

SimEvents[®] Release Notes



MATLAB[®]&SIMULINK[®]



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

SimEvents® Release Notes

© COPYRIGHT 2005–2023 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.

R2023a

Bug Fixes

R2022b

Create and manipulate string data in discrete-event charts 2-2

R2022a

Bug Fixes

R2021b

Bug Fixes

R2021a

Bug Fixes

R2020b

Use Simulink Bus Creator and Bus Selector blocks to combine and select entity paths 6-2

Functionality being removed or changed	6-2
Behavior change for models containing blocks from R2015b and earlier	6-2
System object authoring	6-2

R2020a

Display entity attributes as port value labels for debugging	7-2
Support for fixed-point data type	7-2
Event actions support calling a scoped Simulink Function block	7-2

R2019b

MATLAB Discrete-Event System Block: Exchange data with Data Store Memory block	8-2
Conveyor System Block: Use data from incoming entities to specify the conveyor speed	8-2
Entity Transport Delay Block: Model variable transportation delay of an entity	8-2
Unified animation for SimEvents and Simulink Functions	8-2
Entity Queue Block Updates	8-3
Message Receive Block Updates	8-3
Functionality being removed or changed	8-3
Behavior change for models containing blocks from R2015b and earlier	8-3
Behavior change for declaring global variables in MATLAB Discrete-Event System block	8-3
Behavior change in Message Receive, Message Send, and Entity Queue blocks	8-4
Behavior change in Message Receive block internal queue	8-4

R2019a

Resource Pool Block: Scope your resources to be available at certain levels of the model hierarchy	9-2
---	------------

New matlab.DiscreteEventSystem methods for resource management	9-2
MATLAB Discrete-Event System Block: Call Simulink functions from a MATLAB Discrete-Event System block to create customized blocks and behavior	9-2
New examples of creating custom blocks using MATLAB Discrete-Event System block	9-3
New example of modeling a store inventory management system	9-3
New example for modeling traffic intersections as a queueing network	9-3
Resource Releaser Block Updates	9-3
Number of Matched Entities: Specify the number of entities to be matched from each incoming stream by the Entity Selector block	9-4

R2018b

Entity Find Block: Find entities that use a specific resource to modify or extract them	10-2
New matlab.DiscreteEventSystem class method	10-2
Sequence Viewer Block updates	10-2
New example for modeling machine failure	10-2

R2018a

Entity Store Block: Hold unordered entities in this block that serves as a container or bin	11-2
Entity Selector Block: Select entities using a reference entity	11-2
Hit Crossing Messages: Send messages to SimEvents to indicate events in Simulink for hybrid system modeling	11-2
New matlab.DiscreteEventSystem methods	11-2

R2017b

Conveyor System Block: Simulate manufacturing and transportation of goods using the block to model entities moving along a conveyor . . .	12-2
MATLAB Discrete-Event System Acceleration: Speed up your simulations using code generation mode in the MATLAB Discrete-Event System block	12-2
matlab.DiscreteEventSystem class methods	12-2

R2017a

Event Action Patterns: Automatically insert MATLAB code that generates random numbers and repeating sequences	13-2
Entity Gate: Permit or prevent entity arrival based on entity attributes	13-2
Message Viewer: View states as lifelines	13-2

R2016b

Batch and Unbatch Blocks: Group entities into fixed batch sizes and access batch elements and attributes through event actions	14-2
Parameters in Event Actions: Access workspace variables through event actions	14-2
Message Viewer: Inspect structured data values and function call sequencing during model execution	14-2
Variable resource pools	14-2
MATLAB Discrete Event System block updates	14-2
Nested buses	14-2
Update your SimEvents models	14-3

Event Actions: Modify entity attributes, service, and routes on events such as entity generation, entry, and exit	15-2
MATLAB Discrete Event System Block: Author custom SimEvents blocks using MATLAB	15-2
Discrete Event Chart: Create Stateflow state transition diagrams that process entities, react to entity events, and follow precise timing for temporal operations	15-2
Entity Multicast: Wirelessly broadcast copies of entities to multiple receive queues	15-2
Domain Transitions: Automatically switch between time-based and event-based signals	15-3
Simulink Integration: Use Simulink features, such as Fast Restart to speed up simulation runs and Simulation Stepper to debug	15-3
Unified Entity Type: Define entity types that are consistent across Simulink, Stateflow, and SimEvents products	15-3
Major changes to discrete-event modeling	15-3
Discrete Event Visualization: Create custom animation and inspection mechanisms using MATLAB APIs	15-3
SimEvents Debugger	15-3
Changes in Blocks	15-4
Block Icons: Create presentation-quality models using new SimEvents blocks with improved icons	15-5
New SimEvents Examples	15-5

Bug Fixes

R2015a

Resource management blocks to define, acquire, and release resources	17-2
Data type control of entity attributes	17-2
API for integration with custom visualization	17-2
Updated examples	17-2

R2014b

Models Using SimEvents Blocks from a Release Prior to V4.0 No Longer Run	18-2
---	------

R2014a

Bug Fixes

R2013b

Bug Fixes

R2013a

Drag-and-drop block insertion and one-click connection for entity lines	21-2
Drag-and-drop block insertion	21-2
One-click entity line connection	21-2
Available Attributes list for Set Attribute and Get Attribute blocks	21-2
Entity attribute names visible in SimEvents blocks while editing the model	21-4
Timing tags visible in SimEvents blocks while editing the model	21-5

Context-sensitive help on block dialog boxes	21-6
---	-------------

R2012b

Port Data Types display option that provides in-model view of SimEvents entity types and structures	22-2
Model update and improvement checks in Upgrade Advisor and Model Advisor	22-2
N-Server block options to pause, force service completion, and monitor server occupancy	22-4
Event-based signal port connecting directly to Simulink Constant block without Gateway block	22-8
New examples	22-8

R2012a

New Configuration Parameter for Prevention of Implicit Event Duplication	23-2
Support for Fixed-Step Solvers	23-2
Enhanced Migration Workflow	23-2
Model Advisor Checks for SimEvents Models	23-2
Visual Appearance of Legacy Blocks	23-3
Enhanced seupdate Migration Utility	23-3
Enhanced Visibility of Impact when Saving Models at Earlier Versions ..	23-3
Changes to Modeling Semantics	23-3
Gateway Blocks Support Buses	23-3
Enhanced Support for Simulink Subsystems	23-4
Changed Use of Event Priority Parameter	23-4
Block Library Changes	23-5
New Blocks	23-5
Changes to Block Parameters	23-5
Functions Being Removed	23-6
Demos	23-6
New Demo	23-6
Demos Removed	23-6

Major Changes to Discrete-Event Modeling	24-2
Modeling Syntax Enhancements	24-2
Gateway Blocks	24-2
Input Port Styles	24-3
Simulink Blocks in Event-Based Computations	24-3
Atomic Subsystem and Event Filter for Better Control on Event-Based Execution	24-3
Animation and Debugging Enhancements	24-3
Changes in Blocks and Modeling Semantics	24-4
Change in Entity Arrivals at Empty Queue	24-4
Arrival of Entity Placed at Head of Queue	24-5
Change to Sorted Order of Hybrid Systems	24-6
Discrete Event Signal to Workspace Block Records Initial Value	24-6
Statistical Output Signals Have Updates Throughout Simulation	24-6
Configuration Parameter Race Condition Detection Changes	24-6
Block, Event, and Entity Identifiers	24-7
SimEvents Support for Variant Subsystems	24-7
Discrete Event Signal to Workspace Block Can Reside in Atomic Subsystem	24-7
Set Attribute and Get Attribute Blocks Can Process at Most 32 Attributes	24-7
Attribute Propagation Changes	24-8
Parameter Handling Change for Routing and Entity Management Blocks	24-8
Change in Execution Order of Simultaneous Events with Same Priorities	24-8
Model Acceleration Changes	24-8
Rapid Simulation Support	24-8
New Blocks Do Not Support Accelerator Mode	24-8
Block Library Changes	24-9
Updated SimEvents Library	24-9
Library Changes	24-9
New Blocks	24-9
Blocks Being Removed	24-9
Functions Being Removed	24-10
Demos	24-10
Updated Demos in the Product	24-10
Updated Demos and Examples in the Documentation	24-11

R2011a

Changes in Menu of Scope Figure Window	25-2
---	------

R2010b

Bug Fixes

R2010a

Block-Based Breakpoints in Debugger	27-2
Block Operations Information in Debugger	27-2
Changes in Behavior of Pending Entity Signals	27-2
Renaming of Parameter to Enable Pending Entity Signal	27-6
Expanded Options for Opening Release Gate	27-6
Blocks in Attributes Library Must Get or Set at Least One Attribute ...	27-6
Parameters and Parameter Values Being Removed	27-7

R2009b

Support for Batch Simulation Using Rapid Simulation Target	28-2
Expanded Options for Resetting Entity Departure Counter	28-2

R2009a

Debugger Supports Stepping, Breakpoints, and Querying	29-2
Event Logging Options Removed from Configuration Parameters Dialog Box	29-3

Discrete Event Signal to Workspace Block Clarifies Timing	29-4
--	-------------

R2008b

New Demos for Modeling Architectures and Manufacturing Processes	30-2
Attribute Name Incrementing in Set Attribute and Get Attribute Blocks	30-2
Change in Parameter Name of Event-Based Entity Generator Block ...	30-2

R2008a

Initial Value Block in Signal Management Library	31-2
Discrete Event Subsystem Supports Complex and Nonscalar Values ...	31-2
Seed Management for Random Number Generators	31-2
Configuration Parameters for Diagnostics	31-2
“What’s This?” Context-Sensitive Help Available for Simulink Configuration Parameters Dialog	31-2
New Demos	31-3

R2007b

Attribute Computations Using MATLAB Code	32-2
Simplifying a Model Using the Attribute Function Block	32-2
Attributes Support Complex Values	32-3
Enhanced Visibility and Logging of Events	32-3
New Demos for Shared-Resource Applications and Advanced Techniques	32-4
Consolidation and Removal of Some Tutorial Demos	32-4
Changes in Categorization, Titles, and Content of Some Demos	32-5
Subsystem Connection Port for Entity Paths	32-5

Configuration Parameters to Control Livelock	32-5
Processing Events Via the Event Calendar Instead of Immediately	32-6
Enhanced Support for Multiple Simultaneous Transitions in Switches and Gate	32-7
Change in Indexing in Attribute Scope Block	32-8

R2007a

Attributes Support Multidimensional Values	33-2
Combining and Splitting Entities	33-2
Timeout Feature Establishes Entity Time Limits	33-2
New Demos for Video Processing, Communications, and Architecture Modeling	33-3
Change in ARQ Demo	33-3
Output Switch Block Options for Storage and Initial Condition	33-4
Entity Departure Counter Block Can Create Attribute	33-5
Changes in Names of Parameters Related to Event Priorities	33-5
Change in Default Entity Type of Entity Generators	33-6
Obsolete Blocks	33-6

R2006b

Event-Based Sequence Generator Block	34-2
New Tutorial and Application Demos	34-2
Event Translation Block Supports Delay from a Signal	34-2
Routing Blocks Support Unlimited Entity Ports	34-3
Initial Outputs of SimEvents Blocks	34-3
History Options and Other Changes in Scope Blocks	34-5
Other Changes in Scope Blocks	34-5

Parameters for Lognormal Distribution	34-6
SimEvents Blocks Compatible with Accelerator Mode	34-6
Livelock Detection	34-6

R2006a

Replicate Block Supports Partial Replication	35-2
---	-------------

R14SP3+

Introduction to SimEvents	36-2
--	-------------

R2023a

Version: 5.14

Bug Fixes

R2022b

Version: 5.13

New Features

Create and manipulate string data in discrete-event charts

Discrete-Event Chart blocks now support string data. To see a list of valid operators, see the **MATLAB® Action Language Operators** section on String Data. For more information about using string data in charts, see Manage Textual Information by Using Strings.

R2022a

Version: 5.12

Bug Fixes

R2021b

Version: 5.11

Bug Fixes

R2021a

Version: 5.10

Bug Fixes

R2020b

Version: 5.9

New Features

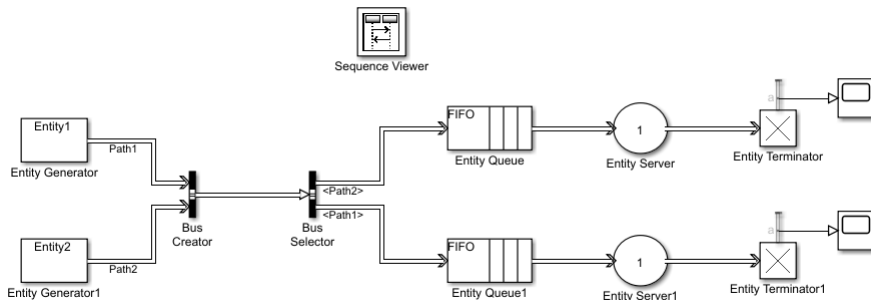
Bug Fixes

Compatibility Considerations

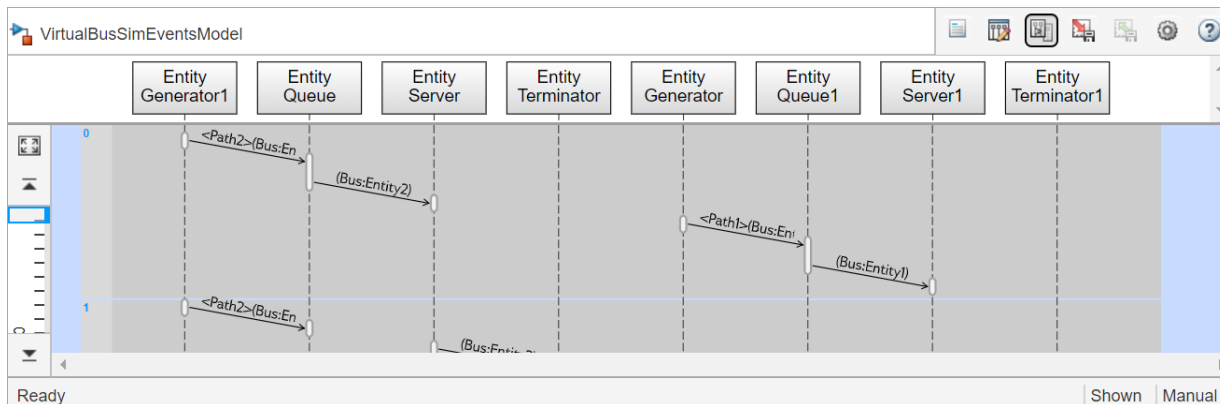
Use Simulink Bus Creator and Bus Selector blocks to combine and select entity paths

You can use Simulink® Bus Creator and Bus Selector blocks to group multiple independent entity paths and virtual buses to simplify entity flows.

This example shows two Entity Generator blocks that generate entities with different intergeneration times. The Bus Creator block combines entity paths while maintaining the separate paths in the group. The Bus Selector block directs entity paths to the corresponding Entity Queue blocks.



You can use animation or the Sequence Viewer block to visualize the flow of buses in your model.



Functionality being removed or changed

Behavior change for models containing blocks from R2015b and earlier

Behavior change in future release

Starting in R2021b, models containing blocks from SimEvents version R2015b and earlier will not be supported. For more information, see Migration Considerations.

System object authoring

Behavior change in future release

- The class `matlab.system.StringSet` will be removed in a future release to bring System object™ authoring closer to MATLAB classes. Use `sysobjupdate` to update your System object to the latest syntax.

For more information, see Limit Property Values to Finite List (MATLAB).

-
- The System object property attributes `Logical` and `PositiveInteger` will be removed in a future release to bring System object authoring closer to MATLAB classes. Use `sysobjupdate` to update your System object to the latest syntax.

For more information, see [Validate Property and Input Values \(MATLAB\)](#).

- To simplify System object authoring, the functionality and methods from these classes are directly included with the base System object class, `matlab.System`.
 - `matlab.system.mixin.CustomIcon`
 - `matlab.system.mixin.Propagates`
 - `matlab.system.mixin.SampleTime`
 - `matlab.system.mixin.Nondirect`

Use `sysobjupdate` to update your System object to the latest syntax.

R2020a

Version: 5.8

New Features

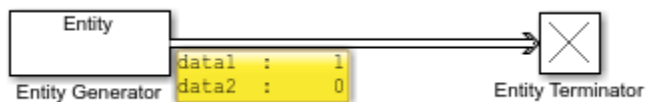
Bug Fixes

Display entity attributes as port value labels for debugging

You can display entity attributes as port value labels on the block diagram during or after a simulation.

To display entity attributes, right-click the entity path emerging from a block, and click **Show Value Label of Selected Port**, and select the attributes to be displayed.

In this example, an Entity Generator block generates entities with two attributes, `data1` and `data2`, which are displayed as port values.



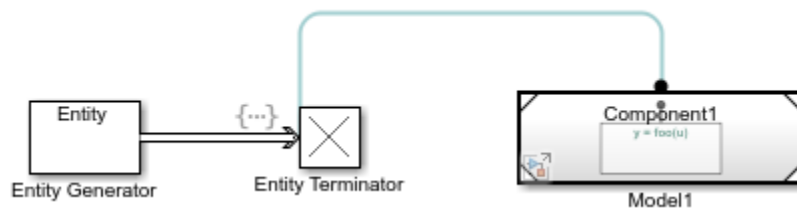
Support for fixed-point data type

Entities and attributes support fixed-point data type. To learn more about the fixed-point data type, see [Specify Fixed-Point Data Types \(Simulink\)](#).

Event actions support calling a scoped Simulink Function block

You can call Simulink functions from event actions when the **Function visibility** parameter of the Simulink Function block is set to `scoped`. You can use the name of the subsystem or the Model block together with the function name to call the scoped Simulink function.

In this example, a Simulink Function block with the function signature `foo()` is in a Model block called `Model1`. The **Function visibility** parameter of the Simulink Function block is set to `scoped`. When an entity enters the Entity Terminator block, the Simulink function `foo()` is called by using the code `Model1.foo()` in the entry event action.



For an example, see [Model a Message Receive Interface that Runs on Message Availability \(Simulink\)](#).

R2019b

Version: 5.7

New Features

Bug Fixes

Compatibility Considerations

MATLAB Discrete-Event System Block: Exchange data with Data Store Memory block

Create custom blocks using the MATLAB Discrete-Event System block to read data from or write data to the Data Store Memory block from the Simulink library.

Declare a global variable in a discrete-event System object, which is also defined as the **Data store name** in the Data Store Memory block. Use the global variable in an event action method to exchange data between the MATLAB Discrete-Event System block and the Data Store Memory block.

If multiple MATLAB Discrete-Event System blocks are exchanging information with the same Data Store Memory block, these events execute based on their respective event priorities in the event calendar.

For an example, see Resource Scheduling Using MATLAB Discrete-Event System and Data Store Memory Blocks.

Conveyor System Block: Use data from incoming entities to specify the conveyor speed

The Conveyor System block has a new drop-down parameter, **Conveyor speed source**, which determines the source to specify the conveyor speed. You can choose **Dialog** or **Port** as the source.

- **Dialog** — The speed of the conveyor is constant.
- **Port** — The speed of the conveyor is specified by the incoming anonymous entities.

When **Port** is selected, a new input port appears on the block to accept anonymous entities. Anonymous entities carry data that specifies the speed of the conveyor. The data must be a value between 0 and ∞ . The block does not accept new entities to the conveyor belt when the speed is set to 0. The new speed is applied to all entities in the Conveyor System block and also the incoming entities. You can track the speed of the conveyor belt by using the **Conveyor speed, s** statistic.

Entity Transport Delay Block: Model variable transportation delay of an entity

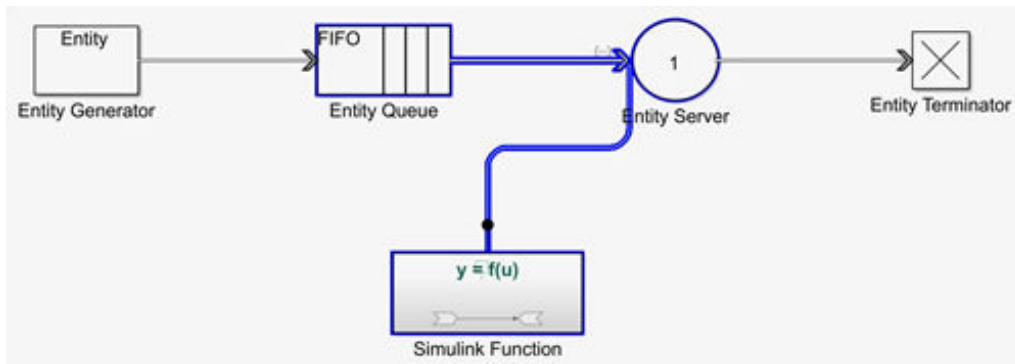
Use the Entity Transport Delay block to apply a transport delay to entities.

The block accepts entities at its first input port and the instantaneous delay for the entity at its second input port. The block connects SimEvents to Simulink by allowing transport delays in a discrete event process to be modeled as a continuous-time process in Simulink.

For an example usage, see Modeling Cyber-Physical Systems.

Unified animation for SimEvents and Simulink Functions

Animation now supports visualization of Simulink function connectors. This is an example animation that visualizes the entity flow and the Simulink function call.



To access animation, in a model window, right-click and select **Animation Speed**.

Entity Queue Block Updates

The Entity Queue block has a new option, **Overwrite the oldest element if queue is full**. Use the check box to specify entity overwriting policy when the queue is full:

- When the check box is selected, an incoming entity overwrites the existing oldest entity after the queue reaches its capacity. In this mode, you can specify the queue type as first-in-first-out (FIFO) or last-in-first-out (LIFO).
- When the check box is cleared, an incoming entity is not accepted when the queue is full.

Message Receive Block Updates

The Message Receive block has a new option, **Overwrite the oldest element if queue is full**. Use the check box to specify the message overwriting policy when the internal queue is full.

- When the check box is selected, an incoming message overwrites the existing oldest entity in the block. In this mode, you can specify the queue type as first-in-first-out (FIFO) or last-in-first-out (LIFO).
- When the check box is cleared, new messages are blocked from entering.

Functionality being removed or changed

Behavior change for models containing blocks from R2015b and earlier

Behavior change

Starting in R2019b, models containing blocks from SimEvents version R2015b and earlier have a new warning for migrating legacy SimEvents models. See [Migrate Legacy SimEvents Models](#), for more information.

Behavior change for declaring global variables in MATLAB Discrete-Event System block

Behavior change

Starting in R2019b, the discrete-event System object that is included in the MATLAB Discrete-Event System block does not support global variable declarations unless the name of the variable is used by a Data Store Memory block.

To share and refer to the global variables in your model, migrate your models that include MATLAB Discrete-Event System block using a Data Store Memory block.

Behavior change in Message Receive, Message Send, and Entity Queue blocks

Behavior change

Starting in R2019b, when you use `get_param()` function to get the block types for the Message Receive, Message Send, and Entity Queue blocks, you get a different value for the `BlockType` parameter.

- For Message Receive block, you get 'Receive' instead of 'MessageReceive'.
- For Message Send block, you get 'Send' instead of 'MessageSend'.
- For Entity Queue block, you get 'Queue' instead of 'EntityQueue'.

All other features related to the `BlockType` continue to work as expected.

Behavior change in Message Receive block internal queue

Behavior change

Starting in R2019b, the Message Receive block behaves differently when the queue is full:

- The internal queue accepts a message and overwrites the oldest message when the **Overwrite the oldest element if queue is full** is checked.
- The internal queue does not accept messages if the queue is full when the **Overwrite the oldest element if queue is full** is cleared.

Prior to R2019b, the Message Receive block accepted and dropped messages when the internal queue was full.

R2019a

Version: 5.6

New Features

Bug Fixes

Resource Pool Block: Scope your resources to be available at certain levels of the model hierarchy

The Resource Pool block has a new drop-down parameter, **Scope**, which determines the availability of resources in a model hierarchy. You can choose `Global` or `Scoped` resources in the pool.

- `Global` — Resources can be referenced from anywhere in a model hierarchy.
- `Scoped` — Resources are locally visible and can be referenced only from the subsystem that contains the Resource Pool block and all the subsystems inside.

New `matlab.DiscreteEventSystem` methods for resource management

The `matlab.DiscreteEventSystem` class has ten new methods:

- `getResourceNamesImpl` — Define resource pools from which the discrete-event system acquires the resources.
- `resourceType` — Specify an entity type and the name of the resources to be acquired by the specified entity.
- `eventAcquireResource` — Create a resource-acquiring event.
- `eventReleaseResource` — Create an event to release previously acquired resources. This method allows for partial resource release.
- `eventReleaseAllResources` — Create an event to release all the resources acquired by an entity.
- `cancelAcquireResource` — Cancel previously scheduled resource acquisition event.
- `resourceSpecification` — Specify the type and amount of resources for `eventAcquireResource` or `eventReleaseResource` requests.
- `initResourceArray` — Initialize a `resourceSpecification` array, required for code generation.
- `resourceAcquired` — Specify event actions upon successful resource acquisition.
- `resourceReleased` — Specify event actions upon successful resource release.

For an example that uses resource management methods, see [Create a Custom Resource Acquirer Block](#).

MATLAB Discrete-Event System Block: Call Simulink functions from a MATLAB Discrete-Event System block to create customized blocks and behavior

When you author discrete-event System objects, you can call a Simulink function to generate a desired custom behavior. Use the `getSimulinkFunctionNamesImpl()` method to declare the name of the Simulink function and call the function using this name when authoring a discrete-event System object. For more information, see [Call Simulink Function from a MATLAB Discrete-Event System Block](#).

New examples of creating custom blocks using MATLAB Discrete-Event System block

There are six new examples to help you learn how to create custom blocks using the MATLAB Discrete-Event System block.

- 1 Delay Entities with a Custom Entity Storage Block
- 2 Create a Custom Entity Storage Block with Iteration Event
- 3 Custom Entity Storage Block with Multiple Timer Events
- 4 Custom Entity Generator Block with Signal Input
- 5 Build a Custom Block with Multiple Storages
- 6 Create a Custom Resource Acquirer Block

New example of modeling a store inventory management system

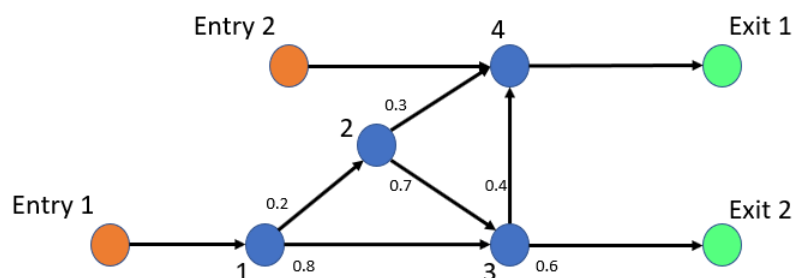
A new example shows how to model a store that requires periodic product orders for inventory replacement. In the model, the Entity Selector block matches multiple products to a corresponding order. The Entity Find block finds and extracts expired products from the inventory. Scoping the resources in the Resource Pool block determines resource availability in the model hierarchy.

Use the model to track the number of customers entering the store, sold and expired products, available supplies, and profitability of the store. Use the profitability analysis to determine the required quantity of products for inventory replacement. For more information, see Inventory Management.

New example for modeling traffic intersections as a queueing network

A new example shows how to model traffic intersections as a queueing network. The model represents a vehicle traffic network including multiple route intersections. The vehicles in the network probabilistically choose a route when they depart an intersection.

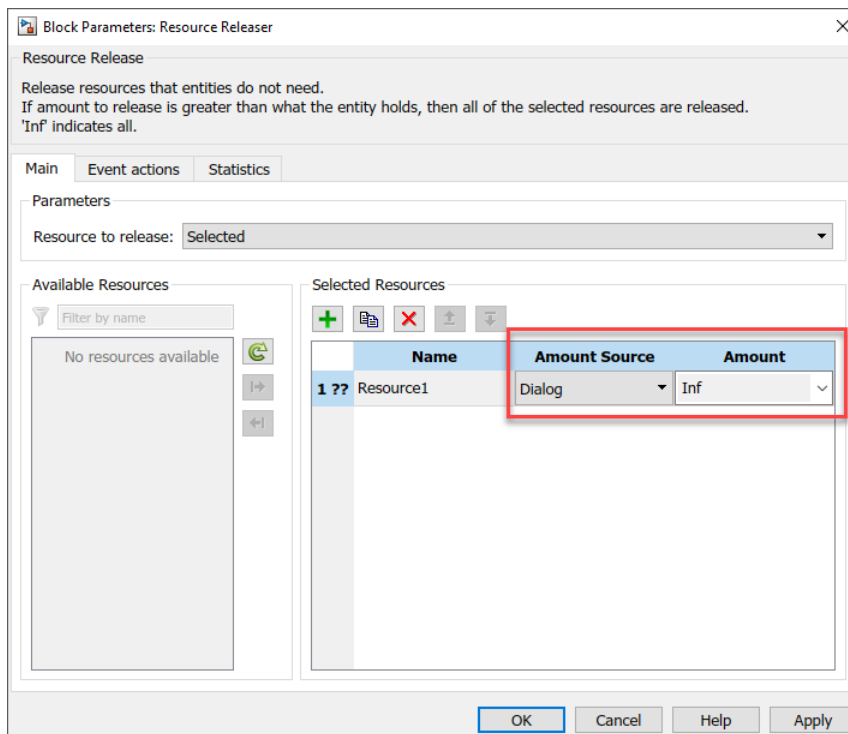
Use the model to investigate the mean waiting time of vehicles in each intersection by comparing theoretical results with the results from the simulations. For more information about the example, see Model Traffic Intersections as a Queuing Network.



Resource Releaser Block Updates

The Resource Releaser block has these updates.

- The block can hold one entity after the entity releases its resource and waits to depart the block. The entity departs the block immediately, provided it is accepted by the next block or extracted by the Entity Find block.
- You can define event actions when an entity enters the block, departs from the block, or when an entity's departure is blocked.
- You can output **Number of entities departed, d**, **Number of entities in block, n**, and **Number of entities extracted, ex** statistics from the block.
- You can choose partial resource release from entities. In the block, if you set the **Resource to release** parameter to **Selected**, you can specify the source to determine the amount of resources to be released.



Number of Matched Entities: Specify the number of entities to be matched from each incoming stream by the Entity Selector block

The Entity Selector block has a new drop-down parameter, **Number of entities source**. You can choose **Dialog** or **Attribute** to determine the number of entities matched from each stream by the block.

- **Dialog** — Specify the number of matched entities.
- **Attribute** — Specify the name of the attribute that determines the number of entities to be matched.

R2018b

Version: 5.5

New Features

Bug Fixes

Entity Find Block: Find entities that use a specific resource to modify or extract them

The Entity Find block finds entities in a model that use a specific resource. You can specify the name of the reference resource that is used by the entities in the model. You can use additional search filters along with the reference resource when finding entities.

The Entity Find block can modify entities in their original location or it can extract entities from the model and reroute them through its output port. The block manipulates found entities in these ways.

- It finds and examines entities without modification.
- It modifies entities in their original location in the model.
- It extracts found entities from their location and reroutes them through its output port.
- It extracts and modifies found entities and reroutes them through its output port.

The Entity Find block has an input port through which messages enter and trigger entity finding. For more information, see Find and Extract Entities in SimEvents Models.

New matlab.DiscreteEventSystem class method

Use the new `matlab.DiscreteEventSystem.modified` method to specify event actions upon entity modification by the Entity Find block.

Sequence Viewer Block updates

The Sequence Viewer block has these updates.

- The block allows visualizing events between blocks in referenced models.
- The block allows resizing a lifeline header by clicking and dragging its right-hand side.

New example for modeling machine failure

The Modeling Machine Failure example shows how to model random failures and scheduled maintenance of a machine during regular operation.

R2018a

Version: 5.4

New Features

Bug Fixes

Entity Store Block: Hold unordered entities in this block that serves as a container or bin

The Entity Store block stores unordered entities that are ready to leave the block immediately, provided they are accepted by the next block.

Entity Selector Block: Select entities using a reference entity

The Entity Selector block selects entities from multiple streams of ready-to-leave entities and matches them to a reference entity.

The block first accepts a **Key** entity and then selects a matching entity from each of the other input ports. The match is based on the equality of the specified attribute values.

The Entity Store block can be used as a temporary container for entities to be selected by the Entity Selector block. For an example, see Match Entities Based on Attributes in SimEvents Models.

Hit Crossing Messages: Send messages to SimEvents to indicate events in Simulink for hybrid system modeling

The Hit Crossing block supports message as a new **Output type**. You can specify the hit crossing output as a signal or a message sent to a SimEvents environment.

New matlab.DiscreteEventSystem methods

The `matlab.DiscreteEventSystem` class has two new methods.

- Use `matlab.DiscreteEventSystem.testEntry` to specify event actions that test if a MATLAB Discrete-Event System storage can accept an entity for entry.
- Use `matlab.DiscreteEventSystem.eventTestEntry` to create an event that unblocks a MATLAB Discrete-Event System storage and to retest the entry condition.

R2017b

Version: 5.3

New Features

Bug Fixes

Conveyor System Block: Simulate manufacturing and transportation of goods using the block to model entities moving along a conveyor

The Conveyor System block models the movement of entities along a conveyor system. The block allows for convenient modeling and simulation of production and logistical systems.

MATLAB Discrete-Event System Acceleration: Speed up your simulations using code generation mode in the MATLAB Discrete-Event System block

The MATLAB Discrete-Event System block now supports simulation using code generation with the **Simulate using** parameter `Code generation` option. This code generation mode reduces simulation time of SimEvents models. On the first model run, the MATLAB Discrete-Event System block simulates and generates code using only MATLAB functions supported for code generation. If the structure of the block does not change, subsequent model runs do not regenerate the code. MATLAB Discrete-Event System blocks also support code reuse for models that have multiple MATLAB Discrete-Event System blocks using the same System object source file. For more information, see [Generate Code for MATLAB Discrete-Event System Blocks](#).

matlab.DiscreteEventSystem class methods

The `matlab.DiscreteEventSystem` algorithm methods listed in the table have been renamed.

Old Method Name	New Method Name
<code>blockedImpl</code>	<code>blocked</code>
<code>destroyImpl</code>	<code>destroy</code>
<code>entryImpl</code>	<code>entry</code>
<code>exitImpl</code>	<code>exit</code>
<code>generateImpl</code>	<code>generate</code>
<code>iterateImpl</code>	<code>iterate</code>
<code>setupEventsImpl</code>	<code>setupEvents</code>
<code>timerImpl</code>	<code>timer</code>

Existing applications continue to work with the **Simulate using** parameter of the Discrete Event System block set to `Interpreted execution`. If you want to generate code for the block by using MATLAB discrete-event system acceleration, see [Replace Renamed matlab.DiscreteEventSystem Methods](#).

Associated with simulation using code generation, the `matlab.DiscreteEventSystem` class has the new method, `matlab.DiscreteEventSystem.initEventArray`.

R2017a

Version: 5.2

New Features

Bug Fixes

Event Action Patterns: Automatically insert MATLAB code that generates random numbers and repeating sequences

Event actions allow you to use MATLAB code to modify entity attributes, service, and routes on events such as entity generation, entry, and exit.

Starting in R2017a, you can select from a list of statistical distributions that generate template code for simulating stochastic event actions. Also, you can now automatically generate MATLAB code that allows for simulating repeated sequences of event actions.

Entity Gate: Permit or prevent entity arrival based on entity attributes

In R2017a, you can create conditions to dictate entity arrivals at the Entity Gate block based on entity attributes. To permit specify arrival of pending entities using a signal at the control port set the **Operating mode** parameter of the Entity Gate to Selection gate.

Message Viewer: View states as lifelines

The Message Viewer has been updated to allow either manual or automatic layouts of lifelines. Also, you can elect whether to hide or show inactive lifelines via a toggle button on the message viewer toolbar.

R2016b

Version: 5.1

New Features

Bug Fixes

Batch and Unbatch Blocks: Group entities into fixed batch sizes and access batch elements and attributes through event actions

Entity Batch Creator and Entity Batch Splitter are new blocks that let you group entities into fixed batch sizes.

Parameters in Event Actions: Access workspace variables through event actions

You can now access workspace variables and parameters, including those from mask-specific parameters and parameters defined using the `Simulink.Parameter` object.. For more information, see Parameters in Event Actions.

Message Viewer: Inspect structured data values and function call sequencing during model execution

The Message Viewer block has been updated to display the values of structured data during model execution and sequencing of function calls for these function call types:

- Calls to Simulink Function blocks — Fully supported.
- Calls to Stateflow[®] graphical or Stateflow MATLAB functions — With this support:
 - Scoped — Select the **Export chart level functions** chart option. Use the `chartName.functionName` dot notation.
 - Global — Select the **Treat exported functions as globally visible** chart option. Do not need the dot notation.

For more information, see Work with Message Viewer.

Variable resource pools

Use the Resource Pool block to send the value to a message port on the block and vary the capacity of the resource.

MATLAB Discrete Event System block updates

The MATLAB Discrete-Event System block has the following updates:

- The `matlab.DiscreteEventSystem.getEntityStorageImpl` method has been updated to return the connections between output ports and entity storage elements as a cell array.
- The MATLAB Discrete-Event System block now uses customer-provided entity type information to automatically propagates data types, dimensions, and complexity on entity ports. Previously, the block had limited support for type propagation.

Nested buses

Entity Generator blocks can now contain nested buses. For more information on entity types, see Entity Types.

Update your SimEvents models

To take advantage of SimEvents features, migrate legacy SimEvents models (pre-R2016a). For more information, see [Migrate Legacy SimEvents Models](#).

R2016a

Version: 5.0

New Features

Bug Fixes

Compatibility Considerations

Event Actions: Modify entity attributes, service, and routes on events such as entity generation, entry, and exit

Use MATLAB code and Simulink Function block to create event actions based on events such as:

- Entity generation
- Entity exit
- Entity pre-emption
- Service completion
- Entity block
- Entity pre-emption

Blocks for which you can create events include:

- Entity Generator
- Entity Queue
- Entity Server
- Entity Terminator
- Resource Acquirer

MATLAB Discrete Event System Block: Author custom SimEvents blocks using MATLAB

The MATLAB Discrete-Event System block uses System objects to create a custom SimEvents block in your model. This capability is useful for defining new algorithms to be introduced into the system creating custom blocks.

Use the `matlab.DiscreteEventSystem` class and its methods to implement a discrete-event System object. For more information, see [Implement Custom Discrete-Event Systems with System Objects](#).

Discrete Event Chart: Create Stateflow state transition diagrams that process entities, react to entity events, and follow precise timing for temporal operations


Use the Discrete Event Chart block to bring SimEvents entities into the Stateflow environment. However, to simulate the model, you must have a Stateflow license. For more information, see [Implement Discrete-Event Systems with Charts](#).

Entity Multicast: Wirelessly broadcast copies of entities to multiple receive queues

Use the Entity Multicast block to broadcast entities to multiple destinations. You do not need to connect any lines to perform the broadcast. Instead, configure the Entity Queue to accept broadcast entities with matching broadcast tags.

For your convenience, the Multicast Receive Queue block is the Entity Queue configured to receive broadcast entities.

Domain Transitions: Automatically switch between time-based and event-based signals

SimEvents can now automatically transition between time-based and event-based signals. When this transition occurs, the badge  appears on the signal line.

In previous releases, you inserted gateway blocks, such as Timed to Event Signal or Event to Timed Signal, to transition between these signals.

Simulink Integration: Use Simulink features, such as Fast Restart to speed up simulation runs and Simulation Stepper to debug

The SimEvents product now has tighter integration with Simulink. For example, you can use Fast Restart to perform iterative simulations (see [How Fast Restart Improves Iterative Simulations](#)).

To debug the simulation, use the Simulink Simulation Stepper. This feature lets you run the simulation step by step to identify problems. You can step forward or backward in your simulation. For more information, see [Simulation Stepper](#).

Unified Entity Type: Define entity types that are consistent across Simulink, Stateflow, and SimEvents products

An entity type is a class of entities that share a common set of data specifications for attributes (dimensions, data type, and complexity) and a common set of event action methods. You can define entity types using the Entity Generator block or use an existing bus object.

Major changes to discrete-event modeling

The SimEvents software has a new discrete-event simulation and modeling engine. Use the new engine with event actions, MATLAB discrete-event system object authoring and Simulink and Stateflow automatic domain transitions to create new discrete-event simulations

Discrete Event Visualization: Create custom animation and inspection mechanisms using MATLAB APIs

You can create custom animation, visualize entity movement through a model, and inspect entities and events. For more information, see [Create Custom Visualization](#).

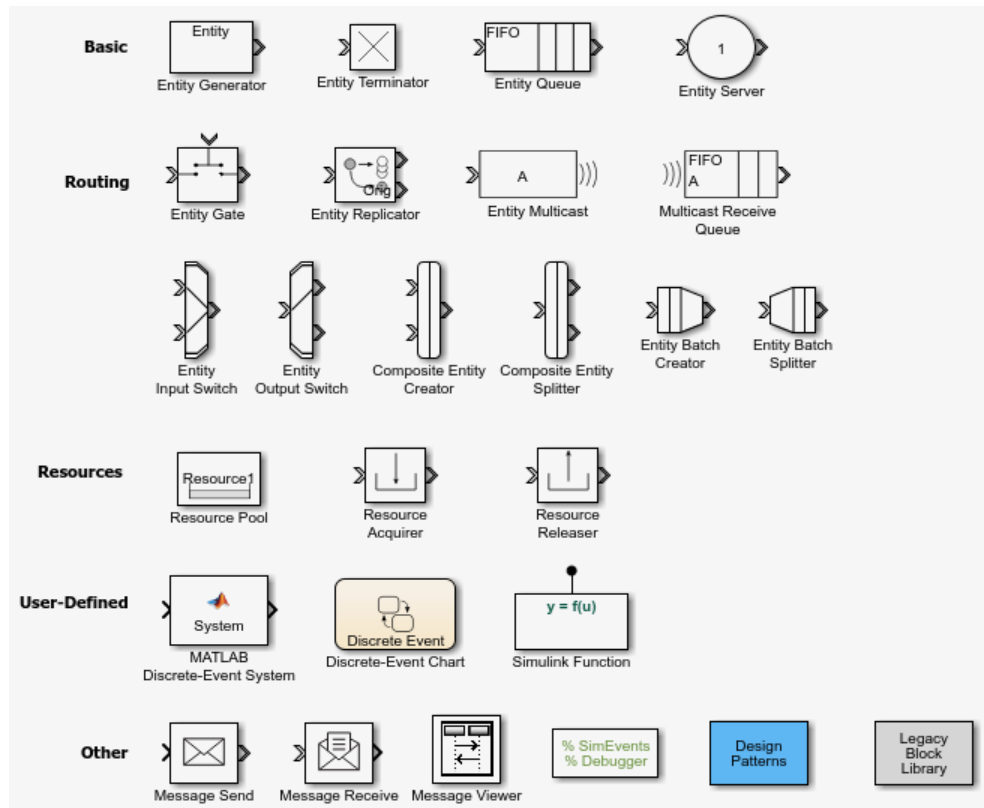
SimEvents Debugger

Debug SimEvents with the SimEvents Debugger block. Insert this block in your SimEvents model. When you start the debugger through the model, a debugger interface opens. This is a preliminary version of the debugger.

Changes in Blocks

The entire SimEvents block library is new. To access this new library, at the MATLAB Command Window, type

```
simevents
```



To access blocks from releases prior to R2016a, double-click the Legacy Blocks Library block. This sublibrary allows you to continue working on your legacy SimEvents models.

Compatibility Considerations

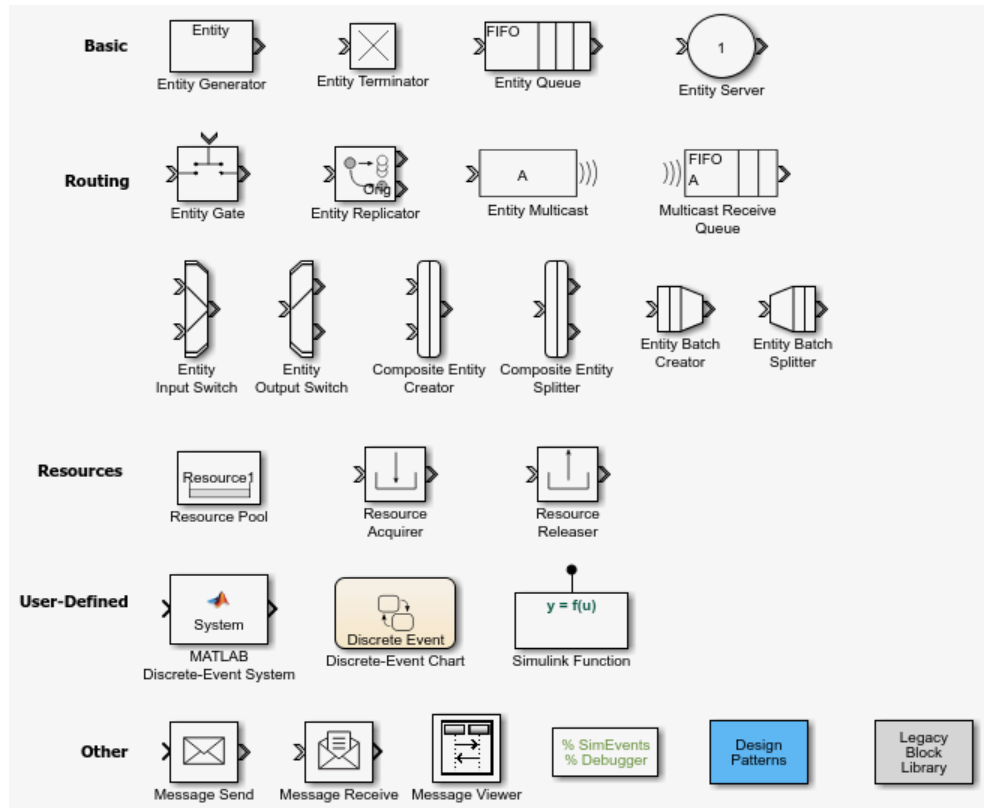
The SimEvents block library prior to R2016a is available as the Legacy Blocks sublibrary in the new SimEvents library.

Legacy models continue to work as before. To take advantage of the new SimEvents modeling environment, model using the new blocks.

Note Do not use legacy blocks in models with new SimEvents blocks and do not use new SimEvents blocks in legacy models.

Block Icons: Create presentation-quality models using new SimEvents blocks with improved icons

The new SimEvents block library has distinctive block icons that help you identify the functionality of the block.



In addition, output ports for statistics no longer appear on the block. Instead, when you select one or more statistics to display, a vertical line appears on top of the block that contains each statistic port you request.

New SimEvents Examples

See the SimEvents Examples tab for new SimEvents examples.

R2015b

Version: 4.4.1

Bug Fixes

R2015a

Version: 4.4

New Features

Bug Fixes

Resource management blocks to define, acquire, and release resources

Resource Pool, Resource Acquire, and Resource Release are new blocks that let you define and manage resources for SimEvents applications. For more information, see Model with Resources.

See “Updated examples” on page 17-2 for a list of examples updated to use these blocks.

Data type control of entity attributes

You can now set entity attributes to any built-in data type. For examples, see “Updated examples” on page 17-2.

API for integration with custom visualization

You can create an observer that attaches to a model and monitors entities and events. For more information, see Interface for Custom Visualization.

The example `Using Custom Visualization for Entities` shows how you can use the API to create applications that let you visualize entities and events as they occur in a model.

Updated examples

These examples were updated to use the new resource blocks.

Name	Model
Sharing Resources in a Batch Production Process	sedemo_batch_production
Modeling a Kanban Production System	sedemo_kanban_system
Resource Allocation from Multiple Pools	sedemo_resource_allocation

These examples were updated to take advantage of built-in data type support.

Name	Model	Change
Customizing Entities with Multidimensional Data	sedemo_nd_attributes	Original: Image data (uint8) was changed to double. Change: Kept original data type (uint8).
Controlling Entity Flow Using Release Gates	sedemo_triggergate_v_val uechangeagate	Original: Explicit Data Type Conversion block was used to convert output (int8) of Limited Counter block to double before feeding to Release Gate. Change: Removed data type conversion block.

Name	Model	Change
Bandwidth-Limited Communication Channel Modeling	sedemo_video_streaming	Issue: Video data (uint8) is an entity attribute that was converted to double. Change: Kept original data type (uint8).

R2014b

Version: 4.3.3

Bug Fixes

Compatibility Considerations

Models Using SimEvents Blocks from a Release Prior to V4.0 No Longer Run

Models created in releases prior to V4.0 (R2011b) or that contain blocks from releases prior to V4.0 (R2011b) can no longer run.

Note Starting in R2015a, you will no longer be able to load these models.

Compatibility Considerations

Use the `seupdate` command to convert the model to a later version.

R2014a

Version: 4.3.2

Bug Fixes

R2013b

Version: 4.3.1

Bug Fixes

R2013a

Version: 4.3

New Features

Drag-and-drop block insertion and one-click connection for entity lines

In R2013a, SimEvents adds capabilities that enable you to quickly insert blocks and connect entity lines.

Drag-and-drop block insertion

To add a new block to an existing entity line, you can now drag the block from the library directly into the existing line. SimEvents automatically inserts the block and connects entity lines. For more information, see [Insert Blocks](#).

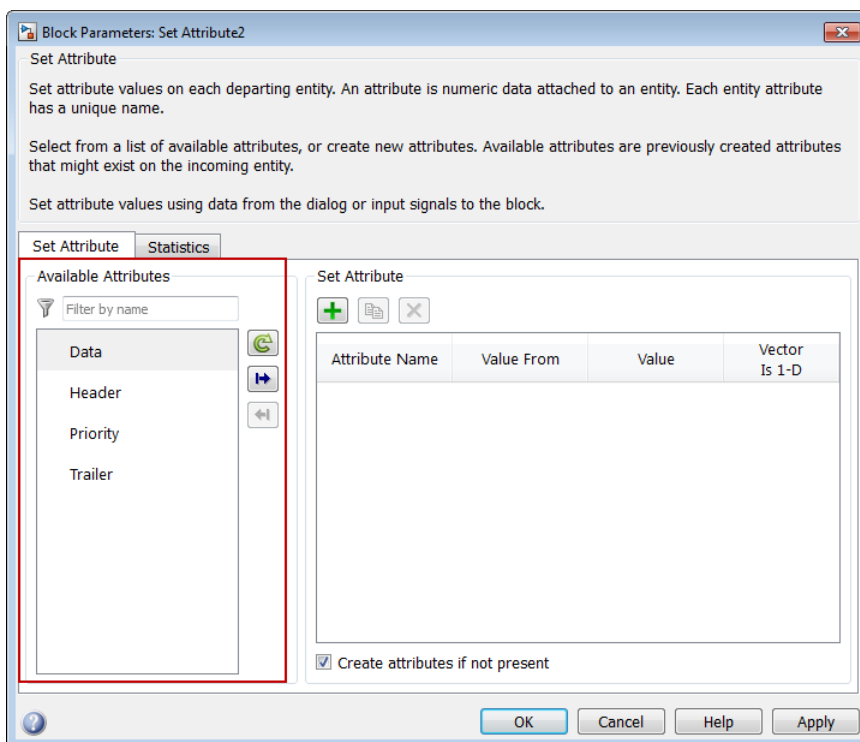
One-click entity line connection

You can now connect an entity line between two blocks with a mouse click. To connect an entity line, select the source block. **Ctrl**+click the destination block. SimEvents connects the blocks. If necessary, the software also routes the connecting line around intervening blocks or lines. For more information, see [Connect Blocks](#).


Available Attributes list for Set Attribute and Get Attribute blocks

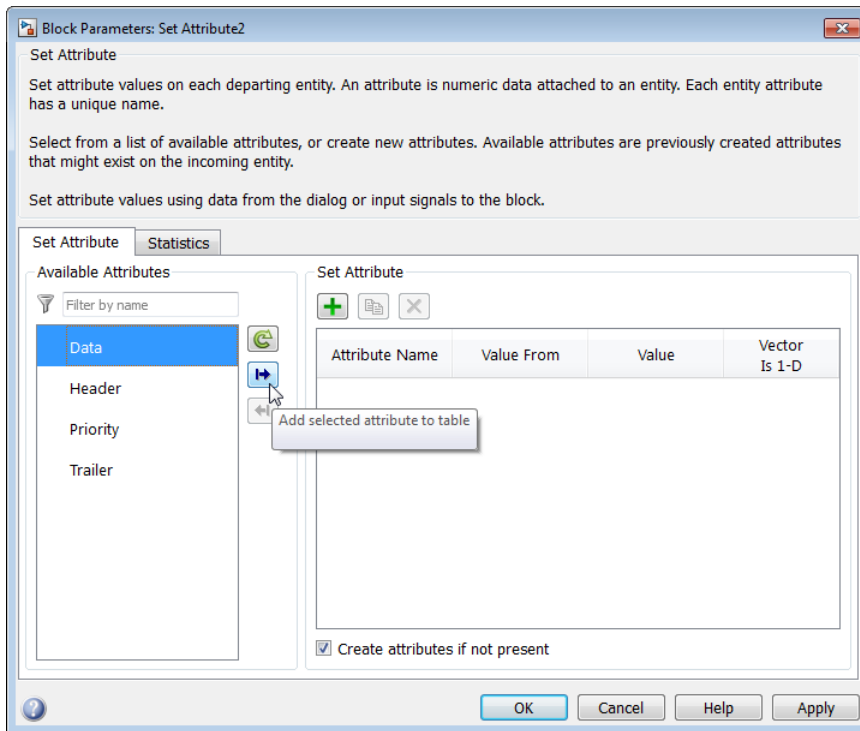
For the Set Attribute and Get Attribute blocks, you can now select attributes that you want to access on the incoming entity from a list. In previous releases, you manually specified existing attributes that you wanted to access.

The new list—**Available Attributes**—displays attributes from entity paths entering the block.

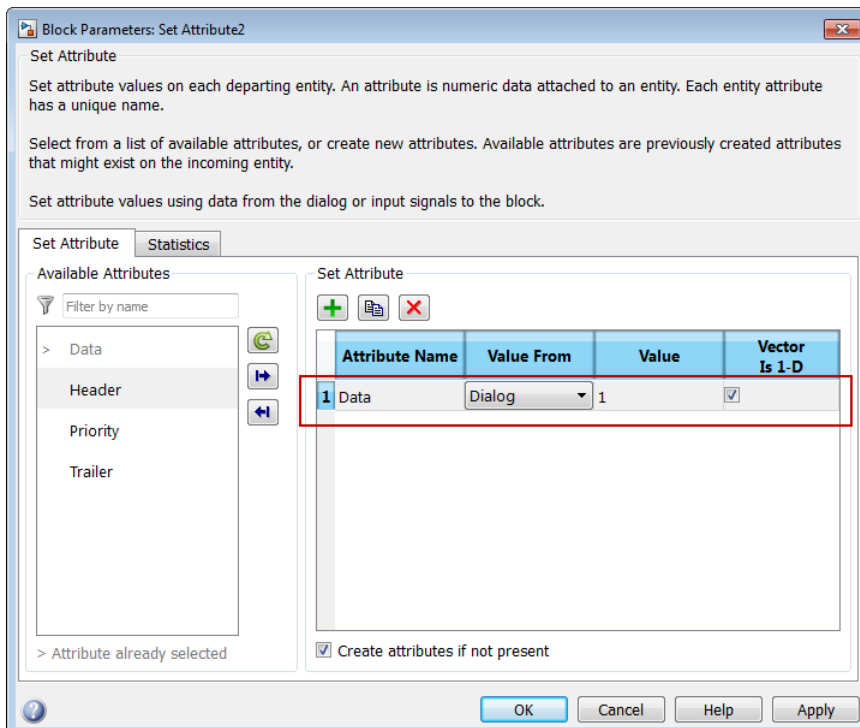


If no attributes exist on entity paths entering the block, the list is empty.

To access an attribute on the incoming entity, select the attribute in the list and click the **Add selected attribute to table** button, .



The selected attribute moves to the **Set Attribute** table.



For more information, see Set Attribute and Get Attribute.

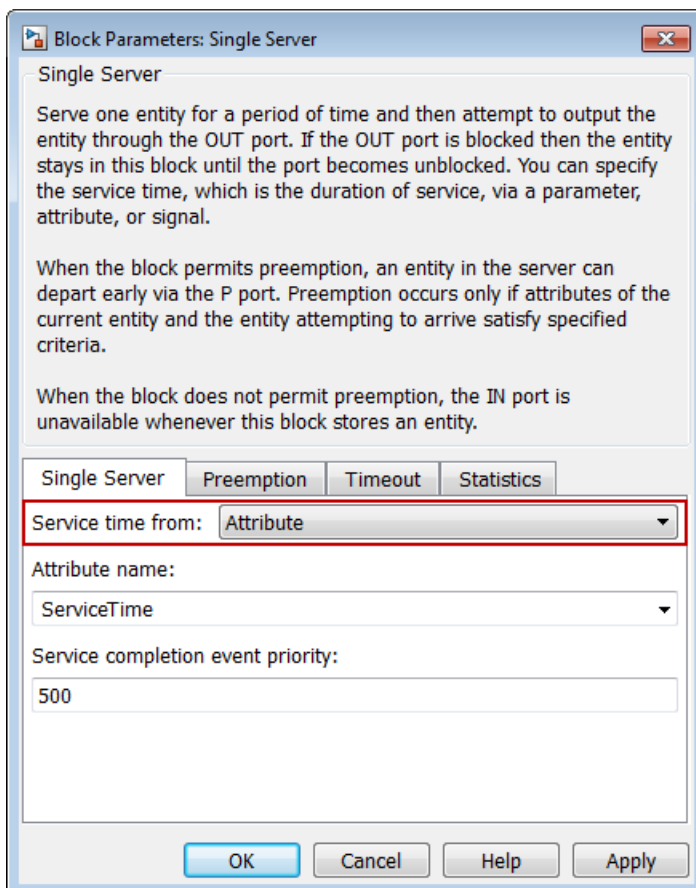
Entity attribute names visible in SimEvents blocks while editing the model

Some SimEvents blocks can use entity attributes as parameter values. To specify attributes for such blocks to use, in the block dialog box you can now select attributes on incoming entity paths from a list. In previous releases, you manually entered the attribute name.

The enhancement applies to the following blocks that can use attributes as parameter values:

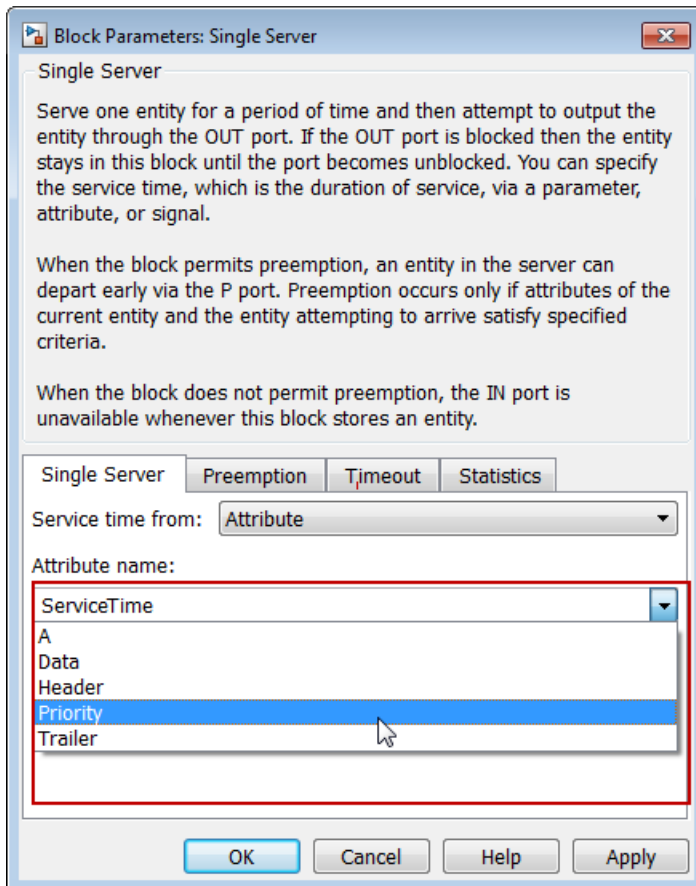
- Infinite Server
- N-Server
- Single Server
- Attribute Scope
- X-Y Attribute Scope
- Output Switch
- Schedule Timeout

Each of these blocks has a parameter that specifies that the next parameter in the dialog box uses an attribute. In this example, the Single Server block **Service time from** parameter is set to **Attribute**.



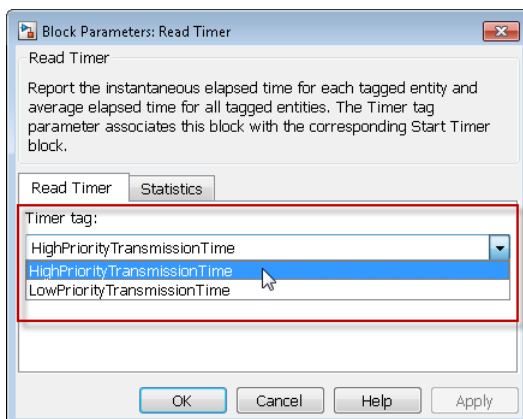
Each of the listed blocks has a parameter that you set to `Attribute` or `From Attribute`, similar to the Single Server block **Service time from** parameter.

The **Attribute name** list displays attributes that you can access on the incoming entity.



Timing tags visible in SimEvents blocks while editing the model

Some SimEvents blocks access timing tags from incoming entity paths. To specify a timing tag for the block to access, you can now select from a list of available timing tags.



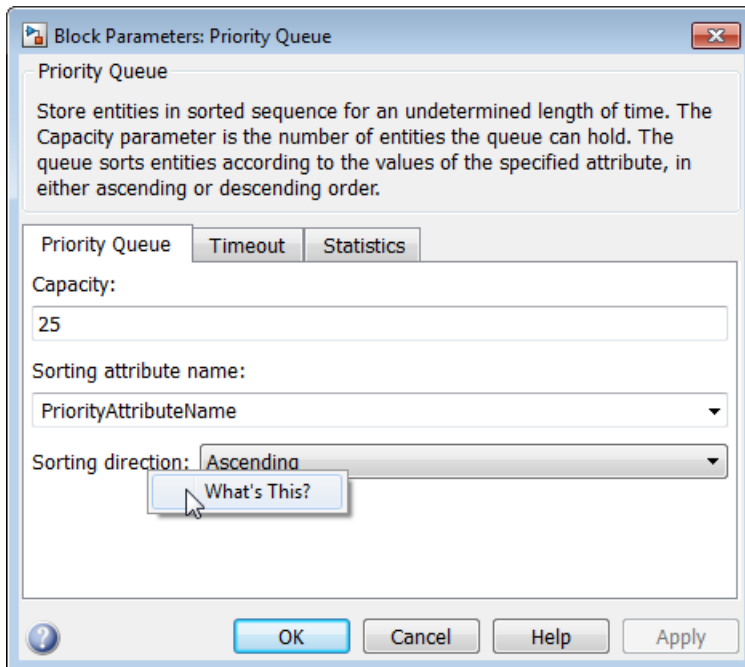
In previous releases, you manually entered the timing tag.

The enhancement applies to the following blocks:

- Cancel Timeout
- Schedule Timeout
- Read Timer
- Start Timer

Context-sensitive help on block dialog boxes

SimEvents block dialog boxes now have context-sensitive help. To access context-sensitive help, right-click a parameter label and select **What's This?**



R2012b

Version: 4.2

New Features

Bug Fixes

Port Data Types display option that provides in-model view of SimEvents entity types and structures

When you compile your SimEvents model, you can now visually inspect the progress of each entity type in the model, and any changes to its structure, as it progresses through the model.

An entity type is the identification tag associated with any block in your SimEvents model that generates new entities. Each new entity that originates in such a block receives this tag.

If an entity has acquired attribute, timeout or timer data, the entity has a structure. If entities pass through multiple Set Attribute blocks that attach attribute data, or if you use an Entity Combiner block to combine data-carrying entities, entity structures might quickly become complex. Therefore, before simulation, you might want to inspect entity structures at various points in the model.

To help you to inspect entity structures, SimEvents uses the existing **Port Data Types** display option in Simulink. If you enable this option for your model, the software shows all data types in the model, including SimEvents entity types. When you hover your cursor over an entity type label, the software displays a tooltip that shows the structure of the entity type at that point in the model.

To show labels for entity types, in the model editor, select **Display > Signals & Ports > Port Data Types**.

Model update and improvement checks in Upgrade Advisor and Model Advisor

The existing Model Advisor checks for SimEvents are renamed.

Old Name	New Name
Check model for legacy SimEvents blocks	Check model and local libraries for legacy SimEvents blocks
Check SimEvents model for implicit event duplication	Check for implicit event duplication caused by SimEvents blocks

For more information about:

- The SimEvents Model Advisor checks, see Model Advisor and Upgrade Advisor Checks in the SimEvents documentation.
- Running the Model Advisor for your model, see Consult the Model Advisor in the Simulink documentation.

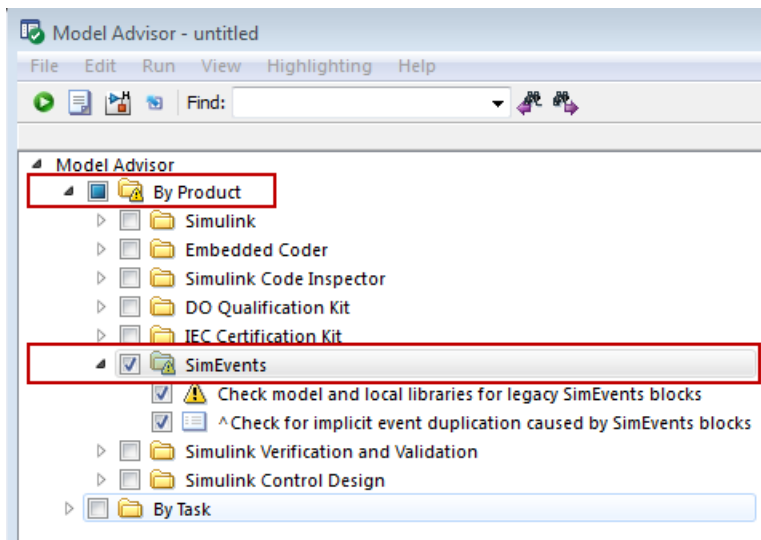
Also, one of the ways in which you use the Model Advisor to access the SimEvents checks has changed.

The Model Advisor groups all checks in two ways:

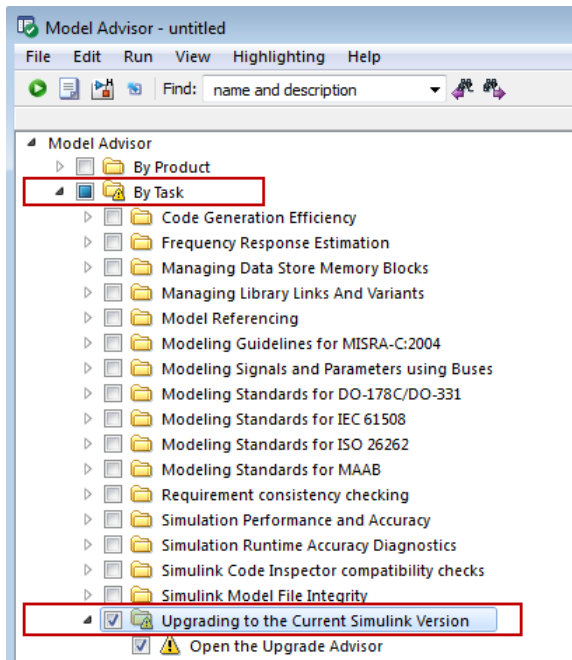
- By product
- By task

As in the previous release, you can access the SimEvents checks from either group. However, when you use the **By Task** group to access the SimEvents checks, you see different behavior.

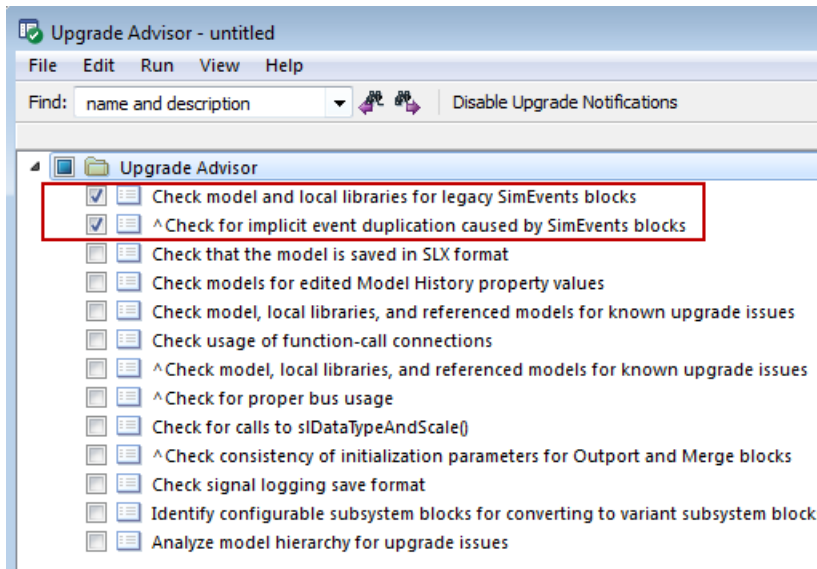
To access the SimEvents checks from the **By Product** group, select **Model Advisor > By Product > SimEvents**.



To access the SimEvents checks from the **By Task** group, first select **Model Advisor > By Task > Upgrading to the Current Simulink Version**.



Upgrading to the Current Simulink Version now contains a single task, **Open the Upgrade Advisor**. If you choose this option, a new Simulink tool, the Upgrade Advisor, opens.

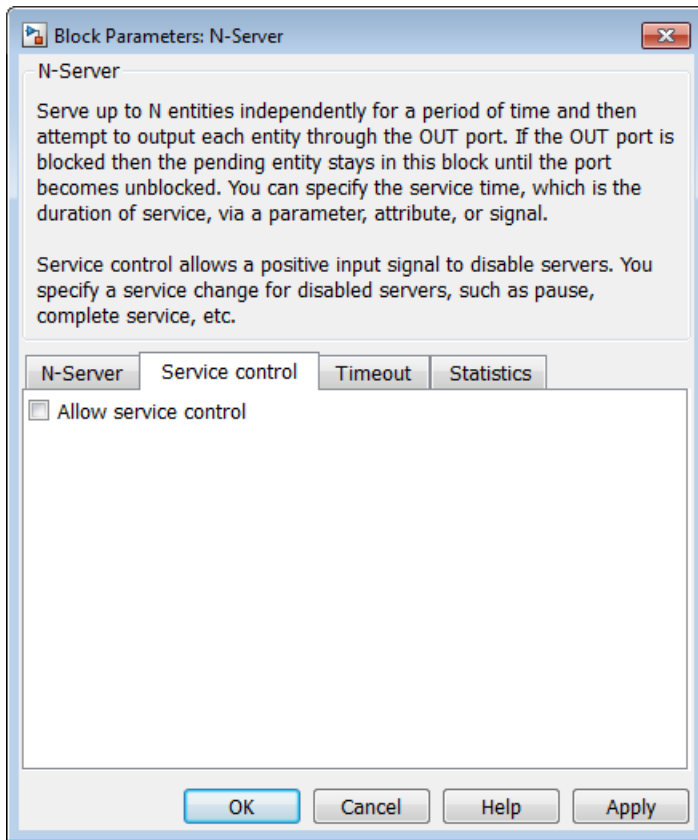


The graphic shows the SimEvents checks in Upgrade Advisor. In the previous release, these checks were directly listed under **Upgrading to the Current Simulink Version** in the Model Advisor.

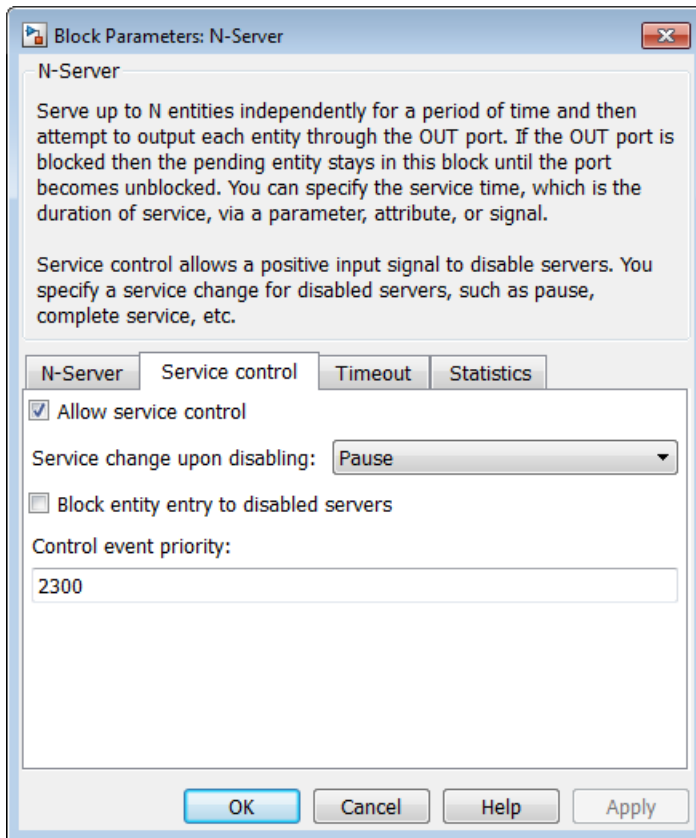
For more information about the Upgrade Advisor, see Consult the Upgrade Advisor in the Simulink documentation.

N-Server block options to pause, force service completion, and monitor server occupancy

The dialog box for the N-Server block has a new tab, **Service control**. The service control feature lets you use an input signal to disable servers in the block, and, while they are disabled, apply a service change such as pausing or forcing service completion.



When you select **Allow service control**, the block dialog box displays additional options. You can use these options to specify the behavior of disabled servers.



By default, **Service change upon disabling** is set to **Pause**. When you click **OK**, the N-Server block displays an additional input port, **pause**.



When you input a positive signal to the **pause** port, the software disables all servers in the block. While this input signal remains positive, any occupied servers retain their entities and the software pauses the remaining service time for each server. When the signal at the input port becomes nonpositive, each server resumes service.

You can also set **Service change upon disabling** to **Force complete**. In this case, when you click **OK**, the N-Server block displays an additional input port, **complete**.



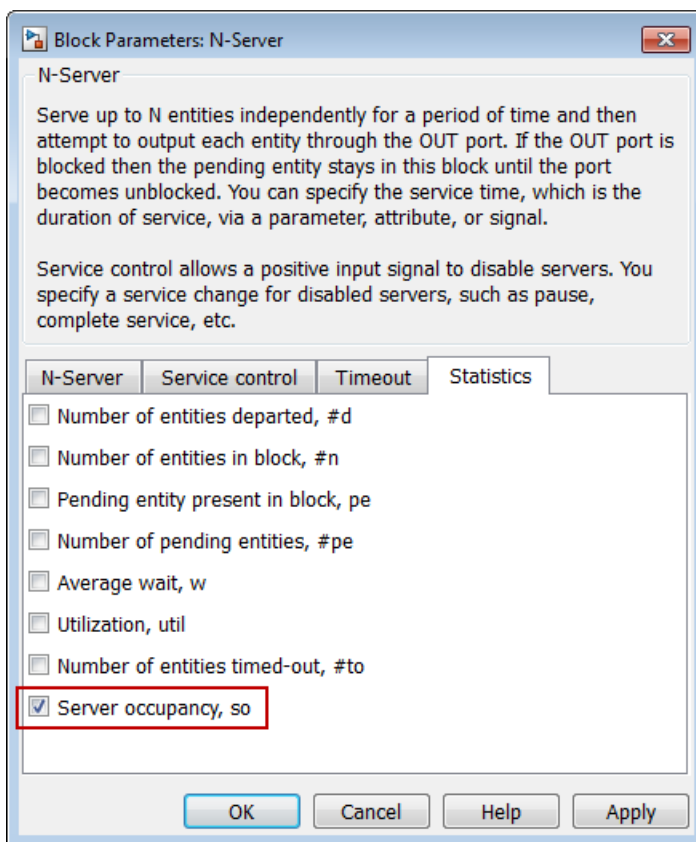
When the signal entering the **complete** port becomes positive, the software:

- Disables all servers.
- Immediately completes service in all occupied servers.
- Resets the remaining service time in all servers.

If no blockage exists at the entity output port of the N-Server block, entities immediately advance from occupied servers to downstream blocks. When the signal at the input port becomes nonpositive, normal behavior of the N-Server block resumes.

If you select **Block entity entry to disabled servers**, while the signal at the **pause** or **complete** port is positive, servers cannot accept entities.

To help you to monitor server occupancy during the simulation, the **Statistics** tab of the N-Server block has a new parameter.



Selecting **Server occupancy, so** adds a signal output port labeled **so** to the block.



The **so** port outputs a vector of values. Each element of the vector indicates the status of the corresponding server in the N-Server block. If a server is unoccupied, the value of the corresponding vector element is 0. If a server is occupied, the vector element has a value of 1.

Event-based signal port connecting directly to Simulink Constant block without Gateway block

You can connect the signal input port of a SimEvents block directly to a Simulink Constant block that has a constant (Inf) sample time. In previous releases, to make this connection, you needed to insert a Timed to Event Signal block from the SimEvents **Gateways** library between the Constant block and the SimEvents block.

To connect any other time-based source to a SimEvents block, a gateway block is still required.

New examples

In this release, the following examples were added:

Name	Model
Supervisory Control of a Conveyor Belt System	sedemo_airport_conveyor

R2012a

Version: 4.1

New Features

Bug Fixes

Compatibility Considerations

New Configuration Parameter for Prevention of Implicit Event Duplication

A new configuration parameter, **Prevent duplicate events on multiport blocks and branched signals**, has been added in this release. This parameter eliminates a condition known as multifiring that might exist in SimEvents models with particular block configurations. Multifiring is an implicit result of the way that the software executes such block configurations and causes duplication of events when the model is simulated. You can find **Prevent duplicate events on multiport blocks and branched signals** in the Configuration Parameters dialog box of your model.

Compatibility Considerations

Compatibility Considerations

If you use the R2012a software to open a SimEvents model created in a release prior to R2011b, you see the configuration parameter **Prevent duplicate events on multiport blocks and branched signals** and can select it, but the configuration parameter does not function. To enable the functionality of this parameter in a model created in a release prior to R2011b, first use the `seupdate` function to migrate the model to the latest version of the software. When the migration process is complete, in the **SimEvents** pane of the Configuration Parameters dialog box, select the **Prevent duplicate events on multiport blocks and branched signals** check box.

Support for Fixed-Step Solvers

Support for fixed-step solvers in SimEvents models has been added in this release. In previous releases, SimEvents models could be used with only variable-step solvers.

Compatibility Considerations

Compatibility Considerations

- If you do not select the configuration parameter **Prevent duplicate events on multiport blocks and branched signals** in your model, you cannot set the solver type to Fixed-step.
- If you use the R2012a software to open a SimEvents model created in a release prior to R2011b, you see the configuration parameter **Prevent duplicate events on multiport blocks and branched signals** and can select it, but the configuration parameter does not function. To enable the functionality of this parameter in a model created in a release prior to R2011b, first use the `seupdate` function to migrate the model to the latest version of the software. When the migration process is completed, in the **SimEvents** pane of the Configuration Parameters dialog box, select the **Prevent duplicate events on multiport blocks and branched signals** check box.

Enhanced Migration Workflow

Model Advisor Checks for SimEvents Models

The Simulink Model Advisor now checks your SimEvents model for certain issues that can be resolved either by migrating the model to the latest version of the software or by enabling functionality that is available in the latest version. You can use these Model Advisor checks as a starting point in the migration workflow for your model.

Use the Model Advisor checks for SimEvents to:

-
- Check whether your model contains legacy blocks. Legacy blocks are blocks from a version of SimEvents prior to 4.0 (R2011b). If you have legacy blocks in your model, your model does not have the benefit of feature updates and modeling syntax changes in the latest version of the software. If this Model Advisor check detects that your model contains legacy blocks, it instructs you to use the `seupdate` function to migrate your model to the latest version of the software.
 - Check whether you have selected the configuration parameter **Prevent duplicate events on multiport blocks and branched signals** in your model. **Prevent duplicate events on multiport blocks and branched signals** resolves issues related to event duplication in SimEvents models. If the Model Advisor check detects that you have not selected this configuration parameter in your model, you can instruct the Model Advisor to select it.

Visual Appearance of Legacy Blocks

Legacy blocks are blocks from a version of SimEvents prior to 4.0 (R2011b). If your model contains legacy blocks, the software now visually labels these blocks so that when you view the model in the model editor, you can easily identify them. The software marks legacy blocks by placing an orange label, **legacy**, in the lower left-hand corner of the block. This visual marker helps you assess whether you need to use the `seupdate` function to migrate your model to the latest version of SimEvents.

Enhanced `seupdate` Migration Utility

The function `seupdate` was introduced in R2011b. The `seupdate` function is a migration utility that migrates a SimEvents model from a release prior to R2011b to the current release. This migration utility updates SimEvents and Simulink blocks. As necessary, it might also add gateway blocks to indicate transitions between event-based and time-based computational components of your model.

In this release, the following enhancements have been made to the `seupdate` function:

- Prior to model migration, software automatically backs up your model and associated libraries.
- Enhanced update report identifies conditions in your model that cause migration to fail and provides you with instructions to resolve the failure.
- Prior to running `seupdate`, you no longer have to open your model. The software automatically opens the model.
- Software checks for legacy queue blocks in your model and provides you with instructions to see if simulation results produced by these queue blocks will change as a result of updating the model.

Enhanced Visibility of Impact when Saving Models at Earlier Versions

If you use the `Save as type:` drop-down list to save your model at a version of SimEvents prior to 4.0 (R2011b), you now receive a warning in the MATLAB command window. This new warning informs you that the software has replaced unsupported blocks with empty subsystem blocks and that a loss of functionality in the model is likely to result. The message provides you with instructions to resolve the problem.

The software also now marks any unsupported blocks that it replaces with the text **Replaced** so that you can easily identify the replaced blocks when you open your model.

Changes to Modeling Semantics

Gateway Blocks Support Buses

You can now connect Simulink bus signals to gateway blocks in a SimEvents subsystem. In previous releases, if you connected a bus signal to a gateway block you saw an error message.

You can also now use the following blocks from the Simulink block library to manipulate bus signals in a SimEvents subsystem:

- Bus Creator
- Bus Selector
- Bus Assignment

Compatibility Considerations

- In this release, you can connect a Simulink bus signal to only a SimEvents gateway block.
- If the SimEvents subsystem contains blocks from a previous release, you cannot connect a bus signal to a SimEvents gateway block. Before using bus signals in your model, first use the `seupdate` function to migrate your model to the latest version of the software.
- You can only use a bus signal in a SimEvents subsystem if you select the configuration parameter **Prevent duplicate events on multiport blocks and branched signals** in your model. After you use the `seupdate` function to migrate your model to the latest version of the software, you can enable this configuration parameter.

Enhanced Support for Simulink Subsystems

You can now use SimEvents blocks in Simulink Nonvirtual and Variant Subsystems, observing some specific guidelines, and in Simulink Virtual Subsystems without restriction. For more information, see SimEvents Support for Simulink Subsystems.

Compatibility Considerations

- The only type of nonvirtual subsystem in which you can place SimEvents blocks is an Atomic Subsystem. You can only place SimEvents blocks in an Atomic Subsystem if you first select the configuration parameter **Prevent duplicate events on multiport blocks and branched signals**.
- If you use the R2012a software to open a SimEvents model created in a release prior to R2011b, you see the configuration parameter **Prevent duplicate events on multiport blocks and branched signals** and can select it, but the configuration parameter does not function. To enable the functionality of this parameter in a model created in a release prior to R2011b, first use the `seupdate` function to migrate the model to the latest version of the software. When the migration process is completed, in the **SimEvents** pane of the Configuration Parameters dialog box, select the **Prevent duplicate events on multiport blocks and branched signals** check box.

Changed Use of Event Priority Parameter

When you select the configuration parameter **Prevent duplicate events on multiport blocks and branched signals** in your model, the behavior of the **Event priority** parameter of the following blocks changes.

- Event Filter
- Signal-Based Function-Call Generator
- Signal Latch

When you select the configuration parameter **Prevent duplicate events on multiport blocks and branched signals**, the software uses the **Event priority** parameter of the blocks in the preceding

list to help Simulink sort blocks in the model. For these blocks, the software no longer schedules an event that you can view on the SimEvents event calendar.

Compatibility Considerations

- You do not see the changed behavior of the **Event priority** parameter for the listed blocks unless you select the configuration parameter **Prevent duplicate events on multiport blocks and branched signals** in your model.
- If you use the R2012a software to open a SimEvents model created in a release prior to R2011b, you see the configuration parameter **Prevent duplicate events on multiport blocks and branched signals** and can select it, but the configuration parameter will not function. To enable the functionality of this parameter in a model created in a release prior to R2011b, first use the `seupdate` function to migrate the model to the latest version of the software. When the migration process completes, in the **SimEvents** pane of the Configuration Parameters dialog box, select the **Prevent duplicate events on multiport blocks and branched signals** check box.
- If you do not select the configuration parameter **Prevent duplicate events on multiport blocks and branched signals** in your model, there is no change in the behavior of the **Event priority** parameter from previous releases. In previous releases, the software used the **Event priority** of each block to determine the order in which it processed events. You can use the SimEvents event calendar to view the events that the software has scheduled.

Block Library Changes

New Blocks

The Time-Based Function-Call Generator block has been added to the Generators/Function-Call Generators library.

Compatibility Considerations

To use the Time-Based Function-Call block in your SimEvents model, use the `seupdate` function to migrate the model to the latest version of the software.

Changes to Block Parameters

An extra diagnostic parameter, **Use of signals awaiting update during function call:** has been added to the Entity Departure Function-Call Generator block. This new parameter is visible in the Block Parameters dialog box, when you set the value of the **Timing of function call f1:** parameter of the block to `Before entity departure`. This new parameter does not function. however, if you do not also select the configuration parameter **Prevent duplicate events on multiport blocks and branched signals** in your model. When you configure the block in this way to generate a function-call before an entity departs the block, the function-call event might use signal values that are still awaiting update. If this situation arises, the new diagnostic parameter **Use of signals awaiting update during function call:** specifies the action you want the software to take.

You can select the following values for this parameter:

Value of Use of signals awaiting update during function call: parameter	Software Action
none	No action. Simulation runs normally.

Value of Use of signals awaiting update during function call: parameter	Software Action
warning	Warning generated in MATLAB command window. Simulation still runs.
error	Error generated in MATLAB command window. Simulation does not run.

The default value of the **Use of signals awaiting update during function call:** parameter of the Entity Departure Function-Call Generator block is `error`.

Compatibility Considerations

The configuration parameter **Use of signals awaiting update during function call:** of the Entity Departure Function-Call Generator block is not available in versions of this block from previous releases. To use this parameter, use the `seupdate` function to migrate the model that contains the Entity Departure Function-Call Generator to the latest version of the software.

Functions Being Removed

Affected Function	What Happens When You Use the Function?	Use This Instead
<code>simeventsstartup</code>	No longer works. You see a message in the MATLAB command window to tell you that no changes are made to your model.	In the Configuration Parameters dialog box, manually choose settings appropriate for your discrete-event model.
<code>simeventsconfig</code>	Still works. Function will be removed in a future release.	In the Configuration Parameters dialog box, manually choose settings appropriate for your discrete-event model.

Demos

New Demo

The demo listed in the following table has been added in this release.

Name	Model
Effects of Communication Delays on an ABS Control System	<code>sedemo_absbrake_can.mdl</code>

Demos Removed

The demos listed in the following table have been removed in this release.

Name	Model
Asynchronous Clock Domains	<code>sedemo_async_clocks.mdl</code>
Asynchronous Execution of a Stateflow Chart	<code>sedemo_async_stateflow.mdl</code>

R2011b

Version: 4.0

New Features

Bug Fixes

Compatibility Considerations

Major Changes to Discrete-Event Modeling

This release contains a number of improvements and changes in the software:

- Enhanced modeling syntax clearly differentiates event-driven from time-driven components.
- Opt-in upgrade workflow helps migrate models to leverage enhanced modeling syntax.
- Entity and signal animation enable better insight into event-based execution.
- Improved simulation speed and memory performance scales to support large models.
- Increased support for Simulink blocks from Math and Logic libraries directly in event-driven computations.
- Support for MATLAB Function and Stateflow Chart blocks to integrate MATLAB algorithms and control logic into discrete-event models.
- New Event Filter block can be combined with Atomic Subsystem blocks to define custom event-triggered computational components.

Compatibility Considerations

To take advantage of the new modeling syntax in models and libraries built prior to V4.0 (R2011b), you must follow a conversion procedure using the new command, `seupdate`. This command:

- Runs `slupdate` to perform all upgrades related to Simulink in the models.
- Replaces SimEvents and Simulink blocks with their R2011b counterparts.
- Replaces instances of the Discrete Event Subsystem block with the Simulink Atomic Subsystem block.
- Inserts new gateway blocks to delineate the boundaries between event-based portions and time-based portions of the model.

After running `seupdate`, you might notice small changes in model behavior as a result of the refinements to the event-based modeling semantics. Verify the operation of your model and, if necessary, modify it. For further information, see [Migrating SimEvents Models](#).

Note The modeling paradigm from releases prior to R2011b continues to work. You can upgrade your models at any appropriate time. To continue to create or work with models from previous releases, use the `simevents('3')` command to access the block library from the previous release.

Modeling Syntax Enhancements

Gateway Blocks

The SimEvents library has new blocks that help delineate the boundary between event-based and time-based signals, and event-based based and time-based function-calls. The new Gateways sublibrary contains these blocks:

- Event to Timed Signal
- Event to Timed Function-Call
- Timed to Event Signal
- Timed to Event Function-Call

Input Port Styles

After you compile a model, all blocks with event-based signal inputs show an empty arrowhead to indicate that the blocks are executing in the event-based domain.

Simulink Blocks in Event-Based Computations

The following Simulink blocks can now operate directly on event-based computations. In previous releases, you had to contain these blocks in a discrete-event subsystem.

- Subset of blocks from Math Operations library
- Subset of blocks from Logic and Bit Operations library
- Atomic Subsystem block
- MATLAB Function block
- Stateflow Chart block
- Scope block
- To Workspace block
- Display block

In addition, you can connect a block from the SimEvents Signal Generator library to a supported Simulink block that has only one input port.

Atomic Subsystem and Event Filter for Better Control on Event-Based Execution

The Simulink Atomic Subsystem block replaces the Discrete Event Subsystem block. The Atomic Subsystem blocks ensures that a set of blocks participate in event-based execution. Additionally, you can use the new Event Filter block to conditionalize, suppress, and prioritize execution of Atomic Subsystem blocks participating in event-based execution.

Animation and Debugging Enhancements

The SimEvents Debugger has been updated to incorporate an animation capability that shows both entity movement and event-based signal execution. The following command facilitates animation in the debugger:

- The new `sedb.animate` command facilitates animation in the debugger. `sedb.animate` enables and disables animation, and controls the animation speed.

The debugger has several new and refined capabilities:

- Sets entity breakpoints.
- Supports multiple event calendars, one for each discrete-event system that your model might have.
- Entries for gateway, event filter, function-call generation blocks
- Head-of-queue-change event for queue blocks (`NewHeadOfQueue`).
- Sorted order for blocks is displayed. You can now view the sorted order for blocks either in the debugger or the Simulink editor. To enable the sorted order of blocks in a model, in the Simulink editor, select **Format > Block Displays > Sorted Order**.
- Debugger session function updates are:

- `sedb.enbreak`, a new command, that breaks on any operation involving a given entity.
- `sedb.blklist` lists gateway blocks and Simulink blocks.
- `sedb.detail` has an option to show signal updates.
- `sedb.evcal` supports multiple event calendars.

Note You cannot set breakpoints for Simulink controlled blocks.

For more information, see Animating Signals and Entities.

Changes in Blocks and Modeling Semantics

Change in Entity Arrivals at Empty Queue

The following blocks behave differently when an entity arrives at an empty queue, and process a new type of event on the event calendar:

- FIFO Queue
- LIFO Queue
- Priority Queue

In V4.0 (R2011b), when an entity arrives at a queue block:

- The queue stores the entity.
- The `#n` output signal, if present, is update to indicate the arrival of the entity.
- The queue block schedules a `NewHeadOfQueue` event on the event calendar. The event time is the current simulation time. When executing this event, the queue attempts to advance the entity to the next block.

This behavior does not depend on whether the queue is empty or whether the arriving entity can advance. Even if the entity can advance to the next block with no time delay, the simulation might process other operations between the arrival and departure of the entity. For example, the sample time hit registered in the `#n` signal might cause other operations to occur before the entity departs.

In earlier versions, when an entity arrived at an empty queue and advanced to the next block, the entity advanced immediately. The simulation did not process other operations between the arrival and departure of the entity. From this release forward, the `#n` output signal from the queue block has a sample time hit, with a value of 1 to indicate the arrival of the entity. The SimEvents debugger does not show a `NewHeadOfQueue` event.

Compatibility Considerations

The queue blocks available from the SimEvents library include the R2011b behavior changes. If you do not upgrade your legacy models, the models use older blocks that do not include the R2011b behavior changes.

If you upgrade your legacy models, you might observe changes in simulation behavior as a result of the queue upgrades.

- When an entity arrives at an empty queue and leaves the queue immediately, there is an additional sample time hit of the `#n` signal.

-
- When an entity arrives at an empty queue, there is the potential for other operations to occur between the arrival and departure of the entity. The rules described in Processing Sequence for Simultaneous Events are still in effect. However, when an empty queue has an arrival followed by a departure at the same value of the simulation clock, sequences described there as arbitrary might be different between this version and earlier versions.
 - The new `NewHeadOfQueue` event changes the way the SimEvents debugger steps through your simulation. For example, if you execute `sedb.step` 10 times and those steps include a `NewHeadOfQueue` event, the 10th step reaches a different point in the simulation compared to an earlier version.

Arrival of Entity Placed at Head of Queue

When an entity arrives at a queue and becomes the new head of queue entity, the following blocks behave differently from previous releases. These blocks now process a new type of event on the event calendar:

- FIFO Queue
- LIFO Queue
- Priority Queue

In V4.0 (R2011b), when an entity arrives at a queue block:

- The queue stores the entity.
- The `#n` output signal, if present, is updated to indicate the arrival of the entity.
- The queue block schedules a `NewHeadOfQueue` event on the event calendar. The event time is the current simulation time. When executing this event, the queue attempts to advance the entity to the next block.

This behavior does not depend on whether the queue is empty or whether the arriving entity can advance. Even if the entity can advance to the next block with no time delay, the simulation might process other operations between the arrival and departure of the entity. For example, the sample time hit in the `#n` signal might cause other operations to occur before the entity departs.

In earlier versions, when an entity arrived at a queue and did advance to the next block, the entity advanced immediately. The simulation did not process other operations between the arrival and departure of the entity. The `#n` output signal from the queue block had a sample time hit, with a value of 1, to indicate the arrival of the entity. The SimEvents debugger did not show a `NewHeadOfQueue` event.

Compatibility Considerations

Only the queue blocks available from the SimEvents library include the R2011b behavior changes. If you do not upgrade your legacy models, the models use older blocks that do not include these behavior changes.

If you upgrade your legacy models, you might observe changes in simulation behavior:

- When an entity arrives at a queue and leaves the queue immediately, there is an additional sample time hit of the `#n` signal.
- When an entity arrives at a queue, there is the potential for other operations to occur between the arrival and departure of the entity. The rules described in Processing Sequence for Simultaneous Events are still in effect. However, when a queue has an arrival followed by a departure at the

same value of the simulation clock, sequences described there as arbitrary might be different between this version and earlier versions.

- The new `NewHeadOfQueue` event changes the way that the SimEvents debugger steps through your simulation. For example, if you execute `sedb.step` 10 times and those steps include a `NewHeadOfQueue` event, the 10th step reaches a different point in the simulation compared to an earlier version.

Change to Sorted Order of Hybrid Systems

SimEvents blocks now simulate in the appropriate order in hybrid systems. In previous releases, SimEvents blocks were scheduled to simulate at the end of a sorted list.

Discrete Event Signal to Workspace Block Records Initial Value

When the Discrete Event Signal to Workspace block creates a workspace variable, it includes the initial value of the signal, if any. In earlier versions, the workspace variable did not include the initial value of the signal.

Compatibility Considerations

One or more initial values are now recorded. There is at least one more data entry at time 0.

Statistical Output Signals Have Updates Throughout Simulation

When a SimEvents block has built-in capabilities to produce statistical output signals, such as the number of entities that have departed from the block, the corresponding parameter in the block dialog box has a check box to indicate on and off. The blocks available from the SimEvents library no longer offer the `Upon stop` or `pause` option.

Compatibility Considerations

If you use the `seupdate` function to update a legacy model in which a SimEvents block has a parameter set to `Upon stop` or `pause`, the function sets the corresponding parameter in the replacement block to `On`. As a result, the signal has sample time hits at simulation times that do not necessarily correspond to the end of the simulation or a time at which you pause the simulation.

Tip If you want to create a variable in the MATLAB workspace containing the final value of the signal, use the To Workspace or Discrete Event Signal to Workspace block. In either block, set the **Limit data points to last** parameter to 1.

Configuration Parameter Race Condition Detection Changes

The default values of the **SimEvents > Diagnostics** parameters have changed. You can now create better models with better race condition checking.

Parameter	New Value	Old Value
Attribute output delayed relative to entities	error	warning

Parameter	New Value	Old Value
Response to function call delayed relative to entities	error	warning
Statistical Output Delayed Relative to Entities	none	warning

Related to these changes, the `simeventsconfig` has been updated. The `simeventsconfig` command now displays a list of all the parameter values that it can change, and then prompts you to enable these changes.

Compatibility Considerations

Because of the more stringent default race conditions diagnostics, existing models that generated warnings now generate errors and do not work. Modify your models as appropriate.

Block, Event, and Entity Identifiers

When you use `seupdate` to update a model from a previous release, the block, event, and entity identifiers might change from those in the original model.

Compatibility Considerations

If you have existing scripts or applications that use block, event, or entity identifiers in the debugger, update them to use the new identifiers. To determine the new identifiers, manually debug the model with the debugger.

SimEvents Support for Variant Subsystems

A model cannot contain a Variant Subsystem block that contains SimEvents blocks. In a discrete-event system, a Variant Subsystem block can only contain Simulink blocks that the SimEvents software supports. See [Blocks That Support Event-Based Input Signals](#) for a list of these blocks.

Discrete Event Signal to Workspace Block Can Reside in Atomic Subsystem

The Discrete Event Signal to Workspace block available from the SimEvents library can reside in any atomic subsystem. In previous releases, the block had the restriction described in “Discrete Event Signal to Workspace Block Clarifies Timing” on page 29-4.

Set Attribute and Get Attribute Blocks Can Process at Most 32 Attributes

The Set Attribute and Get Attribute blocks can set and get, respectively, a maximum of 32 attributes per block instance.

Compatibility Considerations

To set more than 32 attributes, connect two or more Set Attribute blocks. Configure each block to set at most 32 attributes.

To get more than 32 attributes, connect two or more Get Attribute blocks. Configure each block to get at most 32 attributes.

Attribute Propagation Changes

A named attribute now has fixed dimensions and complexity within a discrete-event system. Multiple discrete-event systems within a model can have their own definitions of the same named attribute. In previous releases, attribute value characteristics were consistent across the whole model.

Parameter Handling Change for Routing and Entity Management Blocks

For blocks in the Routing and Entity Management libraries, the **Number of entity input ports** and **Number of entity output ports** parameters are no longer restricted to literal values. Instead, the parameter value can be any of the following:

- Literal value
- Name of a workspace variable
- Expression that the block evaluates

Change in Execution Order of Simultaneous Events with Same Priorities

In both current and previous releases, the execution order of simultaneous events with the same priorities has changed.

For example, if the configuration parameter **SimEvents > Execution order** of your model is configured to **Arbitrary** for the processing of simultaneous events, and this model has multiple entity generator blocks, each generator generates entities at the start of simulation with the same priority. In previous releases, the entity generator block that generated the first entity might be different.

Compatibility Considerations

To control the execution order of simultaneous events, in the block configuration, change the priority values of the events so that they are all different.

Model Acceleration Changes

Rapid Simulation Support

SimEvents software does not support rapid simulation after you perform the upgrade process using the `seupdate` function. In this release, performance was greatly enhanced, removing the need for the performance enhancement of rapid simulation.

New Blocks Do Not Support Accelerator Mode

The blocks that support the new paradigm do not support Accelerator Mode.

Compatibility Considerations

If Accelerator Mode is essential to your work with SimEvents models, you should not use the `seupdate` function to update your SimEvents models and the custom libraries on which those models depend. Instead, you should continue using the older set of SimEvents blocks, from Releases R2009b through R2011a.

Block Library Changes

Updated SimEvents Library

This release provides an updated SimEvents library. After you convert your model with `seupdate`, you can use blocks from this library after you convert your model. The `simeventslib` command displays the updated library.

If you want to continue to use your model from a previous release and add blocks to it, use `simevents('3')`.

Note You cannot use blocks from the updated library in a model from a previous release. You also cannot use blocks from a previously released SimEvents library in an R2011b model.

Library Changes

The Event Translation, Probes, and SimEvents User-Defined Functions sublibraries no longer exist. The contents of these sublibraries have either been removed or moved to other sublibraries.

Libraries	Current Status
Event Translation	The blocks in this library have been replaced. See “Blocks Being Removed” on page 24-9.
Probes	Entity Departure Counter has moved to the Entity Management sublibrary.
SimEvents User-Defined Functions	Attributes Function has moved to the Attributes sublibrary.

New Blocks

New gateway blocks (“Gateway Blocks” on page 24-2) and the Event Filter block have been added to the Gateways and the SimEvents Ports and Subsystem libraries, respectively.

Blocks Being Removed

Affected Blocks	What Happens When You Use the Block?	Use This Instead	Compatibility Considerations
<ul style="list-style-type: none">Discrete Event SubsystemDiscrete Event InportDiscrete Event OutportSubsystem Configuration	<p>If the model contains only blocks from earlier versions, the simulation still runs.</p> <p>If the model contains any SimEvents blocks from this version, the simulation produces errors.</p>	Atomic Subsystem, with inports potentially connected to Event Filter blocks	To learn how to update your legacy models using the <code>seupdate</code> function, see <i>Migrating SimEvents Models</i> .

Affected Blocks	What Happens When You Use the Block?	Use This Instead	Compatibility Considerations
<ul style="list-style-type: none"> Entity Departure Event to Function-Call Event Entity Departure Event to Function-Call Event Generator 	<p>If the model contains only blocks from earlier versions, the simulation still runs.</p> <p>If the model contains any SimEvents blocks from this version, the simulation produces errors.</p>	Entity Departure Function-Call Generator	To learn how to update your legacy models using the <code>seupdate</code> function, see “Compatibility Considerations” on page 24-2.
<ul style="list-style-type: none"> Signal-Based Event to Function-Call Event Signal-Based Event to Function-Call Event Generator 	<p>If the model contains only blocks from earlier versions, the simulation still runs.</p> <p>If the model contains any SimEvents blocks from this version, the simulation produces errors.</p>	Signal-Based Function-Call Generator	Replace block

Functions Being Removed

Affected Function	What Happens When You Use the Function?	Use This Instead	Compatibility Considerations
<ul style="list-style-type: none"> <code>simeventsstartup</code> 	Still works. You see a warning message about future removal.	<code>simeventsconfig</code>	Replace existing instances of <code>simeventsstartup</code> with <code>simeventsconfig</code> .

Demos

Updated Demos in the Product

The SimEvents demos have been updated to V4.0. The demos listed in the table have been removed.

Name	Model
Anti-Lock Braking System (ABS) Model Using CAN Communications	<code>sedemo_absbrake_can.mdl</code>
Anti-Lock Braking System (ABS) Model with Queuing Delay	<code>sedemo_absbrake_delay.mdl</code> , <code>sedemo_absbrake_nodelay.mdl</code>
Time-Driven and Event-Driven Addition	<code>sedemo_add_num_in_two_queues.mdl</code>
Astable Multivibrator Circuit	<code>sedemo_astable_multivibrator.mdl</code>
Event Priorities	<code>sedemo_event_priorities.mdl</code>
FIFO Buffer: Architectural Model	<code>sedemo_fifo_architectural.mdl</code>
Rate-Based Shared Processor	<code>sedemo_rate_based_proc.mdl</code>
Shared Access Communications Media	<code>sedemo_shared_access_media.mdl</code>

Name	Model
Shared Communication Buffer Management	sedemo_shared_buffer_mgmt.mdl
Tank Filling Station	sedemo_tank_filling_station.mdl
Batch Discrete-Event Simulations Using Rapid Simulation Target	sedemo_rsim_resource_alloc_m

Updated Demos and Examples in the Documentation

SimEvents examples in the documentation have been updated to V4.0.

R2011a

Version: 3.1.2

New Features

Changes in Menu of Scope Figure Window

In the figure window that corresponds to any SimEvents scope block, the following menu options no longer appear:

- **File > Page Setup**
- **File > Print Setup**

Instead, if you select the **File > Print Preview** menu option, a dialog box opens with similar page and print setup functionality.

R2010b

Version: 3.1.1

Bug Fixes

R2010a

Version: 3.1

New Features

Bug Fixes

Compatibility Considerations

Block-Based Breakpoints in Debugger

With the SimEvents debugger, you can investigate the behavior of particular blocks using block-based breakpoints. After you establish a breakpoint on a block, the debugger suspends the simulation when that block is about to perform certain operations. For details, see these resources:

- `sedb.blkbreak` function reference page
- Defining a Breakpoint
- Using Breakpoints During Debugging
- Block Operations Relevant for Block Breakpoints

Block Operations Information in Debugger

These blocks now include their operations in the simulation log:

- Discrete Event Signal to Workspace
- Instantaneous Event Counting Scope
- Signal Scope
- X-Y Signal Scope

These blocks now appear in the output of the `sedb.blklist` function and are valid as inputs to the `sedb.blkinfo` function:

- Discrete Event Signal to Workspace
- Discrete Event Subsystem
- Instantaneous Event Counting Scope

The `sedb.blklist` function sorts its Command Window output and cell array output by block names instead of by block identifiers.

Compatibility Considerations

If you have code that manipulates or indexes into the output cell array from `sedb.blklist`, you might need to update the code to reflect new rows or a different sequence of rows.

Changes in Behavior of Pending Entity Signals

The blocks in the following table have an optional **pe** or **#pe** signal output port. The signals at these ports provide information about pending entities in the block. The port behaviors are now simpler and more consistent across the various blocks.

Block	Has Optional pe Port	Has Optional #pe Port
Event-Based Entity Generator	Yes	No
Infinite Server	Yes	Yes
N-Server	Yes	Yes

Block	Has Optional pe Port	Has Optional #pe Port
Output Switch	Yes	No
Single Server	Yes	No
Time-Based Entity Generator	Yes	No

In V3.1 (R2010a), a pending entity is an entity that has tried and failed to depart from the block in which the entity resides.

When a block produces a **pe** output signal, the signal has an update (that is, a sample time hit) whenever there is a change in the set of pending entities that the block stores. The signal value is:

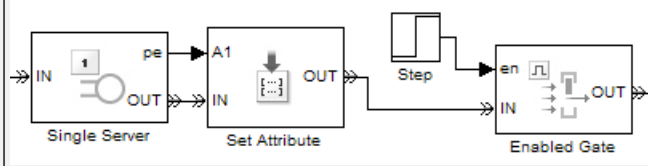
- 1, if the block stores one or more pending entities
- 0, if the block does not store any pending entities

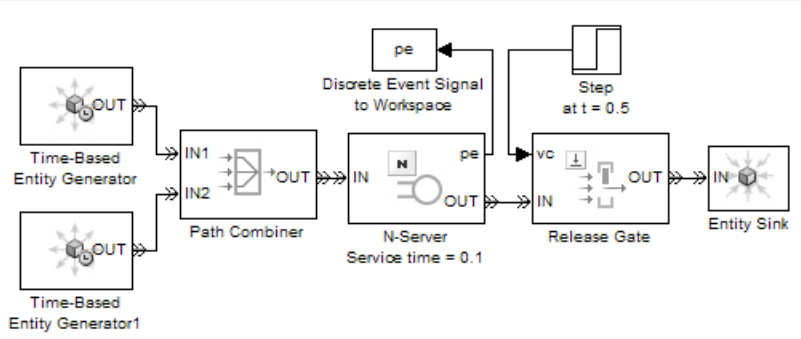
When a block produces a **#pe** output signal, the signal has an update whenever there is a change in the set of pending entities that the block stores. The signal value is the number of pending entities that the block stores.

Compatibility Considerations

If your models use the **pe** or **#pe** signal to control simulation behavior, perform computations, or return results, your models might behave differently. The table summarizes the behavioral changes most likely to affect your models. For typical uses of the **#pe** signal, in which redundant sample time hits with the same value do not matter, the behavioral changes do not change the simulation results.

Affected Blocks	Change in Behavior
<ul style="list-style-type: none"> • Event-Based Entity Generator • Infinite Server • N-Server • Output Switch • Single Server • Time-Based Entity Generator 	<p>The pe signal does not have a sample time hit to reflect that an entity departs the first time it tries to depart. Such an entity is not a pending entity because it does not fail to depart.</p> <p>The same information is true for #pe in blocks that offer this port.</p> <p>Example</p> <p>In V3.1 (R2010a), the next model produces a plot containing no points because no entity fails to depart from the server.</p> <div data-bbox="516 625 1192 919" style="border: 1px solid black; padding: 5px; margin: 10px 0;"> </div> <p>In V3.0 (R2009b), the same model produces a plot that shows sample time hits in the pe signal when entities depart.</p> <div data-bbox="509 1045 982 1354" style="border: 1px solid black; padding: 5px; margin: 10px 0;"> </div>

Affected Blocks	Change in Behavior
<ul style="list-style-type: none"> • Event-Based Entity Generator • Infinite Server • N-Server • Output Switch • Single Server • Time-Based Entity Generator 	<p>When pe reflects the departure or other removal of a pending entity, the sample time hit occurs after the pending entity is no longer in the block. In earlier versions, when the pending entity is the only pending entity in the block, the sample time hit occurs when the departure or other removal is imminent. The sample time hit occurs at the same simulation time, but in a different sequence compared to other simultaneous events.</p> <p>Example</p> <p>The model, whose fragment appears in the next figure, behaves differently in different versions. Suppose a pending entity is in the server when the gate opens. In V3.1 (R2010a), the entity advances and acquires an attribute value of 1 from the pe signal. In earlier versions, the pe signal changes from 1 to 0 before the entity departs. As a result, when the entity advances, it acquires an attribute value of 0.</p> 
<ul style="list-style-type: none"> • Event-Based Entity Generator • Time-Based Entity Generator 	<p>When you configure the block to produce an error if an entity fails to depart, the error situation does not cause a sample time hit in pe. In this configuration, the block cannot store any pending entities, so there is no storage action to cause a sample time hit in pe.</p> <p>You see the effect of this change if, after the error occurs, you examine pe in the workspace or in a plot.</p>

Affected Blocks	Change in Behavior
<ul style="list-style-type: none"> Infinite Server N-Server 	<p>If a pending entity departs and one or more pending entities remain in the block, the pe signal has a single sample time hit of 1. In earlier versions, in this situation, the signal has a sample time hit of 0 followed by a sample time hit of 1.</p> <p>Example</p> <p>In the next model, at $T = 0.5$, one pending entity departs from the server and one pending entity remains. In V3.1 (R2010a), pe has a sample time hit to indicate the departure of the pending entity. The value is 1 because the block still contains another pending entity. In earlier versions, pe has a sample time hit of 0 followed by a sample time hit of 1.</p> 

Renaming of Parameter to Enable Pending Entity Signal

Blocks that have an optional **pe** signal output port rename the parameter that you use to enable the port. The name in V3.0 (R2009b) is **Status of pending entity departure** or **Status of pending entity**. The new name in V3.1 (R2010a) is **Pending entity present in block**. The affected blocks are:

- Event-Based Entity Generator
- Infinite Server
- N-Server
- Output Switch
- Single Server
- Time-Based Entity Generator

Expanded Options for Opening Release Gate

You can configure the Release Gate block to open upon each sample time hit of an input signal. Set the **Open gate upon** parameter to **Sample time hit from port ts**.

Blocks in Attributes Library Must Get or Set at Least One Attribute

These blocks no longer support a configuration in which the table in the dialog box is empty:

- Get Attribute
- Set Attribute

Parameters and Parameter Values Being Removed

Affected Block	Affected Parameter	What Happens When You Use the Parameter?	Use This Instead	Compatibility Considerations
<ul style="list-style-type: none"> • FIFO Queue • LIFO Queue • Priority Queue 	Status of pending entity departure is being removed.	<p>In legacy models in which the parameter is set to On, the corresponding pe signal output port is inactive.</p> <p>In the library block, the parameter is unavailable.</p>	Number of entities in queue	To update legacy models, set Status of pending entity departure to Off. For more information, see the technique in Determining Whether a Queue Is Nonempty. The technique yields similar, but not identical, information.
<ul style="list-style-type: none"> • FIFO Queue • LIFO Queue • Priority Queue 	Capacity must have a positive value.	Warns if you set the value to 0.	A positive value. Alternatively, remove the block.	Remove queue blocks whose capacity is zero.

R2009b

Version: 3.0

New Features

Bug Fixes

Support for Batch Simulation Using Rapid Simulation Target

SimEvents blocks support code generation using the Rapid Simulation target.

You can now perform these tasks:

- Accelerating Discrete-Event Simulations Using Rapid Simulation
- Varying Parameters Between Simulation Runs
- Sharing Executables for Discrete-Event Simulations

The Batch Discrete-Event Simulations Using Rapid Simulation Target demo illustrates this feature by varying parameters between simulation runs.

This feature requires Real-Time Workshop® software and uses the Rapid Simulation target.

Expanded Options for Resetting Entity Departure Counter

The Entity Departure Counter block offers you more options for resetting the entity count during the simulation. The new options and corresponding values of the **Reset counter upon** parameter are listed in the table.

Option	Value of “Reset counter upon” Parameter
Reset counter upon each sample time hit of an input signal	Sample time hit from port <i>ts</i>
Reset counter upon each function call	Function call from port <i>fcn</i>

R2009a

Version: 2.4

New Features

Bug Fixes

Compatibility Considerations

Debugger Supports Stepping, Breakpoints, and Querying

The new SimEvents debugger lets you use MATLAB functions to suspend a simulation at each step or breakpoint, and query simulation state to assess behavior. The debugger includes these functions:

<code>help</code>	Display help for debugger functions
<code>se_getdbopts</code>	SimEvents debugger options structure
<code>sedb.bdelete</code>	Delete breakpoints in discrete-event simulation
<code>sedb.blkinfo</code>	Block information in discrete-event simulation
<code>sedb.blklist</code>	Blocks and their identifiers in discrete-event simulation
<code>sedb.breakpoints</code>	List breakpoints in discrete-event simulation
<code>sedb.cont</code>	Continue simulation until next breakpoint
<code>sedb.currentop</code>	Current operation in discrete-event simulation
<code>sedb.detail</code>	Customize debugger display in discrete-event simulation
<code>sedb.disable</code>	Disable breakpoints in discrete-event simulation
<code>sedb.enable</code>	Enable breakpoints in discrete-event simulation
<code>sedb.eninfo</code>	Entity information in discrete-event simulation
<code>sedb.evbreak</code>	Set breakpoint for execution or cancellation of event
<code>sedb.evcal</code>	Event calendar of discrete-event simulation
<code>sedb.evinfo</code>	Event information in discrete-event simulation
<code>sedb.gceb</code>	Name of currently executing block in discrete-event simulation
<code>sedb.gcebid</code>	Identifier of currently executing block in discrete-event simulation
<code>sedb.gcen</code>	Identifier of entity currently undergoing operation
<code>sedb.gcev</code>	Identifier of current event
<code>sedb.quit</code>	Quit discrete-event simulation debugging session
<code>sedb.runtoend</code>	Run until end of discrete-event simulation
<code>sedb.simtime</code>	Current time in discrete-event simulation
<code>sedb.step</code>	Single step in discrete-event simulation
<code>sedb.tbreak</code>	Set timed breakpoint in discrete-event simulation
<code>sedebug</code>	Debug discrete-event simulation

For more information, see these resources:

- Overview of the SimEvents Debugger in the SimEvents user guide documentation
- A video tutorial on the Web, in two parts:
 - Basic Single Stepping and Querying
 - Breakpoints and Advanced Querying
- Building a Simple Discrete-Event Model in the SimEvents getting started documentation (Exploring the D/D/1 System Using the SimEvents Debugger section)
- Building a Simple Hybrid Model in the SimEvents getting started documentation (Confirming Event-Based Behavior Using the SimEvents Debugger section)

Event Logging Options Removed from Configuration Parameters Dialog Box

The SimEvents pane of the Configuration Parameters dialog box no longer contains event logging options. The behavior of the event logging options in earlier versions is like the behavior of the new debugger on page 29-2.

Compatibility Considerations

The debugger behavior produces slightly different information, with a different format, compared to the information produced by the event logging parameters. The closest approximations to the previous behavior use the `detail` and `runtoend` functions of the debugger.

Event Logging Parameter Removed	Similar Behavior in Debugger
Display events in event calendar	<ol style="list-style-type: none"> At the MATLAB command prompt, start the debugger on the model called <code>model</code>: <code>sedebg(model)</code> At the <code>sedebg>></code> prompt, configure the debugger and run the simulation until the end: <code>detail none</code> <code>detail('ev',1,'cal',1)</code> <code>runtoend</code>
Log events when executed Log events when scheduled	<ol style="list-style-type: none"> At the MATLAB command prompt, start the debugger on the model called <code>model</code>: <code>sedebg(model)</code> At the <code>sedebg>></code> prompt, configure the debugger and run the simulation until the end: <code>detail none</code> <code>detail('ev',1)</code> <code>runtoend</code> <p>The resulting log includes both execution messages and scheduling messages. Execution messages are not indented; scheduling messages are indented.</p>
Log entities advancing from block to block	<ol style="list-style-type: none"> At the MATLAB command prompt, start the debugger on the model called <code>model</code>: <code>sedebg(model)</code> At the <code>sedebg>></code> prompt, configure the debugger and run the simulation until the end: <code>detail none</code> <code>detail('en',1)</code> <code>runtoend</code> <p>The resulting log includes entity advancement messages and other information. Entity advancement messages appear indented.</p>

To approximate the behavior that results from setting multiple parameters for the same model, you can concatenate input arguments in the `detail(...)` command. For example, `detail('ev',1,'en',1)` is like logging event scheduling, event execution, and entity operations.

Discrete Event Signal to Workspace Block Clarifies Timing

You can no longer place the Discrete Event Signal to Workspace block in an atomic subsystem. The atomic subsystem executes at time instants that conflict with the time instants at which the event-based block in the subsystem executes. In earlier versions, placing the Discrete Event Signal to Workspace block in an atomic subsystem can produce unexpected results. Function-Call Subsystem and Enabled Subsystem are examples of atomic subsystems.

Compatibility Considerations

If your legacy model includes a Discrete Event Signal to Workspace block in an atomic subsystem, update the model as follows:

- 1** Move the Discrete Event Signal to Workspace block outside the atomic subsystem.
- 2** Connect the block to an output signal from the subsystem.

R2008b

Version: 2.3

New Features

Bug Fixes

New Demos for Modeling Architectures and Manufacturing Processes

Version 2.3 (R2008b) introduces these new demonstrations:

Tutorial Demos

- Translating Events to Functions Calls


Advanced Technique Demos

- Asynchronous Execution of a Stateflow Chart
- Building an Arrival Rate Estimator
- Interfacing with External File Formats
- Resource Allocation from Multiple Pools

Application Demos

- FIFO Buffer: Functional Model
- FIFO Buffer: Architectural Model
- Anti-Lock Braking System (ABS) Overview
- Batch Production Process
- Kanban Production System

Attribute Name Incrementing in Set Attribute and Get Attribute Blocks

The Add button  in the Set Attribute and Get Attribute blocks adds an attribute named `AttributeN` where N is a positive integer. In earlier versions, the button always adds an attribute named `Attribute1`.

Change in Parameter Name of Event-Based Entity Generator Block

The **Type of value change** parameter in the Event-Based Entity Generator block is now called **Type of change in signal value**. The new name is consistent with other blocks that have a parameter by that name.

R2008a

Version: 2.2

New Features

Bug Fixes

Initial Value Block in Signal Management Library

The new Initial Value block is in a new library called Signal Management. This block sets a signal value before the first event occurs.

Also, the Signal Latch block has moved from the Gates library to the new Signal Management library.

Discrete Event Subsystem Supports Complex and Nonscalar Values

In Version 2.2 (R2008a), input signals to the Discrete Event Subsystem block can be real or complex signals of any dimension. In earlier versions, input signals to the block must be real scalars.

Seed Management for Random Number Generators

New functions and diagnostics help you ensure uniqueness of seeds of random number generators and manage sets of seeds in a series of simulation runs. For details, see these sections:

- Varying Simulation Results by Managing Seeds
- `se_getseeds` function reference page
- `se_setseeds` function reference page
- `se_randomize_seeds` function reference page
- Identical seeds for random number generators

Configuration Parameters for Diagnostics

The Configuration Parameters dialog box has a new SimEvents Diagnostics pane to advise you of race conditions and help you manage seeds of random number generators.

For more information, see SimEvents Diagnostics Pane.

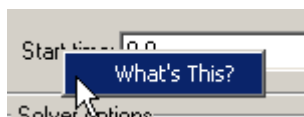
“What’s This?” Context-Sensitive Help Available for Simulink Configuration Parameters Dialog

R2008a introduces “What's This?” context-sensitive help for parameters that appear in the Simulink Configuration Parameters dialog. This feature provides quick access to a detailed description of the parameters, saving you the time it would take to find the information in the Help browser.

To use the “What's This?” help, do the following:

- 1 Place your cursor over the label of a parameter.
- 2 Right-click. A **What's This?** context menu appears.

For example, the following figure shows the **What's This?** context menu appearing after a right-click on the **Start time** parameter in the **Solver** pane.



-
- 3 Click **What's This?** A context-sensitive help window appears showing a description of the parameter.

New Demos

SimEvents software Version 2.2 (R2008a) introduces these new demos:

- Managing Race Conditions
- Avoiding Identical Seeds for Random Number Generators
- Seed Management Workflow for Random Number Generators
- Asynchronous Clock Domains
- Rate-Based Shared Processor

Also, the demo formerly named “Explicit Routing for Distributed Processing” is now called “Distributed Processing for Multi-Class Jobs.”

R2007b

Version: 2.1

New Features

Bug Fixes

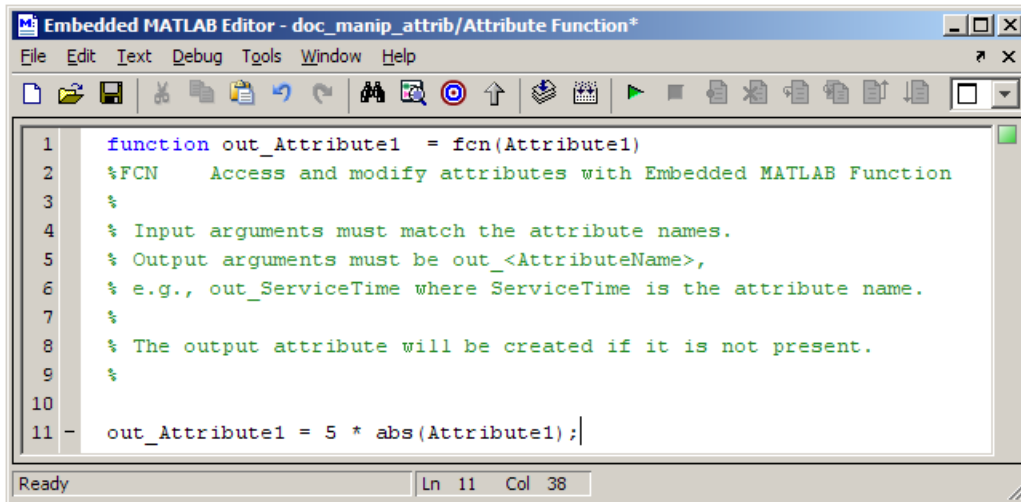
Compatibility Considerations

Attribute Computations Using MATLAB Code

The Attribute Function block lets you conveniently set and modify attributes using MATLAB code. For details, see Writing Functions to Manipulate Attributes.

Simplifying a Model Using the Attribute Function Block

The following figures indicate recommended ways to multiply the absolute value of an attribute by a constant in SimEvents software Version 2.1 (R2007b) and earlier versions. The earlier version is more complicated because of necessary steps to ensure correct timing. By contrast, the Attribute Function block ensures correct timing automatically.

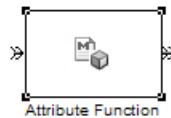


```

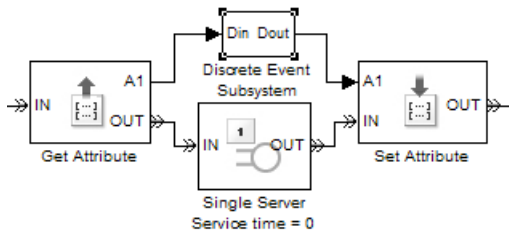
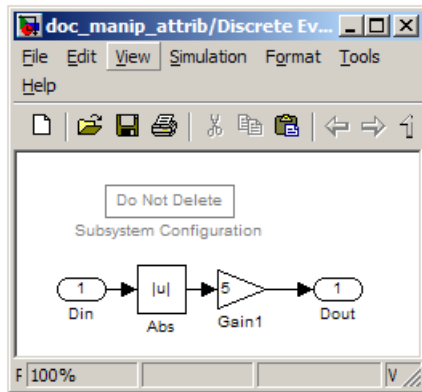
Embedded MATLAB Editor - doc_manip_attrib/Attribute Function*
File Edit Text Debug Tools Window Help
[Icons]
1  function out_Attribute1 = fcn(Attribute1)
2  %FCN    Access and modify attributes with Embedded MATLAB Function
3  %
4  % Input arguments must match the attribute names.
5  % Output arguments must be out_<AttributeName>,
6  % e.g., out_ServiceTime where ServiceTime is the attribute name.
7  %
8  % The output attribute will be created if it is not present.
9  %
10
11 - out_Attribute1 = 5 * abs(Attribute1);

```

Ready Ln 11 Col 38



Manipulating Attribute Value Using Attribute Function Block



Manipulating Attribute Value in Earlier Versions

Attributes Support Complex Values

In Version 2.1 (R2007b), attributes can assume complex values, not only real values.

Enhanced Visibility and Logging of Events

In Version 2.1 (R2007b), SimEvents software changes the set of events that appear in event logs:

- Event logs show a new kind of event, called an entity request event. This event is a notification that an entity input port of a block has become available. To understand the name entity request event, think of the block as requesting an entity from a preceding block. For example, upon becoming empty, a single server requests an entity from a preceding block. A preceding block's response to the notification might result in an entity advancement.

In earlier versions, entity request events do not appear in event logs.

- Event logs show a new kind of event, called a storage completion event. This event exists only in an Output Switch block with the **Store entity before switching** parameter selected. When an entity arrives at the block, the block schedules a storage completion event at the current simulation time. Upon execution of the storage completion event, the block determines whether the entity can advance to a subsequent block.

In earlier versions, storage completion events do not appear in event logs.

- Event logs always show the events listed in the following table, regardless of how you set the **Resolve simultaneous signal updates according to event priority** parameter in the corresponding blocks. This parameter determines whether the event priority is a number you specify in the block dialog box or a system-level category denoted by SYS1.

Events That Affect Entities and Are Caused By Signal-Based Events

Event	Block
Entity generation	Event-Based Entity Generator
Counter reset	Entity Departure Counter
Gate event (gate opening or gate closing)	Enabled Gate
Release	Release Gate
Port selection	Input Switch, Output Switch, Path Combiner

In earlier versions, event logs show these events only if you select **Resolve simultaneous signal updates according to event priority** in the block dialog box.

Also, event logs and entity logs in Version 2.1 (R2007b) are more readable and contain hyperlinks that highlight the corresponding blocks.

New Demos for Shared-Resource Applications and Advanced Techniques

SimEvents software Version 2.1 (R2007b) introduces these new demonstration models:

Tutorial Demos

- Server Blocks and Service Time
- Input and Output Switching
- Schedule Timeout and Cancel Timeout Blocks

Advanced Technique Demos

- Buffering for Variable-Size Messages
- Delayed Function Calls in Pulse Width Modulation
- Explicit Routing for Distributed Processing
- Markov-Modulated Poisson Process
- Queue with Flushing Capability
- Variable Entity Replication
- Variable-Capacity Queue

Application Demos

- Shared Communication Buffer Management
- Processor Sharing Via Time Slicing

Consolidation and Removal of Some Tutorial Demos

The new Server Blocks and Service Time demo replaces these earlier demos:

- Service Time from Attribute
- Specifying Service Time in Single Server

- Specifying Service Time in Infinite Server Block
- Single Server Block Versus Infinite Server Block

The new Input and Output Switching demo replaces these earlier demos:

- Input Switching Using Signal
- Output Switching Using Signal

Changes in Categorization, Titles, and Content of Some Demos

SimEvents demos have been recategorized in the Help browser. Some demos have changed their titles or content.

Title in Version 2.1 (R2007b)	Title in Earlier Versions
Task Sharing with Two Levels of Priority and Preemption	Preemptive Operating System with Two Levels of Priority
Multitasking with Dependent Tasks	Multitasking Model with Dependent Tasks
Operating System with Prioritized Task Execution	Operating System Model with Prioritized Task Execution
Entity Combiner for Assembling Components (with simpler design using the Entity Combiner and Attribute Function blocks)	Aggregation: Assembling a Vehicle Chassis

Also, the G/G/1 Queuing System and Little's Law demo has a simpler design using the Attribute Function and Embedded MATLAB Function blocks.

Subsystem Connection Port for Entity Paths

The Conn block represents an entity input port or entity output port in a virtual subsystem. The model window's **Edit > Create Subsystem** menu option automatically creates connection ports. Copying the Conn block from its library is a convenient way to add more entity ports to an existing subsystem.

Configuration Parameters to Control Livelock

The SimEvents pane of the Configuration Parameters dialog box offers new parameters for setting thresholds related to livelock. Also, the **Execution order of simultaneous events** parameter has been renamed **Execution order**.

New Parameter	Description
Maximum events per block	Limit the number of entity generation, service completion, subsystem execution, and function-call events that each SimEvents block performs at each fixed time instant.
Maximum events per model	Limit the total number of events scheduled via the event calendar at each fixed time instant.

For more information, see Livelock Prevention or the configuration parameter descriptions.

Processing Events Via the Event Calendar Instead of Immediately

In Version 2.1 (R2007b), SimEvents software changes its processing of each event in the next table when you do not select **Resolve simultaneous signal updates according to event priority** in the corresponding block. In this case, the event has a system-level event priority denoted by *SYS1*, and the application processes the event via the event calendar. Using the event calendar decouples the scheduling and the execution of events. Event Sequencing describes how the application processes multiple simultaneous events.

Events That Affect Entities and Are Caused By Signal-Based Events

Event	Block
Entity generation	Event-Based Entity Generator
Counter reset	Entity Departure Counter
Gate event (gate opening or gate closing)	Enabled Gate
Release	Release Gate
Port selection	Input Switch, Output Switch, Path Combiner

Also, each entity request event has a system-level priority denoted by *SYS2*, and the application processes the event via the event calendar.

In earlier versions, the application applies “immediate” processing for entity requests by storage blocks, as well as for events in the table when you do not select the **Resolve simultaneous signal updates according to event priority** parameter in the corresponding block.

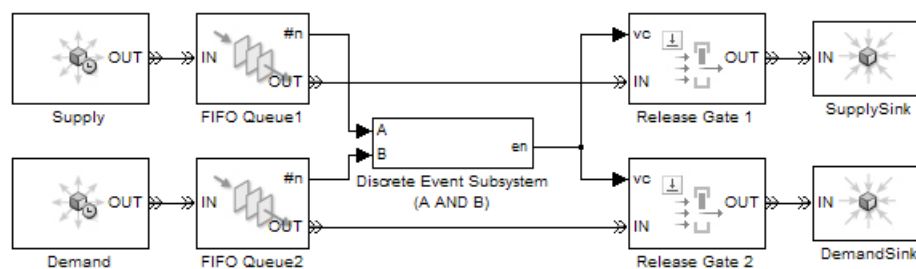
For details about supported events and the processing of simultaneous events, see Working with Events and Managing Simultaneous Events.

Compatibility Considerations

Most models are unaffected by the change in event processing. However, some models might behave differently, because events processed immediately in earlier versions are deferred to the event calendar in SimEvents software Version 2.1 (R2007b). Models that behave differently tend to involve cycles in simulation processing and cascades of simultaneous events (for example, an event has multiple consequences that occur at time *T*, each of which has further consequences also at time *T*).

Example 32.1. Example Showing Change in Behavior

The model below attempts to simultaneously advance one entity from each queue, whenever both queues are nonempty.



Suppose that the top queue contains one entity and an entity arrives at the previously empty bottom queue. Assuming that no block in the model has its **Resolve simultaneous signal updates according to event priority** parameter selected, the entity arrival has the following cascade of consequences:

- 1 The bottom queue updates its **#n** output signal to 1.
- 2 The discrete event subsystem evaluates the condition (A AND B) and returns a value of 1. The previous value of this signal was 0.
- 3 Each of the two Release Gate blocks detects a value-change event at its **vc** signal input port. In SimEvents software Version 2.1 (R2007b), each of the two gates schedules a release event on the event calendar.
- 4 One gate opens, which has these consequences:
 - a An entity advances from the corresponding queue to the sink.
 - b The corresponding queue updates its **#n** output signal to 0.
 - c The discrete event subsystem reevaluates the condition (A AND B) and returns a value of 0.
- 5 In Version 2.1 (R2007b), the other gate opens, which has these consequences:
 - a An entity advances from the corresponding queue to the sink.
 - b The corresponding queue updates its **#n** output signal to 0.
 - c The discrete event subsystem reevaluates the condition (A AND B) and returns a value of 0.

In earlier versions, the gates do not schedule release events on the event calendar if the corresponding **Resolve simultaneous signal updates according to event priority** parameter is not selected. As a result, step 4c negates the value-change event at the other gate and step 5 does not occur. This example involves cycles in simulation processing, because an event at the gate affects the value of the **#n** signal of a preceding block. This example involves cascades of simultaneous events, because the new value of 1 for the condition (A AND B) causes two release events, each of which causes the condition (A AND B) to assume the value 0.

Enhanced Support for Multiple Simultaneous Transitions in Switches and Gate

The blocks in this table model the effects of all transitions in their input signals, even if multiple transitions occur simultaneously.

Block	Input Signal
Enabled Gate	en
Input Switch	p
Output Switch	p
Path Combiner	p

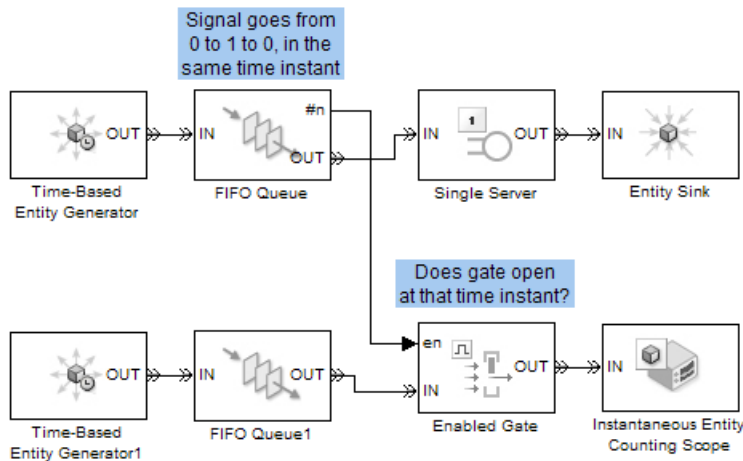
In earlier versions of SimEvents software, selecting the **Resolve simultaneous signal updates according to event priority** option causes the blocks to model only the last transition at a given value of the simulation clock.

Compatibility Considerations

The behavior of some simulations changes depending on whether the application models intermediate transitions in an **en** or **p** input signal in the blocks in the table above.

Example 32.2. Example Showing Change in Behavior

In the model below, the **en** signal transitions from 0 to 1 and then from 1 to 0 in the same time instant. Earlier versions model only the latter transition, so the gate does not open. Version 2.1 (R2007b) models both transitions, so the gate opens and then closes in the same time instant.



Change in Indexing in Attribute Scope Block

When you set the **X value from** parameter to **Index** in the Attribute Scope block, it creates a plot that reflects 1-based indexing. That is, the first entity corresponds to a data point whose value on the horizontal axis is 1. In earlier versions, the plot reflects 0-based indexing.

R2007a

Version: 2.0

New Features

Bug Fixes

Compatibility Considerations

Attributes Support Multidimensional Values

Version 2.0 (R2007a) introduces new versions of the Get Attribute and Set Attribute blocks in a new Attributes library. The new blocks offer these enhancements compared to the earlier versions:

- Attributes can assume values that are vectors, matrices, or multidimensional arrays with up to 32 dimensions, not just scalars. This enhancement facilitates modeling dense payloads via attributes.
- Each instance of a Set Attribute block can assign an arbitrary number of attributes, and each instance of a Get Attribute block can retrieve an arbitrary number of attributes.
- The dialog boxes use a grid on a single tab, making it easier for you to see the entire list of attributes that a block sets or gets.

Compatibility Considerations

If your legacy models contain Get Attribute or Set Attribute blocks from the earlier library, those blocks continue to work in Version 2.0 (R2007a). However, the blocks are considered obsolete, as described in “Obsolete Blocks” on page 33-6.

Combining and Splitting Entities

The new Entity Combiner block lets you combine entities, analogous to combining components to create a larger whole. The block provides options for managing information (attributes and timers) associated with the component entities, so you can think of the operation as bundling the information that entities carry with them.

You can configure the Entity Combiner block to make the combining operation reversible via the Entity Splitter block.

The Entity Combiner and Entity Splitter blocks reside in the new Entity Management library.

Timeout Feature Establishes Entity Time Limits

You can model point-to-point timing constraints by limiting the amount of time an entity spends during the simulation on designated entity paths. Exceeding the limit causes the entity to depart immediately from the storage block where it resides, such as a queue, when the clock reaches the time limit. To learn how to use this feature, see Forcing Departures Using Timeouts in the SimEvents user guide documentation.

The timeout feature involves new blocks, as well as new parameters in existing blocks.

New Block	Purpose
Schedule Timeout	Schedule timeout event for each entity
Cancel Timeout	Cancel timeout event for each entity

Existing Blocks with New Timeout-Related Parameters

- FIFO Queue
- LIFO Queue
- Priority Queue

- Infinite Server
- N-Server
- Single Server
- Output Switch

New Parameter of Existing Blocks	Purpose
Enable TO port for timed-out entities on Timeout tab	Provide a TO entity output port through which an entity departs upon timing out
Number of entities timed out on Statistics tab	Output a signal, #to , that indicates the number of entities that have timed out from the block since the start of the simulation

Compatibility Considerations

If you save a model containing a queue, server, or Output Switch block using V2.0 (R2007a), then opening the model in V1.2 (R2006b) produces warnings like these:

Warning: In instantiating linked block 'mysys/FIFO Queue' :
FIFO Queue block (mask) does not have a parameter named 'EnableTOPort'.

Warning: In instantiating linked block 'mysys/FIFO Queue' :
FIFO Queue block (mask) does not have a parameter named 'StatNumberTimedout'.

Saving the model in the earlier version prevents the warnings from reappearing, but causes the block to omit timeout-related ports and behavior if you later open the model in V2.0 (R2007a).

New Demos for Video Processing, Communications, and Architecture Modeling

Version 2.0 (R2007a) introduces these new demonstration models:

Tutorial Demos

- Transporting Multidimensional Data Using Attributes
- Packet Creation, Transmission and Error Analysis

Application Demos

- Distributed Video Processing
- Distributed Processing Resource Modeling
- Video Streaming Over Bandwidth-Limited Communication Channel
- Bit Timing Recovery Using Fixed-Rate Resampling and SimEvents

Change in ARQ Demo

The Selective-Repeat Automatic Repeat Request demo reverses the interpretation of the CRC check compared to V1.2 (R2006b). The interpretation now matches that of the similar Go-Back-N Automatic Repeat Request demo. In V2.0 (R2007a), both demos use a CRC check value of 1 to correspond to an ACK message.

Output Switch Block Options for Storage and Initial Condition

The Output Switch block offers enhancements that can prevent the need for additional blocks to set initial conditions or to prevent latency. The new parameters apply to signal-based switching and are available only when you set **Switching criterion** to From signal port **p**. The new parameters are in the table below. For details, see Output Switching Based on a Signal and the block's reference page.

New Parameter	Purpose
Specify initial port selection	Determine whether the block uses an initial port selection from the dialog box.
Initial port selection	The entity output port that the block selects when the simulation begins. The block uses this value instead of the p signal until the signal has its first sample time hit.
Store entity before switching	If you select this option, the block can store one entity at a time. Furthermore, the block decouples its arrival and departure processing to give other blocks in the simulation an opportunity to update the p signal if appropriate. If you do not select this option, the block processes an arrival and departure as an atomic operation and assumes that the p signal is already up to date at the given time.
Status of pending entity on Statistics tab	Output a signal, pe , that indicates when the block stores an entity after trying and failing to output it. A value of 0 indicates when the storage location is empty.

For other changes in this release that affect parameters of the Output Switch block, see “Timeout Feature Establishes Entity Time Limits” on page 33-2 and “Changes in Names of Parameters Related to Event Priorities” on page 33-5.

Compatibility Considerations

In some cases, the block enhancements let you optionally simplify models that you do not need to share with users of earlier versions:

- If your model precedes an Output Switch block with a Signal Latch block to create an initial condition for the **p** signal, and if the **p** signal does not branch to become an input for another block, then you can remove the Signal Latch block and instead use the new **Specify initial port selection** option in the switch block.
- If your model precedes an Output Switch block with a Single Server block whose **Service time** parameter is zero and whose sole purpose was to ensure an up-to-date **p** signal, then you can remove the Single Server block and instead use the new **Store entity before switching** option in the switch block.

If you save a model containing an Output Switch block using V2.0 (R2007a), then opening the model in V1.2 (R2006b) produces warnings like these:

```
Warning: In instantiating linked block 'mysys/Output Switch' :
Output Switch block (mask) does not have a parameter named
'InitialConditionsOpt'.
Warning: In instantiating linked block 'mysys/Output Switch' :
Output Switch block (mask) does not have a parameter named
```


'InitialConditions'.
Warning: In instantiating linked block 'mysys/Output Switch' :
Output Switch block (mask) does not have a parameter named
'EntityBufferOpt'.
Warning: In instantiating linked block 'mysys/Output Switch' :
Output Switch block (mask) does not have a parameter named
'StatPendingEntity'.

Saving the model in the earlier version prevents the warnings from reappearing, but causes the block to omit ports and behavior related to the V2.0 (R2007a) enhancements if you later open the model in V2.0 (R2007a).

Entity Departure Counter Block Can Create Attribute

If you configure the Entity Departure Counter block to write the count to an attribute, then you can select the new **Create attribute if not present** parameter to have the block create the attribute if it does not already exist. The block then sets the value of the attribute according to the entity count.

In earlier versions, the block sets the value of the attribute but does not create it.

Changes in Names of Parameters Related to Event Priorities

Parameters related to optional priorities of events have been renamed to be more suggestive of how the option works. The name **Resolve simultaneous signal updates according to event priority** replaces names that start with **Specify event priority**. In a subset of affected blocks, the name **Event priority** replaces similar names. For more information about what the parameters mean, see Choosing How to Resolve Simultaneous Signal Updates.

The table below itemizes the blocks and parameters that have changed.

Block	Parameter Name in V1.2 (R2006b)	Parameter Name in V2.0 (R2007a)
Discrete Event Inport	Specify event priority for executing subsystem	Resolve simultaneous signal updates according to event priority
	Subsystem execution event priority	Event priority
Enabled Gate	Specify event priority for gate opening and closing	Resolve simultaneous signal updates according to event priority
Entity Departure Counter	Specify event priority for counter reset	Resolve simultaneous signal updates according to event priority
Event-Based Entity Generator	Specify event priority for entity generation	Resolve simultaneous signal updates according to event priority
	Generation event priority	Event priority

Block	Parameter Name in V1.2 (R2006b)	Parameter Name in V2.0 (R2007a)
Input Switch	Specify event priority for port selection	Resolve simultaneous signal updates according to event priority
Output Switch	Specify event priority for port selection	Resolve simultaneous signal updates according to event priority
Path Combiner	Specify event priority for port precedence selection	Resolve simultaneous signal updates according to event priority
Release Gate	Specify event priority for gate opening	Resolve simultaneous signal updates according to event priority
Signal Latch	Specify event priority for writing to memory	Resolve simultaneous signal updates according to event priority on Write tab
	Specify event priority for reading from memory	Resolve simultaneous signal updates according to event priority on Read tab
Signal-Based Event to Function-Call Event	Specify event priority for function-call generation	Resolve simultaneous signal updates according to event priority
	Function-call event priority	Event priority
Signal-Based Function-Call Event Generator	Specify event priority for function-call generation	Resolve simultaneous signal updates according to event priority
	Function-call event priority	Event priority

This change merely renames parameters and does not change the behavior of affected blocks.

Change in Default Entity Type of Entity Generators

The default value of **Entity type** in the Time-Based Entity Generator and Event-Based Entity Generator block is **Blank**. In earlier versions, the default value is **Standard**. This change in default value does not affect blocks in a saved model but only affects new instances of the block that you copy from the library to a model.

Obsolete Blocks

The table below indicates blocks that are obsolete as of the current version or that are planned to be removed in a future version.

Obsolete Block	Removed from Version	Replacement
Get Attribute block from simeventsattributes1 library	Future version	Get Attribute block from simeventsattributes2 library

Obsolete Block	Removed from Version	Replacement
Set Attribute block from simeventsattributes1 library	Future version	Set Attribute block from simeventsattributes2 library

R2006b

Version: 1.2

New Features

Bug Fixes

Compatibility Considerations

Event-Based Sequence Generator Block

The new Event-Based Sequence block provides data to an event-driven process by producing a scalar event-based output signal whose values come from a vector. The block selects the next value from the vector upon each notification from a port of a subsequent block. For example, if you connect the Event-Based Sequence block to the **t** input port of a Single Server block, the values in the vector become the service times for the entities arriving at the server. You provide the values in the vector, but do not need to know in advance when the entities arrive at the server.

New Tutorial and Application Demos

Version 1.2 (R2006b) introduces these new demonstration models:

Tutorial Demos

- Entity Combiner for Assembling Components
- Task Sharing with Two Levels of Priority and Preemption
- Multitasking with Dependent Tasks

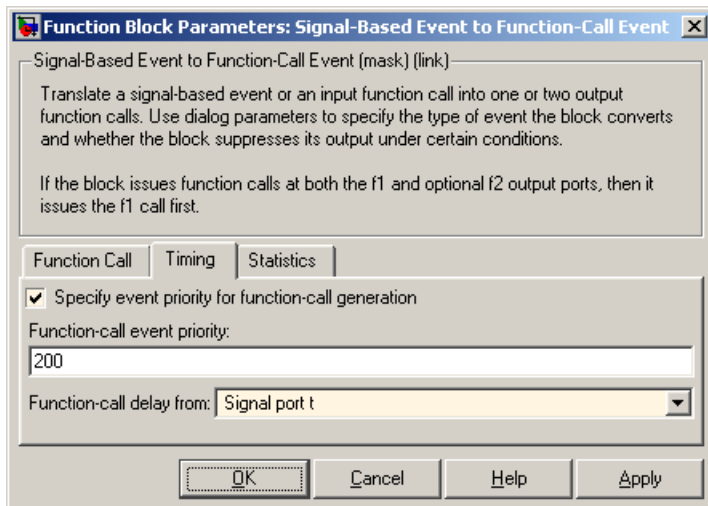
Application Demos

- Go-Back-N Automatic Repeat Request
- Operating System with Prioritized Task Execution
- Ethernet Local Area Network
- Anti-Lock Braking System (ABS) Model
- Anti-Lock Braking System (ABS) Model with Queuing Delay
- Anti-Lock Braking System (ABS) Model Using CAN Communications

Event Translation Block Supports Delay from a Signal

The Signal-Based Event to Function-Call Event block can delay its generation of a function call by an amount of time that you specify using either an input signal or the **Function-call time delay** parameter. In V1.1 (R2006a), the block lets you specify the delay amount using the parameter, but not an input signal.

To access the new feature, select **Specify event priority for function-call generation** (or, in V2.0 (R2007a), select **Resolve simultaneous signal updates according to event priority**). Then set the new **Function-call delay from** parameter to **Signal port t**, as shown. Then connect a nonnegative-valued signal to the **t** signal input port that appears on the block.



Compatibility Considerations

If you save a model containing the Signal-Based Event to Function-Call Event or Discrete Event Subsystem block using V1.2 (R2006b), then opening the model in V1.1 (R2006a) produces warnings like these:

```
Warning: In instantiating linked block 'mysys/Signal-Based
Event to Function-Call Event' : Signal-Based Event to Function-
Call Event block (mask) does not have a parameter named
'FunctionCallDelayFrom'.
```

Saving the model in the earlier version prevents the warnings from reappearing, but causes the Signal-Based Event to Function-Call Event block to omit the **t** input port if you later open the model in V1.2 (R2006b).

Routing Blocks Support Unlimited Entity Ports

The **Number of entity input ports** parameter of the Input Switch and Path Combiner blocks can be any positive integer. The **Number of entity output ports** parameter of the Output Switch and Replicate blocks also can be any positive integer. In V1.1 (R2006a), these parameters can assume only the values 1, 2, 3, and 4.

Compatibility Considerations

If you save a model in which one of the blocks listed above has more than four entity input ports or more than four entity output ports, then the model will not work in V1.1 (R2006a).

Initial Outputs of SimEvents Blocks

All SimEvents blocks now have well-defined initial values for any numerical output signals they produce.

The initial value of an output signal of a SimEvents block is in effect from the start of the simulation until the block updates the output signal for the first time during the simulation. For example, if an N-

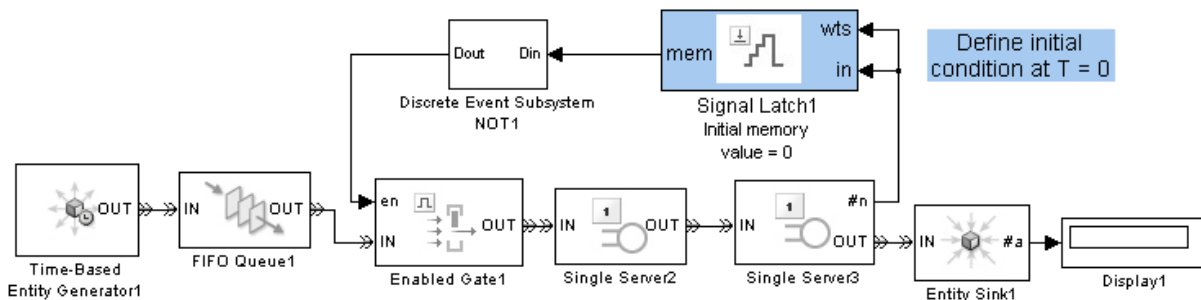
Server block is configured to produce a **#n** output signal representing the number of entities in the server, then **#n** has a well-defined initial value of 0 at the start of the simulation. The initial value persists until the first arrival of an entity at the N-Server block, which could occur well after the start of the simulation, if at all.

The block reference pages indicate the initial values of the block output signals.

Compatibility Considerations

If you connect the Signal Latch block to a **ts**, **tr**, or **vc** signal input port of a SimEvents block, the input port might detect an event at the start of the simulation in V1.1 (R2006a) that no longer occurs in V1.2 (R2006b). This is because the Signal Latch block assumes its initial condition in a true initialization stage in V1.2 (R2006b) rather than slightly after the simulation start in V1.1 (R2006a). If your model relies on an event at the start of the simulation (to invoke a discrete event subsystem or generate an event or an entity, for example), then you might see a change in simulation behavior between the two versions.

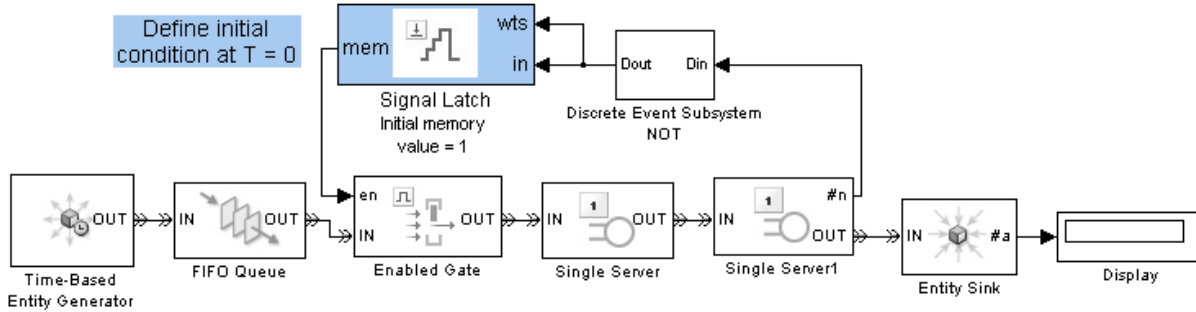
For example, the model below uses a Discrete Event Subsystem block to compute a signal that indicates whether a gate is open or closed.



Subsystem Invoked at Simulation Start in V1.1 (R2006a), but Not V1.2 (R2006b)

In V1.1 (R2006a), the Signal Latch block's output signal has a sample time hit at the start of the simulation. This sample time hit invokes the subsystem, which initializes the gate's **en** input signal to 1. As a result, the gate is open at the start of the simulation. In V1.2 (R2006b), the Signal Latch block does not have a sample time hit at the start of the simulation, so the initial condition of the subsystem's output determines the initial condition of the gate's **en** input signal. As a result, the gate is closed at the start of the simulation.

An alternative approach that works in both versions is to move the Signal Latch block so that it follows the Discrete Event Subsystem block. The Signal Latch block directly provides the gate's initial condition.



Correct Gate Initialization in Both V1.1 (R2006a) and V1.2 (R2006b)

History Options and Other Changes in Scope Blocks

The following blocks include new **Store data when scope is closed** and **Limit data points to** parameters on the new **Data History** tab of the dialog box:

- Attribute Scope
- Instantaneous Entity Counting Scope
- Instantaneous Event Counting Scope
- Signal Scope
- X-Y Attribute Scope
- X-Y Signal Scope

The parameters determine how much data the blocks cache, letting you balance data visibility with simulation efficiency. Caching data lets you view it later, even if the scope is closed during part or all of the simulation. Caching less or no data accelerates the simulation and uses less memory. In V1.1 (R2006a), if you have the scope closed for the first T seconds of simulation and then open the scope, you can view only the data for $t > T$.

Other Changes in Scope Blocks

Version 1.2 (R2006b) changes some aspects of the way you interact with the scope blocks:

- A Pan toolbar button lets you move your view of a plot.
- A Parameters toolbar button opens the block dialog box.
- Double-clicking on the block opens the plot if it is not already open. In V1.1 (R2006a), double-clicking on the block opens the block dialog box. To open the block dialog box in V1.2 (R2006b), click the Parameters toolbar button on the plot.
- The autoscale feature no longer changes the initial axis limits that you specify in the block dialog box. A new **Save axes limits** menu option lets you update the initial axis limits to match the plot's current limits. The current limits might differ from their initial values due to stretching, shifting, panning, zooming, or autoscaling operations that occurred since the initial values were last set.
- The former **Open at start of simulation** parameter is now called **Open scope at start of simulation** and has moved from the **Figure** tab of the dialog box to the **Plotting** tab.

The scope blocks also plot initial conditions without a plotting marker. In V1.1 (R2006a), initial conditions typically do not appear in plots.

Finally, the scope blocks run significantly faster in V1.2 (R2006b).

Compatibility Considerations

If your legacy models contain scope blocks that plot more than 1000 points, then the default values of the new **Store data when scope is closed** and **Limit data points to** parameters cause the scope to retain only the last 1000 points. To plot all points, set **Store data when scope is closed** to **Unlimited**.

If you save a model containing a scope block using V1.2 (R2006b), then opening the model in an earlier version produces warnings about the parameters that are not in the earlier block. For example,

```
Warning: In instantiating linked block 'mysys/Attribute
Scope' : Attribute Scope block (mask) does not have a parameter
named 'DataStoreOption'.
Warning: In instantiating linked block 'mysys/Attribute
Scope' : Attribute Scope block (mask) does not have a parameter
named 'DataPointsLimit'.
```

Saving the model in the earlier version prevents the warnings from reappearing, but also causes the block to use default values for the new parameters if you later open the model in V1.2 (R2006b).

Parameters for Lognormal Distribution

The Event-Based Random Number block produces random numbers from a lognormal distribution when you set the **Distribution** parameter to **Lognormal**. Different texts use different parameterizations of the lognormal distribution. V1.2 (R2006b) renames some parameters in this block to clarify the relationship between a lognormal random variable X and the normal random variable $\log(X)$.

V1.1 (R2006a) Parameter Name	V1.2 (R2006b) Parameter Name
Scale	Mu
Shape	Sigma

Compatibility Considerations

The block behaves the same in V1.1 (R2006a) and V1.2 (R2006b) because the change merely renames parameters. However, the parameter names in V1.2 (R2006b) more accurately reflect the block's behavior.

SimEvents Blocks Compatible with Accelerator Mode

All SimEvents blocks are compatible with accelerator mode. Version 1.1 (R2006a) does not support simulating models in accelerator mode if the models contain the Event-Based Random Number block.

Livelock Detection

SimEvents software can detect livelock during a simulation. When it detects livelock, it halts the simulation with an error message that indicates too many simultaneous events. In V1.1 (R2006a), livelock can potentially cause MATLAB software to crash.

For details, see Livelock Prevention.

Compatibility Considerations

It is possible for the application to consider a situation to be livelock when it is actually a large but finite loop. Such simulations might work in V1.1 (R2006a) but not in V1.2 (R2006b).

R2006a

Version: 1.1

New Features

Bug Fixes

Compatibility Considerations

Replicate Block Supports Partial Replication

The Replicate block supports partial replication and offers more flexibility when you choose complete replication. New parameters in the block's dialog box are in the table below.

Parameter	Description
Replicate entity when	Lets you choose whether the block accepts arriving entities for replication only when all entity output ports are not blocked or whenever at least one entity output port is not blocked. The first option is the default.
If an output port becomes blocked during replication	Determines how the block responds if a departure through one entity output port causes another entity output port to become blocked.
Number of entities departed	Toggles the optional output signal #d , representing the number of departed entities.

Compatibility Considerations

By default in V1.1 (R2006a), when a departure through one entity output port causes another entity output port to become blocked, the result is a discarded entity with no error or warning message. If this phenomenon occurs in your legacy models, then the result in V1.0 (R14SP3+) might be an error message or incorrect behavior. If you want to learn when this phenomenon occurs in your legacy models that you simulate using V1.1 (R2006a), then set **If an output port becomes blocked during replication** to either Warn and discard entity, or Error.

The default values of the other new parameters added in V1.1 (R2006a) are consistent with the block's behavior in V1.0 (R14SP3+), so legacy models need no changes to accommodate these new features.

If you save a model containing the Replicate block using V1.1 (R2006a), then opening the model in V1.0 (R14SP3+) produces warnings about the parameters that are not in the V1.0 block. For example,

```
Warning: In instantiating linked block 'mysys/Replicate' :
  Replicate block (mask) does not have a parameter named
  'ReplicateEntityWhen'.
Warning: In instantiating linked block 'mysys/Replicate' :
  Replicate block (mask) does not have a parameter named
  'ActionUponBlocking'.
Warning: In instantiating linked block 'mysys/Replicate' :
  Replicate block (mask) does not have a parameter named
  'StatNumberDeparted'.
```

Also, simulating that model under V1.0 causes the block to exhibit its V1.0 behavior, which is to omit a **#d** output signal and to replicate the arriving entity only when all entity output ports are not blocked. Saving the model in V1.0 prevents the warnings from reappearing in V1.0 but also causes the block to exhibit its V1.0 behavior if you later open the model in V1.1.

R14SP3+

Version: 1.0

New Features

Introduction to SimEvents

SimEvents software extends Simulink software with tools for modeling and simulating discrete-event systems using queues and servers. With SimEvents software you can create a discrete-event simulation model to simulate the passing of entities through a network of queues, servers, gates, and switches based on events. The software provides an integrated environment for modeling hybrid dynamic systems containing continuous-time, discrete-time, and discrete-event components.

A key concept that SimEvents software adds to the Simulink environment is that of *entities*, which are discrete items of interest in a discrete-event simulation. For example, entities could represent messages to be communicated or parts to be assembled. Entities can carry data in one or more scalar structures called *attributes*. For example, attributes could represent destinations of messages or dimensions of parts.

The libraries in SimEvents software contain blocks that can

- Create entities
- Store entities in a queue
- Serve or delay entities
- Forbid or allow entities to depart, depending on specified criteria
- Manipulate the paths on which entities travel
- Attach data or timers to entities
- Create plots using data from entities or statistics gathered during simulation
- Manipulate or generate discrete events that can affect the behavior of blocks and entities
- Control the simulation timing in situations where event-driven behavior and time-driven behavior interact