DDS Blockset Release Notes

MATLAB®&SIMULINK®

MathWorks®

How to Contact MathWorks

Latest news:	`www.mathworks.com`
Sales and services:	`www.mathworks.com/sales_and_services`
User community:	`www.mathworks.com/matlabcentral`
Technical support:	`www.mathworks.com/support/contact_us`
Phone:	508-647-7000

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

DDS Blockset Release Notes

Trademarks

Patents

Contents

R2022b

R2022a

R2021b

R2021a

R2022b

Version: 1.3

New Features

Support for DDS input events

R2022b introduces support for DDS input events. DDS Blockset now supports modeling and code generation for event-driven communication where a subscriber application thread waits for an input event trigger before executing condition-based algorithms.

In Simulink®, you can configure DDS events at the input ports of your model and bind an event to a model partition that contains the response algorithm. You can trigger one or more events based on the flow of data into a root-level input port of a model subsystem. For more information, see “Model DDS Input Events”.

Support for shortened version of DDS type names

In R2022b, DDS Blockset supports the use of a shortened version of DDS user data type names in Simulink and in the generated code.

Before R2022b, type names were represented in Simulink and generated code as the full name, which is a concatenation of the full path of module names defined in the namespace hierarchy for the type. This representation sometimes resulted in object names that exceeded the 63 character limit. The short name for a data type is a version of the long name that is shortened by using a unique identifier representing the namespace hierarchy.

- For new DDS dictionaries created in R2022b, the full name, short name, and base of a DDS type display in the DDS Dictionary. Simulink and code generation use the short name for `Simulink.Bus` objects, enumerated types, and constants, with appropriate C++ aliases for the namespace and full names in the generated code.
- For DDS dictionaries created in previous releases and moved to R2022b, the DDS Dictionary, Simulink, and code generation use the full name for backward compatibility.

For more information, see DDS Dictionary.

Specifying C++ compiler, linker, and shared library linker options

In R2022b, you can specify C++ compiler, C++ linker, and C++ shared library linker options to add to the default options for compiling and linking your DDS model.

To specify DDS C++ compiler, linker, and shared library link options, open the DDS model, and select **Settings** to open the **Configuration Parameters** dialog box. Then under **Code Generation** > **Build process** > **Toolchain settings**, set **Build Configuration** to `Specify` and enter the additional options in the **C++ Compiler**, **C++ Linker**, and **C++ Shared Library Linker** fields, as needed. The specified options are added to the default options and configured in the appropriate definitions in the `CMakeLists.txt` for the project.

2

R2022a

Version: 1.2

New Features

Introduction of new vendor product RTI Connext Micro

R2022a introduces a new out-of-the-box supported vendor product, RTI Connext Micro. Previously, DDS Blockset supported the product RTI Connext. In addition to RTI Connext, DDS Blockset now supports a more compact version of that product with RTI Connext Micro.

For more information about RTI Connext Micro, see RTI Connext Micro. For more information about how to integrate DDS Blockset applications with RTI Connext Micro, see DDS Blockset System Requirements and Deploy DDS Applications.

Configuration of Topics, DataReaders, and DataWriters simplified

Beginning in R2022a, you can open the Code Mappings editor and directly configure the Topic and DataReader or DataWriter for each port. Previously, you had to specify a **Configuration Mode** as `Use Topic and QoS` or `Use Writer/Reader XML path` to indicate the source of the available configuration options. This step has been optimized out of the workflow and the Code Mappings editor has updated its mapping to remove the **Configuration Mode** column. For more information, see Interactively Configure DDS Interface.

QoS configuration available in Code Mappings editor

For easier access and visibility, you can now configure Quality of Service (QoS) within the Code Mappings editor. Previously, QoS properties were configured by using the Property Inspector. For more information, see Interactively Configure DDS Interface and Manage QoS.

Topic filters to selectively receive data

R2022a enables the ability to filter Topics so that applications can receive appropriate data without having to implement additional logic. To apply a filter to a DDS application, use the Property Inspector to configure the properties **Filter Kind**, **Filter Expression**, and **Filter Parameter List**. The filter kind property determines if the filter is specified by using a string or SQL expression. The filter expression and filter parameter list fields are then used to create the filter. For more information, see Interactively Configure DDS Interface.

Default generated C++ class name is the model name

Beginning in R2022a, the default name for a DDS application model class is the name of the model. Previously, the class name in the generated code used a default class name of the form *`model`*`ModelClass`. The new default is of the form *`model`*.

This change to the generated class name may cause integration scripts that use the class names to break. Update integration scripts to the new generated class names. For more information, see Deploy DDS Applications.

Nested namespace support for C++ code generation

You can now customize the generated C++ code from your modeled DDS application to generate the model class in a nested namespace. For more information, see Deploy DDS Applications.

R2021b

Version: 1.1

New Features

Streamlined IDL import for RTI

Previously, to import IDL specifications for RTI based DDS applications you had to convert IDL to XML prior to import. You can now directly import IDL specifications for RTI by using the DDS Application Quick Start. For more information, see Import or Create DDS Definitions.

Easy conversion of Simulink data dictionary data types to DDS data types for Data Samples

You can now convert design data represented in a Simulink data dictionary to a DDS data type that you can use in a DDS Dictionary. In Model Explorer, you can select or create Simulink data types and convert them to DDS data types if you right-click and choose `Convert to DDS Type` from the context menu. For more information, see Manage Types.

Ability to edit QoS profiles in the DDS Dictionary

The DDS Dictionary now enables you to duplicate and delete QoS profiles that are available for your application. For more information, see Manage QoS.

Enhanced modeling pattern for Publishers and Subscribers

For R2021b, you can explicitly model Publishers and Subscribers as virtual buses composed of bus element ports. In this modeling pattern, the bus represents a Publisher or Subscriber and the individual bus element ports represent their DataWriters or DataReaders respectively. This modeling pattern increases the modeling visibility of Publishers and Subscribers and reduces the complexity of simulation.

When you use this modeling pattern, the model elements use these naming conventions:

- Port name: The name of the Publisher or Subscriber
- Bus element name: The name of the Topic for the DataWriter or DataReader

For an example that shows this modeling pattern, see DDS Blockset Shapes Demo.

Improved C++ code generation

For R2021b, the generated C++ code from DDS application models has been updated in the following ways:

- **Class name and namespace control** – You can now control the generated C++ class names and namespaces for your DDS applications.
- **Code generation for RTI uses C++ API** – Previously, code generation for RTI used the C API provided by RTI to connect to DDS. Now the generated code uses the C++ API.
- **String support** – DDS code generation for RTI now supports the string data type.

For more information about DDS code generation, see Deploy DDS Applications.

R2021a

Version: 1.0

New Features

Introducing DDS Blockset

DDS Blockset provides apps and blocks for modeling and simulating software applications that publish or subscribe to Data Distribution Service (DDS) middleware. The blockset includes a DDS dictionary that lets you manage, create, and edit your DDS definitions in Simulink. You can import DDS specifications as XML files to create a skeleton Simulink model as a starting point for developing algorithms for DDS applications.

DDS Blockset provides blocks for publishing and subscribing samples to DDS, including their corresponding Quality of Service (QoS). It fully integrates with the RTI Connext DDS and eProsima DDS stacks.

DDS Blockset generates C++ code and XML files from Simulink models (with Embedded Coder®).

IDL/XML import and export

You can use the DDS Application Quick Start to import IDL/XML DDS definitions or create definitions to get started making DDS applications in the Simulink environment. For more information, see Import or Create DDS Definitions.

You can also export IDL/XML for your modeled application when you build your application model. For more information, see Deploy DDS Applications.

Manage Domains, Topics, Data Samples, and Quality of Service (QoS)

You can use a **DDS Dictionary** to manage the DDS aspects of your applications in the Simulink environment. The DDS Dictionary is an intuitive graphical interface that enables you to quickly create and edit the DDS Domains, Topics, Data Samples, and Quality of Service (QoS) you need to configure the Publisher, Subscriber, DataReader, and DataWriter aspects of your model. For more information, see Manage DDS Definitions.

Model and simulate Publishers and Subscribers

In Simulink, you can use the Take DDS Sample and Write DDS Sample blocks to construct or adapt Simulink models to behave as DDS Publishers and Subscribers. For more information, see Model DDS Applications.

You can use a **Code Mappings Editor** to apply DDS definitions like Domains, Topics, Types, and Quality of Service (QoS) to your modeled Publishers and Subscriber applications. For more information, see Interactively Configure DDS Interface.

Deploy DDS applications with RTI and eProsima and generate C++ code

DDS Blockset supports out-of-the box integration with the DDS vendors RTI and eProsima. Additionally, DDS Blockset supports C++ code generation. (requires Embedded Coder). For more information, see Deploy DDS Applications.