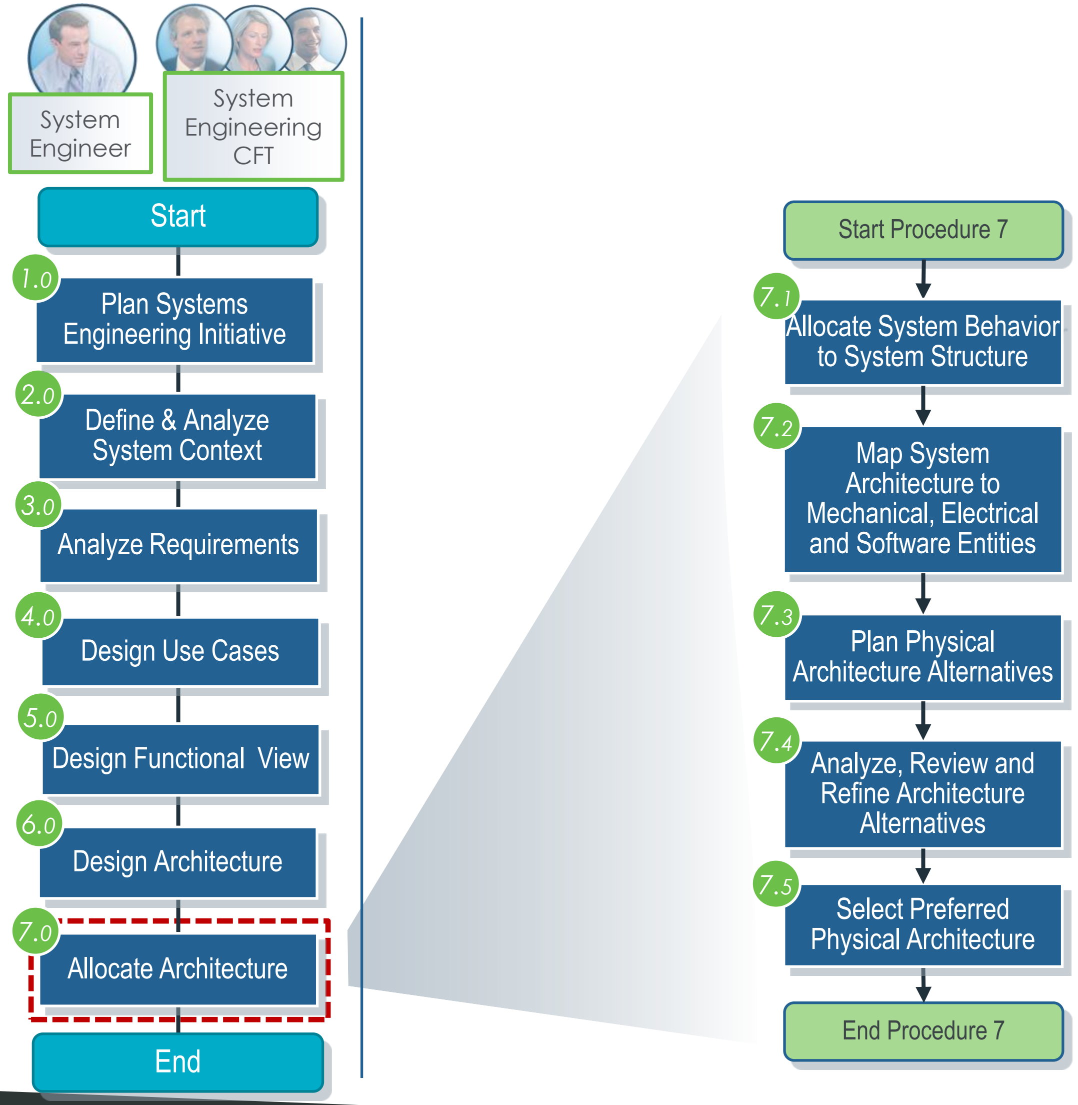
## ► Allocate Architecture



- 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

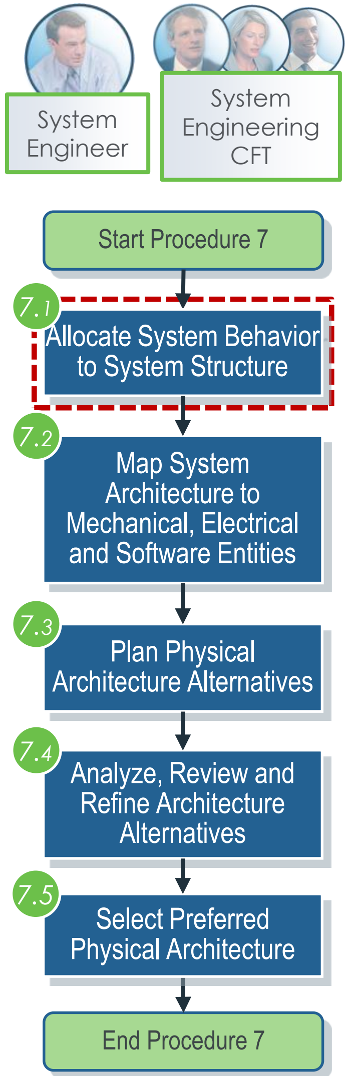
# MODEL BASED SYSTEM ENGINEERING

## ▶ Allocate Architecture

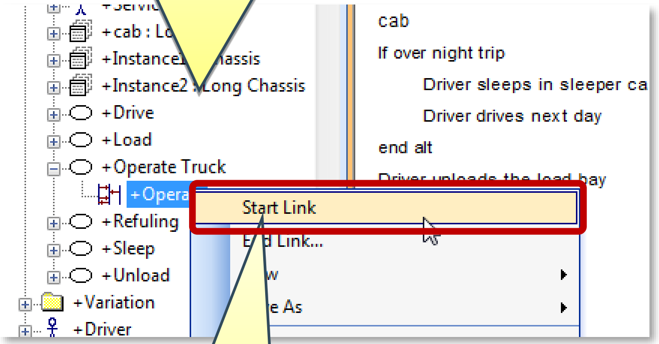

System Engineer
System Engineering CFT
Start
1.0
Plan Systems Engineering Initiative
2.0
Define & Analyze System Context
3.0
Analyze Requirements
4.0
Design Use Cases
5.0
Design Functional View
6.0
Design Architecture
7.0
Allocate Architecture
End
Start Procedure 7
7.1
Allocate System Behavior to System Structure
7.2
Map System Architecture to Mechanical, Electrical and Software Entities
7.3
Plan Physical Architecture Alternatives
7.4
Analyze, Review and Refine Architecture Alternatives
7.5
Select Preferred Physical Architecture
End Procedure 7

# MODEL BASED SYSTEM ENGINEERING

## ► Link System Behavior to System Structure

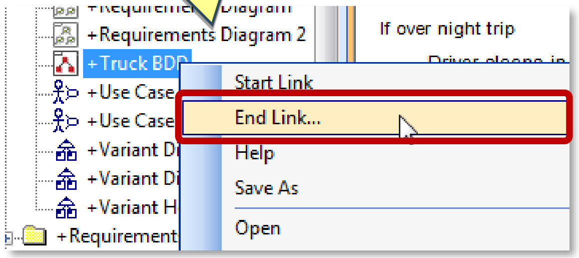

System Engineer
System Engineering CFT
Start Procedure 7
7.1
Allocate System Behavior to System Structure
7.2
Map System Architecture to Mechanical, Electrical and Software Entities
7.3
Plan Physical Architecture Alternatives
7.4
Analyze, Review and Refine Architecture Alternatives
7.5
Select Preferred Physical Architecture
End Procedure 7

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

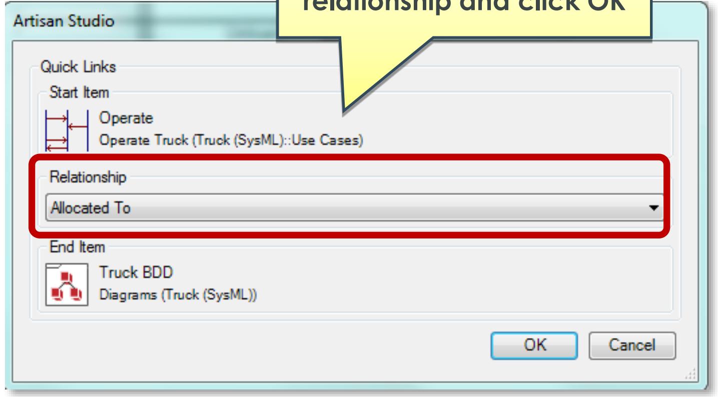


Right click on a sequence diagram and select Start Link…

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



Select Allocated To as the relationship and click OK



Quick Links

Start Item

Operate
Operate Truck (Truck (SysML)::Use Cases)

Relationship

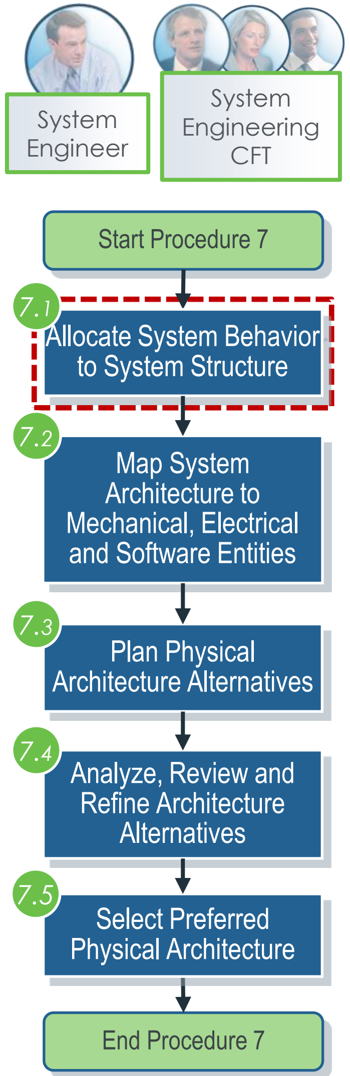
Allocated To

End Item

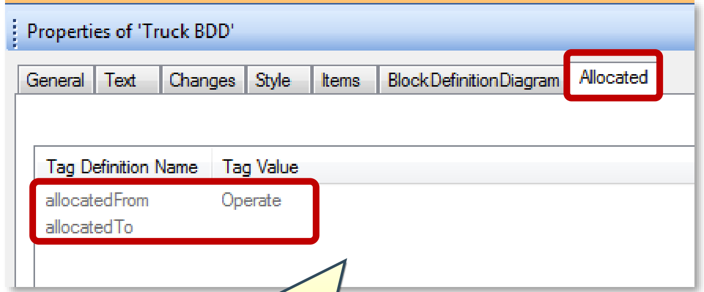
Truck BDD
Diagrams (Truck (SysML))

# MODEL BASED SYSTEM ENGINEERING

## ▶ Link Requirements to System Architecture



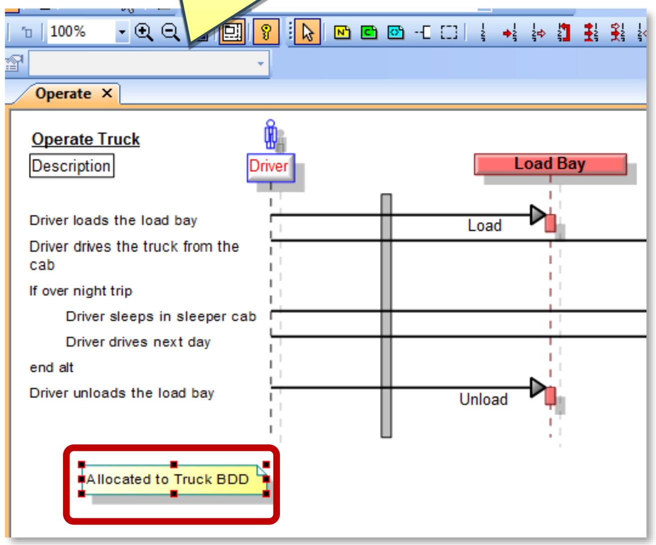
System Engineer
System Engineering CFT
Start Procedure 7
7.1
Allocate System Behavior to System Structure
7.2
Map System Architecture to Mechanical, Electrical and Software Entities
7.3
Plan Physical Architecture Alternatives
7.4
Analyze, Review and Refine Architecture Alternatives
7.5
Select Preferred Physical Architecture
End Procedure 7



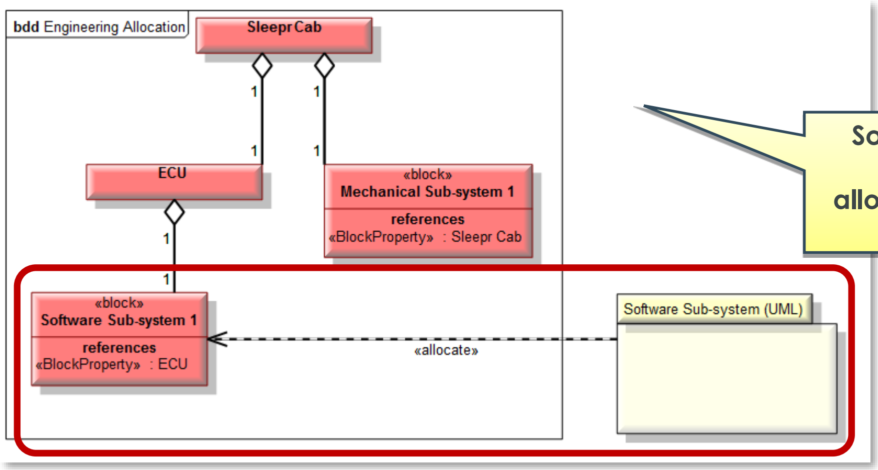
| Tag Definition Name | Tag Value |
|---|---|
| allocatedFrom | Operate |
| allocatedTo | |

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

A note can be added to the Sequence Diagram to indicate the relationship



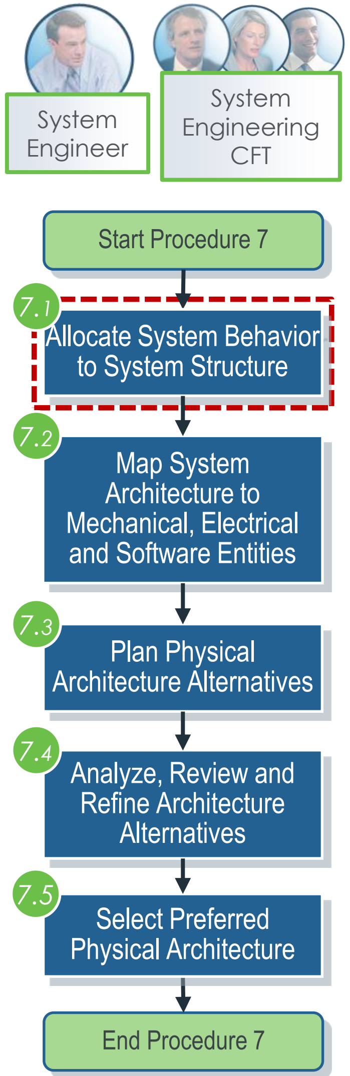
Driver loads the load bay
Driver drives the truck from the cab
If over night trip
Driver sleeps in sleeper cab
Driver drives next day
end alt
Driver unloads the load bay



Software designs can also be allocated to system blocks

# MODEL BASED SYSTEM ENGINEERING

## ► Link Requirements to System Architecture


System Engineer

System Engineering CFT

Start Procedure 7

7.1 Allocate System Behavior to System Structure

7.2 Map System Architecture to Mechanical, Electrical and Software Entities

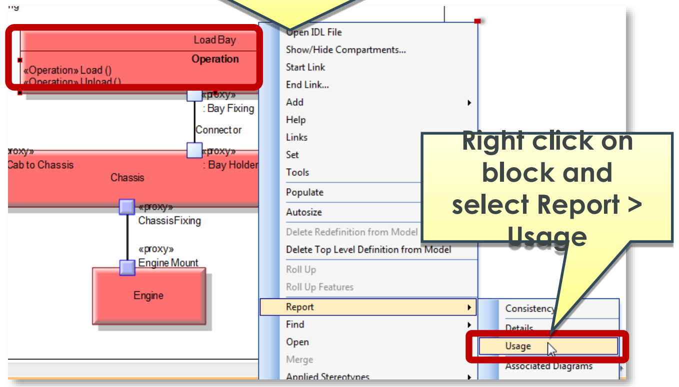
7.3 Plan Physical Architecture Alternatives

7.4 Analyze, Review and Refine Architecture Alternatives

7.5 Select Preferred Physical Architecture

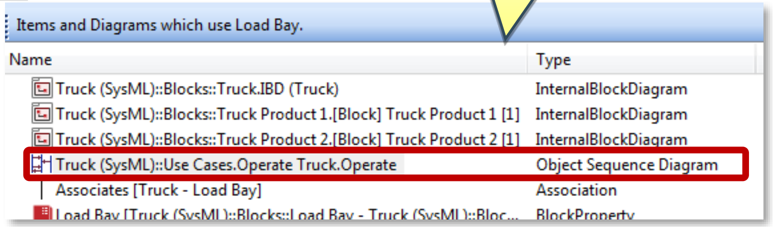
End Procedure 7

Navigate model relationships to view traceability from blocks back to requirements



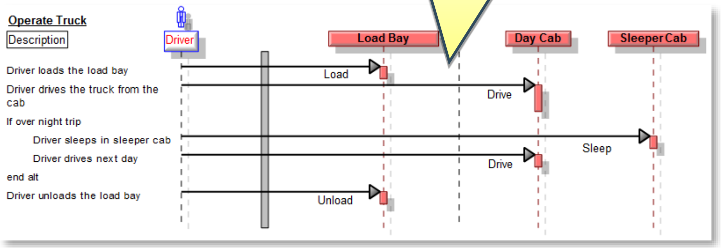
Right click on block and select Report > Usage

Review results to identify sequence diagram

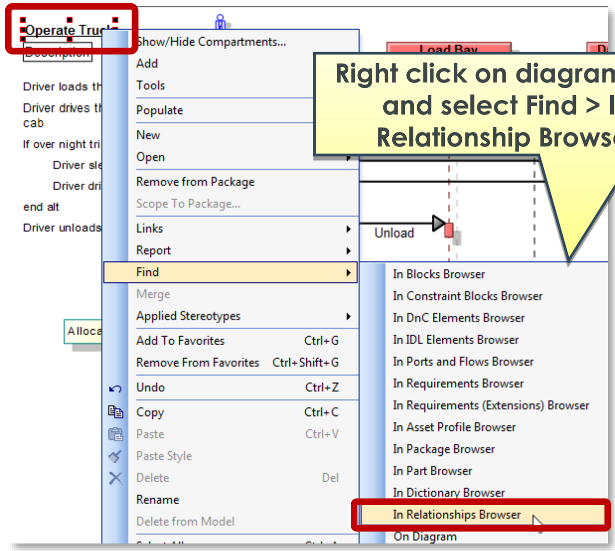

Items and Diagrams which use Load Bay.

| 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 |

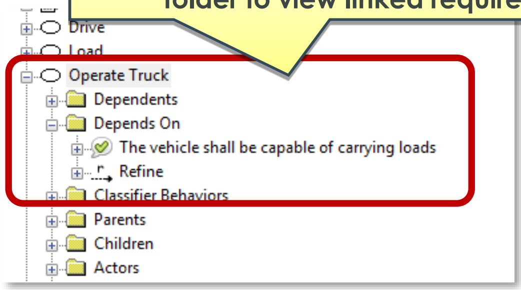
Double click to open sequence diagram



Right click on diagram title and select Find > In Relationship Browser

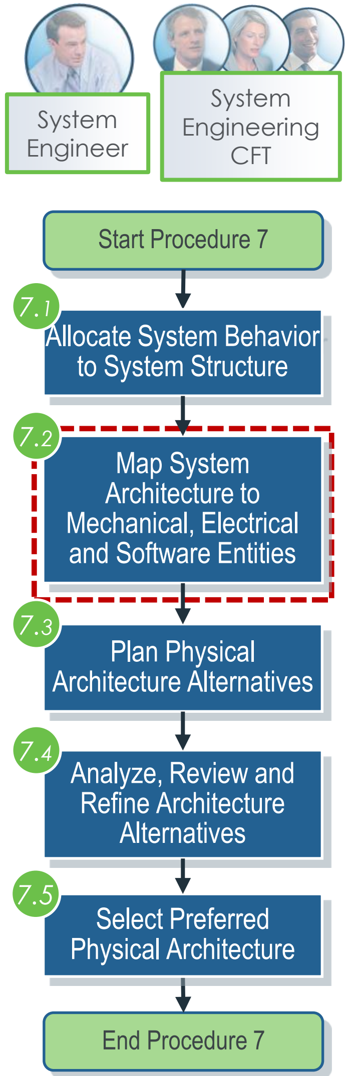


Use case will be highlighted in Relationship browser. Expand and select Depends On folder to view linked requirements



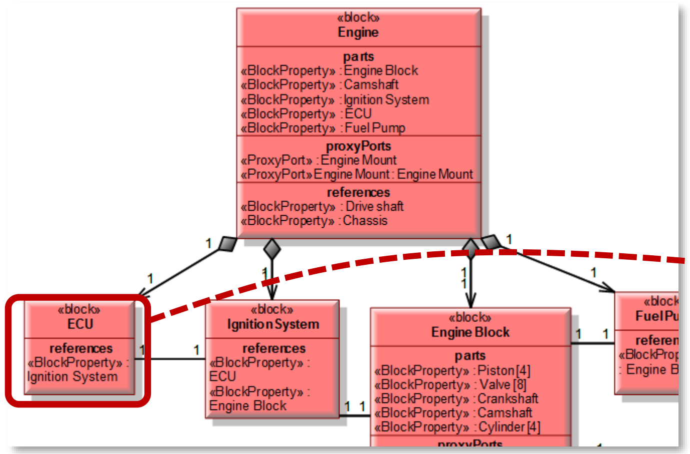
# MODEL BASED SYSTEM ENGINEERING

- Map System Architecture to Mechanical, Electrical and Software Entities



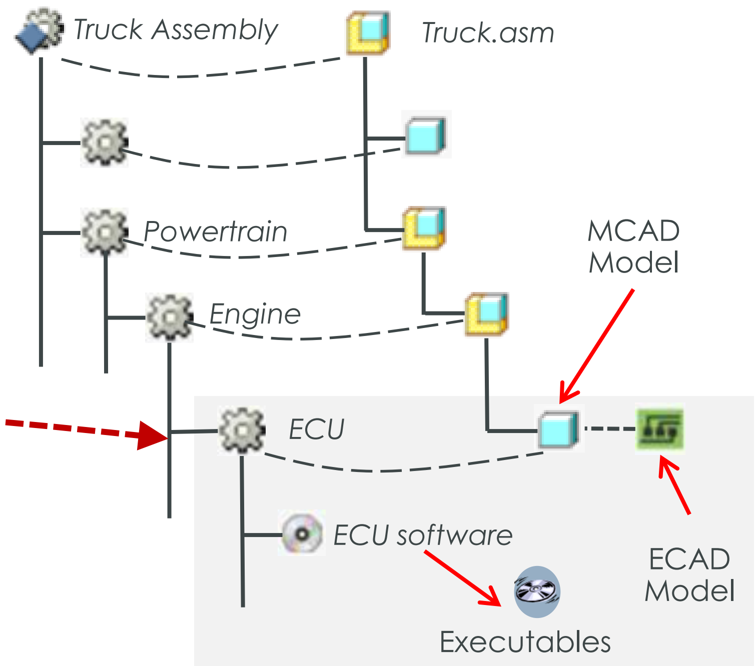
System Engineer
System Engineering CFT
Start Procedure 7
7.1
Allocate System Behavior to System Structure
7.2
Map System Architecture to Mechanical, Electrical and Software Entities
7.3
Plan Physical Architecture Alternatives
7.4
Analyze, Review and Refine Architecture Alternatives
7.5
Select Preferred Physical Architecture
End Procedure 7

**Determine sub-systems or parts that require mechanical, electrical or software components**



«block»
Engine
parts
«BlockProperty» : Engine Block
«BlockProperty» : Camshaft
«BlockProperty» : Ignition System
«BlockProperty» : ECU
«BlockProperty» : Fuel Pump
proxyPorts
«ProxyPort» : Engine Mount
«ProxyPort» Engine Mount : Engine Mount
references
«BlockProperty» : Drive shaft
«BlockProperty» : Chassis
«block»
ECU
references
«BlockProperty» : Ignition System
«block»
Ignition System
references
«BlockProperty» : ECU
«BlockProperty» : Engine Block
«block»
Engine Block
parts
«BlockProperty» : Piston [4]
«BlockProperty» : Valve [8]
«BlockProperty» : Crankshaft
«BlockProperty» : Camshaft
«BlockProperty» : Cylinder [4]
proxyPorts
«block
Fuel Pu
referen
«BlockProp
: Engine B
1

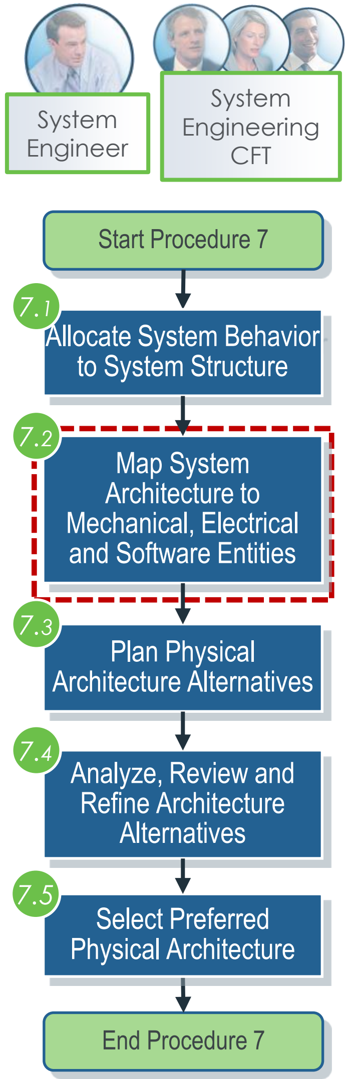
***Preliminary Design Structure***

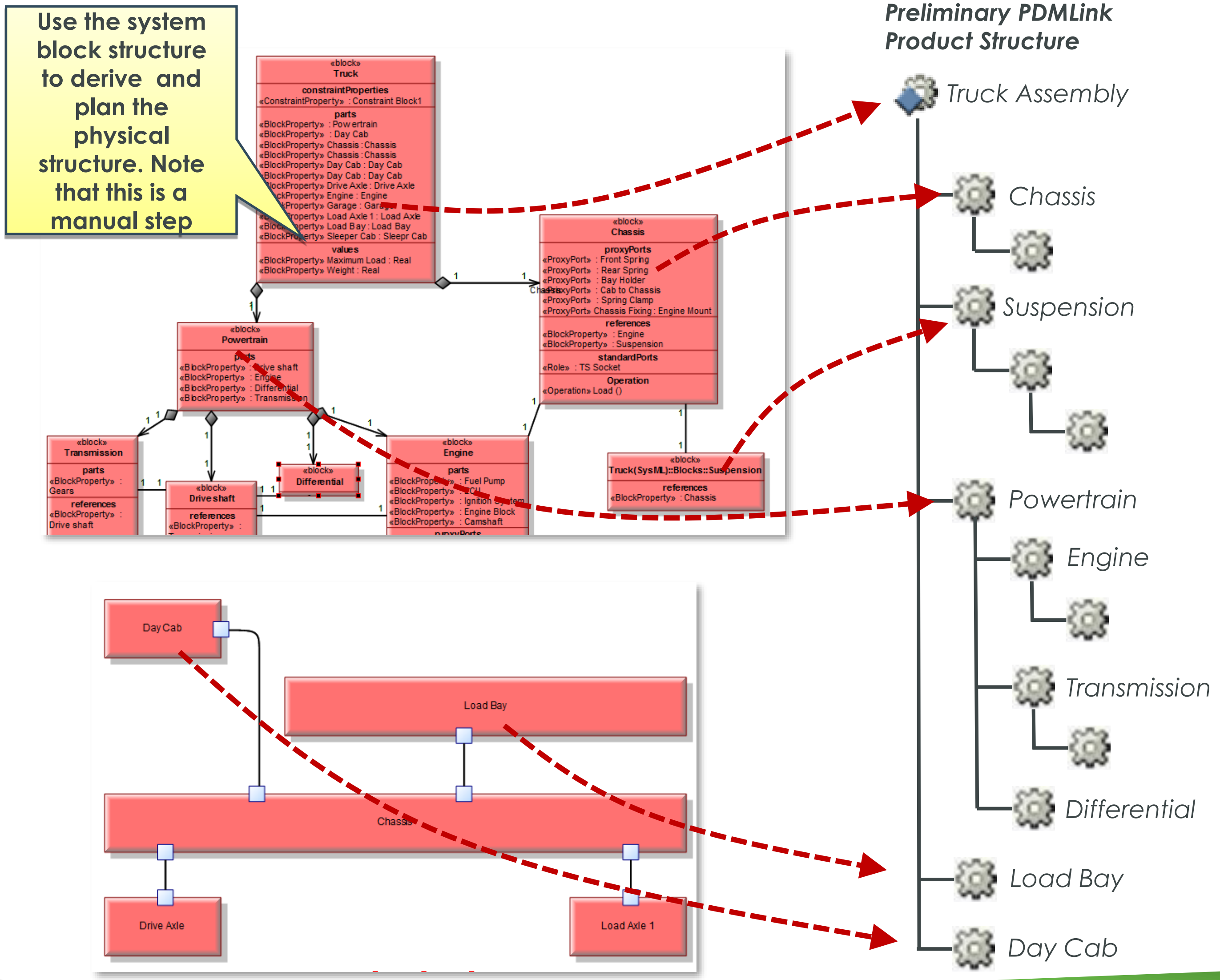


**ECU block will require a physical casing, electrical circuits and embedded software**

# MODEL BASED SYSTEM ENGINEERING

- Map System Architecture to Mechanical, Electrical and Software Entities


System Engineer
System Engineering CFT
Start Procedure 7
7.1
Allocate System Behavior to System Structure
7.2
Map System Architecture to Mechanical, Electrical and Software Entities
7.3
Plan Physical Architecture Alternatives
7.4
Analyze, Review and Refine Architecture Alternatives
7.5
Select Preferred Physical Architecture
End Procedure 7

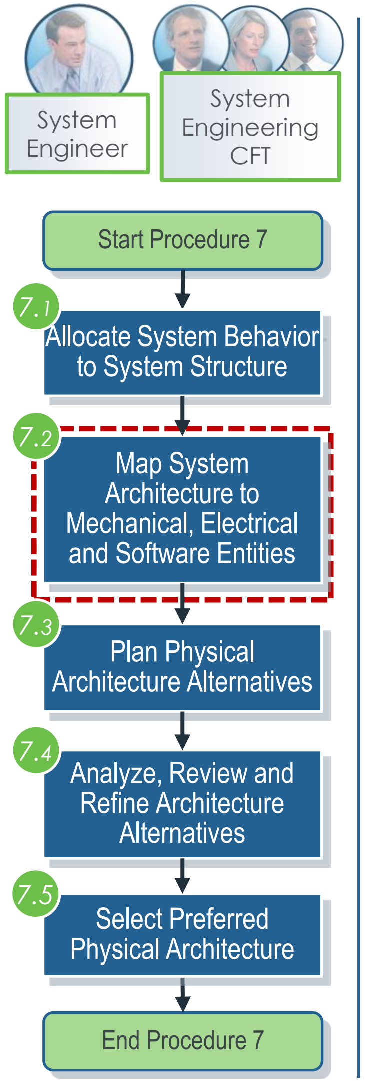

«block»
Truck
constraintProperties
«ConstraintProperty» : Constraint Block1
parts
«BlockProperty» : Powertrain
«BlockProperty» : Day Cab
«BlockProperty» Chassis : Chassis
«BlockProperty» Chassis : Chassis
«BlockProperty» Day Cab : Day Cab
«BlockProperty» Day Cab : Day Cab
«BlockProperty» Drive Axle : Drive Axle
«BlockProperty» Engine : Engine
«BlockProperty» Garage : Garage
«BlockProperty» Load Axle 1 : Load Axle
«BlockProperty» Load Bay : Load Bay
«BlockProperty» Sleeper Cab : Sleepr Cab
values
«BlockProperty» Maximum Load : Real
«BlockProperty» Weight : Real
«block»
Chassis
proxyPorts
«ProxyPort» : Front Spring
«ProxyPort» : Rear Spring
«ProxyPort» : Bay Holder
«ProxyPort» : Cab to Chassis
«ProxyPort» : Spring Clamp
«ProxyPort» Chassis Fixing : Engine Mount
references
«BlockProperty» : Engine
«BlockProperty» : Suspension
standardPorts
«Role» : TS Socket
Operation
«Operation» Load ()
Chassis
«block»
Powertrain
parts
«BlockProperty» : Drive shaft
«BlockProperty» : Engine
«BlockProperty» : Differential
«BlockProperty» : Transmission
«block»
Transmission
parts
«BlockProperty» : Gears
references
«BlockProperty» : Drive shaft
«block»
Drive shaft
references
«BlockProperty» :
«block»
Differential
«block»
Engine
parts
«BlockProperty» : Fuel Pump
«BlockProperty» : ECU
«BlockProperty» : Ignition System
«BlockProperty» : Engine Block
«BlockProperty» : Camshaft
proxyPorts
«block»
Truck(SysML)::Blocks::Suspension
references
«BlockProperty» : Chassis
1
Day Cab
Load Bay
Chassis
Drive Axle
Load Axle 1
Truck Assembly
Chassis
Suspension
Powertrain
Engine
Transmission
Differential
Load Bay
Day Cab

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

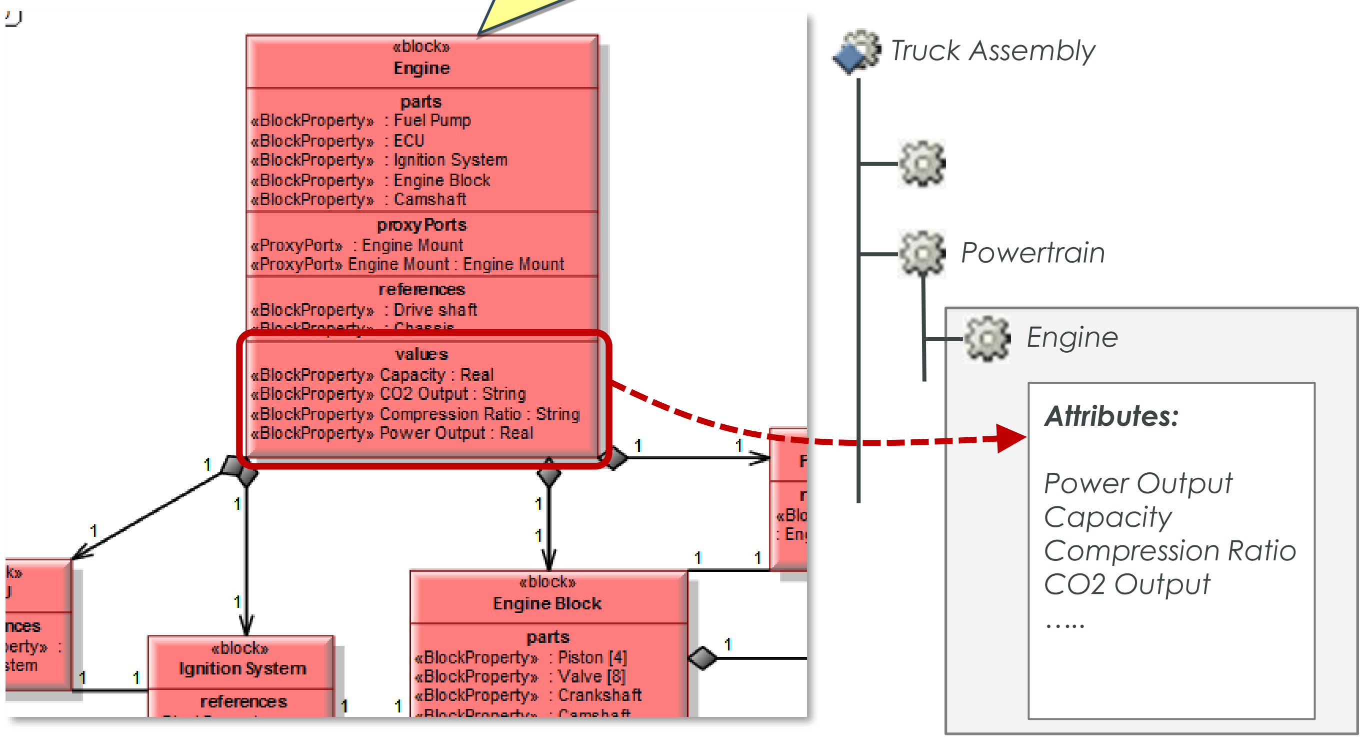
***Preliminary PDMLink Product Structure***

# MODEL BASED SYSTEM ENGINEERING

► Map System Architecture to Mechanical, Electrical and Software Entities


System Engineer
System Engineering CFT
Start Procedure 7
7.1
Allocate System Behavior to System Structure
7.2
Map System Architecture to Mechanical, Electrical and Software Entities
7.3
Plan Physical Architecture Alternatives
7.4
Analyze, Review and Refine Architecture Alternatives
7.5
Select Preferred Physical Architecture
End Procedure 7

**Block properties can be mapped to attributes on the physical models (Parts, CAD Documents or ECAD data)**


«block»
Engine

parts
«BlockProperty» : Fuel Pump
«BlockProperty» : ECU
«BlockProperty» : Ignition System
«BlockProperty» : Engine Block
«BlockProperty» : Camshaft

proxyPorts
«ProxyPort» : Engine Mount
«ProxyPort» Engine Mount : Engine Mount

references
«BlockProperty» : Drive shaft
«BlockProperty» : Chassis

values
«BlockProperty» Capacity : Real
«BlockProperty» CO2 Output : String
«BlockProperty» Compression Ratio : String
«BlockProperty» Power Output : Real

«block»
Ignition System

references

«block»
Engine Block

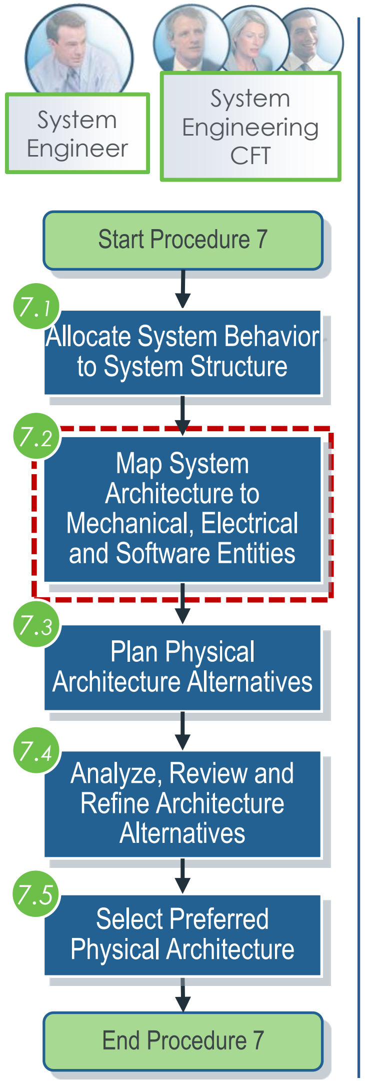
parts
«BlockProperty» : Piston [4]
«BlockProperty» : Valve [8]
«BlockProperty» : Crankshaft
«BlockProperty» : Camshaft

*Attributes:*

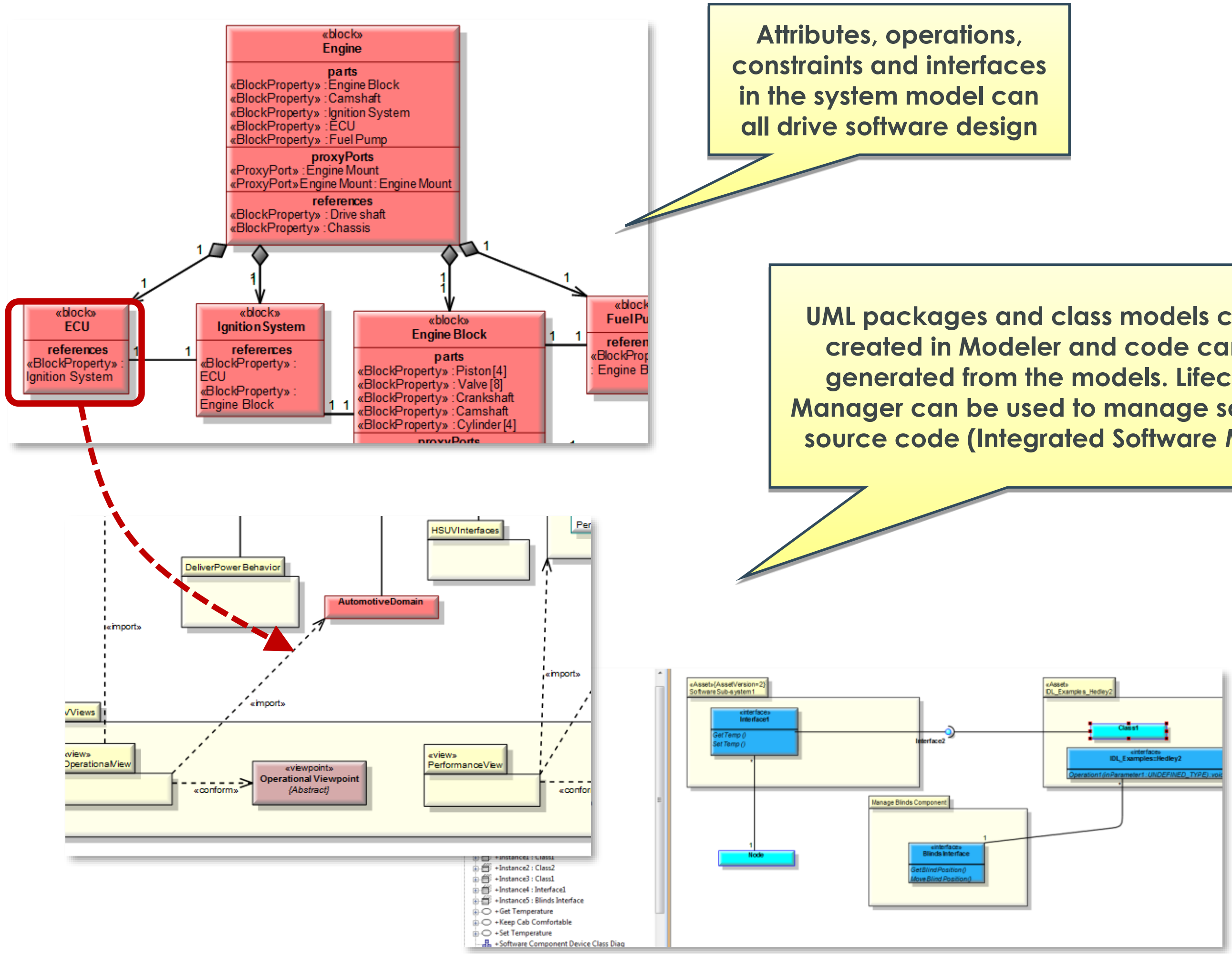
*Power Output*
*Capacity*
*Compression Ratio*
*CO2 Output*
*.....*

# MODEL BASED SYSTEM ENGINEERING

► Map System Architecture to Mechanical, Electrical and Software Entities



System Engineer
System Engineering CFT
Start Procedure 7
7.1
Allocate System Behavior to System Structure
7.2
Map System Architecture to Mechanical, Electrical and Software Entities
7.3
Plan Physical Architecture Alternatives
7.4
Analyze, Review and Refine Architecture Alternatives
7.5
Select Preferred Physical Architecture
End Procedure 7

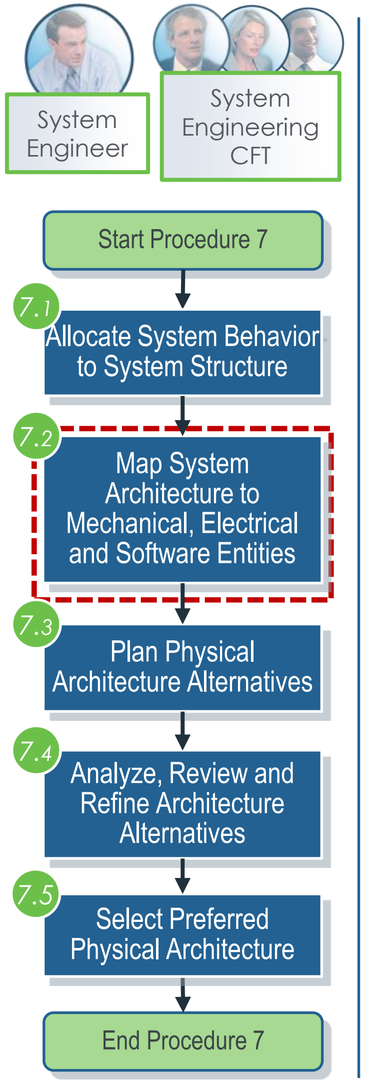


Attributes, operations, constraints and interfaces in the system model can all drive software design

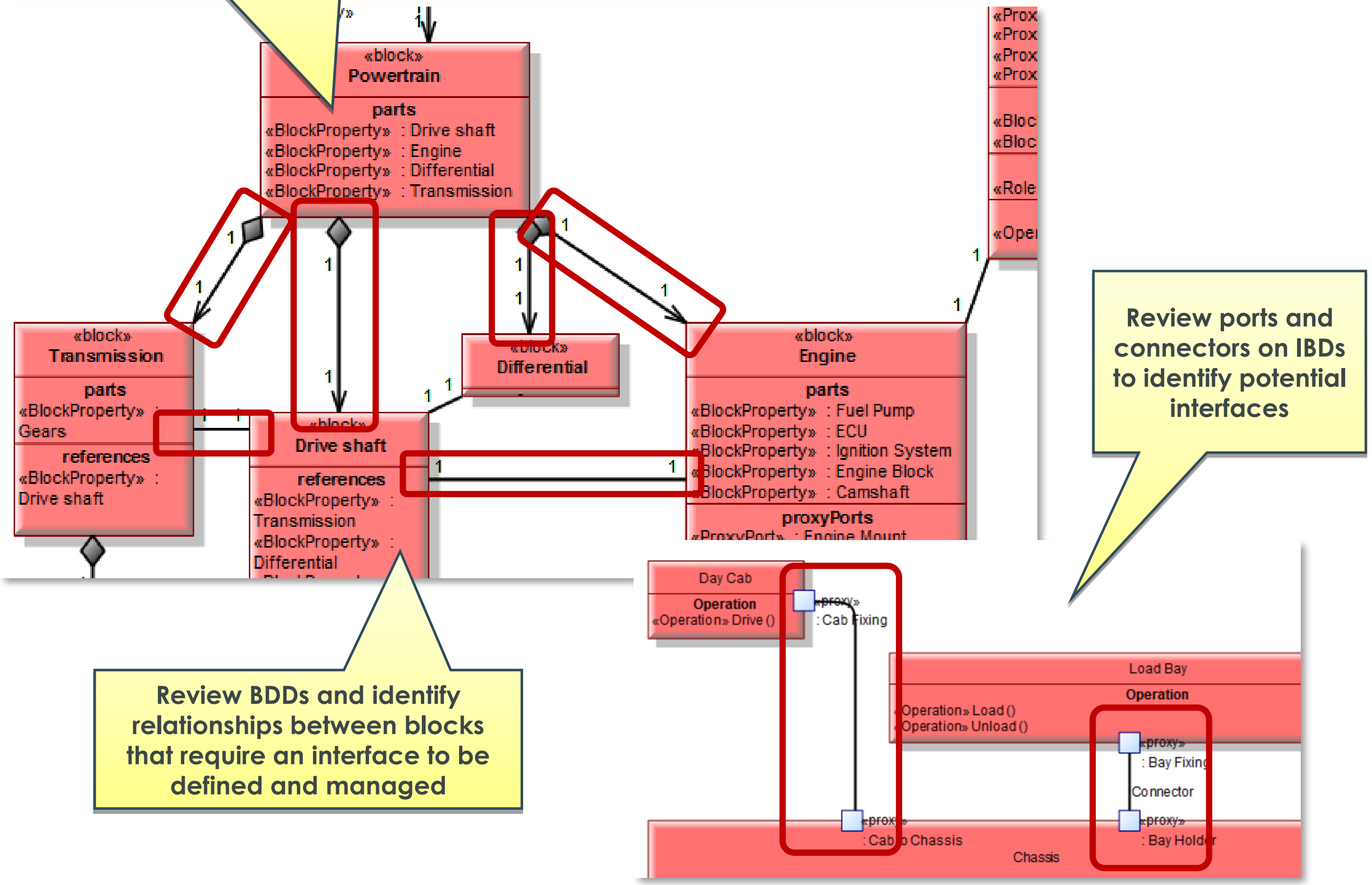
UML packages and class models can be created in Modeler and code can be generated from the models. Lifecycle Manager can be used to manage software source code (Integrated Software Mgmt)

# MODEL BASED SYSTEM ENGINEERING

- Map System Architecture to Mechanical, Electrical and Software Entities


System Engineer
System Engineering CFT
Start Procedure 7
7.1
Allocate System Behavior to System Structure
7.2
Map System Architecture to Mechanical, Electrical and Software Entities
7.3
Plan Physical Architecture Alternatives
7.4
Analyze, Review and Refine Architecture Alternatives
7.5
Select Preferred Physical Architecture
End Procedure 7

Identify interfaces by reviewing associations in the structure model

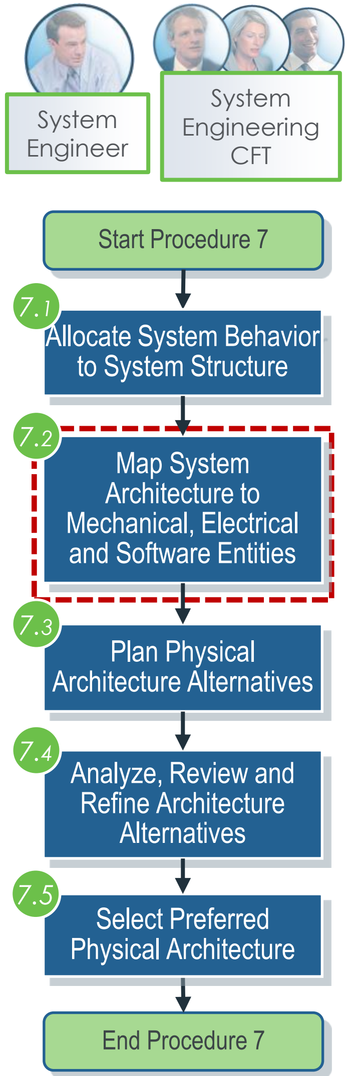


Review ports and connectors on IBDs to identify potential interfaces

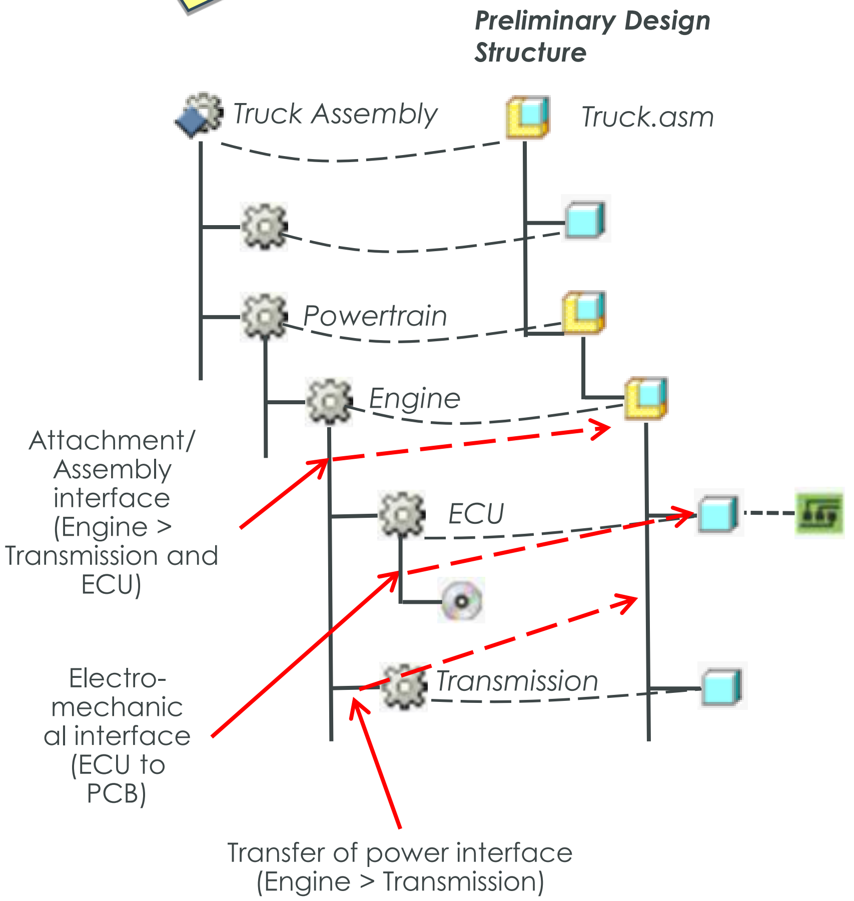
Review BDDs and identify relationships between blocks that require an interface to be defined and managed

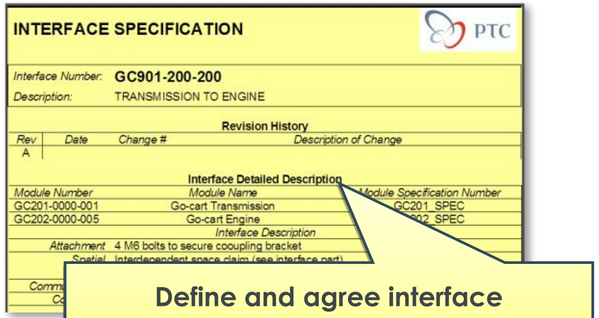
# MODEL BASED SYSTEM ENGINEERING

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


System Engineer
System Engineering CFT
Start Procedure 7
7.1
Allocate System Behavior to System Structure
7.2
Map System Architecture to Mechanical, Electrical and Software Entities
7.3
Plan Physical Architecture Alternatives
7.4
Analyze, Review and Refine Architecture Alternatives
7.5
Select Preferred Physical Architecture
End Procedure 7

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


Preliminary Design Structure
Truck Assembly
Truck.asm
Powertrain
Engine
ECU
Transmission
Attachment/ Assembly interface (Engine > Transmission and ECU)
Electro-mechanical interface (ECU to PCB)
Transfer of power interface (Engine > Transmission)



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 | 02_SPEC |

*Interface Description*

*Attachment* 4 M6 bolts to secure coupling bracket

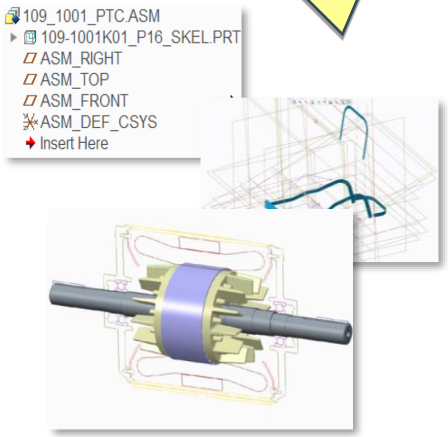
*Spatial* Interdependent space claim (see interface part)

*Comm*

*Co*

**Define and agree interface specifications and assign ownership**

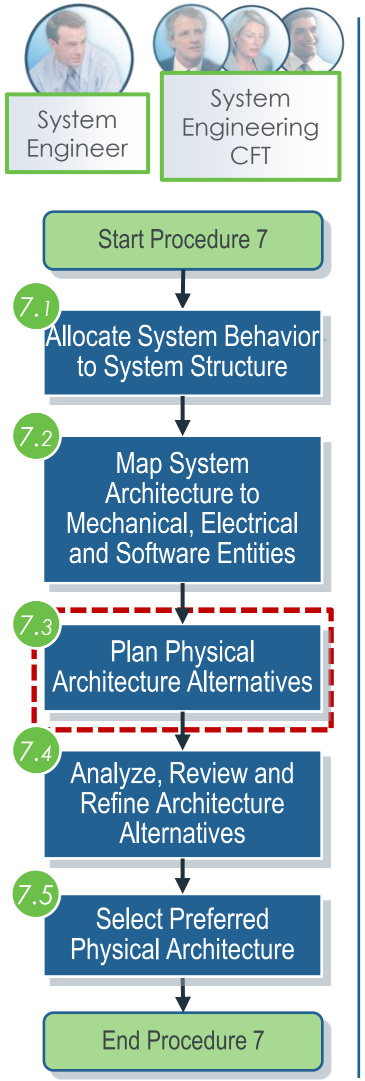
**Plan the realization of interfaces in design models (Refer to Interface Definition and Mgmt best practice)**

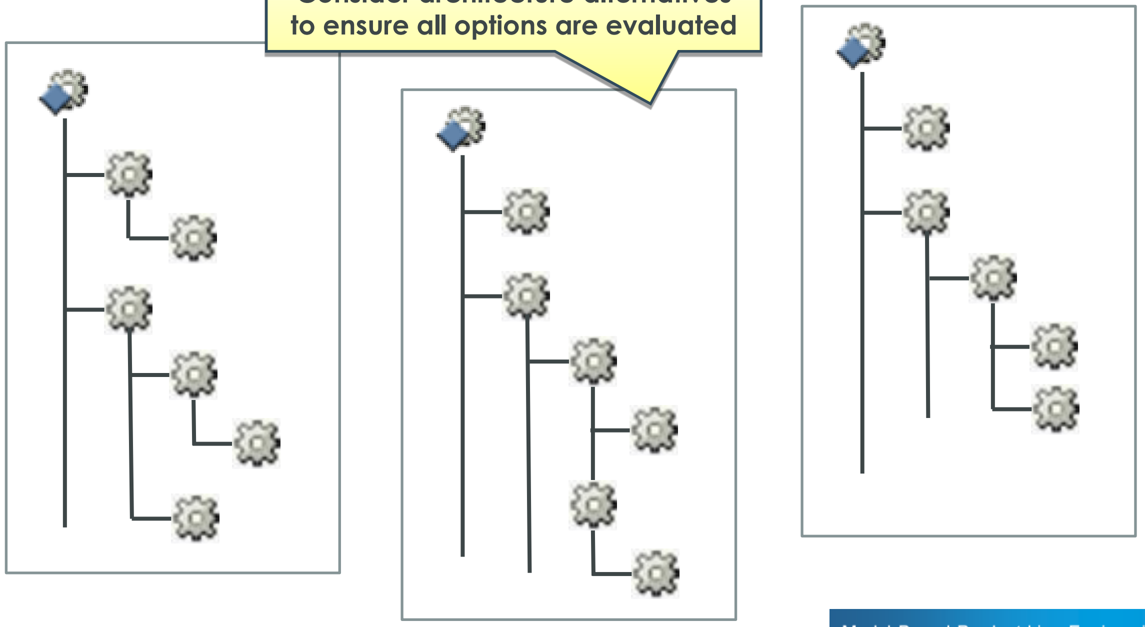


109_1001_PTC.ASM
▸ 109-1001K01_P16_SKEL.PRT
ASM_RIGHT
ASM_TOP
ASM_FRONT
ASM_DEF_CSYS
Insert Here

# MODEL BASED SYSTEM ENGINEERING

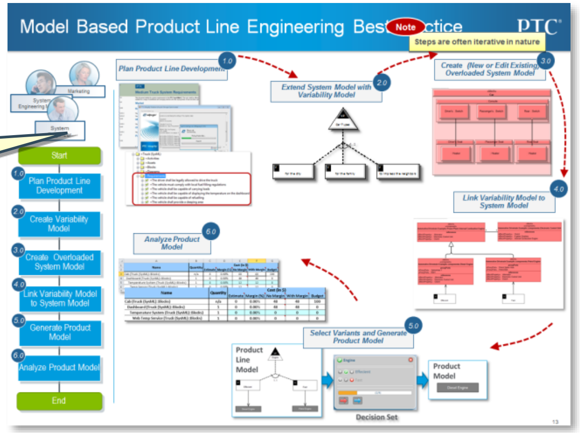
## ▶ Plan Physical Architecture Alternatives


System Engineer
System Engineering CFT
Start Procedure 7
7.1
Allocate System Behavior to System Structure
7.2
Map System Architecture to Mechanical, Electrical and Software Entities
7.3
Plan Physical Architecture Alternatives
7.4
Analyze, Review and Refine Architecture Alternatives
7.5
Select Preferred Physical Architecture
End Procedure 7



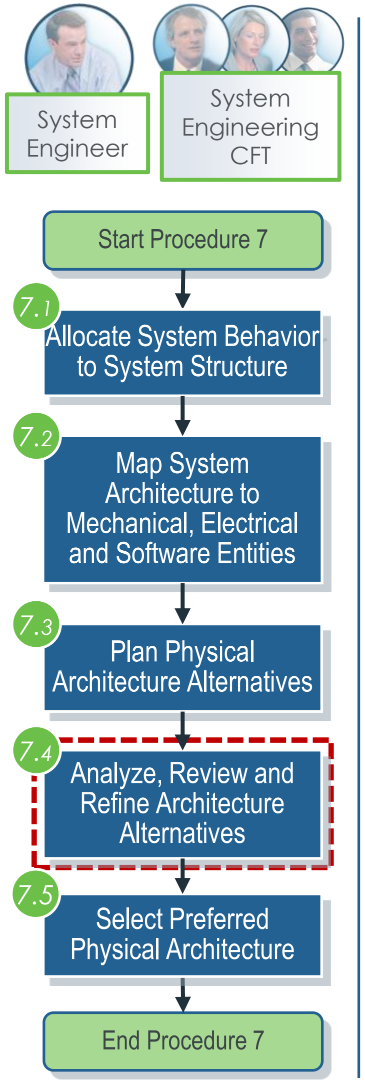
Consider architecture alternatives to ensure all options are evaluated

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

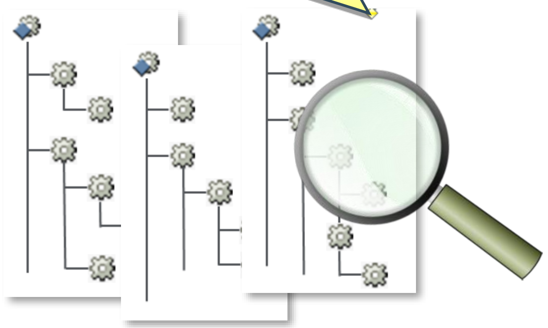

Model Based Product Line Engineering Best Practice
PTC
Note
Steps are often iterative in nature
Marketing
System Engineering
System
Start
1.0 Plan Product Line Development
2.0 Create Variability Model
3.0 Create Overloaded System Model
4.0 Link Variability Model to System Model
5.0 Generate Product Model
6.0 Analyze Product Model
End
Plan Product Line Development 1.0
Extend System Model with Variability Model 2.0
Create (New or Edit Existing) Overloaded System Model 3.0
Link Variability Model to System Model 4.0
Analyze Product Model 6.0
Select Variants and Generate Product Model 5.0
Product Line Model
Decision Set
Product Model
13

# MODEL BASED SYSTEM ENGINEERING

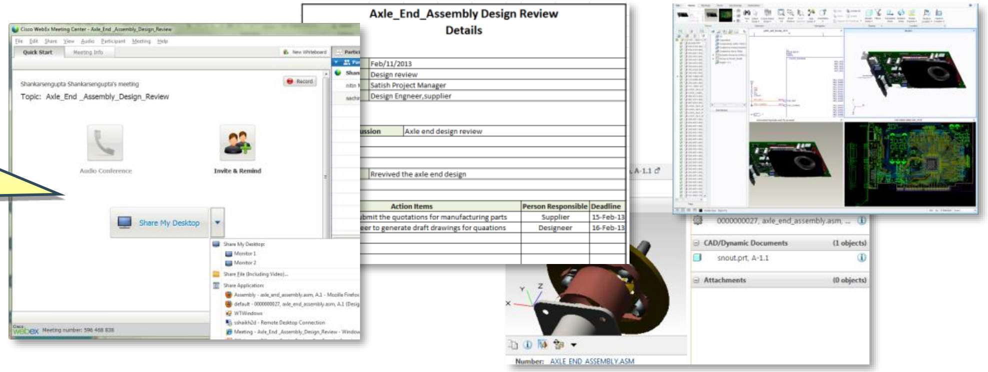
## ► Analyze, Review and Refine Architecture Alternatives


System Engineer
System Engineering CFT
Start Procedure 7
7.1
Allocate System Behavior to System Structure
7.2
Map System Architecture to Mechanical, Electrical and Software Entities
7.3
Plan Physical Architecture Alternatives
7.4
Analyze, Review and Refine Architecture Alternatives
7.5
Select Preferred Physical Architecture
End Procedure 7

**Review and refine architecture alternatives with CFT**



**Refer to the Efficient Design Review best practice for more information**



Shankarsengupta Shankarsengupta's meeting

Topic: Axle_End _Assembly_Design_Review

Meeting number: 596 468 838

**Axle_End_Assembly Design Review Details**

Feb/11/2013

Design review

Satish Project Manager

Design Engneer,supplier

ussion | Axle end design review

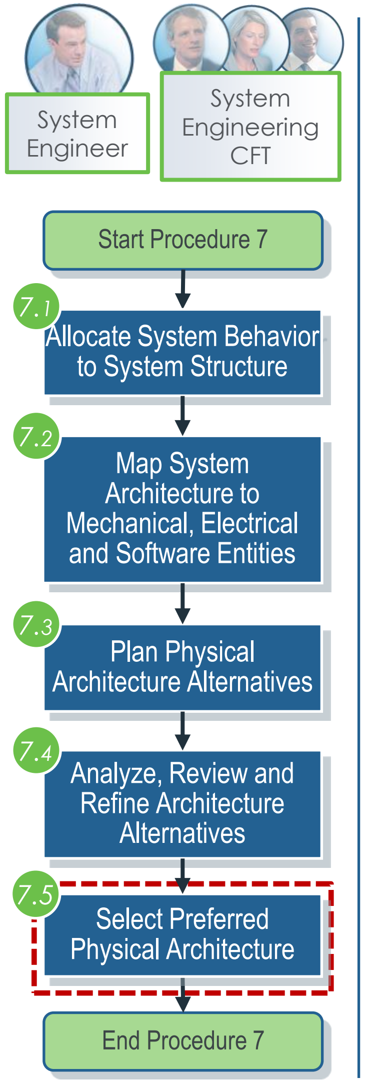
Rrevived the axle end design

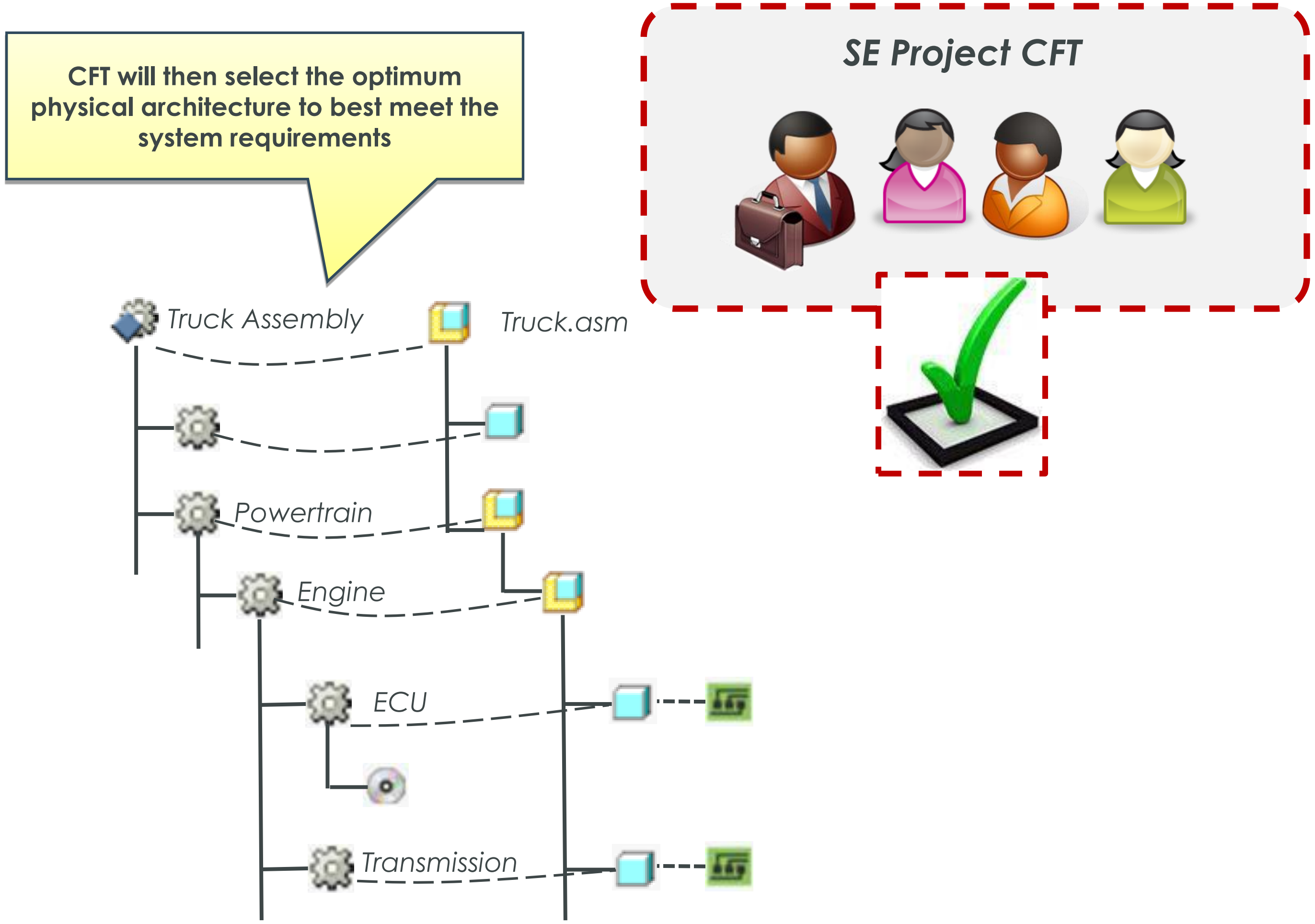
| Action Items | Person Responsible | Deadline |
|---|---|---|
| ubmit the quotations for manufacturing parts | Supplier | 15-Feb-13 |
| eer to generate draft drawings for quaations | Designeer | 16-Feb-13 |

# MODEL BASED SYSTEM ENGINEERING

ptc

## ► Select Preferred Physical Architecture


System Engineer
System Engineering CFT
Start Procedure 7
7.1
Allocate System Behavior to System Structure
7.2
Map System Architecture to Mechanical, Electrical and Software Entities
7.3
Plan Physical Architecture Alternatives
7.4
Analyze, Review and Refine Architecture Alternatives
7.5
Select Preferred Physical Architecture
End Procedure 7



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

# DOCUMENT CONTROL

## 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 | |
| **09/17/2018** | Roman Legat | 1.1 | Layout update to 2018 style |
| | | | |
| | | | |
| | | | |