Enterprise Architect

User Guide Series

SysML Models

Author: Sparx Systems

Date: 15/07/2016

Version: 1.0

CREATED WITH ENTERPRISE ARCHITECT

Table of Contents

Systems Engineering

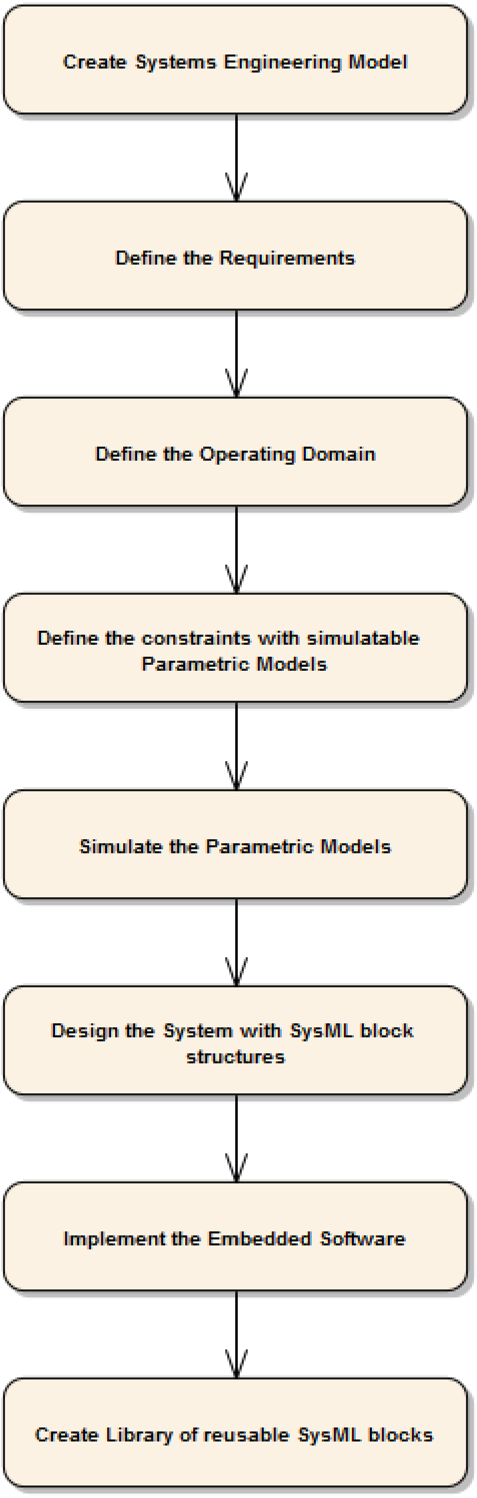
Systems engineering is a field of engineering that deals with how to research, design and manage complex systems over their life time. It focuses on the whole system and typically involves a number of sub-disciplines including requirements, reliability, logistics, design, testing and maintenance; it requires sophisticated project management techniques.

Systems engineering not only deals with the problem of the system itself but also involves processes, optimization and risk management. Enterprise Architect provides a platform for the system engineer using the Systems Modeling Language (SysML) and model based development, and in addition to the engineering models the tool provides support for project and process management.

Model Systems using SysML in Enterprise Architect

Work through these steps:

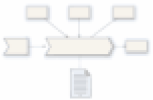
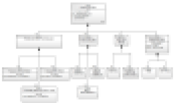
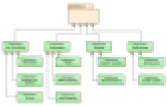
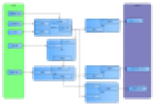
The steps are represented by this flowchart:


Create Systems Engineering Model
Define the Requirements
Define the Operating Domain
Define the constraints with simulatable Parametric Models
Simulate the Parametric Models
Design the System with SysML block structures
Implement the Embedded Software
Create Library of reusable SysML blocks

Step
Create a Systems Engineering model to develop your system
Create a Requirements model to define the systems requirements and expectations
Create an Operational Domain model, which describes the environment that the system operates within, and the entities it interacts with

Create Constraint models to describe the systems operating characteristics using parametric models
Simulate the parametric models to verify their correctness and obtain the desired characteristic
Design the system's composition using SysML Blocks and Parts
Implement the embedded software using UML Classes and behavioral models
Create a Library of reusable SysML blocks, representing subsystems that can be reused on other projects, and other common Type definitions

Areas to consider

Subject	Details
Systems Engineering Modeling 	A step by step guide to creating your first Systems Engineering Model.
SysML 	Get started with SysML with this introduction to the MDG Technology, available toolboxes and more.
SysML Requirements 	The SysML Requirements Model provides the system requirements, the expected abstract behavior, and the operating constraints that the designed system must conform to.
Parametrics 	SysML Parametric models support the engineering analysis of critical system parameters, including the evaluation of key metrics such as performance, reliability and other physical characteristics.

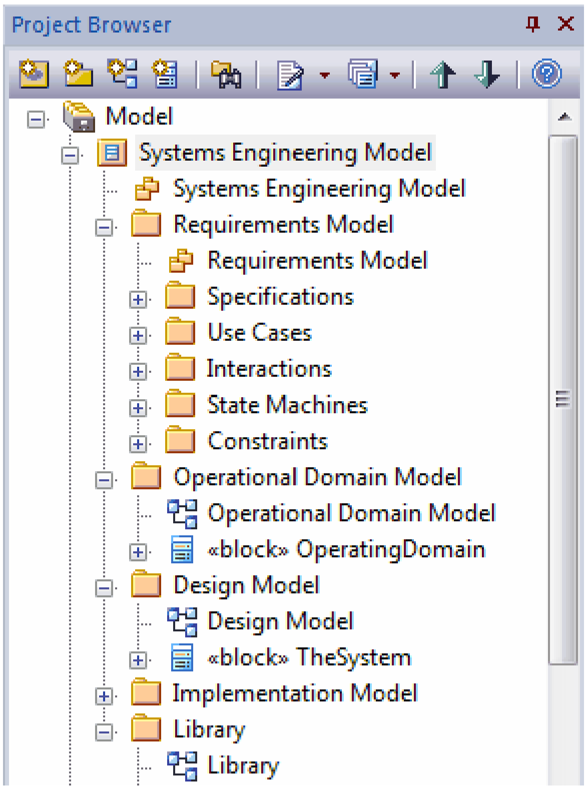
Create a Systems Engineering Model from a template

Step	Action
1	In the **Project Browser**, either: • Click on the 'New Model From Pattern' icon in the toolbar • Right-click on a model root node and select the 'Add a New Model using Wizard' option, or • Right-click on a Package and select the 'Add | Add a New Model using Wizard' option

	The 'Select Model(s)' dialog displays.
2	In the 'Select From' field, click on the drop-down arrow and select 'Systems Engineering Model'. Alternatively, if it is listed in the 'Technology' panel, select the 'Systems Engineering Model' item.
3	In the 'Name' panel, select the checkbox next to the 'Systems Engineering Model' icon.
4	Click on the **OK button**.

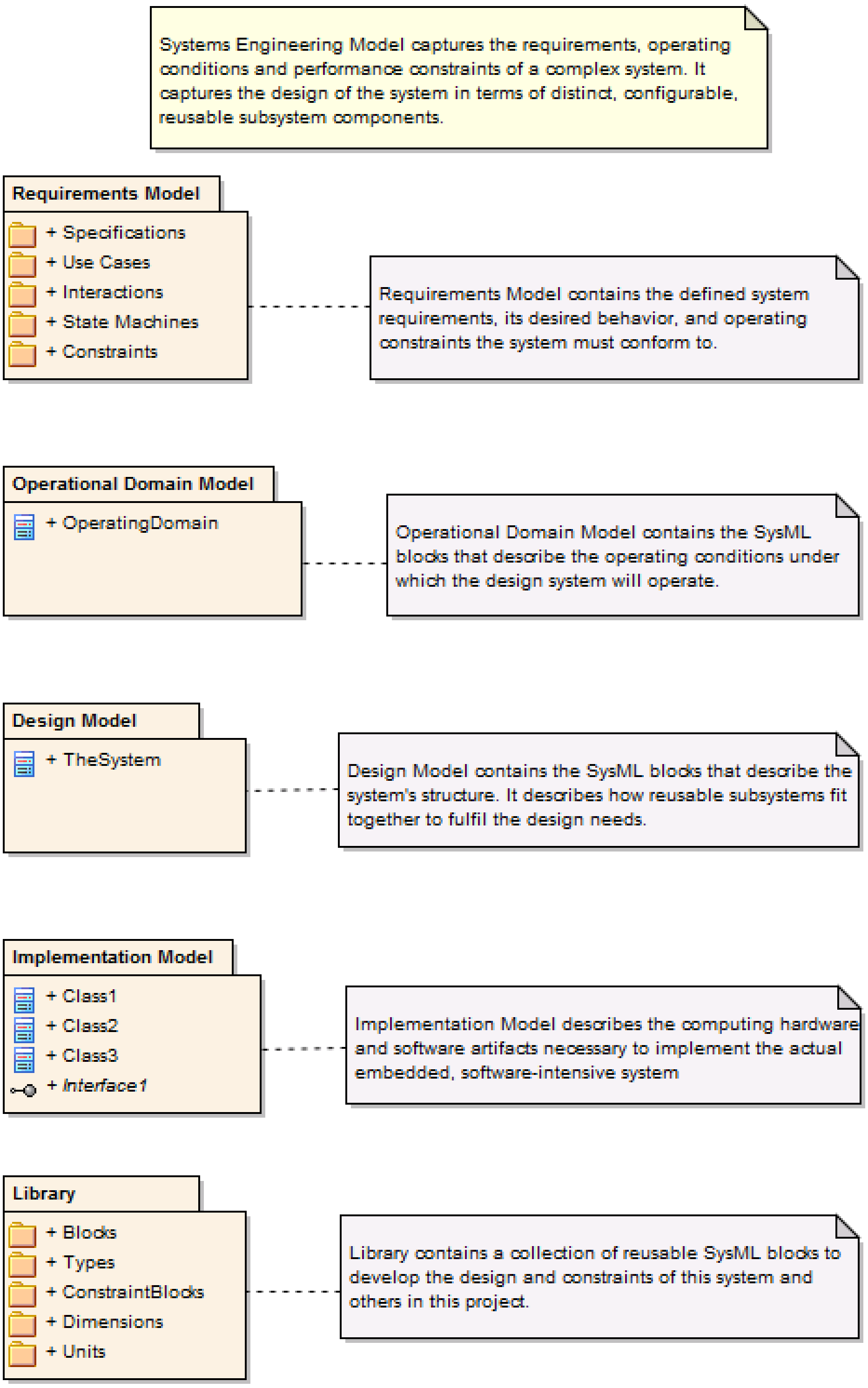
Example

This model structure is created in the **Project Browser**:



- Model
  - Systems Engineering Model
    - Systems Engineering Model
    - Requirements Model
      - Requirements Model
      - Specifications
      - Use Cases
      - Interactions
      - State Machines
      - Constraints
    - Operational Domain Model
      - Operational Domain Model
      - «block» OperatingDomain
    - Design Model
      - Design Model
      - «block» TheSystem
    - Implementation Model
    - Library
      - Library

This Systems Engineering Model diagram encapsulates the key components of the Systems Engineering model.



Systems Engineering Model captures the requirements, operating conditions and performance constraints of a complex system. It captures the design of the system in terms of distinct, configurable, reusable subsystem components.

Requirements Model contains the defined system requirements, its desired behavior, and operating constraints the system must conform to.

Operational Domain Model contains the SysML blocks that describe the operating conditions under which the design system will operate.

Design Model contains the SysML blocks that describe the system's structure. It describes how reusable subsystems fit together to fulfil the design needs.

Implementation Model describes the computing hardware and software artifacts necessary to implement the actual embedded, software-intensive system

Library contains a collection of reusable SysML blocks to develop the design and constraints of this system and others in this project.

Notes

- Systems Modeling Language (SysML) is provided with the Systems Engineering edition and Ultimate edition of Enterprise Architect

Learning Center topics

- **Alt+F1** | Enterprise Architect | Systems Engineering

Systems Modeling Language (SysML)

To develop SysML models quickly and simply, you can use one of four versions of the MDG Technology for SysML, integrated with Enterprise Architect. The available versions correspond to SysML 1.1, SysML 1.2, SysML 1.3 and SysML 1.4.

This text is derived from the official OMG SysML site of the Object Management Group.

'The OMG Systems Modeling Language (OMG SysML™) is a general-purpose graphical modeling language for specifying, analyzing, designing, and verifying complex systems that may include hardware, software, information, personnel, procedures, and facilities. The language provides graphical representations with a semantic foundation for modeling system requirements, behavior, structure, and parametrics, which is used to integrate with other engineering analysis models. SysML was developed in response to requirements developed jointly by the OMG and the International Council on Systems Engineering (INCOSE) by the diverse group of tool vendors, end users, academia, and government representatives.'

Access

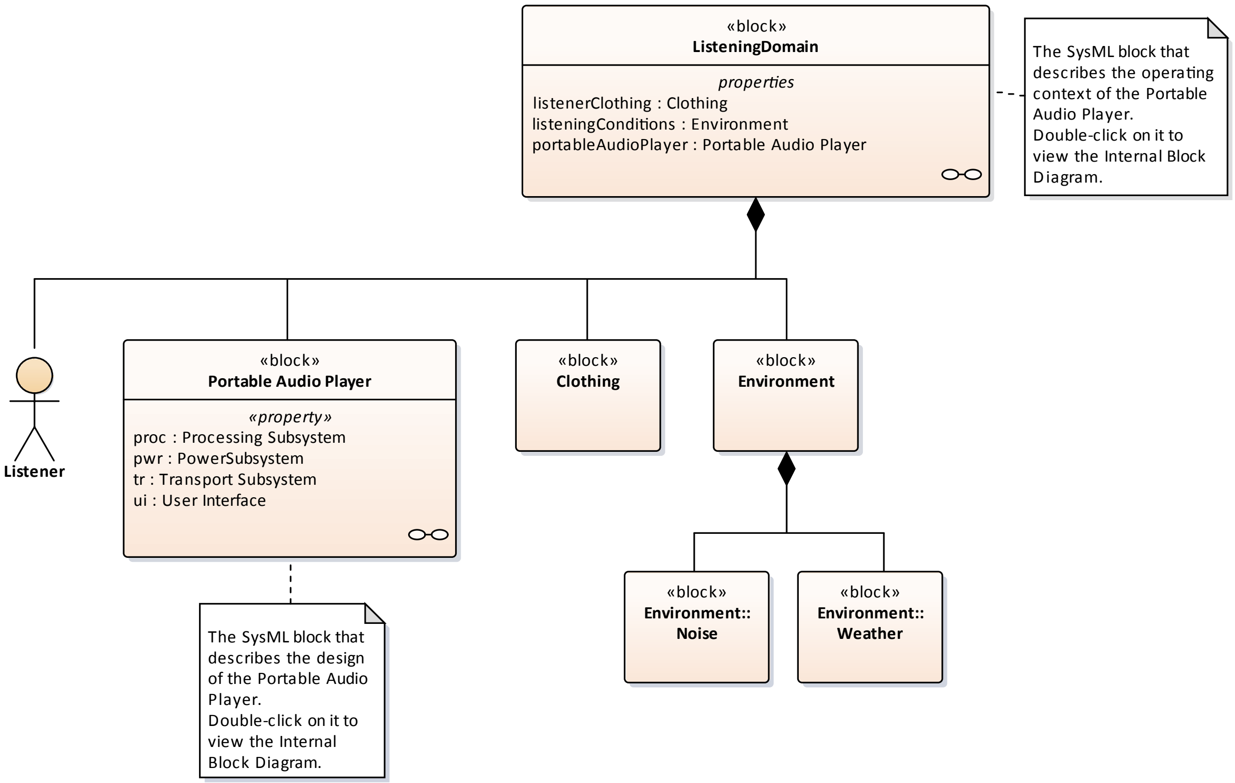
On the **Diagram Toolbox**, click on the More tools button and select:

- SysML 1.1
- SysML 1.2
- SysML 1.3 or
- SysML 1.4

Ribbon	Design > Diagram > Toolbox
Menu	Diagram | Toolbox
Keyboard Shortcuts	**Alt+5**

Example SysML Diagram

This is an example of a **SysML Block** diagram, which can also be found in the EAExample model under Systems Engineering > Listening Domain.


«block»
ListeningDomain
properties
listenerClothing : Clothing
listeningConditions : Environment
portableAudioPlayer : Portable Audio Player
The SysML block that describes the operating context of the Portable Audio Player. Double-click on it to view the Internal Block Diagram.
Listener
«block»
Portable Audio Player
«property»
proc : Processing Subsystem
pwr : PowerSubsystem
tr : Transport Subsystem
ui : User Interface
«block»
Clothing
«block»
Environment
The SysML block that describes the design of the Portable Audio Player. Double-click on it to view the Internal Block Diagram.
«block»
Environment::
Noise
«block»
Environment::
Weather

SysML Integration

Facilities	Detail
SysML In Enterprise Architect	The MDG Technology for SysML provides: - Each of the nine SysML diagram types, accessed through the 'New Diagram' dialog - A collection of SysML pages in the **Diagram Toolbox** that contain the SysML elements and relationships for each of the supported diagram types - SysML element and relationship entries in the 'Toolbox Shortcut Menu' and Quick Linker - Simulation of SysML Parametric diagrams, which supports engineering analysis of critical system parameters including the evaluation of key metrics such as performance, reliability and other physical characteristics
SysML Toolboxes	The MDG Technology for SysML provides **Diagram Toolbox** pages that you can access through the **More Tools button**. If you enable SysML as the active technology, you can also open the SysML Toolbox pages by default. These sets of Toolbox pages are available: - Activity contains the constructs required to construct SysML Activity models - Block Definition contains the constructs required to design SysML blocks, constraint blocks, data and value types - Interaction contains the constructs required to construct SysML interactions and Sequence diagrams - Internal Block contains the constructs required to design SysML block compositions within Internal Block Diagrams

	• Model Elements contains the constructs required to build SysML models, Package structures and views • Parametrics contains the constructs required to construct SysML Parametric Diagrams using constraint blocks • Requirements contains the constructs required to build SysML Requirements models • State Machine contains the constructs required to build SysML State Machines • Use Case contains the constructs required to build SysML Use Case models With the 'Model Elements' pages there is a set of SysML Common elements and relationships; these are also provided with the other SysML Toolbox pages if the active technology is set on the Default Tools toolbar to SysML 1.1, SysML 1.2, SysML 1.3 or SysML 1.4.
Upgrade SysML Models	You can migrate a SysML model (or part of a model) to a later SysML version, using the **Automation Interface**.
Disable SysML	If you prefer not to use SysML in Enterprise Architect, you can disable it (and subsequently re-enable it) using the 'MDG Technologies' dialog ('Extensions | MDG Technologies').

Notes

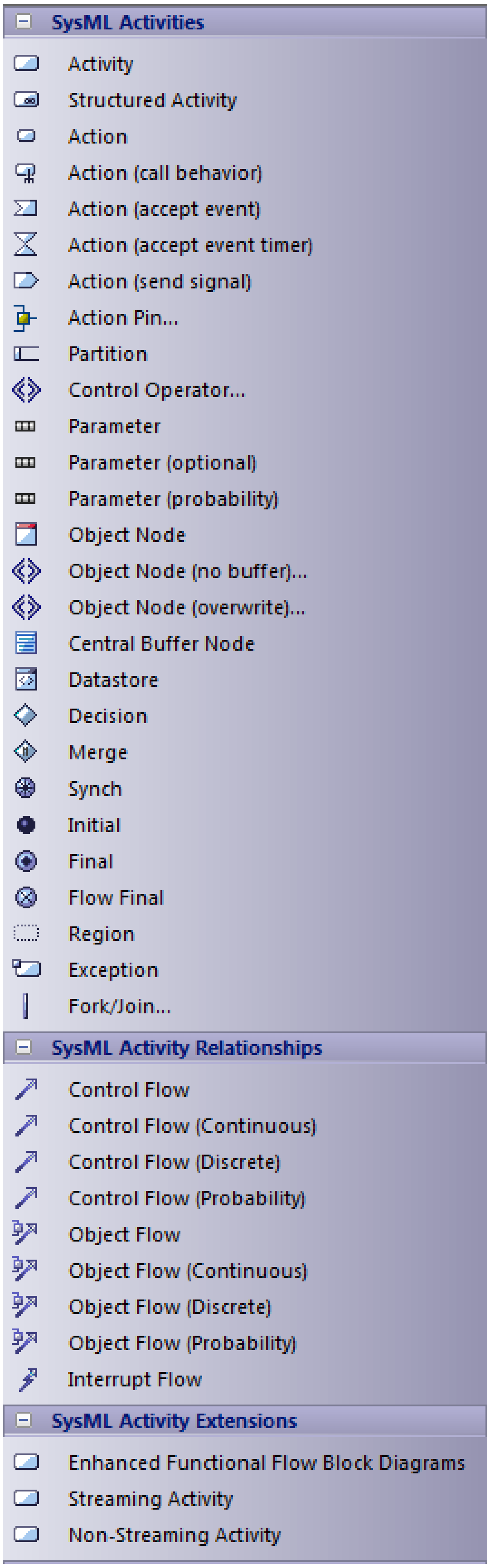
- Systems Modeling Language (SysML) is provided with the Systems Engineering and Ultimate editions of Enterprise Architect

SysML Activity Toolbox

When you are constructing SysML models, you can populate the Activity diagrams using the icons on the 'SysML Activity' pages of the **Diagram Toolbox**.

The Block that owns the Activity diagram is automatically represented by a diagram frame enclosing the Activity diagram elements. You can:

- Hide the frame by right-clicking on the diagram and selecting the 'Hide Diagram Frame' option (and show it again by selecting the 'Show Diagram Frame' option)
- Make the frame selectable to move or resize it, by right-clicking on it and selecting the 'Selectable' option
- Create Ports and Parts on the frame and create connectors between them



SysML Activities

- Activity
- Structured Activity
- Action
- Action (call behavior)
- Action (accept event)
- Action (accept event timer)
- Action (send signal)
- Action Pin...
- Partition
- Control Operator...
- Parameter
- Parameter (optional)
- Parameter (probability)
- Object Node
- Object Node (no buffer)...
- Object Node (overwrite)...
- Central Buffer Node
- Datastore
- Decision
- Merge
- Synch
- Initial
- Final
- Flow Final
- Region
- Exception
- Fork/Join...

SysML Activity Relationships

- Control Flow
- Control Flow (Continuous)
- Control Flow (Discrete)
- Control Flow (Probability)
- Object Flow
- Object Flow (Continuous)
- Object Flow (Discrete)
- Object Flow (Probability)
- Interrupt Flow

SysML Activity Extensions

- Enhanced Functional Flow Block Diagrams
- Streaming Activity
- Non-Streaming Activity

Access

On the **Diagram Toolbox**, click on the More tools button and select 'SysML 1.3 | SysML 1.3 Activity'.

Ribbon	Design > Diagram > Toolbox
Menu	Diagram | Toolbox
Keyboard Shortcuts	**Alt+5**

SysML Activities

Item	Action
Activity	Defines a **SysML Block** of executable behavior as a **UML Activity**.
Structured Activity	Defines a **SysML Block** of executable behavior as a **UML Structured Activity**.
Action	Declares a unit of execution in an Activity as a **UML Action**.
Action (call behavior)	Declares a unit of execution that calls another behavior.
Action (accept event)	Declares a unit of execution that accepts an event raised by the system.
Action (accept event timer)	Declares a unit of execution that accepts an event raised by a time epoch.
Action (send signal)	Declares a unit of execution that sends a signal as an event.
Action Pin	Defines the data values passed out of and into an Action. See also: Action Pin
Partition	Creates an Activity Partition to group execution elements according to the node responsible for their execution.
Control Operator	Controls the execution of an Activity.
Parameter	Provides access to input and output objects within the Activity.
Parameter (optional)	Defines a parameter whose contents are optional in the Activity's execution.
Parameter (probability)	Tags a parameter with the probability of the parameter being in use in the Activity.
Object Node	Declares a variable in the Activity, typed by a ValueType, DataType or Block.
Object Node (no buffer)	Declares an ObjectNode in an Activity that discards unconsumed tokens.
Object Node (overwrite)	Declares an ObjectNode in an Activity that overwrites tokens.
Central Buffer Node	Declares an ObjectNode that stores tokens for consumption throughout the Activity.

Datastore	Defines permanently stored data. See also: Datastore
Decision	Creates a branch of control in an Activity, based on a decision.
Merge	Merges two or more Activity control branches.
Synch	Establishes a rendezvous point for two or more Activity flows, in order to synchronize their execution in the Activity.
Initial	Declares the start of an Activity's execution.
Final	Declares the end of an Activity's execution, and the termination of the Activity.
Flow Final	Declares the end of an Activity's execution path without terminating the Activity.
Region	Groups a subset of an Activity into a common execution context.
Exception	Declares a node of execution that happens outside the normal flow of execution of an Activity.
Fork/Join	Simultaneously branches / joins a set of Control or Object Flows.

SysML Activity Relationships

Item	Action
Control Flow	Establishes a flow of logic between two Activity nodes.
Control Flow (Continuous)	Declares a continuous control flow.
Control Flow (Discrete)	Declares a discrete control flow.
Control Flow (Probability)	Tags a control flow with a probability of the likelihood of the flow's traversal.
Object Flow	Establishes a flow of objects (data) between two Activity nodes.
Object Flow (Continuous)	Declares a continuous object flow.
Object Flow (Discrete)	Declares a discrete object flow.
Object Flow (Probability)	Tags an object flow with the probability of the flow's traversal.
Interrupt Flow	Declares a control flow that interrupts flow within a Region.

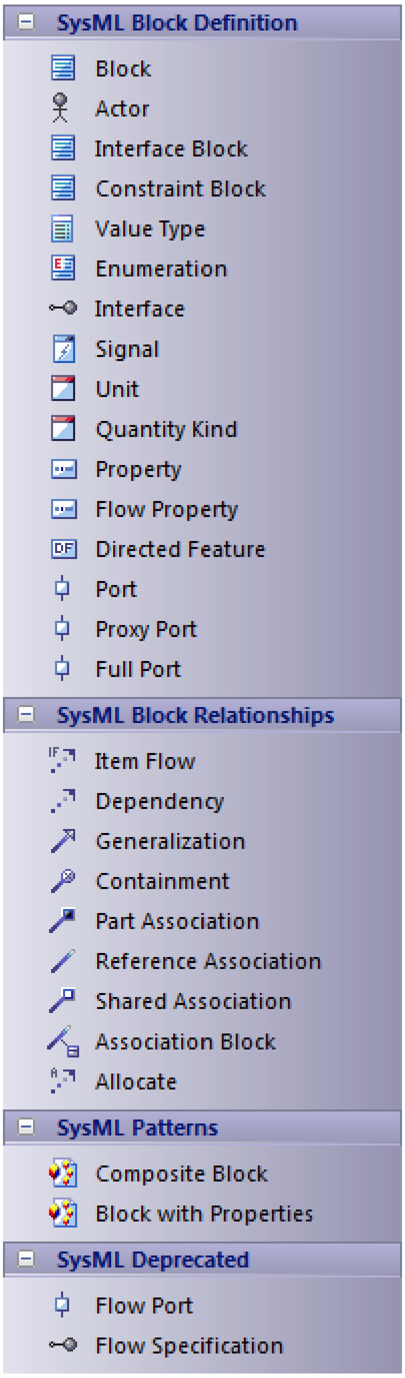
SysML Activity Extensions

Item	Action
Enhanced Functional Flow Block Diagrams	Action: Declares an Activity used to contain an Enhanced Functional Flow Block Diagram (EFFBD).
Streaming Activity	Declares an Activity where the flow of tokens passes through its parameters continuously throughout the Activity's execution.
Non-Streaming Activity	Declares an Activity where the flow of tokens passes through its parameters at the start of the Activity's execution.

SysML Block Definition Toolbox

When you are constructing SysML models, you can populate the Block Definition diagrams using the icons on the **SysML Block** Definition pages of the **Diagram Toolbox**.

You can also generate Property elements on the Block, based on the Association relationships created for the Block element. These Properties (or Parts) are initially created in the **Project Browser**, but you can quickly render them on the Internal Block diagram for the Block.



SysML Block Definition

- Block
- Actor
- Interface Block
- Constraint Block
- Value Type
- Enumeration
- Interface
- Signal
- Unit
- Quantity Kind
- Property
- Flow Property
- Directed Feature
- Port
- Proxy Port
- Full Port

SysML Block Relationships

- Item Flow
- Dependency
- Generalization
- Containment
- Part Association
- Reference Association
- Shared Association
- Association Block
- Allocate

SysML Patterns

- Composite Block
- Block with Properties

SysML Deprecated

- Flow Port
- Flow Specification

Access

On the **Diagram Toolbox**, click on the More tools button and select 'SysML 1.3 | SysML 1.3 Block Definition'.

Ribbon	Design > Diagram > Toolbox
Menu	Diagram | Toolbox
Keyboard Shortcuts	**Alt+5**

SysML Block Definition Objects

Item	Action
Block	Defines a composite system entity in SysML.
Actor	Represents a user that interacts with one or more SysML systems.
Interface Block	A specialized kind of block that has no behaviors or internal parts, which is used to type Proxy Ports.
Constraint Block	Defines a composite constraint as a system of parametric equations.
ValueType	Defines a SysML quantity, expressed as a measurable dimension with specific units.
Enumeration	Defines a data type as a set of symbols or values.
Interface	Defines an element that describes a specification of an interaction point with properties and methods.
Signal	Defines a SysML message, containing attributes, exchanged between system blocks in an interaction.
Unit	Represents a standard unit of measure in SysML.
QuantityKind	Identifies a measurable quantity in SysML.
Property	Describes the decomposition of a **SysML Block** in the context of its whole, using instances of reusable SysML Blocks.
Flow Property	Creates a single kind of Flow element that can flow to or from a block.
Directed Feature	Generates an operation if dropped on a Block, or a DirectedFeature (Property) element if dropped on the diagram, both of which indicate that the associated Block either owns (provided) the feature, uses (required) the feature owned by another Block, or both (providedrequired). The direction value is defined in the 'SysML1.n' page of the element 'Properties' dialog, or the '**Tagged Values**' page of the operation 'Properties' dialog.
Port	Describes a structural interaction point of a **SysML Block**, which in turn connects

	interacting parts of a block.
Proxy Port	Exposes features of the owning Block or its internal parts.
Full Port	Specifies an element of the system separate from the owning Block or its internal parts.

SysML Block Definition Relationships

Item	Action
Item Flow	Specifies the items that flow across a connector in an interaction point. Used in the same way as UML Information Flows. See Using Information Flows
Dependency	Establishes a traceable relationship describing how one element is dependant upon another.
Generalization	Describes an element as a specialized descendant of another element, containing additional properties and behavior.
Containment	Graphically displays ownership of one element within a parent element.
Part Association	Describes the characteristics of a connection between a **SysML Block** and its internal parts, such as the multiplicity and type.
Reference Association	Describes the characteristics of a connection between separate SysML Blocks, such as the multiplicity and type.
Shared Association	Describes the characteristics of a common connection between SysML Blocks, such as the multiplicity and type.
Allocate	Connects model elements to formalize a refinement of behavior, structure, constraints or design expectations.

SysML Patterns

Item	Action
Composite Block	A pattern that creates a Composite Block, made up of Blocks related by Aggregation relationships.
Block with Parts	A pattern that creates a Composite Block made up of nested parts.

SysML Deprecated

Item	Action
Flow Port	Describes what flows in and out of interacting SysML Blocks. See Show Direction on SysML 1.3 Ports
Flow Specification	Defines a set of flow properties that correspond to individual pieces of a common interaction point. See Show Direction on SysML 1.3 Ports

Generate Properties From Block Associations

On a **SysML Block** diagram, an Association between two Block elements has ends that are represented by properties. If an Association end is navigable, the property that represents it is owned by the Block element at the other end of the Association.

In Enterprise Architect you can automatically create Part elements to represent these owned properties, in one of two ways:

- Create a directed Association ('Tools | Options | Links > select 'Association default = source --> target' checkbox'), which generates an anonymous Part on the source Block element
- Open the source Block's internal block diagram, right-click on the diagram and select 'Synchronize Structural Elements'

In both cases, the Part property is considered to be bound to the Association End property – they represent the same property, so changing one updates the other, either automatically or at the next synchronization; that is, if you change the Association Source Role name, multiplicity or Aggregation setting, the Part name, multiplicity and isReference setting are updated; if you change the Part details, the Association properties are updated.

If you want to change the binding of a Property to an Association, or bind existing Associations and Properties that are not yet bound to each other, right-click on the Property in:

- The **Project Browser**, and select 'Add | Bind to Connector Role', or
- The **Internal Block Diagram** and select 'Advanced | Bind to Connector Role'

In each case the 'Choose Connector Role to Bind' dialog displays, listing the Associations issuing from the parent Block element. Select the Association to bind the Property to, and click on the **OK button**.

If you subsequently delete an Association that is bound to a property, when you save the diagram you are prompted to confirm whether to also delete the property or keep it, unbound to a connector. If the property element is locked, it cannot be deleted.

Notes

- You can create an Association connector on a **SysML Block** diagram by dragging the Reference Association icon from the Toolbox or selecting the Association option from the Quick Linker arrow drop-down list
- Associations have direction when it is explicitly selected on the connector Properties, when the connector has assigned Source and Target roles, and/or when the Association default = source --> target checkbox is selected on the 'Options' dialog

Block Element Compartments

On a **SysML Block** diagram, SysML 1.3 Block and Constraint Block elements can display any of a number of element compartments that list their child elements that are not already shown on the diagram. You can select to display or hide these compartments using the 'Feature Visibility' dialog (press **Ctrl+Shift+Y** on the element); however, if there are no child elements corresponding to a type of compartment, that compartment is not shown.

SysML 1.3 Block element compartments

Compartment Name	Child Element Types Listed
values	Parts that are typed by a «valueType» element.
participants	Parts with the «participantProperty» stereotype.
flow properties	Parts with the «flowProperty» stereotype.
references	Parts for which the isReference **Tagged Value** is set to true.
constraints	Parts with the «constraintProperty» stereotype.
properties	Parts without a stereotype.
«stereotype»	Parts with stereotypes other than those identified above.
flow ports	Ports with the «flowPort» stereotype.
full ports	Ports with the «fullPort» stereotype.
proxy ports	Ports with the «proxyPort» stereotype.
ports	Any other Ports not listed in the previous compartments.

SysML 1.3 Constraint Block element compartments

Compartment Name	Child Element Types Listed
constraint properties	Parts with the «constraintProperty» stereotype.
parameters	Any other types of Part.

Show Direction on SysML 1.3 Ports

In SysML 1.3 you can show Ports, Full Ports and Proxy Ports with arrows indicating the flow direction.

Create a new port with direction arrows

Step	Action
1	Create a Port, Full Port or Proxy Port from the Block Definition toolbox for SysML 1.3.
2	Right-click on the Port, select Advanced | Set Property Type. The Select <Item> Dialog opens.
3	Select a Block or Interface Block that owns one or more Flow Property elements. Click the **OK button**. The Port will automatically display arrows based upon the aggregated 'direction' **Tagged Values** of all the Flow Property elements owned by the Block or Interface Block that is set as the Property Type. For example, if the Block contains an 'in' Flow Property and an 'out' Flow Property, the Port typed with this Block will display an 'inout' arrow.

Nested Ports in SysML 1.3

This statement is derived from the SysML 1.3 Specification:

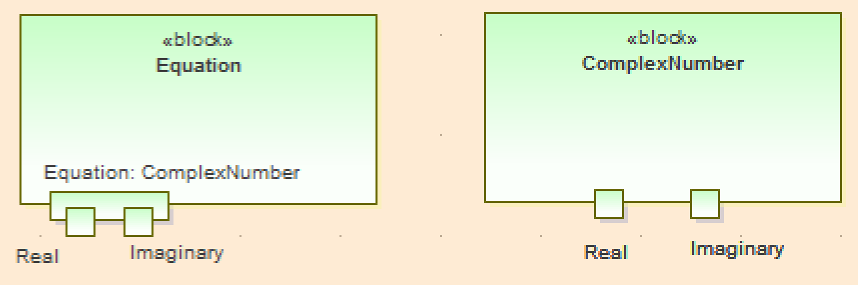
"Ports nest other ports in the same way that blocks nest other blocks. The type of the port is a block (or one of its specializations) that also has ports. For example, the ports supporting torque flows in the transmission example might have nested ports for physical links to the engine or the driveshaft."

In Enterprise Architect, Ports belonging to a specified type can be exposed on other Ports using that type as nested Ports, as defined in the specification. That is, the nested Port element is exposed by the type of the nesting Port element. This is simply achieved using the 'Structural Elements' dialog.

For example, a complex number is made up of two members (of type Real) - the real value and the imaginary value. This is represented by the Class ComplexNumber with two Ports, Real and Imaginary. An Equation Class has a Port that takes a ComplexNumber, but we might want to connect the real and imaginary portions of that variable to different sources. Therefore we have to show them, on the Equation Port, in order to connect them up.

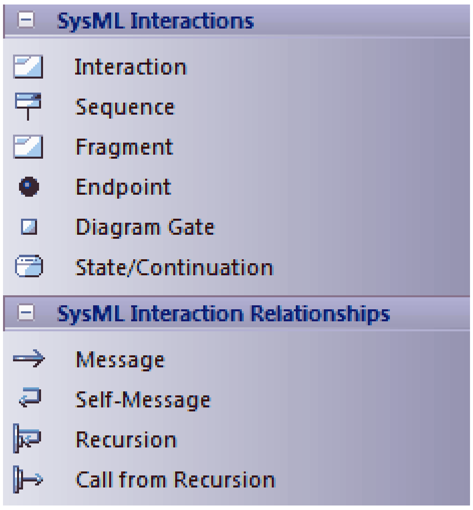
To create a representation of this example

1. Create a **SysML Block** named 'Equation' with a Port named 'Equation'.
2. Create a SysML Block named 'ComplexNumber' with Ports named 'Real' and 'Imaginary'.
3. Double-click on Port 'Equation', select the 'Property' page and set the 'Type' field to 'ComplexNumber'.
4. Right-click on Port 'Equation' and select the 'Structural Elements' option. The 'Structural Elements' dialog displays.
5. Select the 'Show Owned/Inherited' checkbox. Ports 'Real' and 'Imaginary' are shown in the list panel.
6. Select the checkboxes against 'Real' and 'Imaginary', and click on the **Close button**. The two Ports are now nested in Port 'Equation'.



SysML Interaction Toolbox

When you are constructing SysML models, you can populate the Interaction and Sequence diagrams using the icons on the SysML Interaction pages of the **Diagram Toolbox**.



SysML Interactions

- Interaction
- Sequence
- Fragment
- Endpoint
- Diagram Gate
- State/Continuation

SysML Interaction Relationships

- Message
- Self-Message
- Recursion
- Call from Recursion

Access

On the **Diagram Toolbox**, click on the More tools button and select 'SysML 1.3 | SysML1.3 Interaction'.

Ribbon	Design > Diagram > Toolbox
Menu	Diagram | Toolbox
Keyboard Shortcuts	**Alt+5**

SysML Interaction Objects

Item	Action
Interaction	Defines a **SysML Block** of executable behavior as a UML Interaction.
Sequence	References an instance of a **SysML Block** as a Lifeline in the Interaction.
Fragment	Declares a portion of an interaction as a group with specific behavior semantics.
Endpoint	Creates an exit point for the Interaction.
Diagram Gate	Creates an endpoint for the interaction, which bridges between nested interactions.
State/Continuation	Constrains the Interaction with assertions of the state that the lifeline is expected to be in.

SysML Interaction Relationships

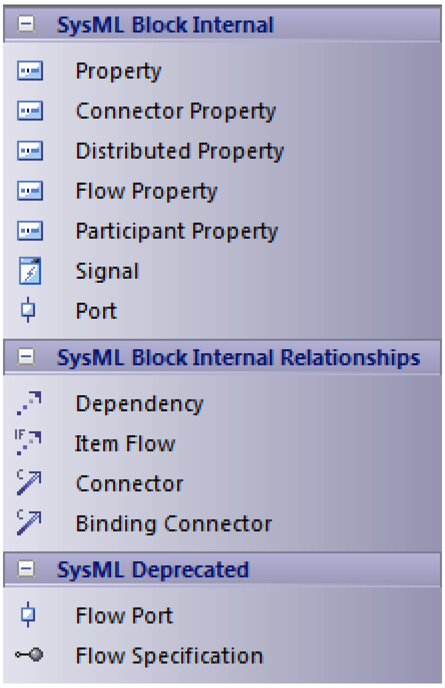
Item	Action
Message	Describes a message exchange between two lifelines in an Interaction.
Self-Message	Describes a message exchange between a lifeline and itself in an Interaction.
Recursion	Describes a recursive message exchange between a lifeline and itself in an Interaction.
Call from Recursion	Describes a message exchange between two lifelines within a recursive exchange.

SysML Internal Block Toolbox

When you are constructing SysML models, you can populate the Internal Block diagrams using the icons on the '**SysML Block** Internal' pages of the **Diagram Toolbox**.

The Block that owns the Internal Block diagram is automatically represented by a diagram frame enclosing the Internal Block diagram elements. You can:

- Hide the frame by right-clicking on the diagram and selecting the 'Hide Diagram Frame' option (and show it again by selecting the 'Show Diagram Frame' option); if the diagram frame is shown, the 'Diagram Frame' settings on the 'Tools | Options | Diagram' page will be ignored
- Make the frame selectable to move or resize it, by right-clicking on it and selecting the 'Selectable' option
- Create Ports and Parts on the frame and create connectors between them
- Generate Property elements on the diagram inside the Block frame, based on the Associations that the Block element has on the Block diagram



Access

On the **Diagram Toolbox**, click on the More tools button and select 'SysML 1.3 | SysML 1.3 Internal Block'.

Ribbon	Design > Diagram > Toolbox
Menu	Diagram | Toolbox
Keyboard Shortcuts	**Alt+5**

SysML Block Internal

Item	Description
Property	Describes the decomposition of a **SysML Block** in the context of its whole using instances of reusable SysML Blocks.
Connector Property	Creates an instance of an Association Block.
Distributed Property	Creates a stereotype of Property, to apply a probability distribution to the values of the property.
Flow Property	Creates a single kind of Flow element that can flow to or from a block.
Participant Property	Creates the end of a connector owned by an Association Block.
Signal	Defines a SysML message, containing attributes, exchanged between system Blocks in an interaction.
Port	Describes a structural interaction point of a **SysML Block** which, in turn, connects between interacting parts of a block.

SysML Block Internal Relationships

Item	Description
Dependency	Establishes a traceable relationship describing how one element is dependant upon another.
Item Flow	Specifies the items that flow across a connector in an interaction point. Used in the same way as UML Information Flows.
Connector	Establishes Communication links between parts.
Binding Connector	Establishes a connection between two parts in a system decomposition.

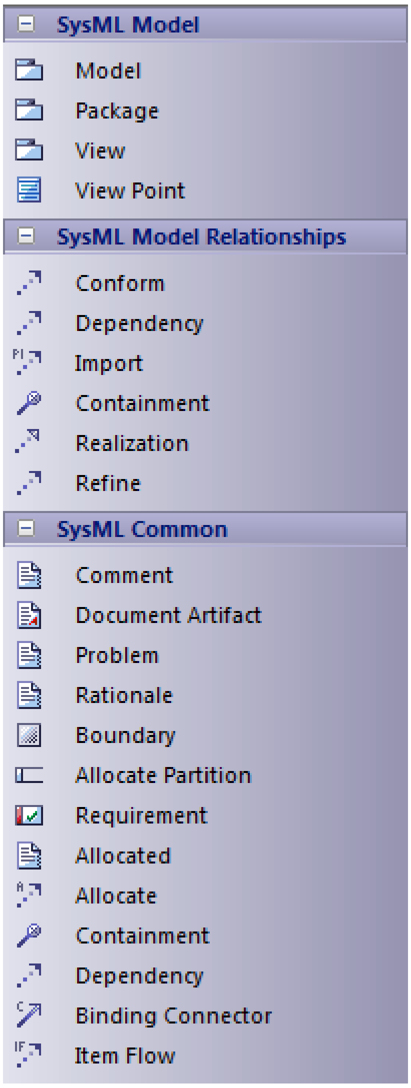
SysML Deprecated

Item	Description
Flow Port	Describes what flows in and out of interacting SysML Blocks.
Flow Specification	Defines a set of flow properties that correspond to individual pieces of a common interaction point.

SysML Model Elements Toolbox

When you are constructing SysML models, you can populate the diagrams with Model, Package and View elements using the icons on the SysML Model pages of the **Diagram Toolbox**.

The SysML Model toolbox includes a page of SysML Common element and relationship icons. You can add this page to all Diagram Toolboxes so that it is always available regardless of what type of diagram you are using; to do this, set the SysML 1.4 Technology to 'Active' ('Configure > Technology > Manage : SysML 1.4: Set Active').



Access

On the **Diagram Toolbox**, click on the More tools button and select 'SysML 1.4 | SysML 1.4 Model Elements'.

Ribbon	Design > Diagram > Toolbox
Menu	Diagram | Toolbox
Keyboard Shortcuts	**Alt+5**

SysML Model Element Objects

Item	Description
Model	Creates a Package containing a SysML Model.
Package	Groups model constructs in a single unit of containment.
View	Creates a stereotyped Package that defines a **SysML View** of a system, from the perspective of a **SysML View Point**.
View Point	Creates a stereotyped Class that defines a **SysML View Point**, which specifies the rules and conventions for the construction and use of Views.

SysML Model Relationships

Item	Description
Conform	Establishes a conformance dependency of a View to the defining View Point.
Dependency	Establishes a traceable relationship describing how one element is dependant upon another.
Import	Represents a reuse of elements from one model Package in another.
Containment	Graphically displays ownership of one element within a parent one.
Realization	Identifies a design fulfillment of a specification between elements.
Refine	Represents a refinement of one element by another.

SysML Common

Item	Description
Comment	Creates a textual annotation that can be attached to a set of elements of any other type. The attachment is created separately, using a Notelink connector.
Document Artifact	Attaches a linked document to the diagram by associating this element with the document.
Problem	A stereotyped Comment that documents the failure of model elements to satisfy a

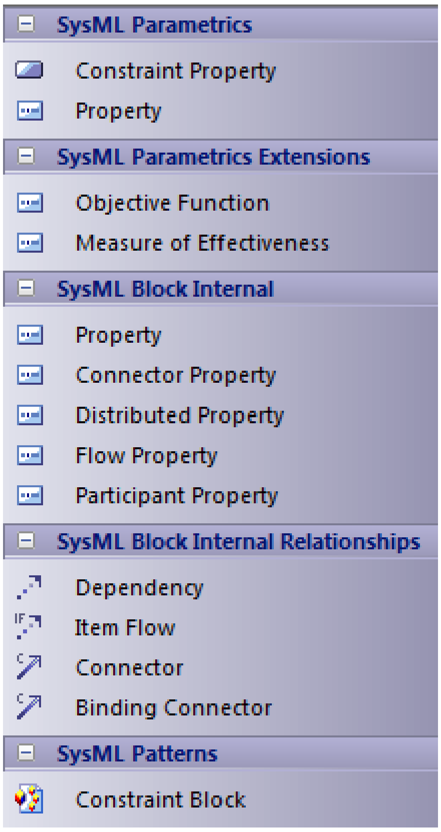
	requirement.
Rationale	A stereotyped Comment that documents the justification for decisions.
Boundary	Defines a conceptual boundary, to visually group logically related elements.
Allocate Partition	A stereotyped Activity Partition that contains elements deemed to be allocated to the classifier of the partition.
Requirement	Specifies the capabilities of the system, or the conditions that it should satisfy.
Allocated	A stereotyped Comment that defines the source element being allocated to the target element in an Allocate relationship.
Allocate	A stereotyped Abstraction that relates model elements to formalize a refinement of behavior, structure, constraints or design expectations. The Allocate relationship points from the element being allocated to the element that is the target of the allocation. The system provides an 'Allocations' search that lists all Allocate abstractions in tabular format, showing the 'To' and 'From' elements. Select 'Edit | **Search in Model**', then select 'SysML 1.4' as the Search Category; 'Allocations' defaults as the Search Type. If a **SysML Block** element has any Allocate relationships to or from elements that are not visible on the same diagram, those elements can be listed in 'AllocatedTo' and 'AllocatedFrom' compartments of the Block element on the diagram. Press **Ctrl+Shift+Y** to display the 'Feature and Compartment Visibility' dialog and select the 'Allocatedto' and/or 'Allocatedfrom' checkboxes. Alternatively, you can list the hidden elements on a linked Note. Create a Note element on the diagram and link it to the visible Block element with a Notelink connector. Right-click on the connector, select the 'Link this Note to an Element feature' option and, on the 'Link note to element feature' dialog, click on the drop-down arrow on the 'Feature Type' field and select 'AllocatedTo' or 'AllocatedFrom'. Click on the **OK button**; the names of the linked elements are now displayed in the Notes element. (To show both 'To' and 'From' elements, create a separate Note for each type.) For either compartments or Notes, you should save the diagram just before setting up the facility, and possibly reload the diagram to activate the facility. If there are no Allocate relationships, or the related elements are on the diagram, the options are not available.
Containment	Graphically displays ownership of one element within a parent one.
Dependency	Establishes a traceable relationship describing how one element is dependant upon another.
Binding Connector	A stereotyped Connector that establishes a connection between two parts in a system decomposition.
Item Flow	A stereotyped Information Flow that specifies the items that flow across a connector in an interaction point.

SysML Parametrics Toolbox

When you are constructing SysML models, you can populate the SysML Parametric diagrams with constraint blocks, using the icons on the 'SysML Parametrics' pages of the **Diagram Toolbox**.

The Block that owns the Parametric diagram is automatically represented by a diagram frame enclosing the Parametric diagram elements. You can:

- Hide the frame by right-clicking on the diagram and selecting the 'Hide Diagram Frame' option (and show it again by selecting the 'Show Diagram Frame' option)
- Make the frame selectable to move or resize it, by right-clicking on it and selecting the 'Selectable' option
- Create Ports and Parts on the frame and create connectors between them



Access

On the **Diagram Toolbox**, click on the More tools button and select 'SysML 1.3 | SysML 1.3 Parametrics'.

Ribbon	Design > Diagram > Toolbox
Menu	Diagram | Toolbox
Keyboard Shortcuts	**Alt+5**

SysML Parametrics Objects

Item	Description
Constraint Property	Instantiates a Constraint Block for use in a Parametric diagram.
Property	Defines a SysML property typed by a DataType, ValueType or Block.

SysML Parametrics Extensions

Item	Description
Objective Function	Defines a **SysML Constraint Block** for use as an objective function to evaluate Measures of Effectiveness (MOEs).
Measure of Effectiveness	Defines a SysML property for use as a Measure of Effectiveness (MOE).

SysML Block Internal

Item	Description
Property	Describes the decomposition of a **SysML Block** in the context of its whole using instances of reusable SysML Blocks.
Connector Property	Creates an instance of an Association Block.
Distributed Property	Creates a stereotype of Property, to apply a probability distribution to the values of the property.
Flow Property	Creates a single kind of Flow element that can flow to or from a block.
Participant Property	Creates the end of a connector owned by an Association Block.

SysML Block Internal Relationships

Item	Description
Dependency	Establishes a traceable relationship describing how one element is dependant upon another.
Item Flow	Specifies the items that flow across a connector in an interaction point. Used in the same way as UML Information Flows.

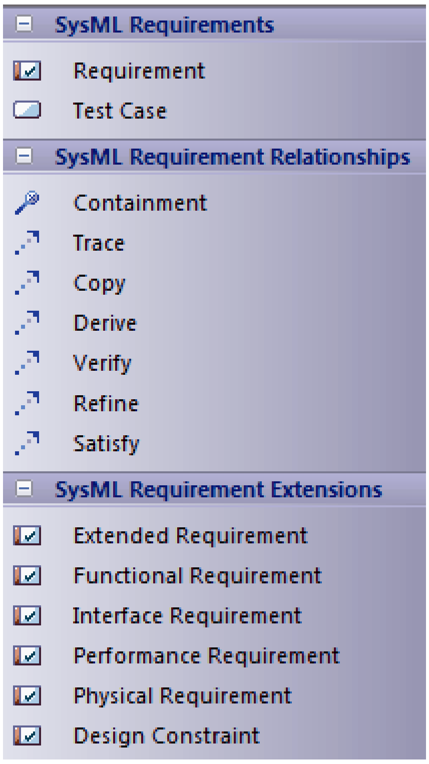
Connector	Establishes Communication links between parts.
Binding Connector	Establishes a connection between two Parts in a system decomposition.

SysML Patterns

Item	Description
Constraint Block	A pattern that creates a typical Constraint property that owns two Parts.

SysML Requirements Toolbox

When you are constructing SysML models, you can populate the Requirements diagrams using the icons on the SysML Requirements pages of the **Diagram Toolbox**.



SysML Requirements

- Requirement
- Test Case

SysML Requirement Relationships

- Containment
- Trace
- Copy
- Derive
- Verify
- Refine
- Satisfy

SysML Requirement Extensions

- Extended Requirement
- Functional Requirement
- Interface Requirement
- Performance Requirement
- Physical Requirement
- Design Constraint

Access

On the **Diagram Toolbox**, click on the More tools button and select 'SysML 1.3 | SysML 1.3 Requirements'.

Ribbon	Design > Diagram > Toolbox
Menu	Diagram | Toolbox
Keyboard Shortcuts	**Alt+5**

SysML Requirement Objects

Page	Item
Requirement	Specifies the capabilities of the system, or the conditions that it should satisfy.
Test Case	Describes the verification of a Requirement through methods of inspection, analysis, demonstration or testing.

SysML Requirement Relationships

Item	Description
Containment	Graphically displays ownership of one element within a parent element.
Trace	Declares a trace relationship between a SysML Requirement and another SysML element.
Copy	Declares a copy of one SysML Requirement by another.
Derive	Derives a SysML Requirement from another.
Verify	Declares a verification of a SysML Requirement by another SysML element.
Refine	Declares a refinement of a SysML Requirement by another SysML element.
Satisfy	Declares that the SysML Requirement is satisfied by another SysML element.

SysML Requirement Extensions

Item	Description
Extended Requirement	Extends a SysML Requirement with additional Tag properties.
Functional Requirement	Declares a SysML Requirement that describes the operation, or behavior, that the system must perform.
Interface Requirement	Declares a SysML Requirement that describes how the system connects, or interfaces with, other systems.
Performance Requirement	Declares a SysML Requirement that describes how the system performs against defined capabilities or conditions.
Physical Requirement	Declares a SysML Requirement that describes the physical characteristics, or physical constraints, of the system.
Design Requirement	Declares a SysML Requirement that specifies a constraint on the implementation of the system.

Notes

- SysML Requirements contain the **Tagged Values** 'Text' and 'ID', the values of which are not immediately visible in the Tagged Values window; you can see the values more easily if you have the **Summary View** window open (View

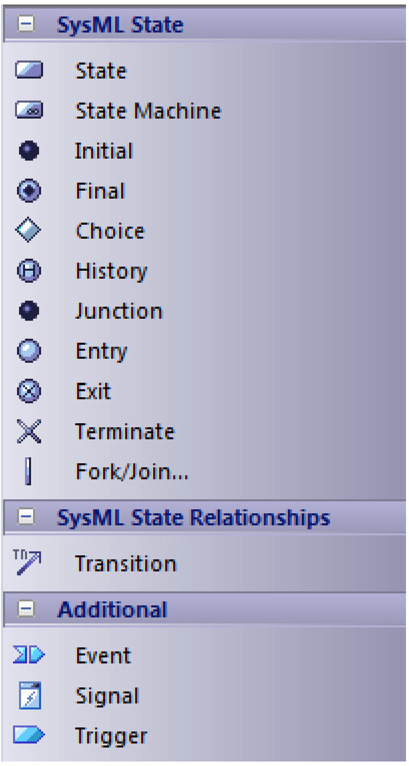
| Summary View) when you click on these elements

SysML State Machine Toolbox

When you are constructing SysML models, you can populate the State Machine diagrams using the icons on the 'SysML State Machine' pages of the **Diagram Toolbox**.

The Block that owns the State Machine diagram is automatically represented by a diagram frame enclosing the State Machine diagram elements. You can:

- Hide the frame by right-clicking on the diagram and selecting the 'Hide Diagram Frame' option (and show it again by selecting the 'Show Diagram Frame' option)
- Make the frame selectable to move or resize it, by right-clicking on it and selecting the 'Selectable' option
- Create Ports and Parts on the frame and create connectors between them



Access

On the **Diagram Toolbox**, click on the More tools button and select 'SysML 1.3 | SysML 1.3 State Machine'.

Ribbon	Design > Diagram > Toolbox
Menu	Diagram | Toolbox
Keyboard Shortcuts	**Alt+5**

SysML State Machine Objects

Item	Description
State	Declares a significant condition in the life of a **SysML Block** within its State Machine.
State Machine	Describes the life-cycle behavior of a **SysML Block** in terms of its states and transitions.
Initial	Declares the starting state of the State Machine.
Final	Declares the ending state of the State Machine, and its completion.
Choice	Declares a Junction with a mandatory 'else' transition.
History	Represents the last active State of the State Machine prior to its interruption.
Junction	Declares a decision point at which a Transition branches out into multiple guarded, alternative paths.
Entry	Declares an Entry point between State Machines, Substate Machines and Regions.
Exit	Declares an Exit point between State Machines, Substate Machines and Regions.
Terminate	Declares a termination State in which the State Machine no longer operates.
Fork/Join	Simultaneously branches and joins a set of Transitions.

SysML State Relationships

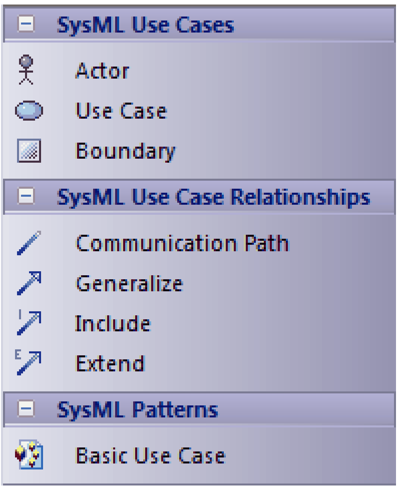
Item	Description
Transition	Establishes a life-cycle path between one State and another, based on its operational conditions.

Additional

Item	Description
Event	Depicts the action of sending a signal.
Signal	A specification of Send request instances communicated between objects.
Trigger	Indicates an event that initiates an action (and might arise from completion of a previous action).

SysML Use Case Toolbox

When you are constructing SysML models, you can populate the Use Case diagrams using the icons on the 'SysML Use Cases' pages of the **Diagram Toolbox**.



Access

On the **Diagram Toolbox**, click on the More tools button and select 'SysML 1.3 | SysML 1.3 Use Cases'.

Ribbon	Design > Diagram > Toolbox
Menu	Diagram | Toolbox
Keyboard Shortcuts	**Alt** + 5

SysML Use Case Objects

Item	Description
Actor	Represents a user that interacts with one or more SysML systems.
Use Case	Describes the expected functionality of a system as a **UML Use Case**.
Boundary	Graphically binds elements in a diagram within a border.

SysML Use Case Relationships

Item	Description

Communication Path	Declares which Actors perform in the Use Case.
Generalize	Describes an element as a specialized descendant of another element, containing additional properties and behavior.
Include	Describes one Use Case as a subset of another.
Extend	Describes one Use Case as an extension of another.

SysML Patterns

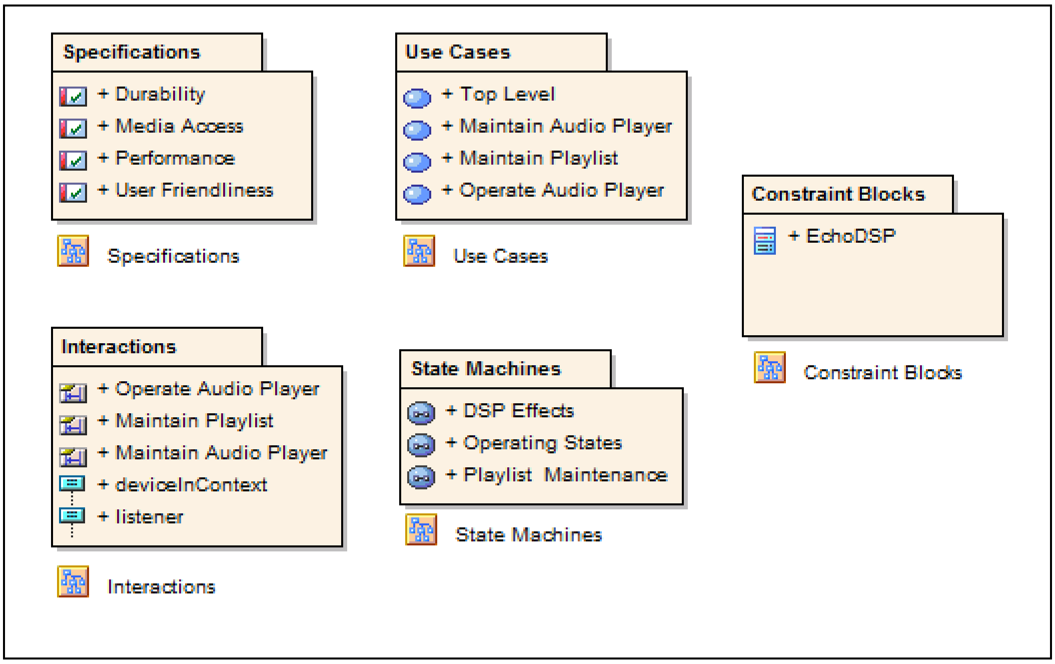
Item	Description
Basic Use Case	A pattern that creates a typical simple Use Case diagram of Actor, Use Case and System Boundary elements.

A SysML Requirements Model

Requirements engineering is a fundamental aspect of a systems engineering model. The discipline focuses on eliciting, analyzing and managing customer requirements early in the process. Once the requirements are understood, trade studies can be conducted to formally assess design options, typically using weighted choices. The requirements are managed as first class citizens and are formally allocated to development items and verification methods.

Enterprise Architect has extensive functionality to assist the modeler with every aspect of the requirements engineering discipline, including elicitation, modeling, management and testing. High quality engineering documentation can be generated out-of-the-box using a wide range of built-in templates, carefully crafted to extract the information in the models and present it in visually compelling and high quality documentation in a wide range of formats, including docx, pdf and html. The documentation engine is highly configurable, and documentation to match any engineering or organizational standard can be generated by creating templates and setting generation options.

The SysML Requirements Model provides the system requirements, the expected abstract behavior and the operating constraints that the designed system must conform to. This diagram shows an example requirements model for a Portable Audio Player.



In this example, each of the child Packages contains child models that capture aspects of the system's requirements:

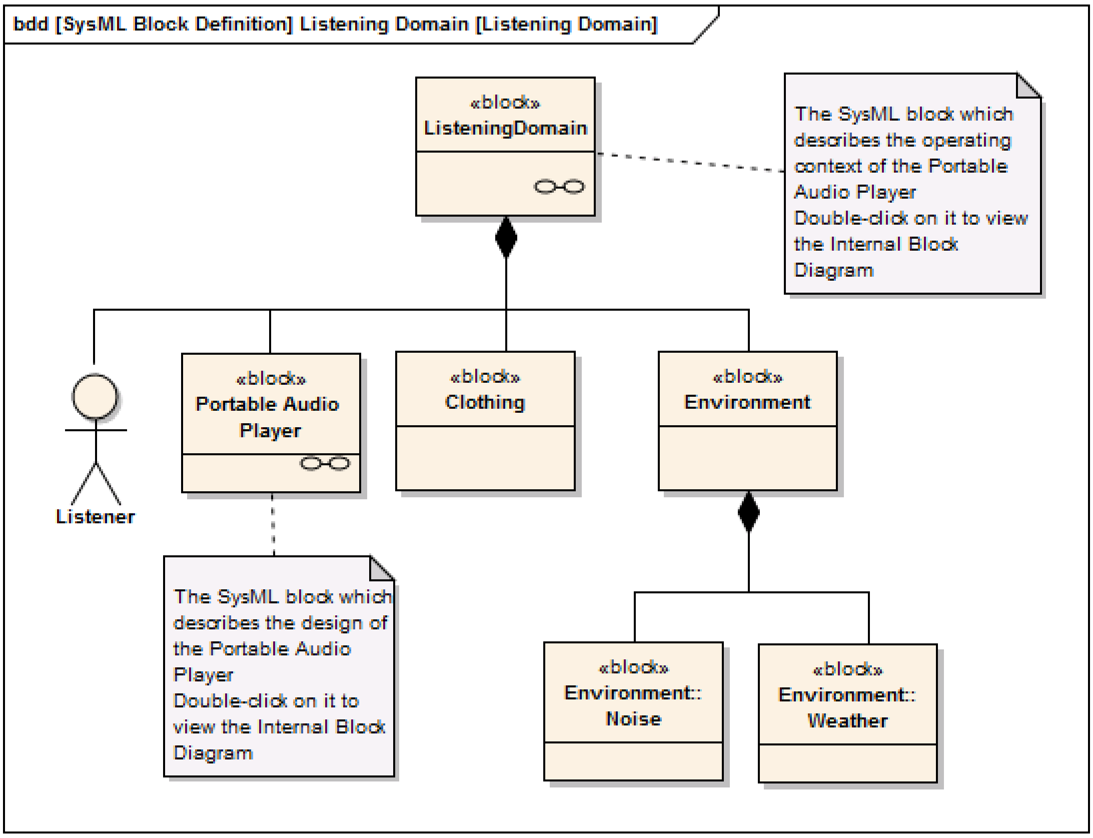
- The Specifications Package contains SysML Requirements describing the overall expectations of the designed system
- The Use Cases Package contains SysML Use Cases that describe the general interaction between the system and its users
- The Interactions Package contains SysML Interactions that describe a detailed sequence of interactions between the system and its users
- The State Machines Package contains SysML State Machines that describe each of the operational states the designed system has
- The Constraint Blocks Package contains SysML ConstraintBlocks that describe the expected performance and operating boundaries of the system

A SysML Operational Domain Model

Systems engineering is an interdisciplinary field of engineering that takes a whole-of-system view of a problem and its solution. The operational domain model is a central part of any model-based approach and describes the system in the context of its environment. This includes the humans that are intended to operate and interact with the system, external objects that might influence the system, and environmental elements that could impact the system. The operational domain model is a useful starting point to get an overview of a system and how it will operate.

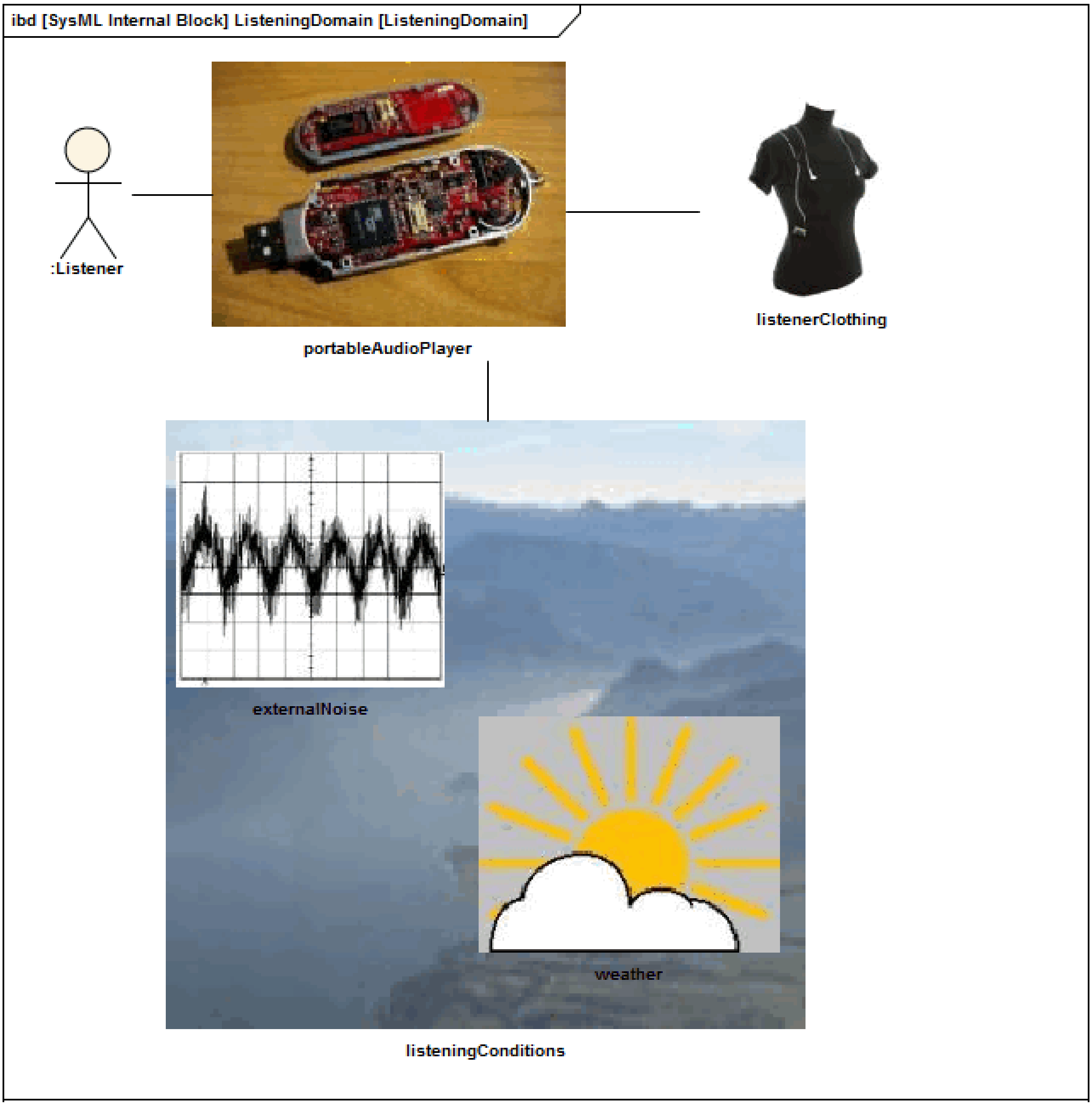
Enterprise Architect provides a range of features that help the engineer to construct an operational domain model, including standard SysML block definition and Internal Block diagrams, and the ability to include pictorial representations of elements that make the diagrams more compelling. The elements can also be hyper-linked, enabling the viewer to use a diagram as a launching pad to more detailed models and diagrams.

The SysML Operational Domain Model defines the system's operating environment, which describes the operating conditions that the system is intended to operate within. This diagram shows an example Operational Domain model for a Portable Audio Player; the **SysML Block** Definition Diagram describes the Operational Domain (in this example - the ListeningDomain) as a system composition.


bdd [SysML Block Definition] Listening Domain [Listening Domain]
«block»
ListeningDomain
The SysML block which describes the operating context of the Portable Audio Player
Double-click on it to view the Internal Block Diagram
Listener
«block»
Portable Audio Player
«block»
Clothing
«block»
Environment
The SysML block which describes the design of the Portable Audio Player
Double-click on it to view the Internal Block Diagram
«block»
Environment:: Noise
«block»
Environment:: Weather

In the example, the ListeningDomain is defined as a system containing other subsystems; the domain contains subsystems that define the Listener (i.e. User), the Portable Audio Player, Clothing (which the user wears), and the External Environment.

Details of the ListeningDomain system are further detailed in the ListeningDomain's **Internal Block Diagram**:



In the example, the ListeningDomain system's detailed composition shows how the Portable Audio Player and other sub-systems fit together to form the Listening Domain; it also describes the binding relationships between the parts, which define how the parts are functionally bound to one another.

Compose System Design

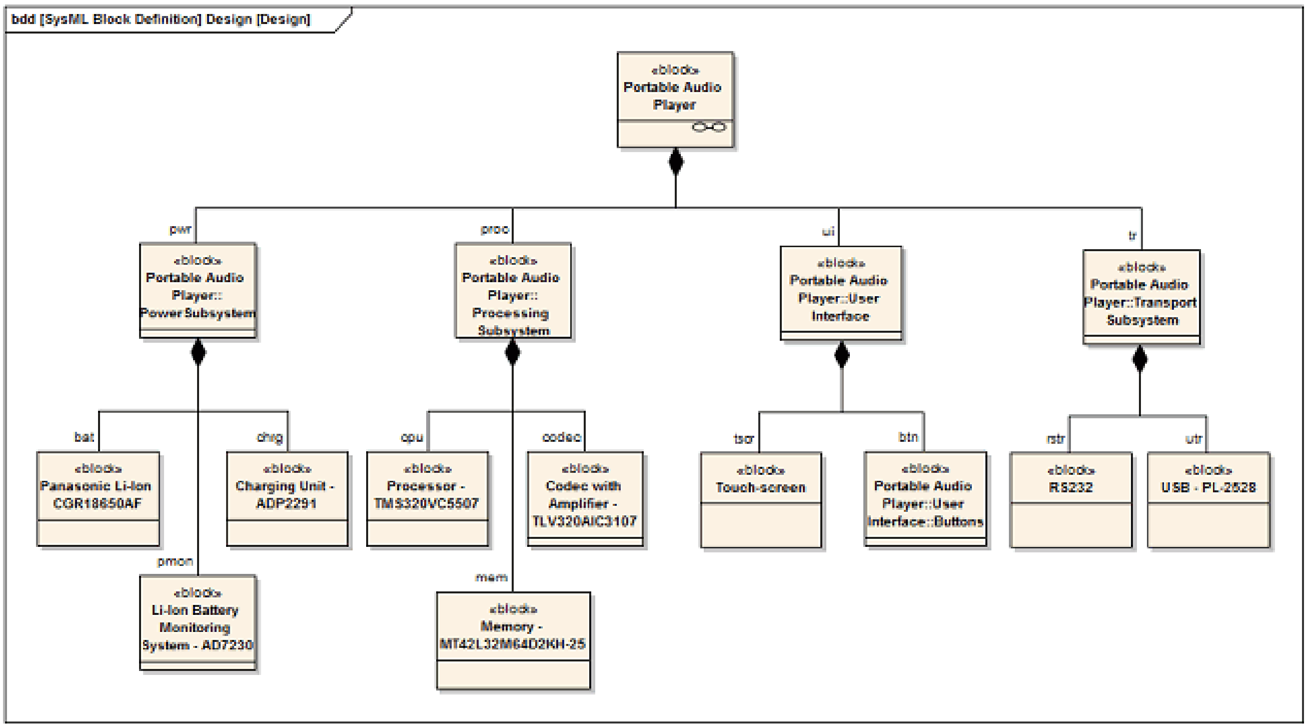
The systems engineering language has a strong focus on design. Once the requirements have been elicited, modeled and analyzed, the attention of the engineer turns to design. The systems are typically complex and must be broken down into a number of Subsystems that will interact with each other through known and published interfaces. The decomposition of the system is modeled using a block definition diagram displaying the subsystems, which in turn may be further decomposed. These subsystems can be traced back to the requirements and then to the stakeholders who own the requirements.

A Block Definition diagram can be created in Enterprise Architect to model the decomposition of a system into a hierarchy of subsystems using, the composition association. Subsystems can be hyper-linked to more detailed diagrams, which allow the viewer to click through from the system level through to all its constituent parts. The subsystems can also be linked back to the requirements they are implementing.

The SysML Design Model

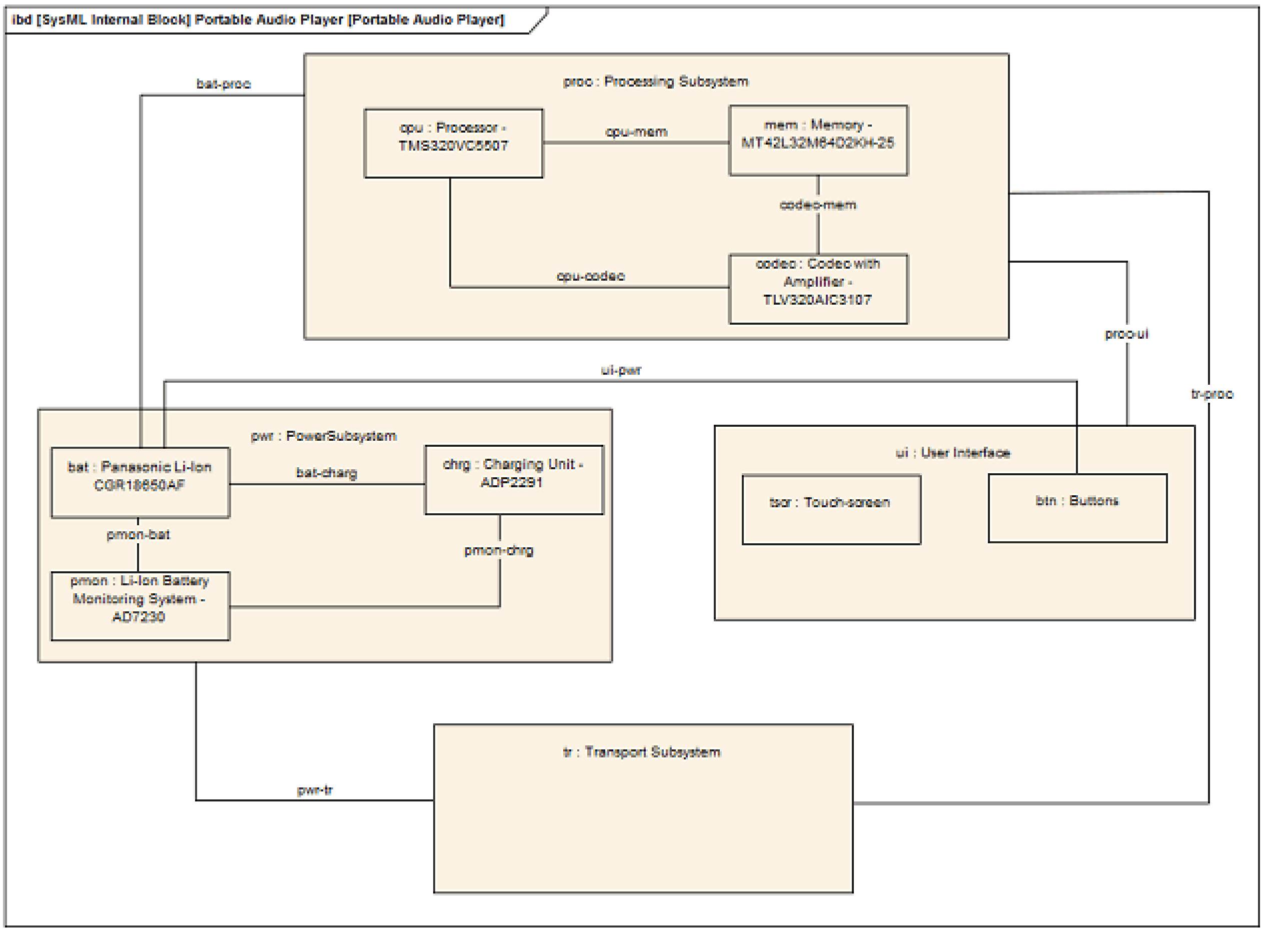
The SysML Design Model contains the blocks that define the system's composition; it describes the manner in which reusable subsystems fit together to fulfill the design requirements.

This diagram shows an example Design Model for a Portable Audio Player; the **SysML Block** Definition Diagram describes the Portable Audio Player as a composition of various reusable off-the-shelf subsystems and in-house designed ones.


bdd [SysML Block Definition] Design [Design]
«block» Portable Audio Player
pwr
«block» Portable Audio Player:: Power Subsystem
proc
«block» Portable Audio Player:: Processing Subsystem
ui
«block» Portable Audio Player::User Interface
tr
«block» Portable Audio Player::Transport Subsystem
bat
«block» Panasonic Li-Ion CGR18650AF
chrg
«block» Charging Unit - ADP2291
pmon
«block» Li-Ion Battery Monitoring System - AD7230
cpu
«block» Processor - TMS320VC5507
codec
«block» Codec with Amplifier - TLV320AIC3107
mem
«block» Memory - MT42L32M64D2KH-25
tscr
«block» Touch-screen
btn
«block» Portable Audio Player::User Interface::Buttons
rstr
«block» RS232
utr
«block» USB - PL-2528

In this example, the Portable Audio Player is defined as a SysML system containing subsystems that perform specific tasks; the design contains subsystems for supplying power, performing playback and audio processing, and interfacing with other devices and the user interface.

Details of the Portable Audio Player's composition are further described in detail within the Portable Audio Player's **Internal Block Diagram**:


ibd [SysML Internal Block] Portable Audio Player [Portable Audio Player]
bat-proc
proc : Processing Subsystem
cpu : Processor - TMS320VC5507
cpu-mem
mem : Memory - MT42L32M64D2KH-25
codec-mem
cpu-codec
codec : Codec with Amplifier - TLV320AIC3107
proc-ui
tr-proc
ui-pwr
pwr : PowerSubsystem
bat : Panasonic Li-Ion CGR18650AF
bat-charg
chrg : Charging Unit - ADP2291
pmon-bat
pmon-chrg
pmon : Li-Ion Battery Monitoring System - AD7230
ui : User Interface
tsor : Touch-screen
btn : Buttons
tr : Transport Subsystem
pwr-tr

The example describes the Portable Audio Player's composition, detailing how each of the sub-systems is structured. The example also describes the binding relationships between the parts, which define how they are functionally bound to one another; for example, the CPU, Memory and Codec are interfaced together in the Processing Subsystem.

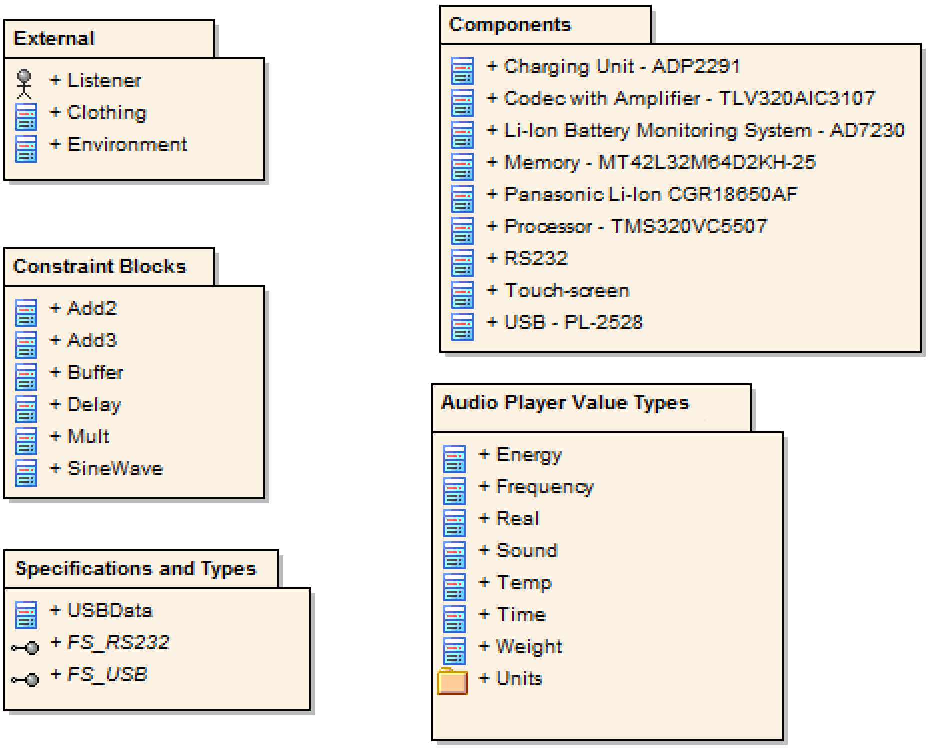
Create Reusable Subsystems

Model-based Systems Engineering provides the flexibility and expressiveness to define complex systems quickly and effectively, by reusing common entities across design projects. Before the model based approach became prevalent, systems were defined using document based methods with little opportunity for re-use. The SysML contains a series of reusable libraries such as the SI Definitions and the SI Value Types, but also supports a modeler in creating additional domain or technology-specific libraries that could be reused within an organization, or published for wider use by a community of users or an entire industry.

Enterprise Architect provides a range of functionality to assist in creating, discovering, visualizing and reusing libraries of elements such as subsystems, parametric constraints, common data types, common value types, dimensions and units. The **Reusable Asset Service** could be used to store these assets in order to provide a canonical set of libraries governed by the respective standards agencies.

SysML Design Model

A Library is a Package containing many reusable subsystems, parametric constraints, common data types and common value types, dimensions and units. This diagram shows an example library model:



In the example Library, each of the child Packages contains child models that capture these reusable entities:

- Blocks defining systems such as those listed in the Components Package, or those defined in the External Package
- ConstraintBlocks defining parametric constraints for use in parametric models
- Value Types describing quantities, expressed as measurable dimensions in specific units
- Data Types and Flow Specifications describing data structures and Flows

Migrate SysML Model to Later SysML Version

Enterprise Architect provides a useful feature to migrate a model from one version of SysML to the next. A model (or part of a model) created in an older version of the SysML MDG Technology can be migrated to the next version using the automation interface. This function updates the **Tagged Values** and, if required, stereotypes to the later version for all elements, attributes, connectors and diagrams under the selected Package or element.

Example Script for Migrating SysML 1.1 to SysML 1.2

This VB script calls the Migrate() function to migrate the SysML 1.1 Package or element to SysML 1.2:

```
Sub MigrateElement (sGUID, lngPackageID)
   Dim proj as EA.Project
   set proj = Repository.GetProjectInterface
   proj.Migrate sGUID, "SysML1.1", "SysML1.2"

   'refresh the model
   If lngPackageID<>0 Then
      Repository.RefreshModelView (lngPackageID)
   End If
End Sub

Sub MigrateSelectedItem
   Dim selType
   Dim selElement as EA.Element
   Dim selPackage as EA.Package

   selType = GetTreeSelectedItemType
   If selType = 4 Then 'means Element
      set selElement = GetTreeSelectedObject
      MigrateElement selElement.ElementGUID, selElement.PackageID
      MsgBox "Element Migration Completed",0,"SysML Migration"
   ElseIf selType = 5 Then 'means Package
      set selPackage = GetTreeSelectedObject
      MigrateElement selPackage.PackageGUID, selPackage.PackageID
      MsgBox "Package Migration Completed",0,"SysML Migration"
   Else
      MsgBox "Select a Package or Element in the Project Browser to initiate migration",0,"SysML Migration"
   End If
End Sub

Sub Main
```

```
    MigrateSelectedItem
End Sub

Main
```

Migrate SysML 1.1 to SysML 1.3

Migrate from SysML 1.1 to SysML 1.2, as explained earlier, then migrate from SysML 1.2 to SysML 1.3, as described in the next section.

Migrate SysML 1.2 to SysML 1.3

The MDG Technology for SysML 1.3 has a built-in script for migrating 1.2 models to 1.3. You must enable both the MDG Technology for SysML 1.2 and the MDG Technology for SysML 1.3. Firstly, you select the SysML 1.2 Package in the **Project Browser**, then you open the **Scripting window** and open the SysML 1.3 script group, and finally you execute the Migrate script.

Notes

- When migrating from SysML1.1 to SysML1.2, the:
    - Stereotype dimension is changed to quantitykind
    - Stereotype dataType is removed from SysML1.2
    - **Tagged Value** dimension is migrated to quantitykind; this applies to stereotypes unit and valueType
    - Tagged Value isConjugated of stereotype flowport is migrated to custom properties

Simple Parametric Simulation (Legacy)

This functionality has been superceded by the Parametric Simulation using OpenModelica.

The following topic is available only for those who have existing models using the previous functionality.

You use the scripting function of Enterprise Architect to simulate a SysML model from a Parametric diagram, using the 'Simulation Configuration' dialog. This is limited when compared to the OpenModelica based simulation because the behavior of each constraint block is represented by a script that calculates outputs from a known set of inputs instead of solving for the unknowns dynamically.

Access

Context Menu	On Parametric diagram | Right-click | SysML | Simulate Diagram...

Simulate a SysML model

Step	Action
1	The 'Parameters' panel lists all of the parameters that can be assigned input. Select each of the required parameters and click on the right **Arrow button** to assign them as input. Parameters designated as input parameters are listed in the 'Inputs' panel on the right. There must be at least one input parameter assigned for the simulation to execute.
2	Assign a set of values for each of the designated input parameters. For each input parameter, in the 'Input Values' panel select one of the two possible value kinds: • Discrete - To enter a constant or a comma-separated range of discrete values • Range - To enter a range of values beginning at the 'From' value and ending at the 'To' value; the input values are incremented by the 'Step' value
3	Specify the classes of output value: • 'Parameters' - To output the parameters' data, select the checkbox • 'Variables' - To output the data generated within each internal variable, select the checkbox; internal variables are automatically generated by the simulator
4	Specify how the simulation results are to be reported. The 'Output Format' panel enables you to choose how the simulation outputs the simulation data: • Plot To Graph: To plot the results on a 2-dimensional graph, select the checkbox; if you select this option, you must specify an input parameter for the plot's X Axis • Title - To enter a title for the graph, type in the title text • Output to File - To output the results to a CSV text file, select the checkbox and type or browse (click on ...) for the file name

5	Click on the **OK button** to execute the simulation.

Notes

- Systems Modeling Language (SysML) Parametric Model Simulation is available in the Systems Engineering and Ultimate editions of Enterprise Architect

Learning Center topics

- Start > Help > Learn | Enterprise Architect | System Engineering | Parametric Models | Getting Started
- Start > Help > Learn | Enterprise Architect | System Engineering | Parametric Models | Parametric Example Models