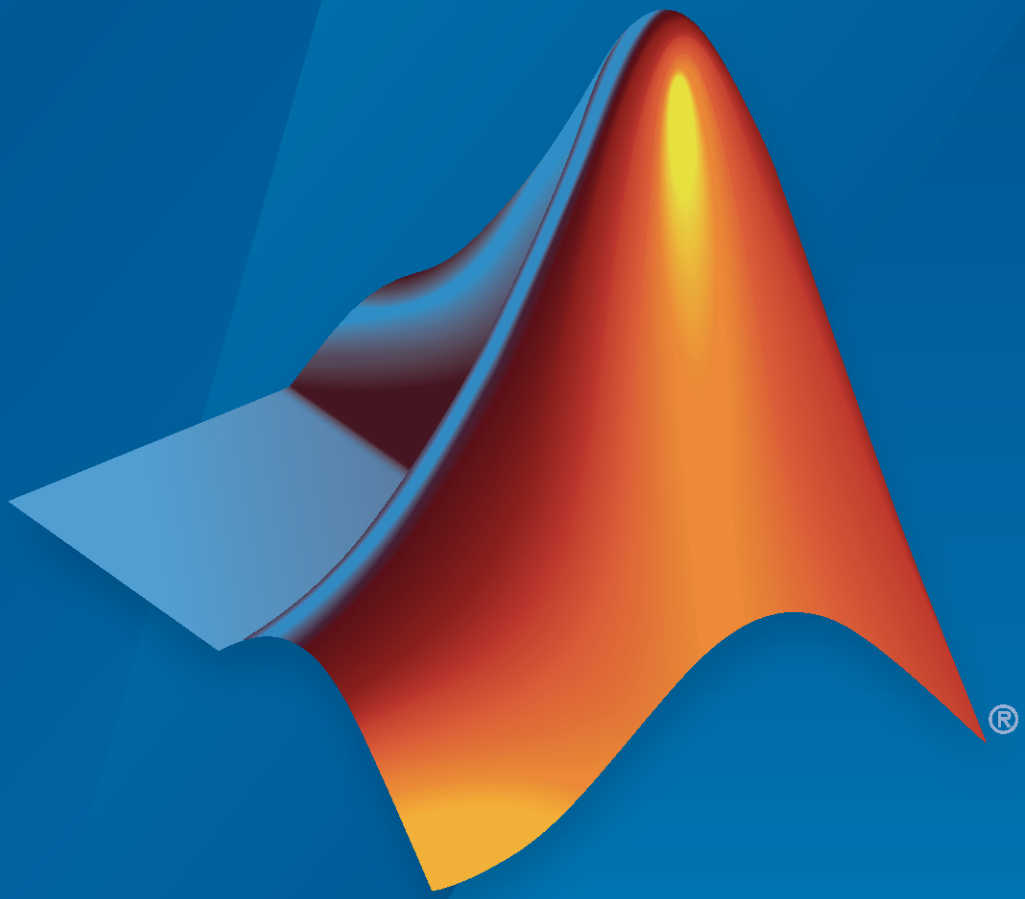
System Composer™ Release Notes



MATLAB® & SIMULINK®

MathWorks®

How to Contact MathWorks

Latest news: `www.mathworks.com`

Sales and services: `www.mathworks.com/sales_and_services`

User community: `www.mathworks.com/matlabcentral`

Technical support: `www.mathworks.com/support/contact_us`

Phone: 508-647-7000

The MathWorks, Inc.
1 Apple Hill Drive
Natick, MA 01760-2098

System Composer™ Release Notes

© COPYRIGHT 2019–2021 by The MathWorks, Inc.

The software described in this document is furnished under a license agreement. The software may be used or copied only under the terms of the license agreement. No part of this manual may be photocopied or reproduced in any form without prior written consent from The MathWorks, Inc.

FEDERAL ACQUISITION: This provision applies to all acquisitions of the Program and Documentation by, for, or through the federal government of the United States. By accepting delivery of the Program or Documentation, the government hereby agrees that this software or documentation qualifies as commercial computer software or commercial computer software documentation as such terms are used or defined in FAR 12.212, DFARS Part 227.72, and DFARS 252.227-7014. Accordingly, the terms and conditions of this Agreement and only those rights specified in this Agreement, shall pertain to and govern the use, modification, reproduction, release, performance, display, and disclosure of the Program and Documentation by the federal government (or other entity acquiring for or through the federal government) and shall supersede any conflicting contractual terms or conditions. If this License fails to meet the government's needs or is inconsistent in any respect with federal procurement law, the government agrees to return the Program and Documentation, unused, to The MathWorks, Inc.

Trademarks

MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See `www.mathworks.com/trademarks` for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.

Patents

MathWorks products are protected by one or more U.S. patents. Please see `www.mathworks.com/patents` for more information.

Contents

R2021a

Sequence Diagrams: Describe system behavior as a sequence of interactions between components 1-2

State Charts: Describe component behavior using Stateflow Charts to represent modes of operation 1-2

Software Architecture: Simulate and deploy software functions from System Composer 1-3

Updates to the Architecture Views Gallery and APIs 1-3

Performance and Scalability Improvements 1-5

Interface Editor enhancements 1-5

R2020b

Model to Model Allocations: Establish traceable and directed relationships between architecture elements in source and target models 2-2

Enhanced workflows with Architecture Views 2-2

Hierarchy Views: Display component hierarchy as a tree diagram 2-3

Referenced Data Dictionaries: Organize interfaces in a hierarchy of data dictionaries 2-3

Interface Editor enhancements 2-3

Support for component behavior using Simulink models with message input and output 2-4

Stereotype-Based Styling: Style connectors based on stereotypes and use custom icons for component stereotypes 2-5

Support for protected models as component behaviors 2-6

Import and export requirement links along with the architectural models **2-6**

R2020a

Stereotype-Based Styling: Associate a color with component stereotypes **3-2**

Interface Stereotypes: Apply stereotypes for custom metadata on interface objects **3-2**

Quick Insert Stereotypes: Insert stereotyped components directly by typing on the canvas **3-3**

Model Templates for Referenced Models: Select from preconfigured templates when creating new referenced architecture or behavior models **3-4**

Requirements on Component Ports: Link requirements to component ports **3-4**

Partial Architecture Model Load: Improved loading for large models . . . **3-5**

R2019b

Architecture Views **4-2**

Interface Adapter **4-2**

Import and Export Architecture Models **4-2**

AUTOSAR Software Architecture **4-2**

R2019a

Introducing System Composer **5-2**

Composition Editor **5-2**

Spotlight Views **5-2**

Linking Simulink Behavior Models **5-2**

Linking and Managing Requirements **5-2**

Stereotypes and Profiles **5-2**

Architecture Model Analysis **5-3**

R2021a

Version: 2.0

New Features

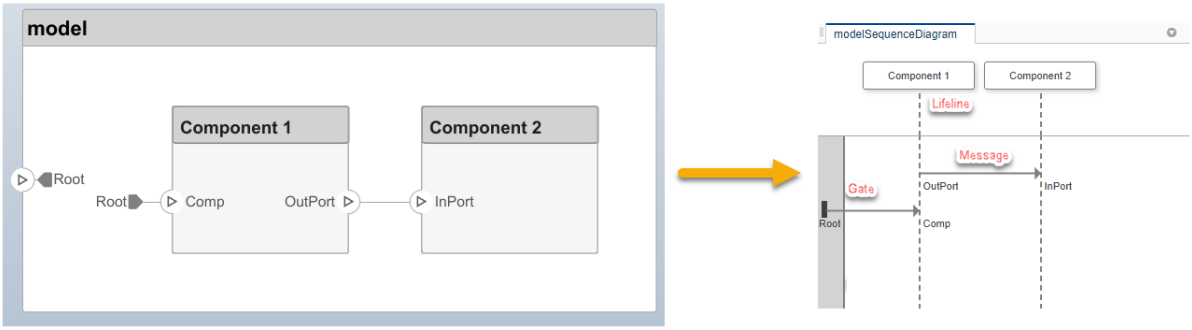
Bug Fixes

Compatibility Considerations

Sequence Diagrams: Describe system behavior as a sequence of interactions between components

You can use Sequence Diagrams in System Composer to describe the expected system behavior as a sequence of interactions between components of a system. You can create multiple sequence diagrams to represent different operational scenarios of the system. New lifelines or interactions authored on the sequence diagram are automatically reflected as new components or connections in the architecture model.

Sequence Diagrams are available in the Architecture Views Gallery and support these features:



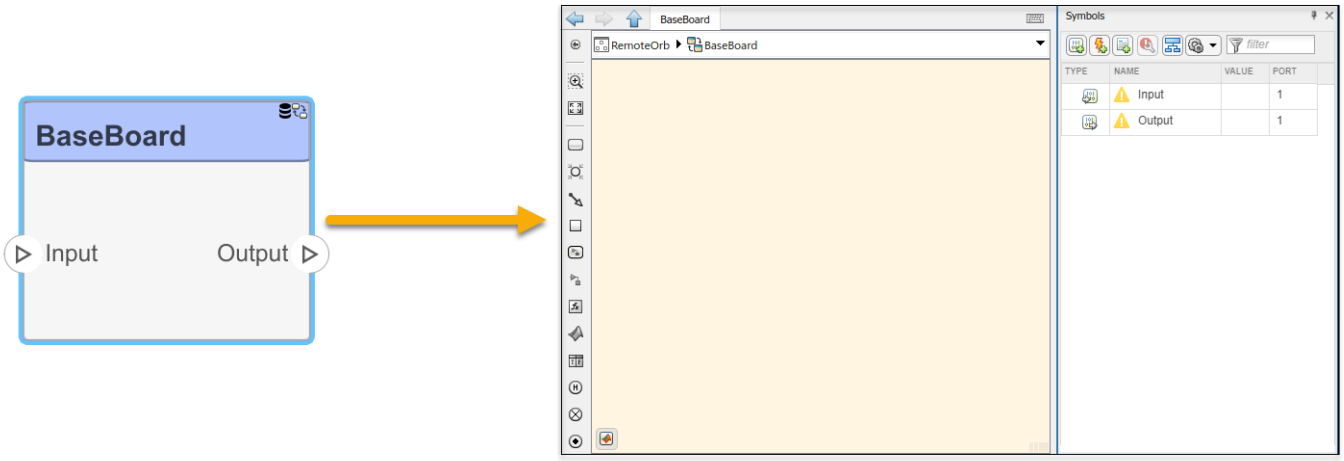
- Add lifelines to represent components and show what the component does in terms of events received and events sent out.
- Create child lifelines for child components.
- Add messages between lifelines that correspond to a connection between ports to show how components interact. These messages can have triggers and constraints attached to them.
- Create messages from root architecture ports to any lifeline component.
- Use the co-creation workflow. The software generates components and connections as you author new lifelines and messages in the Sequence Diagram.
- Make a marquee selection of messages and create a fragment. Click the **Add Operand** button to add operands for more complicated semantics like if-else branching or looping.
- When a mismatch occurs from the architecture model the sequence diagram references, click the **Check Consistency** button to highlight lifelines pointing to missing components for manual reconciliation.

For more information, see “Define Sequence Diagrams”.

State Charts: Describe component behavior using Stateflow Charts to represent modes of operation

In R2021a, you can add Stateflow® Chart behavior to a component in a System Composer architecture model.

- Add Stateflow Chart behavior to a component.



- The new Stateflow Chart behavior for a component is embedded within the same `.slx` file as the parent architecture model, reducing the need for multiple model files.

For more information, see “Add Stateflow Chart Behavior to Architecture Component”.

Software Architecture: Simulate and deploy software functions from System Composer

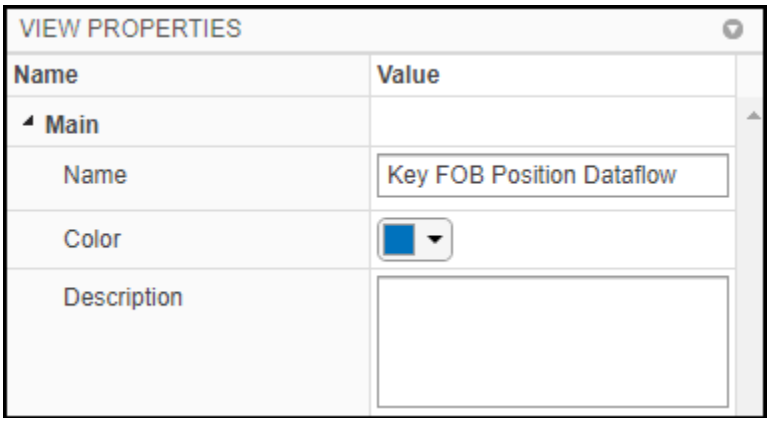
Use System Composer to author software architectures composed of software components, ports, and interfaces. Design your software architecture model, define the execution order of functions from your components, simulate your design at the architecture level, and generate production code. For more information, see “Author Software Architectures”.

For an example that shows how to author the software architecture of a throttle position control system in System Composer, see “Modeling the Software Architecture of a Throttle Position Control System”.

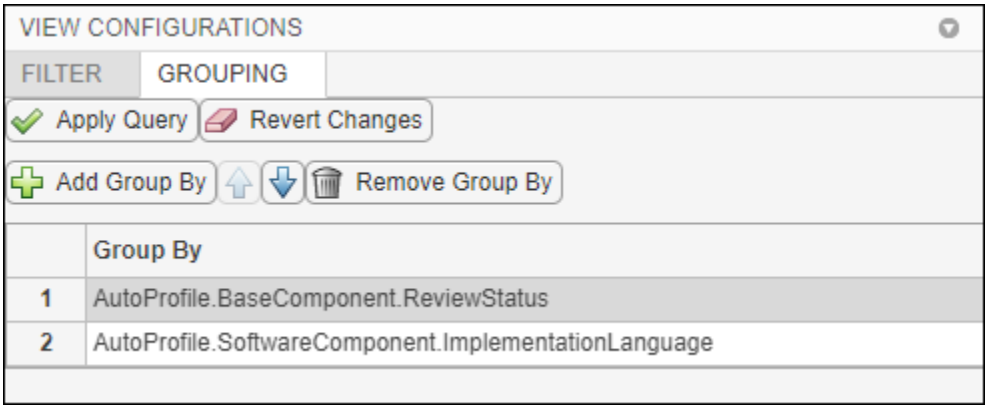
Updates to the Architecture Views Gallery and APIs

The Architecture Views Gallery updates include UI improvements and new APIs for views.

- Use View Properties on the right of the views gallery when a view is selected. Properties include Name, Color, and Description.



- Create and define the **Filter** and **Grouping** for views in **View Configurations** at the bottom of the views gallery.
- **Group By** multiple property values.



- New views APIs are available to streamline creating views programmatically. The new API allows you to add elements to the view independent of ordering and to remove elements cleanly. Existing APIs are deprecated.

For more information, see “Create Architecture Views Interactively” and “Create Architectural Views Programmatically”.

Compatibility Considerations

This table shows new APIs in this release and corresponding APIs that are removed.

New	Removed	Rationale
`createView`	`createViewArchitecture`	The new object `systemcomposer.view.View` is created with a name and optionally with a query, grouping criteria, and the option whether to include references. The `Root` property of a `View` object is a `systemcomposer.view.ElementGroup` object.
`addElement`	`addComponent`	The `addElement` method is used to populate the view manually and exists under the `systemcomposer.view.ElementGroup` object.
`removeElement`	`removeComponent`	The `removeElement` method is used to depopulate the view manually and exists under the `systemcomposer.view.ElementGroup` object.
`createSubGroup`	`createViewComponent`	The `createSubGroup` method is used to add subgroups under the main `systemcomposer.view.ElementGroup` object.

Performance and Scalability Improvements

In R2021a, there are performance and scalability improvements for System Composer features.

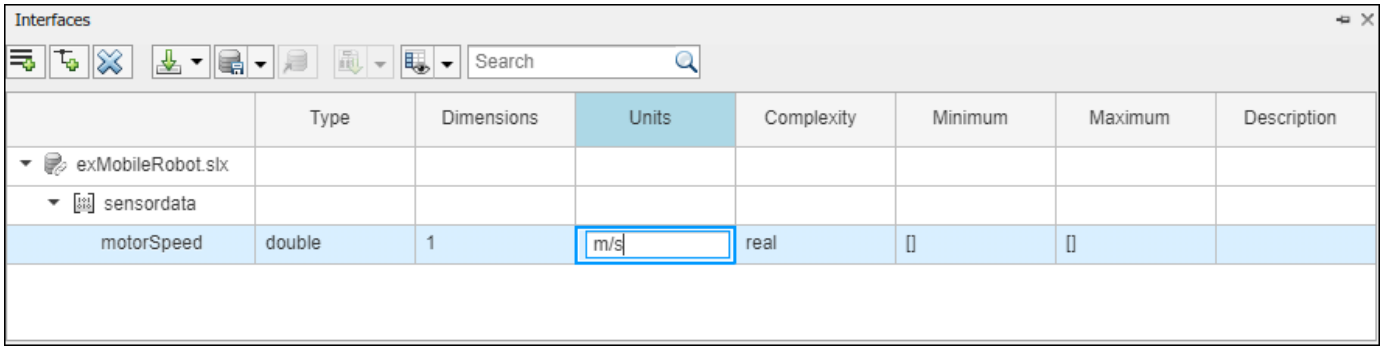
- Model-building APIs including `connect`, `addPort`, `addComponent`, and `addVariantComponent` now perform operations faster.
- Queries that query for model elements with specific stereotypes and model elements that satisfy constraints on stereotype property values now perform better. For examples, see “Find Elements in a Model Using Queries”.
- Interface editor scalability enhancements include loading improvements and an improved user experience on interface search, sort, expand, and collapse.

Interface Editor enhancements

In R2021a, the Interface Editor features three enhancements.

- Use multiple columns to track interface element properties. Column names include: Type, Dimensions, Units, Complexity, Minimum, Maximum, and Description.

- Show and hide columns by toggling the button.
- After creating interface elements, you can edit the properties in the columns from the Interface Editor.



For more information, see “Define Interfaces”.

R2020b

Version: 1.3

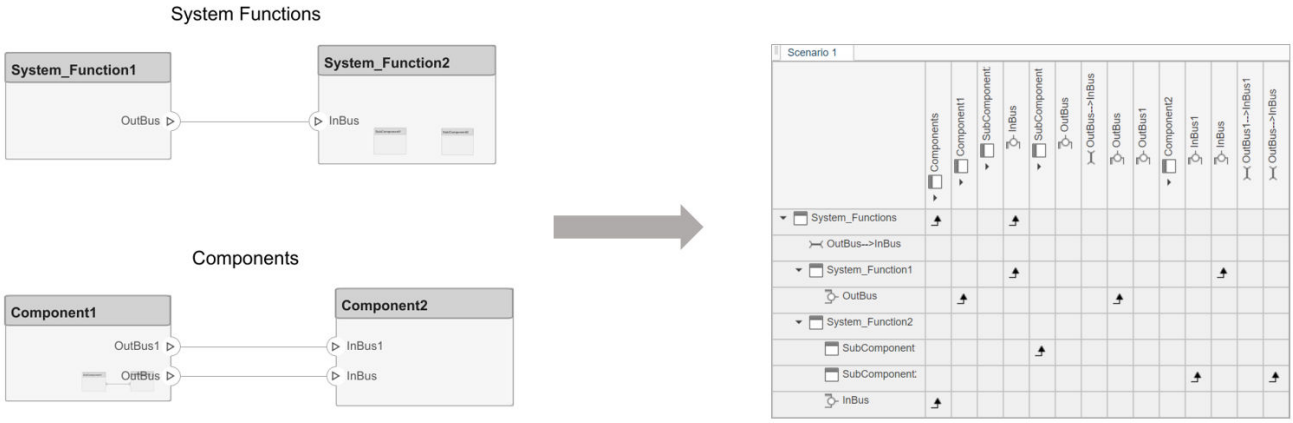
New Features

Bug Fixes

Model to Model Allocations: Establish traceable and directed relationships between architecture elements in source and target models

System Composer allocations allow you to establish a direct relationship from architecture elements in one model such as components, ports, and connectors, to architecture elements in another model. For more information, see Create and Manage Allocations.

You can use allocations to establish relationships from software components to hardware components and to indicate the deployment strategy. You can allocate different instances of components, ports, and connectors and use the allocation to perform various analyses such as resource-based allocation analysis. For an example, see Allocate Architectures in a Tire Pressure Monitoring System.



An allocation set is a collection of allocation relationships between two models. The allocation set is stored with the `.mldatx` file extension. You can create an allocation set in the Allocation Editor. Upon the allocation set creation, a matrix appears where the source model elements are the rows and the target model elements are the columns. You can open the allocation editor from the **Allocation Editor** button in the toolstrip or by typing the `editor` command in the MATLAB command window.

Enhanced workflows with Architecture Views

In R2020b, workflows for authoring and generating multiple views are enhanced to simplify view editing and filtering. There are six enhancements:

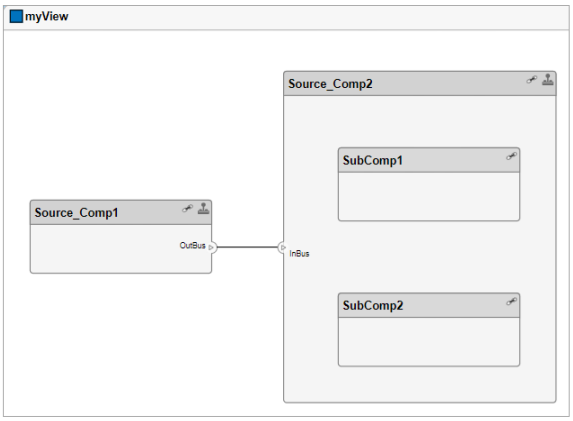
- You can start editing a view after selecting the view in the **View Browser** pane.
- You can add components from the composition tree in the **Model Components** pane by dragging and dropping them from the composition tree.
- You can use the keyboard shortcut **Delete** to delete components from the view.
- You can use the **Group** button in the toolstrip to create a view component based on the selected components.
- You can ungroup components by using the **Ungroup** button in the toolstrip.
- View filtering is now placed in a separate pane in which you can customize the query and apply it to your view.

For more information, see Create Architecture Views Interactively.

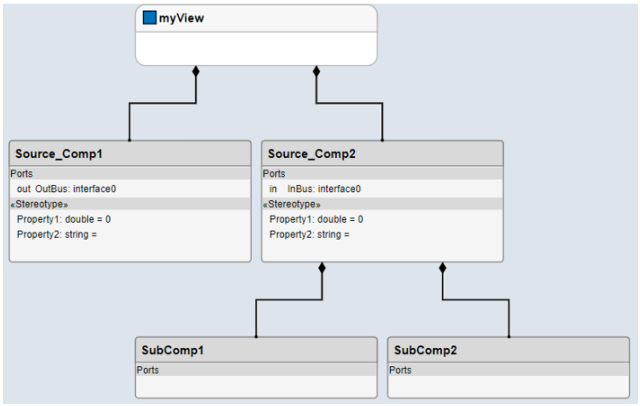
Hierarchy Views: Display component hierarchy as a tree diagram

The Hierarchy View allows you to visualize component hierarchy together with component stereotypes, stereotype properties, and the reference type a component instantiates. Hierarchy diagrams show the same set of components visible in the component diagram view, and components are selected and filtered in the same way. Any component diagram view can be optionally represented as a hierarchy diagram. The Hierarchy View flattens the component diagram view and displays in a tree form. You can see Hierarchy View by selecting **Hierarchy diagram** in the Architecture Views Editor.

Component diagram



Hierarchy diagram



The hierarchy diagrams show a single root, which is the view specification. The root corresponds to the containing system box shown in the component view. The lines between components originate from the parent component with the filled diamond symbols that are used in UML compositions and end with a terminating dot at the child component.

For more information, see Display Component Hierarchy Using Hierarchy Views.

Referenced Data Dictionaries: Organize interfaces in a hierarchy of data dictionaries

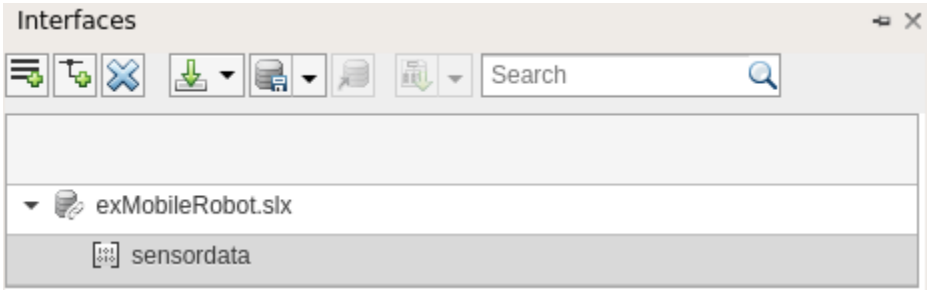
In R2020b, an interface data dictionary can reference other data dictionaries, creating a hierarchy in which the dictionary aggregates interfaces defined in its referenced dictionaries. A System Composer model linked to the root (top-level) dictionary of a hierarchy can access and use interfaces defined in the root as well as any of the dictionaries it references directly or indirectly.

For more information, see Save, Link, and Delete Interfaces.

Interface Editor enhancements

In R2020b, there are three enhancements to the Interface Editor.

- The **Interfaces** pane is now placed at the bottom of the model window.

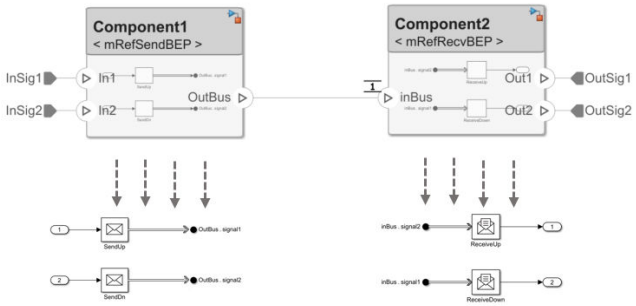


- You can search the interface table on the first column.
- You can sort interfaces and specify the sorting criteria.

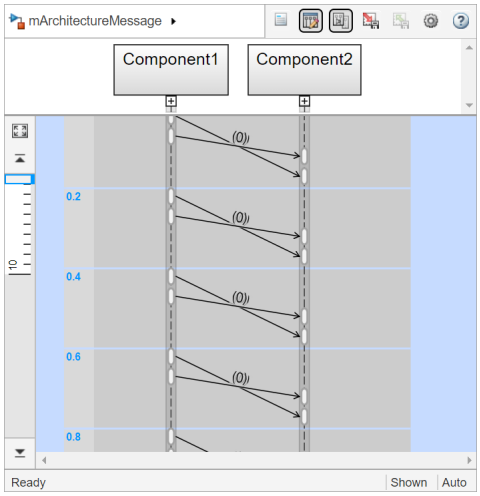
For more information, see Define Interfaces.

Support for component behavior using Simulink models with message input and output

You can define component behavior by linking a Simulink® behavior model with message input and output. For more information about Simulink messages, see Simulink Messages Overview (Simulink).



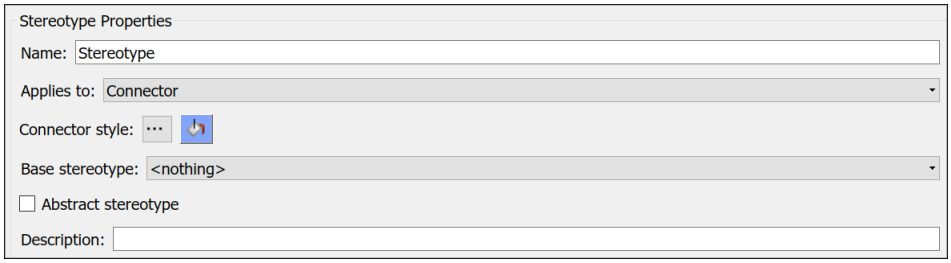
Use the **Sequence Viewer** tool to visualize events that represent the flow of messages, functions calls, and state transitions. To activate event logging, on the **Simulation** tab, in the **Prepare** section, select **Log Events**. To visualize the logged events, go to the **Review Results** section and select the Sequence Viewer.



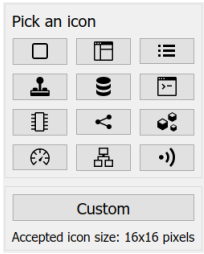
Stereotype-Based Styling: Style connectors based on stereotypes and use custom icons for component stereotypes

In R2020b, there are two stereotype-based styling enhancements:

- You can style architecture connectors using the stereotype settings. You can style connectors by using connector, port, or port interface stereotypes. Customize styling provides various color and line style choices. Connector styles are also reflected in architecture and spotlight views.



- You can use custom icon images for component stereotypes. Custom icons support `.png`, `.jpeg`, or `.svg` image files of size 16-by-16 pixels. The custom icons are displayed as badges on the components for which the stereotypes are applied.



For more information, see Define Profiles and Stereotypes.

Support for protected models as component behaviors

In R2020b, you can link a component in your architecture model to a protected Simulink model to create component behaviors. You can convert an already linked Simulink behavior model to a protected model, and this change is reflected after refreshing the model.

For more information, see Implement Components in Simulink.

Import and export requirement links along with the architectural models

In R2020b, you can:

- Export your architecture models with requirement links to your system requirements.

  When you export your architecture model using the `exportModel` function, the output structure has a `requirementLinks` field that contains all the link information present in the model.
- Import links to requirements and reconstruct them in your System Composer model using `importModel`.

  You can import requirement links to your model by using the new `requirementLinks` argument in `systemcomposer.importModel(modelName,components,ports,connections,interfaces,requirementLinks)`.

  For more information, see Import and Export Architecture Models.

R2020a

Version: 1.2

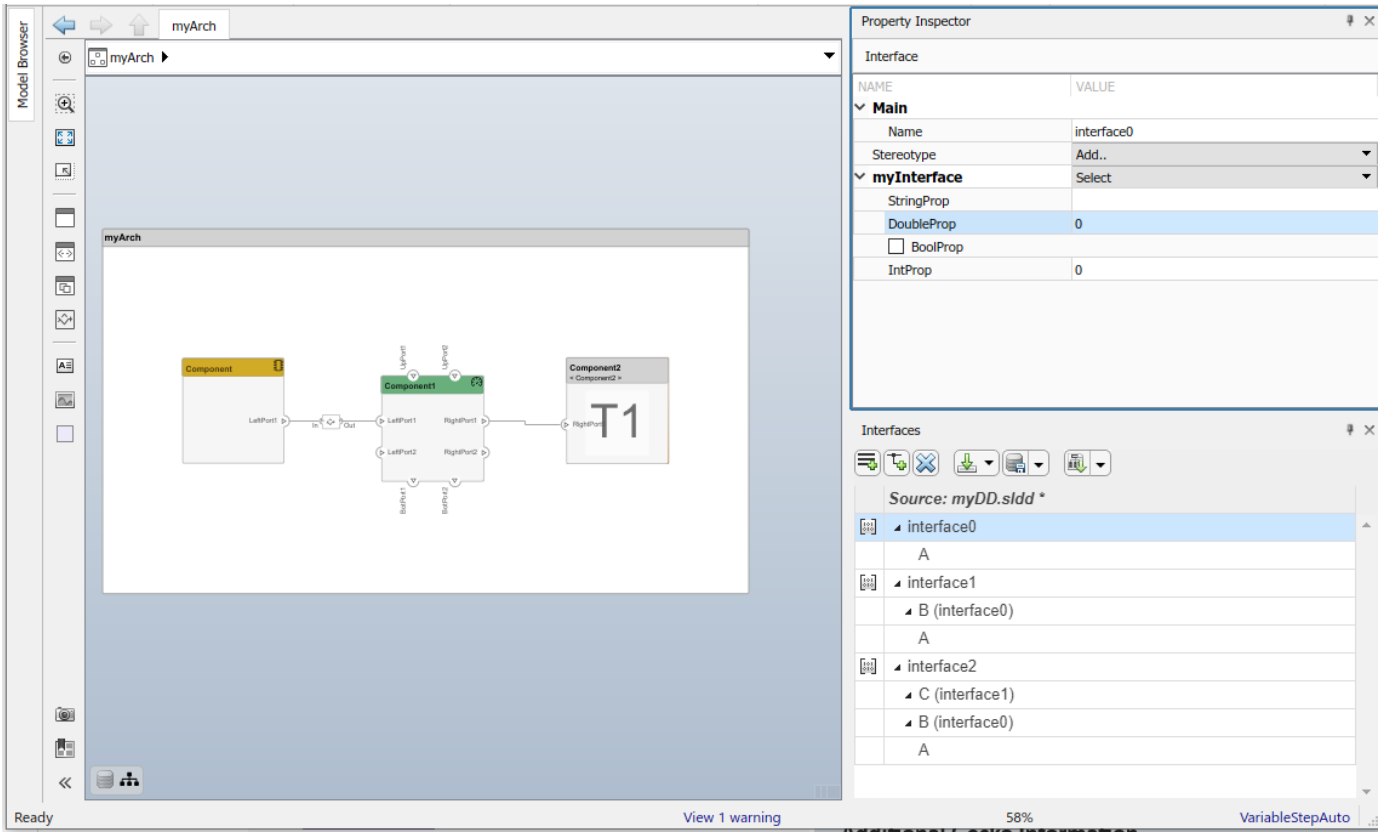
New Features

Stereotype-Based Styling: Associate a color with component stereotypes

Profiles and stereotypes are used to apply custom metadata on the architecture model elements. Element styling is an additional visual cue that indicates applied stereotypes. Use a preconfigured set of color options for component stereotypes to style the architecture component headers.

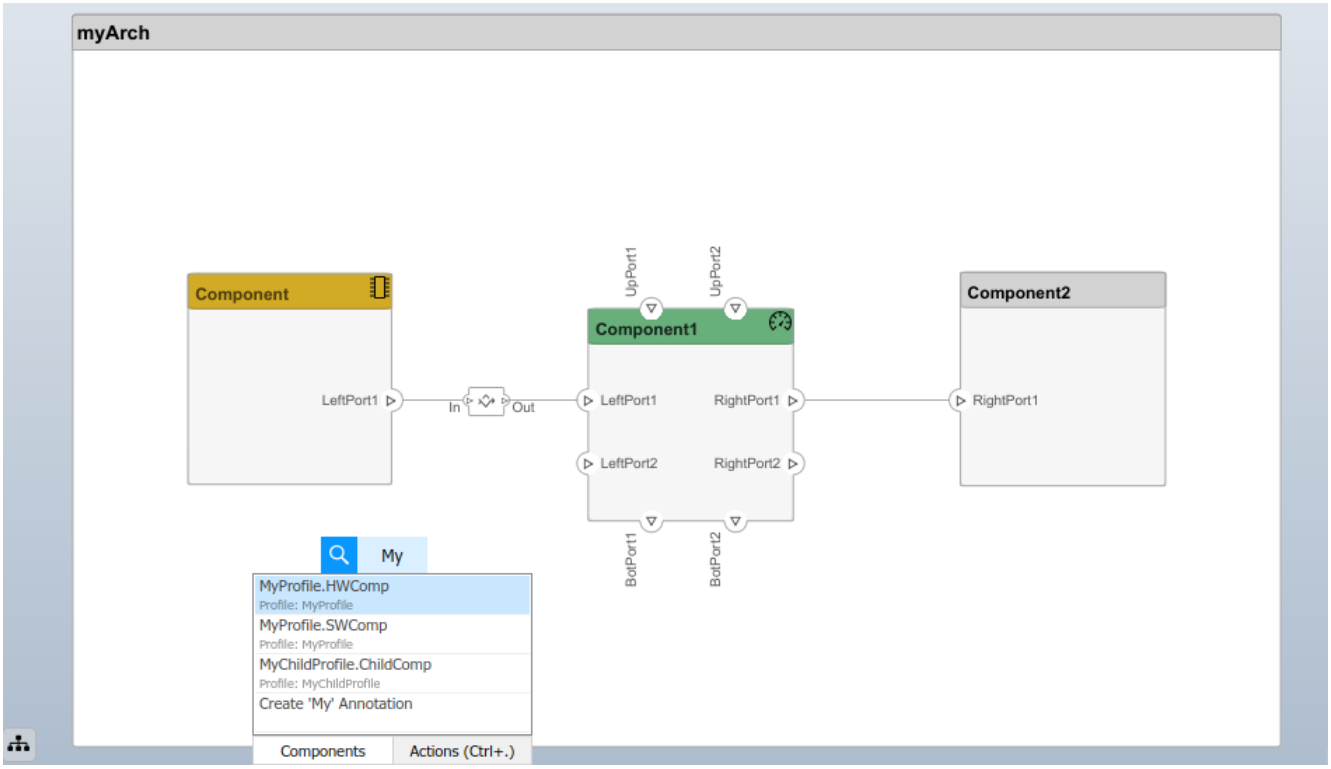
Interface Stereotypes: Apply stereotypes for custom metadata on interface objects

In R2020a, you can add stereotypes containing custom properties to port interfaces. Architecture profiles can be imported into shared dictionaries to create and manage stereotyped interfaces that are shared across multiple models, which enables collaborative workflows. Specify stereotypes that apply to interfaces in the Profile Editor. You can view and edit stereotype properties in the Property Inspector.



Quick Insert Stereotypes: Insert stereotyped components directly by typing on the canvas

You can now insert components preconfigured with all available component stereotypes in the profiles applied to a model using the quick insert capability of the System Composer canvas, which is enabled by typing into an empty location in the canvas.

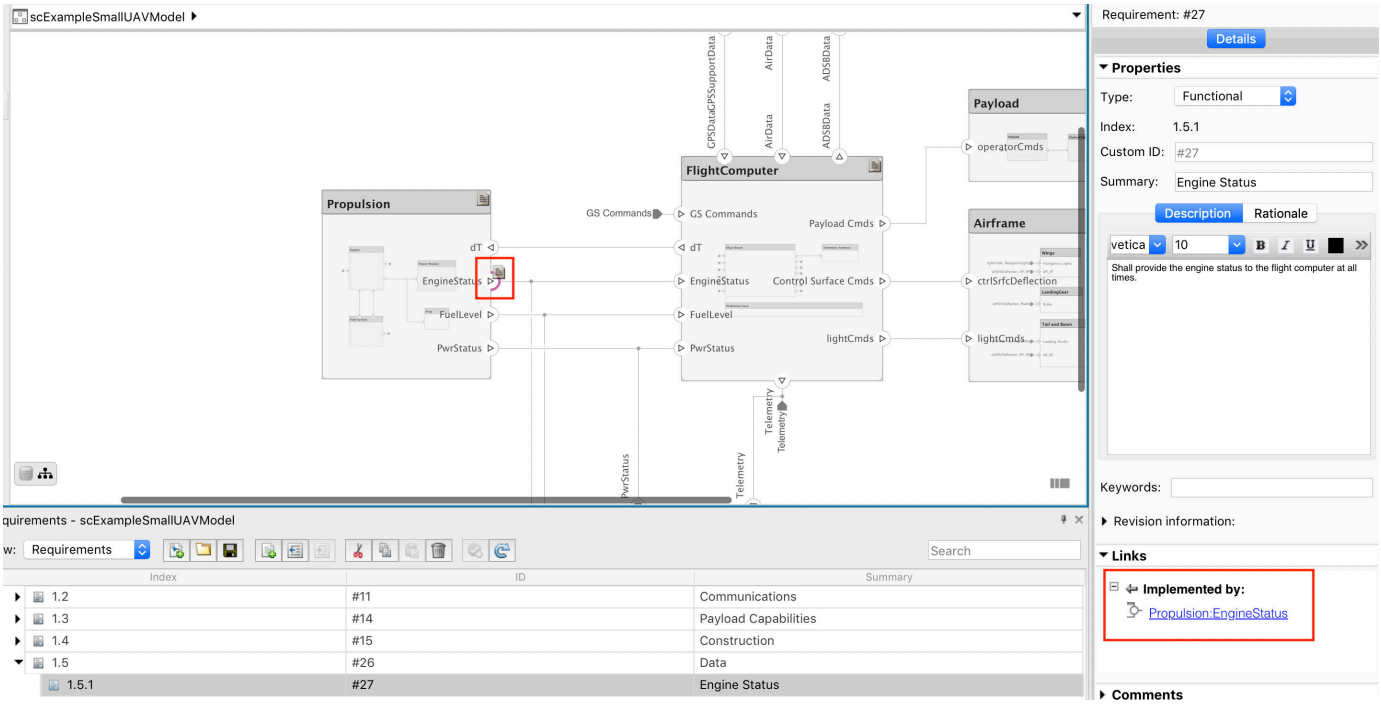


Model Templates for Referenced Models: Select from preconfigured templates when creating new referenced architecture or behavior models

To maintain specialized settings when creating Simulink behaviors or new architecture models linked to components, you can choose to use an existing template model from the palette of available template models on the Simulink start page. Specialized settings can include configuration parameters such as the solver type, diagnostic settings, and code generation settings. Use the template to specify copyright annotations and organizational styles.

Requirements on Component Ports: Link requirements to component ports

In R2020a, you can link requirements to ports in a System Composer model using the Requirements Perspective. Create requirement sets, organize requirements into hierarchies, and link requirements to components and ports using Simulink Requirements. Annotate architecture models with requirement information and navigate to the source requirement.



Partial Architecture Model Load: Improved loading for large models

System Composer now loads the architecture part of the model without completely loading the underlying block diagrams of models that are linked to components. For example, opening an architecture model with hundreds of reusable components and behaviors no longer loads every block diagram in the model hierarchy. Models are fully loaded when System Composer workflows need them or you choose to open them explicitly.

R2019b

Version: 1.1

New Features

Bug Fixes

Architecture Views

You can author and generate multiple views for specific viewpoints for design, analysis or communication. Create a view by writing queries against the model using the System Composer model query language. See Creating Architecture Views Interactively.

Interface Adapter

Visually specify an interface mapping between components with different but compatible interfaces. For example, map a signal with different interface names. See Interface Adapter.

Import and Export Architecture Models

Import a system architecture into System Composer using Excel spreadsheets or MATLAB tables. The tables define components, ports, and connections. See Importing and Exporting Architecture Models.

AUTOSAR Software Architecture

AUTOSAR Composition Editor: Author compositions, simulate functional behavior with basic software services using Composition Editor.

Support for architecture and composition modeling for the AUTOSAR Classic Platform. You can:

- Create AUTOSAR architecture models.
- Use AUTOSAR composition editor to add and connect Composition and Component blocks.
- Link AUTOSAR components to Simulink requirements.
- Define AUTOSAR component behavior by creating or linking Simulink implementation models.
- Configure AUTOSAR scheduling and simulation, including AUTOSAR Basic Software services.
- Generate and package composition AUTOSAR XML description and component code.

R2019a

Version: 1.0

New Features

Introducing System Composer

System Composer enables the definition, analysis, and specification of architectures and compositions for model-based systems engineering and software design. With System Composer, you allocate requirements while refining an architecture model that can then be designed and simulated in Simulink.

System Composer lets you create architecture models that describe a system in terms of components and interfaces. You can also populate an architecture model from the architectural elements of Simulink designs or C/C++ code. You can create custom live views of the model to study specific design or analysis concerns. With these architecture models, you can analyze requirements, capture properties via stereotyping, perform trade studies, and produce specifications and ICDs.

Composition Editor

Author and edit architecture models in the Composition Editor. Model physical and logical architecture of a system. Create a visual representation of components, ports, and connections and specify information exchange between components with interfaces. Decompose components to add detail and define hierarchical relationships. Create reusable architectures.

Spotlight Views

You can create Spotlight views to analyze component dependencies and hierarchy. A Spotlight view is a simplified view of a model that captures the upstream and downstream dependencies of a specific component of interest. You can shift the spotlight to a different component from within the Spotlight view. You can also trace an element from the Spotlight view back to the composition.

Linking Simulink Behavior Models

You can use Simulink models with System Composer to define component behavior by creating or linking to a Simulink behavior model. Take advantage of System Composer architecture editing and analysis capabilities for Simulink behavior models by exporting them as architecture models.

Linking and Managing Requirements

You can link components to requirements using the Requirements perspective. Create requirement sets, organize requirements into hierarchies, and link requirements to components using Simulink Requirements™. Annotate architecture models with requirements information and navigate to the source requirement.

Stereotypes and Profiles

You can create stereotypes as extensions of components, ports, and connections by defining additional properties. Use Profile Editor to define profiles as self-consistent sets of stereotypes. Import profiles into architecture models. Assign stereotypes to model elements.

Architecture Model Analysis

You can use MATLAB® analytics with System Composer API to write scripts to generate data that can be used for trade studies or for verifying nonfunctional requirements.