Interfaces [Functional View] [Logical View]

Docupedia Export

Author:Mukkamala Kiran (XC-AS/EDA2)
Date:29-Jan-2024 11:23

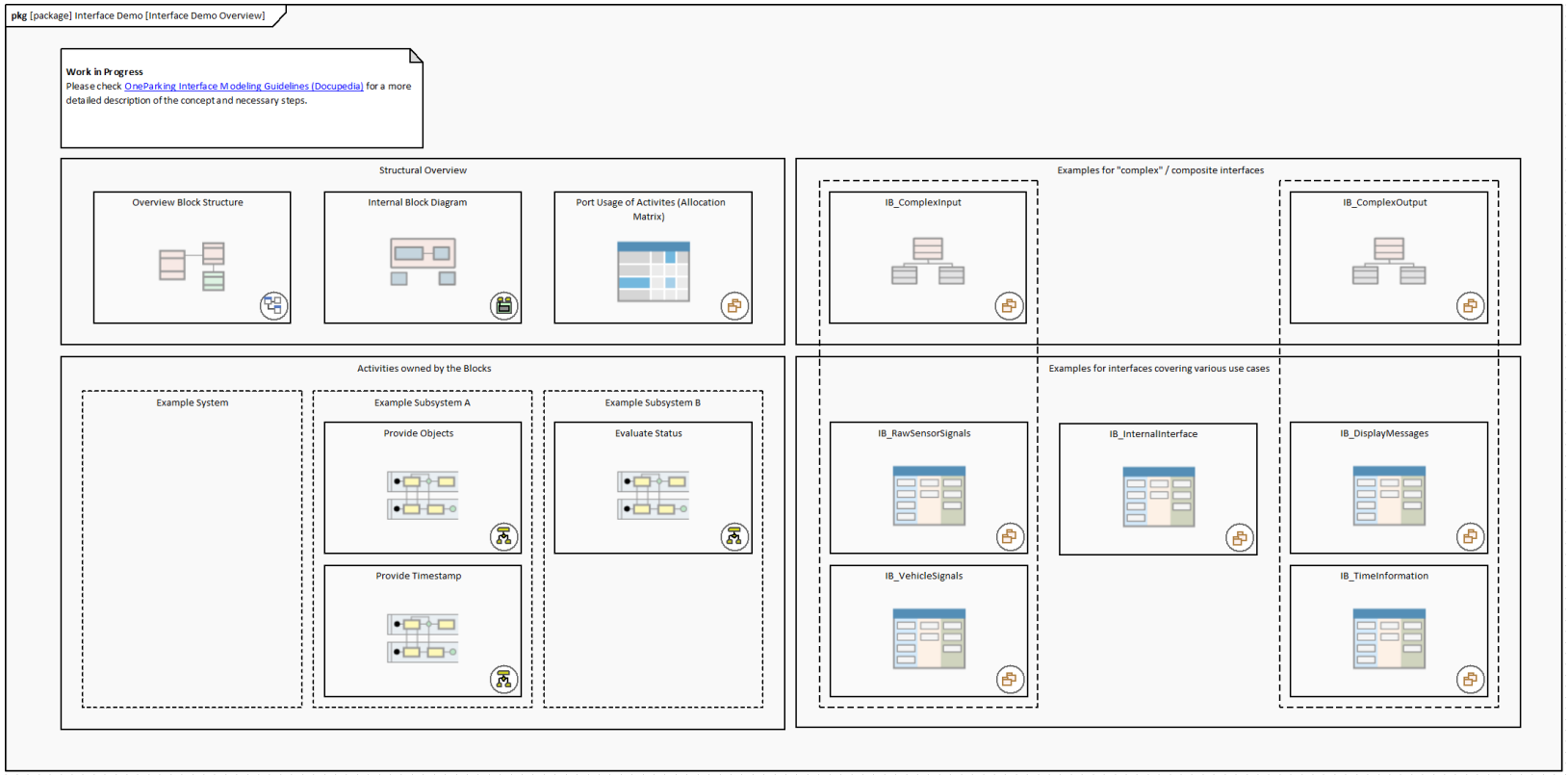
Table of Contents

1 Overview

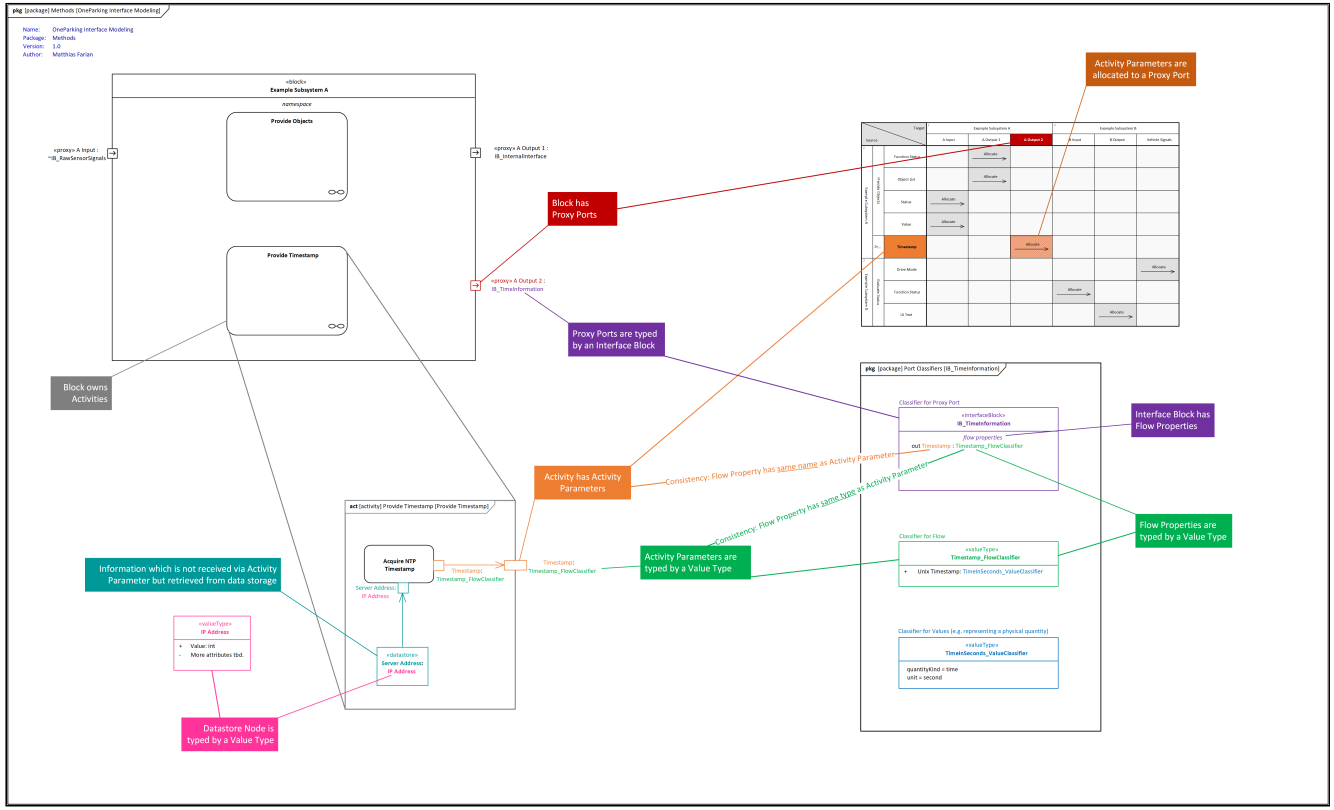
In this guideline, we will cover interfaces in both the functional and the logical view.
We will take a closer look at modeling

- Inputs and outputs of actions and activities
- Ports of logical blocks
- Allocation of inputs and outputs of activities to ports of logical blocks
- Type definition of
    - Ports
    - Flows
    - Value properties
- Examples for
    - Basic interfaces
    - Complex interfaces with inheritance
    - Lists with variable entries
    - Simple types
    - Custom value types
    - Enumerations

You can find this example in WebEA:



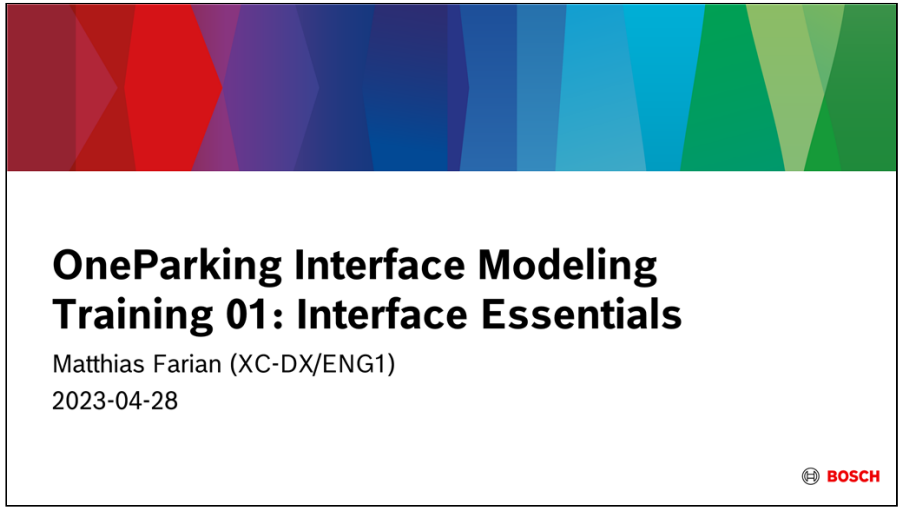
2 Big Picture



Info

This chapter contains a reusable documentation segment `(PageWithExcerpt = Interfaces [Functional View] [Logical View] | MultiExcerptName = 1P_MBSE_Interface_Modeling_Big_Picture)`

3 Training Material



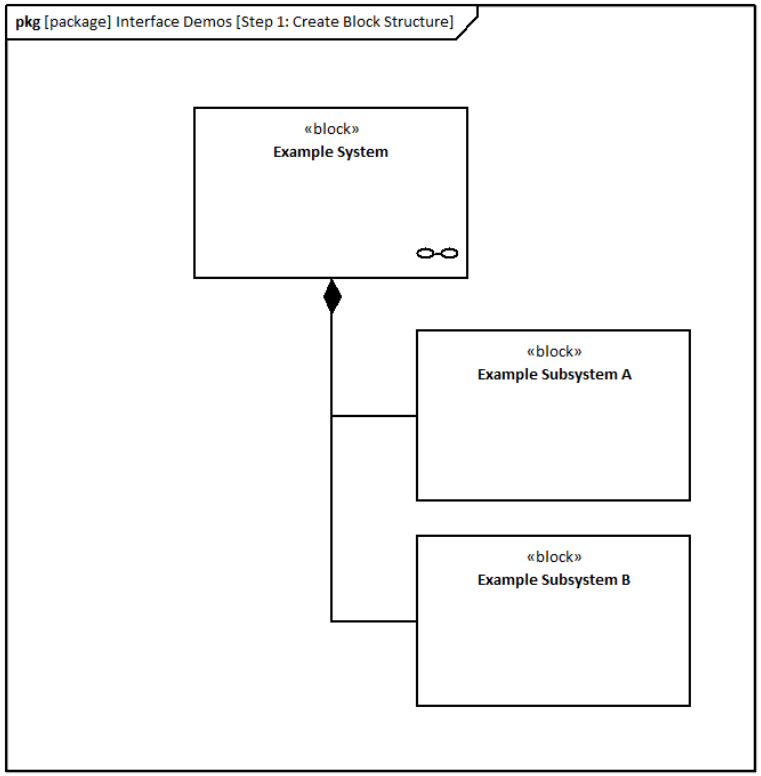
Slides: OneParking_Interface_Modeling_-_01_Interface_Training_Essentials.pptx
Enterprise Architect Files: OneParking_Interface_Modeling_-_01_Interface_Training_Essentials.zip

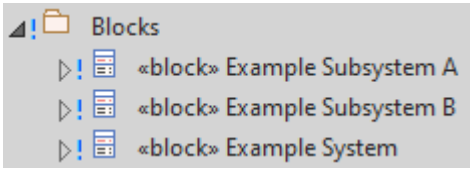
4 Basic structure of example system

This method of interface modeling can be applied to any MBSE level, it might as well be an L3 subsystem consisting of several L4 blocks.

4.1 Block relations

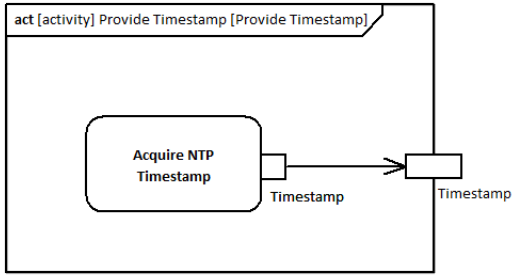
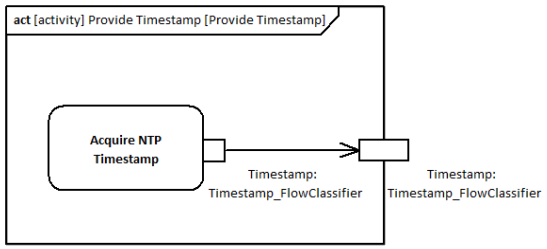
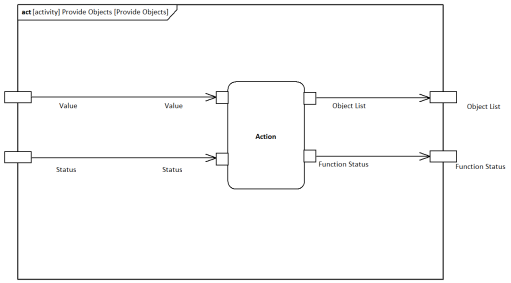
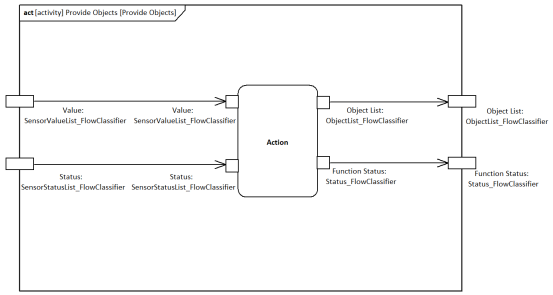
This example consists of a simplified system with two exemplary subsystem blocks.

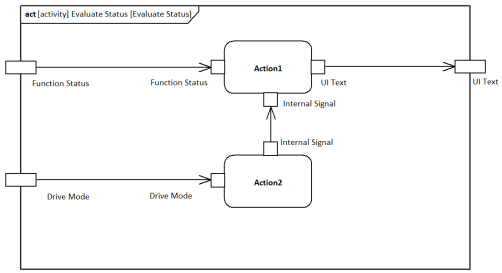
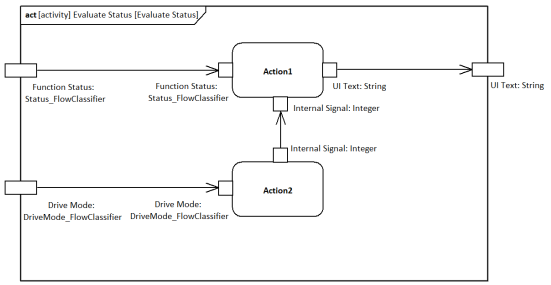


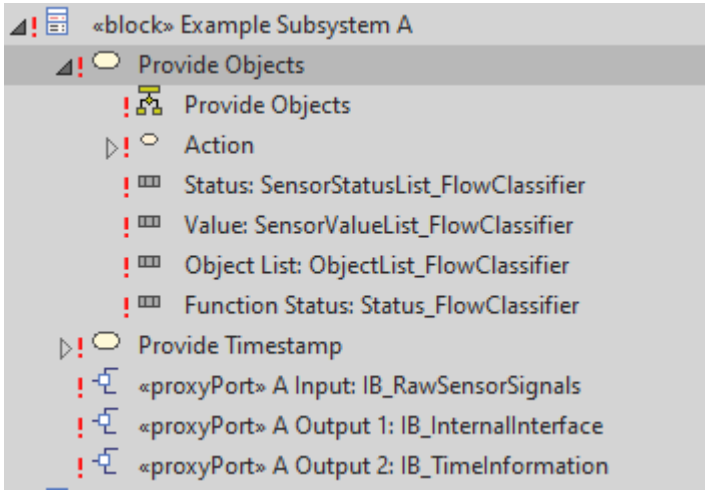


4.2 Activities of subsystem blocks

Each block can own multiple activities, with any number of inputs and outputs. Activity parameter nodes act as interfaces for activities.

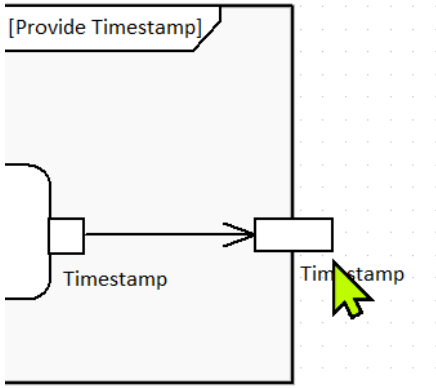
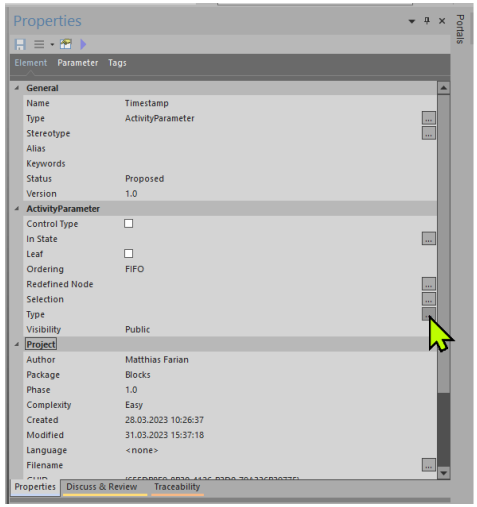
Activity	Step 1: Create Activity Parameters and Action Pins	Step 2: Assign Type Classifier to Pins and Parameter Nodes	
Subsystem A: "Provide Timestamp" Basic Example with just a single output			
Subsystem A: "Provide Objects" Extended example with multiple inputs and outputs, consisting of ordered lists			

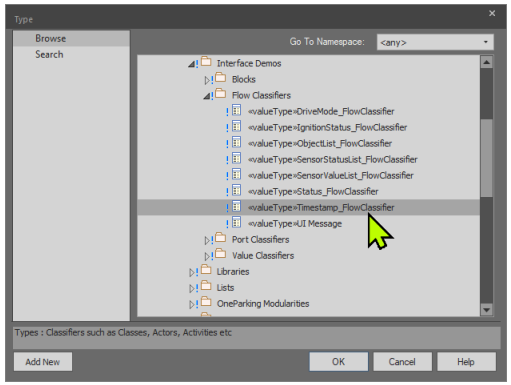
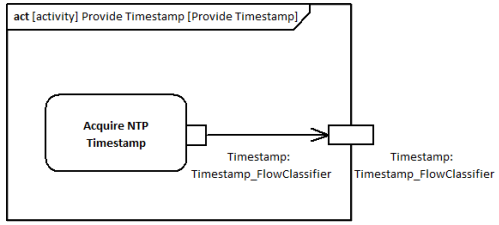
Activity	Step 1: Create Activity Parameters and Action Pins	Step 2: Assign Type Classifier to Pins and Parameter Nodes	
Subsystem B: "Evaluate Status" Extended example with multiple inputs and a single output consisting			



4.2.1 Activities: assigning types to pins and parameters

Assumption: the relevant type classifiers are already available in the model. If not, please compare chapter on specifying type.

Select activity parameter	
Use the "Type" field in the properties window or press <CTRL><L>	

Select type classifier	
Result after assigning types to all action pins and activity parameters	

4.3 Adding proxy ports to blocks

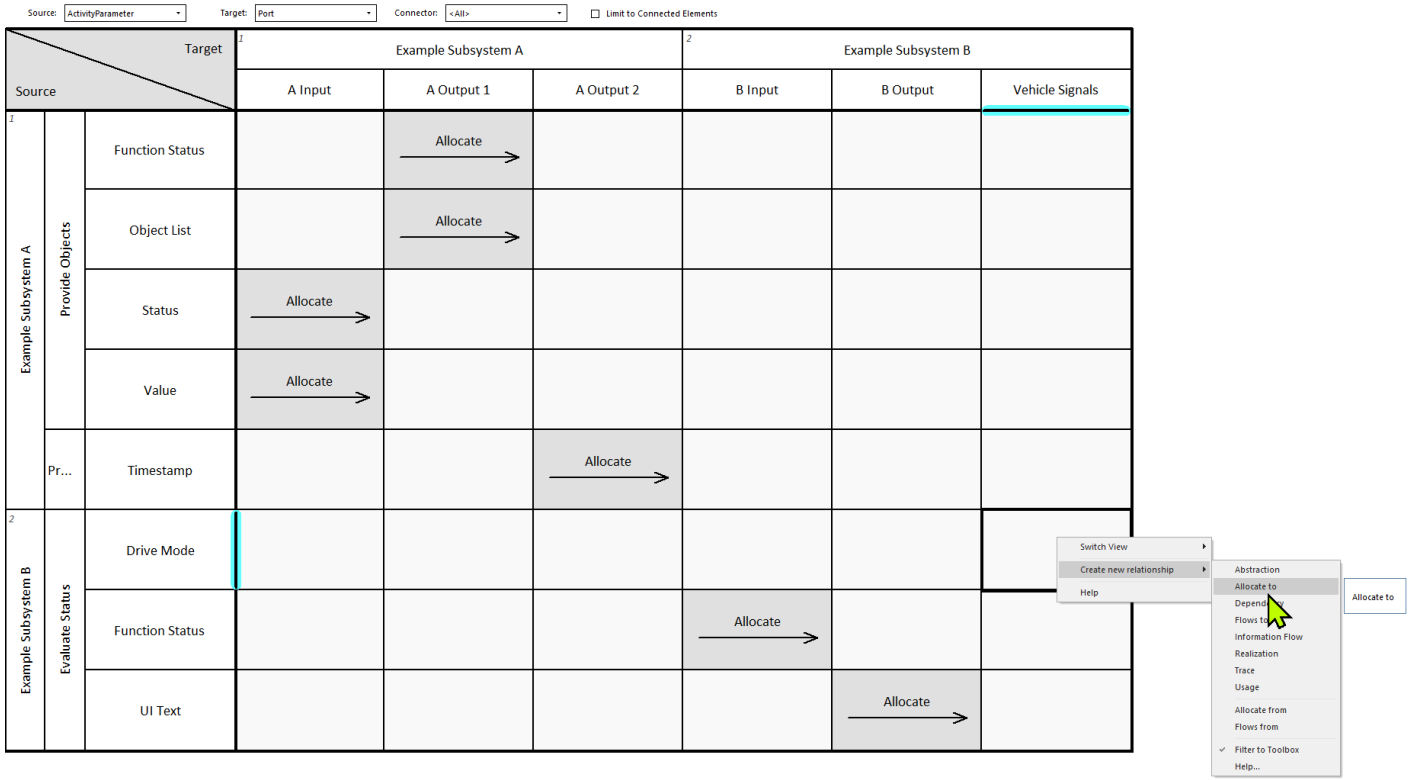
[example will follow soon]

4.4 Allocating activity parameters to proxy ports

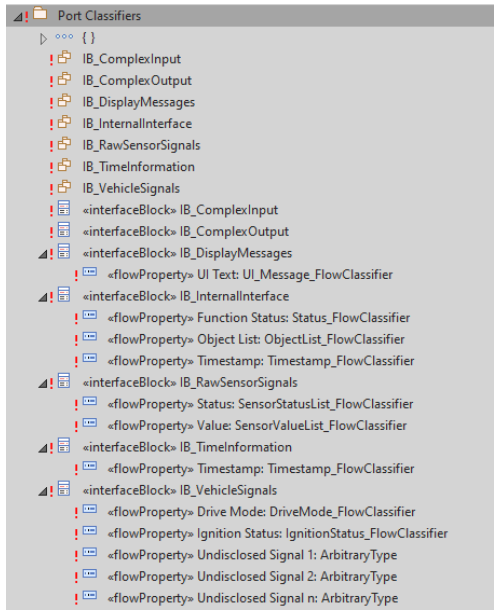
Pre-requisites:

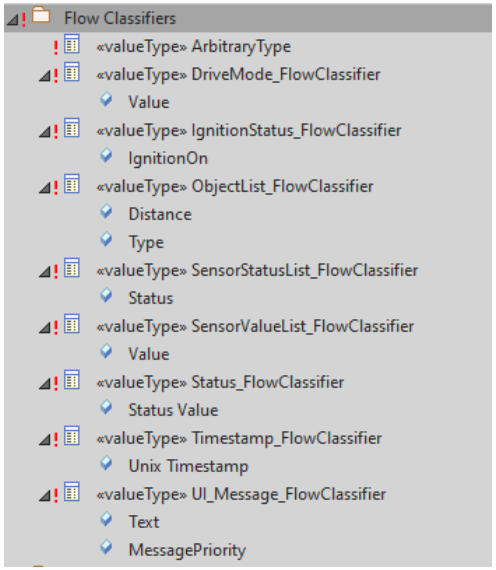
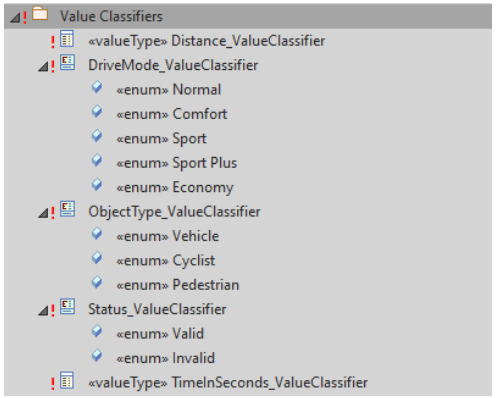
Proxy ports are typed by interface blocks, owning flow properties. The name of each activity parameter matches the name of a flow property. By allocating the activity parameter to a port, their relation becomes unambiguous.

⚠️ Make sure to have consistent names for activity parameters and action pins which are connected by object flow connectors.



4.5 Type classifiers

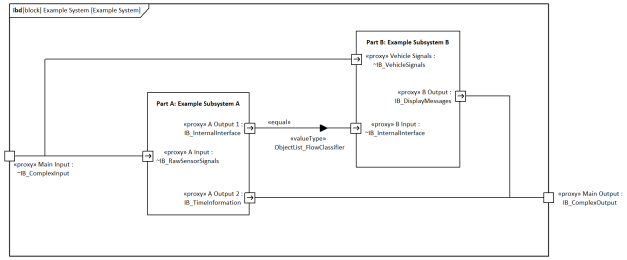
Interface blocks used as type classifiers for proxy ports Flow properties represent the individual signals.	

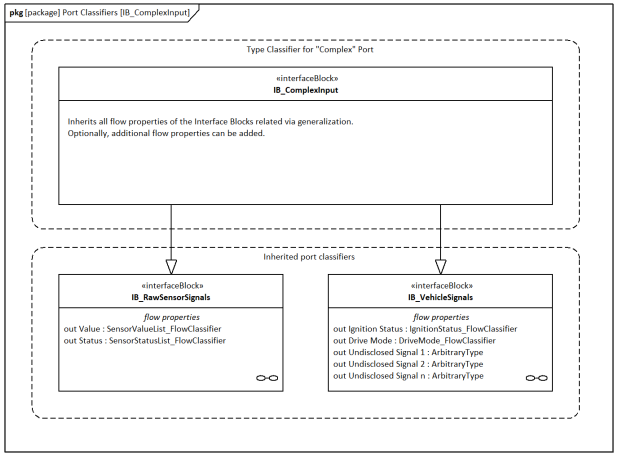
Value types used as type classifiers for flows (individual signals).	
Value types and enumerations used as type classifiers for fine-grained value definition.	

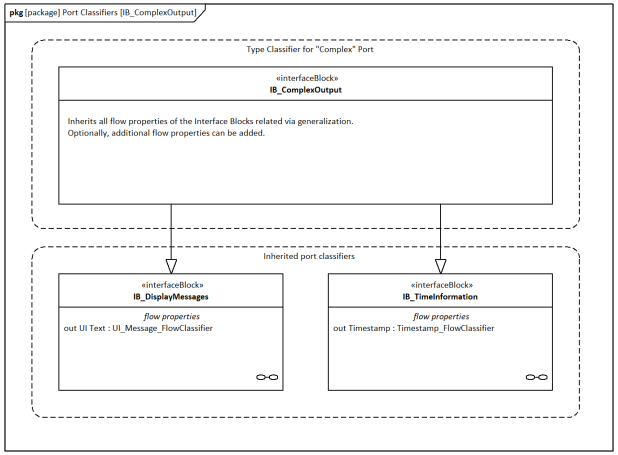
...

4.6 Defining "complex" interface Types

Interface blocks can have generalization relations to other interface blocks. This enables inheriting properties from other interface blocks.



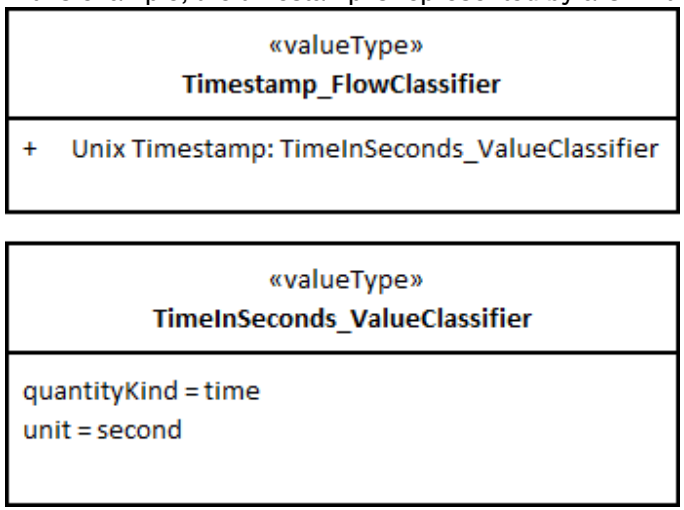




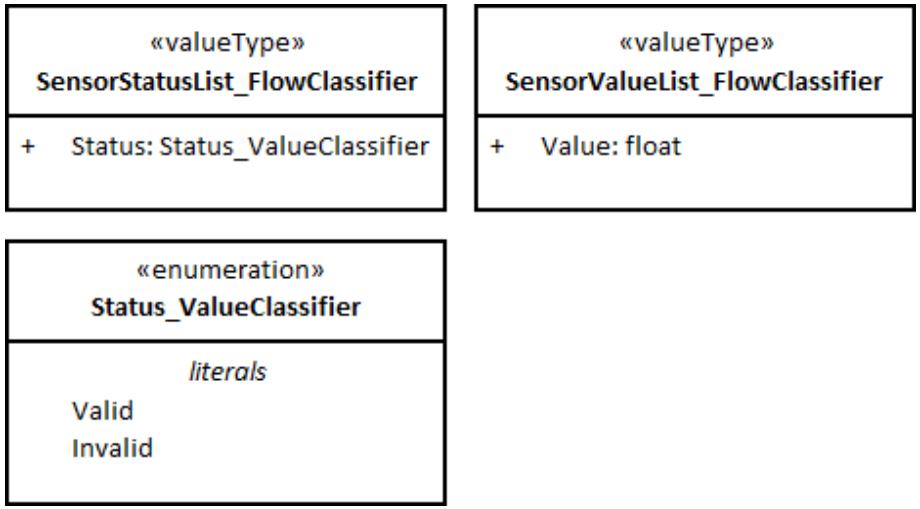
Example for application of complex interface types:

4.6.1 Specifying types

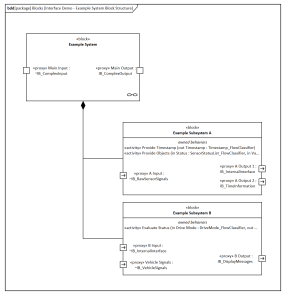
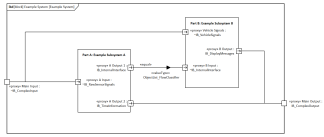
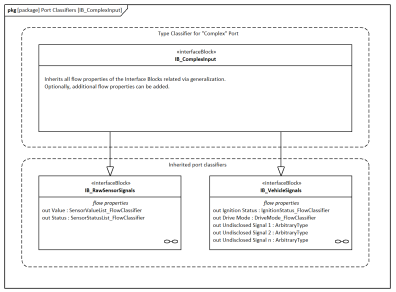
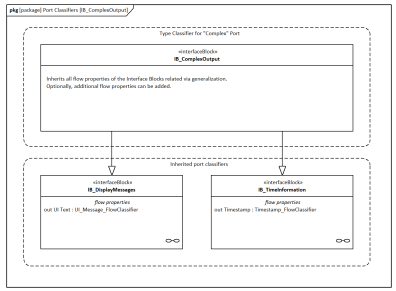
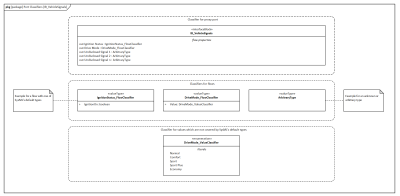
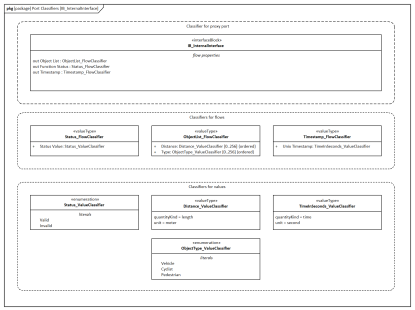
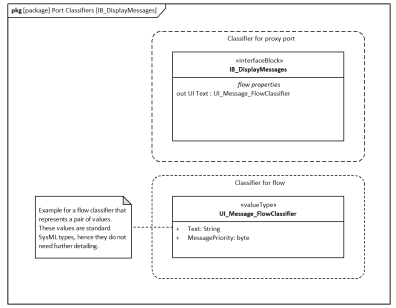
In this example, the timestamp is represented by a Unix timestamp, which itself is a time quantity with unit seconds.

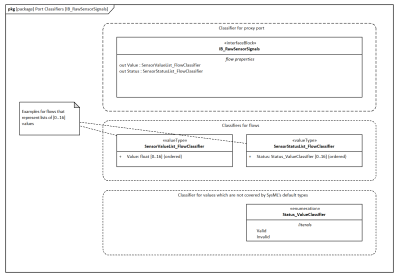
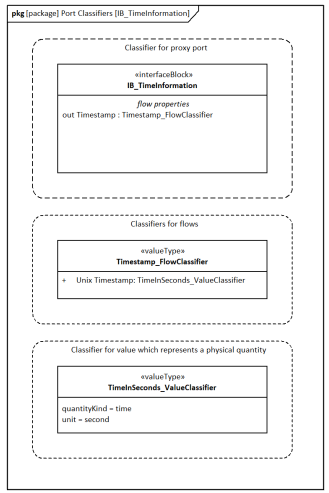


The inputs and outputs of the "Provide Objects" activity are a bit more complicated.

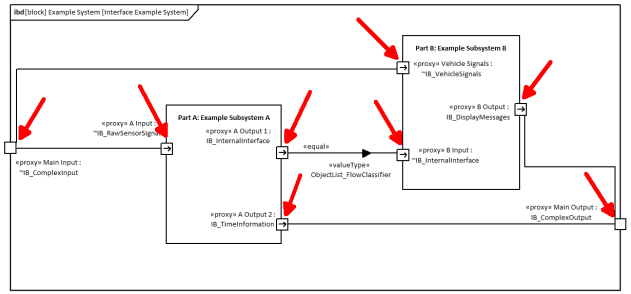


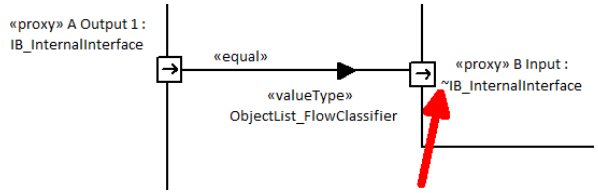
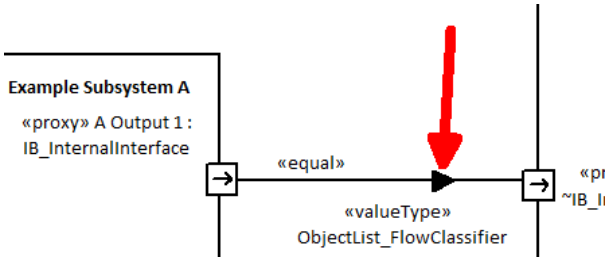
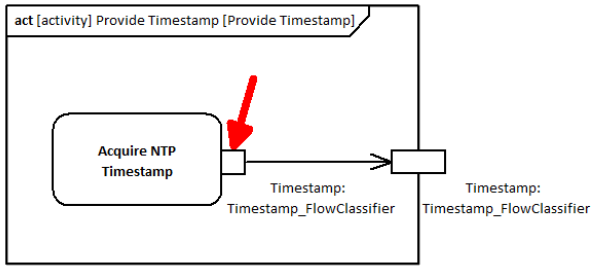
5 Elements and Diagrams

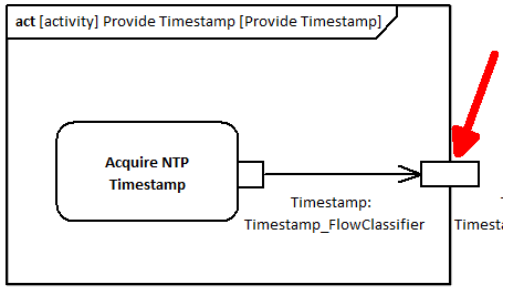
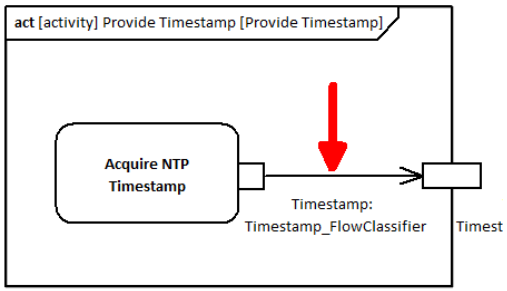
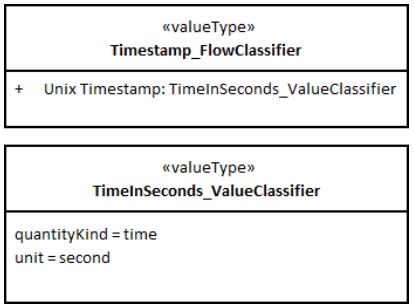
Block structure	Internal block diagram	Interface definition of complex input	Interface definition of internal interface	Interface definition of complex output
				
				

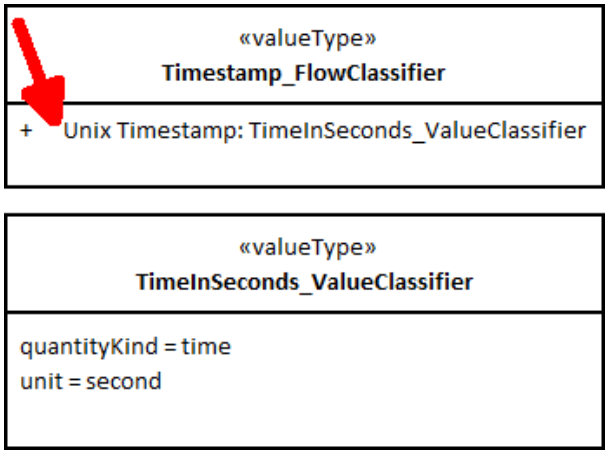
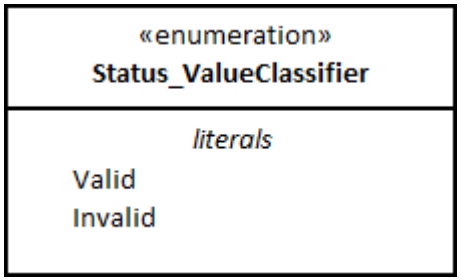
Block structure	Internal block diagram	Interface definition of complex input	Interface definition of internal interface	Interface definition of complex output
				

6 Relevant SysML Terms and Definitions

Name		Description
Proxy Port		Ports are points at which external entities can connect to and interact with a block. A proxy port acts as proxy for their owning blocks or its internal parts. They define the boundary by specifying which features of the owning block or internal parts are visible through external connectors. (1)
Interface Block	«interfaceBlock» **IB_RawSensorSignals** *flow properties* out Value : SensorValueList_FlowClassifier[0..16] {unique, ordered} out Status : SensorStatusList_FlowClassifier[0..16] {unique, ordered}	REPLACE SCREENSHOT: • ambiguous multipliers for flows and value properties Proxy ports are always typed by interface blocks, a specialized kind of block that has no behaviors or internal parts. (1)
Flow Property	«interfaceBlock» **IB_RawSensorSignals** *flow properties* out Value : SensorValueList_FlowClassifier[0..16] {unique, ordered} out Status : SensorStatusList_FlowClassifier[0..16] {unique, ordered}	REPLACE SCREENSHOT: • ambiguous multipliers for flows and value properties Flow properties specify the kinds of items that might flow between a block and its environment, whether it is data, material, or energy. The kind of items that flow is specified by typing flow properties. A FlowProperty signifies a single kind of flow element that *can* flow to/from a block. A flow property's values are either received from or transmitted to another block. (1) Convention in OneParking: Flow Properties shall be modeled with direction "out".

Name		Description
Conjugation		The meaning of the direction is reversed for conjugated ports. (1)
Item Flow		Item flows specify the things that flow between blocks and/or parts and across associations or connectors. *Note: Whereas flow properties specify what "can" flow in or out of a block, item flows specify what "does" flow between blocks and/or parts in a particular usage context.* (1)
Action Pin		An Action Pin is used to define the data values passed out of and into an Action. An Input Pin provides values to the Action, whereas an Output Pin contains the results from that Action. (2)

Name		Description
Activity Parameter (Activity Parameter Node)		An Activity Parameter Node accepts input to an Activity or provides output from an Activity. (3)
Object Flow		
Value Type		

Name		Description
Value Property		
Enumeration		

Name		Description
Quantity Kind	«valueType**Timestamp_FlowClassifier**+ Unix Timestamp: TimeInSeconds_ValueClassifier «valueType**TimeInSeconds_ValueClassifier**quantityKind = time unit = second	
Unit	«valueType**Timestamp_FlowClassifier**+ Unix Timestamp: TimeInSeconds_ValueClassifier «valueType**TimeInSeconds_ValueClassifier**quantityKind = time unit = second	

Sources:
(1): OMG Systems Modeling Language, v1.5, Chapter 9 Ports and Flows
(2): Sparx Systems Enterprise Architect User Guide: Action Pin
(3): Sparx Systems Enterprise Architect User Guide: Activity Parameter Node

7 Links and further Documentation

- BBM Model-Based Systems Engineering: Interface Concept
- BBM Model-Based Systems Engineering: Naming Convention Recommendation