Fast DDS Monitor Documentation

Release 2.0.0

eProsima

Mar 26, 2024

INTRODUCTION

eProsima Fast DDS Monitor is a graphical desktop application aimed at monitoring DDS environments deployed using the *eProsima Fast DDS* library. Thus, the user can monitor in real time the status of publication/subscription communications between DDS entities. They can also choose from a wide variety of communication parameters to be measured (latency, throughput, packet loss, etc.), as well as record and compute in real time statistical measurements on these parameters (mean, variance, standard deviation, etc.).

Furthermore, the user can check the status of the deployed DDS network at any time, i.e. see for each DDS Domain which DomainParticipants are instantiated, as well as their publishers and subscribers and the topics under which they publish or to which they subscribe respectively. It is also possible to see the physical architecture of the network on which the DDS applications that use *Fast DDS* are running.

eProsima Fast DDS Monitor is designed to meet the following criteria:

1. **Monitoring**: real-time tracking of network status and DDS communication.

2. **Intuitive**: graphical user interface developed following a user experience design approach.

3. **Introspection**: easily navigate through the deployed and active DDS entities being able to inspect their configuration and physical deployment.

4. **Troubleshooting**: detect at a glance the possible issues or anomalous events that may occur in the communication.

Warning: In order to monitor a DDS network deployed using *Fast DDS* library, compiling the latter with statistics and explicitly activating the statistics module is required. See *Fast DDS with Statistics module* for more details.

Warning: If Fast DDS has been compiled with statistics and they are explicitly enabled and statistical data are not correctly received, only few data arrive or even none, configure the Fast DDS endpoints publishing statistics data with a less restrictive memory constraints. Please check the following documentation for more details on how to do this.

Find more about us at eProsima's webpage.

Support available at:

- Email: support@eprosima.com

- Phone: +34 91 804 34 48

CONTRIBUTING TO THE DOCUMENTATION

Fast DDS Monitor Documentation is an open source project, and as such all contributions, both in the form of feedback and content generation, are most welcomed. To make such contributions, please refer to the Contribution Guidelines hosted in our GitHub repository.

TWO

STRUCTURE OF THE DOCUMENTATION

This documentation is organized into the sections below.

- *Installation Manual*
- *Getting Started*
- *User Manual*
- *Developer Manual*
- *Release Notes*

eProsima Fast DDS Monitor is a graphical desktop application aimed at monitoring DDS environments deployed using the *eProsima Fast DDS* library. Thus, the user can monitor in real time the status of publication/subscription communications between DDS entities. They can also choose from a wide variety of communication parameters to be measured (latency, throughput, packet loss, etc.), as well as record and compute in real time statistical measurements on these parameters (mean, variance, standard deviation, etc.).

Furthermore, the user can check the status of the deployed DDS network at any time, i.e. see for each DDS Domain which DomainParticipants are instantiated, as well as their publishers and subscribers and the topics under which they publish or to which they subscribe respectively. It is also possible to see the physical architecture of the network on which the DDS applications that use *Fast DDS* are running.

eProsima Fast DDS Monitor is designed to meet the following criteria:

1. **Monitoring**: real-time tracking of network status and DDS communication.

2. **Intuitive**: graphical user interface developed following a user experience design approach.

3. **Introspection**: easily navigate through the deployed and active DDS entities being able to inspect their configuration and physical deployment.

4. **Troubleshooting**: detect at a glance the possible issues or anomalous events that may occur in the communication.

Warning: In order to monitor a DDS network deployed using *Fast DDS* library, compiling the latter with statistics and explicitly activating the statistics module is required. See *Fast DDS with Statistics module* for more details.

> **Warning:** If Fast DDS has been compiled with statistics and they are explicitly enabled and statistical data are not correctly received, only few data arrive or even none, configure the Fast DDS endpoints publishing statistics data with a less restrictive memory constraints. Please check the following documentation for more details on how to do this.

2.1 Contacts and Commercial support

Find more about us at eProsima's webpage.

Support available at:

- Email: support@eprosima.com
- Phone: +34 91 804 34 48

2.2 Contributing to the documentation

Fast DDS Monitor Documentation is an open source project, and as such all contributions, both in the form of feedback and content generation, are most welcomed. To make such contributions, please refer to the Contribution Guidelines hosted in our GitHub repository.

2.3 Structure of the documentation

This documentation is organized into the sections below.

- *Installation Manual*
- *Getting Started*
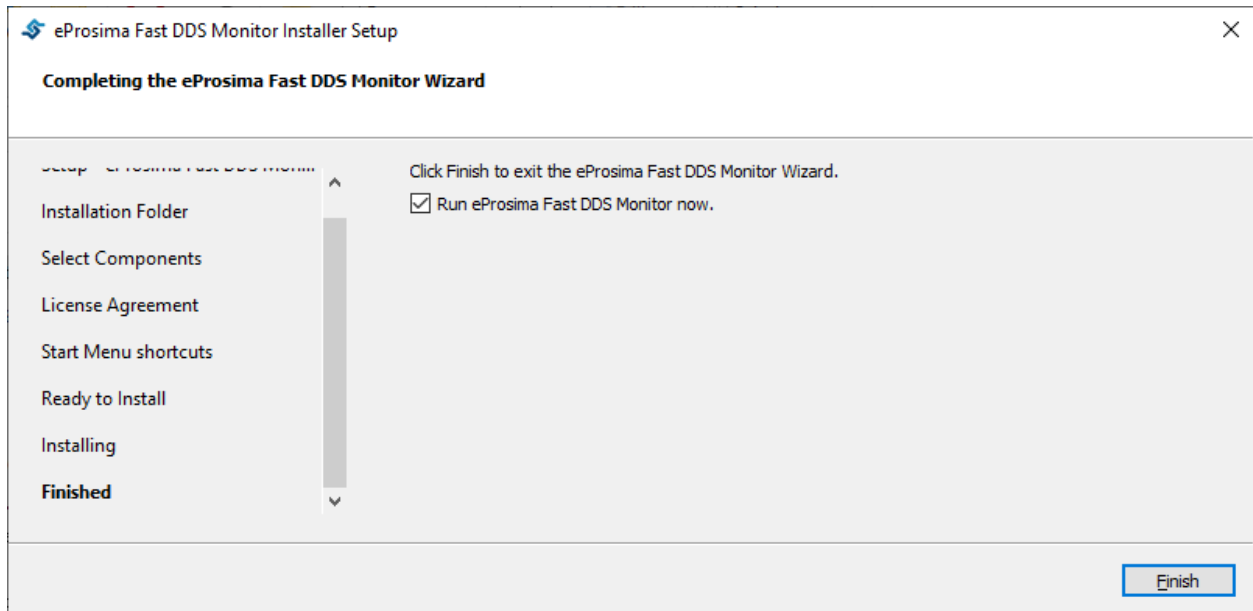- *User Manual*
- *Developer Manual*
- *Release Notes*

2.4 Fast DDS Monitor on Windows

This section provides instructions on how to install the *Fast DDS Monitor* application. This is available on the eProsima website in the Downloads section.

In the list of eProsima official releases, search for *eProsima Fast DDS Monitor* and go to the available files of the latest version available. Then click on the **Download now** button of the Windows installer (*eProsima Fast DDS Monitor x.x.x - Win (32 & 64)*).

Now locate the downloaded file and run the installer. The *Fast DDS Monitor* installer window should open as shown in the following image.

Follow the installation steps until the installation process is complete. A window as the one shown below should be visible on screen after installation, from which you may directly run the *Fast DDS Monitor* application.
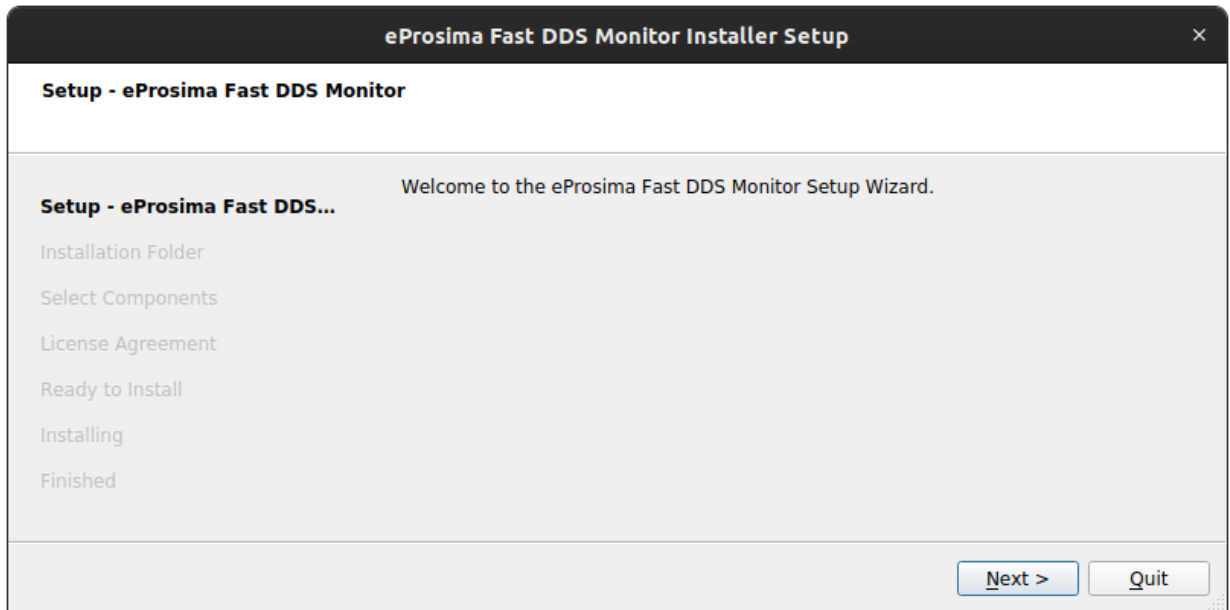
2.5 Fast DDS Monitor on Linux

This section provides instructions on how to install the *Fast DDS Monitor* application. This is available on the eProsima website in the Downloads section.

There are two mechanisms for running the monitor application.

- Through the *Fast DDS Monitor* installer.

- Using the *AppImage* format, which is a portable format of the application software.

2.5.1 *Fast DDS Monitor* installer

The first option is to install the *Fast DDS monitor* application together with all its dependencies. To do so, first run the `FastDDSMonitorInstaller.run` executable and follow the instructions it provides to install the program in a directory on the system.

2.5.2 *Fast DDS Monitor* portable format

eProsima also distributes a portable version of the *Fast DDS Monitor* for Linux in AppImage format. In this case, download this version from the eProsima Downloads website and run the downloaded file to launch the monitor. The name of this file is `Fast_DDS_Monitor-x86_64-X.X.X.AppImage`, where `X.X.X` is the version of the application.

> **Warning:** In case these files are not executed, check that they have executable permissions.

2.6 Entities

The monitoring functions of this application relay on tracking the activity of certain *Entities* retrieved by the *Fast DDS Monitor*, as well as on storing connections and interchanged data between them for their presentation to the user. These entities represent DDS communication entities, or different physical elements related to DDS entities.

In the following diagram one can see the different kinds of entities being tracked by the monitor. The arrows in this schema represent a direct connection (not a 1:n in every case) between both kinds of entities. Those entities (or types of entities) that do not have a direct relationship are also related, as intermediate entities make it possible to relate any entity in the graph to one another indirectly, i.e. one *DomainParticipant* is related with its *User* by the relation with its *Process*.

2.6.1 DDS Entities

These entities represent the DDS entities that manage the communication. That is, the *DomainParticipants*, *DataWriters* and *DataReaders*. Each *DataReader/DataWriter* has one or more associated *Locator* entities. *Locators* are the network addresses through which *DataReaders/DataWriters* communicate in a DDS network.

For further information about each entity, please refer to the DDS specification or visit the Fast DDS Documentation.

DomainParticipant

DomainParticipant is the main entity in the DDS protocol. It represents a collection of *DataReaders/DataWriters*, and manage the whole DDS Discovery of other *DomainParticipants* and *DataReaders/DataWriters* within the DDS Domain to which it belongs. Refer to DomainParticipant Fast DDS Documentation for a more detailed explanation of the *DomainParticipant* entity in DDS.

Each *DomainParticipant* can only communicate under one *Domain* (see *Logical Entities*) and so it exists a direct connection between each *DomainParticipant* and the *Domain* in which it works. From image `fig_entities_diagram` it can be seen that *DomainParticipant* entities are contained in a *Process*, this is because a system process (so-called *Process* entity) executes an application using *Fast DDS* that instantiates *DomainParticipants*. The same applies to *DataReaders/DataWriters* instantiated by a *DomainParticipant* belonging to a specific *Process*. Therefore, a *Process* can contain as many DDS entities as the *Fast DDS* application running in that *Process* has instantiated.

DataWriter

DataWriter is the DDS entity in charge of publishing data. Each *DataWriter* is directly contained in a single *DomainParticipant*. In addition, since a *DataWriter* is associated to the *Topic* under which it publishes, it is possible to define a direct containment relationship of a *DataWriter* in a *Topic*. Thus, the *Topic* will contain all the *DataWriters* that are publishing under it.

Therefore, each *DataWriter* is directly connected with the *DomainParticipant* it belongs, and with the *Topic* under which it publishes. Also, a *DataWriter* is associated with one or multiple *Locators* that would represent the physical communication channel this *DataWriter* is using to send data.

DataReader

DataReader holds the subscribe function of the communication. As for the *DataWriter*, each *DataReader* is directly contained in a single *DomainParticipant*. In addition, since a *DataReader* is associated to the *Topic* to which it is subscribed, it is possible to define a direct containment relationship of a *DataReader* in a *Topic*. Thus, the *Topic* will contain all the *DataReaders* that are subscribed under it. Therefore, each *DataReader* is directly connected with the *DomainParticipant* to which it belongs, and with the *Topic* to which it is subscribed.

Locator

Locator represents the physical address and port that a *DataReader/DataWriter* uses to send or/and receive data. This entity is related with the physical division of the entities, as a *Locator* belongs to a unique *Host* (see section *Physical Entities*). However, the monitor treats this entity as a *DDS Entity* in order to simplify the entities' connection and improve comprehensibility.

A *Locator* is connected with one or multiple *DataReader/DataWriter*, and so it is related with a *Host* by relating each of these *DataReader/DataWriter* with a *DomainParticipant* and each *DomainParticipant* with its *Host*.

2.6.2 Logical Entities

Domain

Domain represents a logical abstraction in DDS protocol that divides the DDS network into partitions, making each *Domain* completely independent and unaware of others. This logical partition depends on the chosen discovery protocol.

In case of using *Simple Discovery Protocol* (discovery protocol by default in Fast DDS) as the default discovery mechanism, the *Domain* will be represented by a number (domain ID), and every DDS entity in that *Domain* will discover the rest of entities deployed on the same *Domain*. In case of using *Discovery Server* as discovery protocol, the partition will be made by the *Discovery Server* or *Discovery Servers Net* to which the monitor connects. Please refer to Fast DDS documentation for more information about this feature. Each entity connected to a *Discovery Server* on the same network will know all other entities with which it needs to communicate.

This entity inside the monitor is related with the *DomainParticipants* that communicate under this same *Domain*, and the *Topics* created in this *Domain*.

Topic

Topic is an abstract DDS entity that represents the channel of communications between publishers and subscribers. Every *DataReader* subscribed to a *Topic* will receive the publications of every *DataWriter* publishing under the same *Topic*.

This entity inside the monitor is directly connected with the *Domain* it belongs to, and with the *DataReaders/DataWriters* communicating under this *Topic*.

2.6.3 Physical Entities

Host

Host makes reference to the physical (or logical i.e. Docker) machine where one or more DDS *Processes* are running. This entity is connected directly with the *Users* running in this *Host*.

User

User makes reference to the different users that could run in a *Host*.

This entity is connected directly with the *Host* it belongs to, and the running *Processes* that are being executed within this *User*.

Process

Process represents each process running an application using *Fast DDS*. Be aware that it is possible to run more than one *DomainParticipant* in the same *Process*, and they do not require to be related with each other, not even under the same *Domain*.

This entity is connected directly with the *User* to which it belongs, and the *Participants* running within it.

2.7 Monitor Domain

This application is able to track different *Entities* that belongs to the DDS communication protocol or are, in some sense, related to these entities. The DDS communication protocol (see DDS specification) divide a DDS network in independent partitions referred to as *Domain*. This Domains must fulfilled that only the entities in the same Domain can discover and communicate with each other, and so this depends on the *Discovery Protocol* that is being use. This application implements two different *Discovery Protocols* that could be use to monitor entities.

Several Domains could be monitored at the same time and the *Logical Entities* and the *DDS Entities* under one Domain would never be shared between others (except for *Locator*). This is not the case with the *Physical Entities* that could be shared between entities in different domains, as the same *Host*, or even the same *Locator* could be related with entities in several Domains.

For the propose of monitoring a Domain, the application gives the button *Initialize Monitor* where a user can manually specify the configuration of a Discovery type. Once a Monitor is initialized in a specific Domain, the entities in this Domain will start to be discovered and its data collected. Every new entity or data discovered will be notified as a callback in *Issues Panel*.

Note: Be aware that the discovery of the *Fast DDS Monitor* entities is made by DDS protocols, and so they will not be instantaneous or simultaneous.

2.7.1 Simple Discovery Monitor

The DDS Simple Discovery Protocol (SDP) remains on the discovery of individual entities by multicast communication. No previous information about the network and its architecture is needed in order to create a new monitor that connects with the *Participants* already running in the same network. In order to configure this kind of Domain monitoring, it is only needed to specify the number of the Domain that is going to be tracked

2.7.2 Discovery Server Monitor

The Discovery Server discovery protocol is a *Fast DDS* feature that allows to centralize the discovery phase in a single or a network of *Discovery Servers*. This has been demonstrated to be very useful in order to reduce the discovery traffic and to avoid certain problems that could appear with the Simple Discovery Protocol and multicast.

In order to configure this kind of Domain monitoring, it is required to insert a string with different network addresses. This string consist in one or several addresses in the format of `ip_address:port`. Each of these are IP-port pair where a Discovery Server is listening. Each network address is separated with `;`. It is only needed to successfully connect to one of the addresses set in the parameter, as a network of Discovery Servers interconnected creates a redundancy that makes the network more robust, but it is not required to connect with all of the Servers in it.

The following command shows an example on how to connect with one Discovery Server in your own localhost listening in port `11811`, one in the same local network in address `192.168.1.2:12000` and a third one in an external network in address `8.8.8.8:12345`.

```
"127.0.0.1:11811;192.168.1.2:12000;8.8.8.8:12345"
```

In order to clarify how to set this parameter, please visit the Discovery Server CLI tutorial. The parameter of the Discovery Server *Init New Monitor* button in this application will be used equally as the input to the CLI command.

> **Warning:** Due to the designed architecture for the communication of the Monitor application and the DDS entities, it is highly recommended not to initialize a *Discovery Server Monitor* with servers in different *Discovery Server Networks* (do not connect a monitor to servers that are not connected).

> **Warning:** Do not initialize a *Discovery Server* monitoring in a *Discovery Server Network* where another *Discovery Server* is already been monitored. This will lead to an undefined behavior.

2.8 Example of usage

This example will show how to monitoring a DDS network using *Fast DDS Monitor* and how to understand the different application features and configurations.

2.8.1 Fast DDS with Statistics module

In order to show the *Fast DDS Monitor* running and monitoring a real DDS network, this tutorial uses a *Fast DDS* example to create a simple and understandable DDS network. The example proposed by this tutorial is using `DDSHelloWorldExample` of *Fast DDS* repository. **Be aware that the statistics module is not compiled and used by default by Fast DDS but it has to be specifically configured to send statistical data of an specific entity.**

In order to execute this minimum DDS scenario where each entity publish its statistical data, follow these steps:

1. Compile *Fast DDS* library with CMake option `FASTDDS_STATISTICS` to activate the statistics module (`-DFASTDDS_STATISTICS=ON`).

2. Compile *Fast DDS* library with CMake option `COMPILE_EXAMPLES` to build the examples (`-DCOMPILE_EXAMPLES=ON`).

3. Have *Fast DDS Monitor* installed or a working environment with *Fast DDS*, *Fast DDS Statistics Backend* and *Fast DDS Monitor* built.

4. Use the environment variable `FASTDDS_STATISTICS` to activate the statistics writers in the DDS execution (see following section).

For further information about the Statistics configuration, please refer to Fast DDS statistics module. For further information about installation of the Monitor and its dependencies, please refer to the documentation section *Fast DDS Monitor on Linux* or *Linux installation from sources*.

Hello World Example

For this tutorial it has being used the *Fast DDS* `DDSHelloWorldExample` to show a minimum DDS network being monitoring. Below are the commands executed in order to run this network. Nevertheless, this tutorial does not start with this DDS network running but it is executed once the monitor has been started. This does not change the Monitor behavior, but would change the data and information shown by the application.

1. Execute a *Fast DDS* `DDSHelloWorldExample` **subscriber** with statistics data active.

```
export FASTDDS_STATISTICS="HISTORY_LATENCY_TOPIC;NETWORK_LATENCY_TOPIC;PUBLICATION_
↪THROUGHPUT_TOPIC;\
SUBSCRIPTION_THROUGHPUT_TOPIC;RTPS_SENT_TOPIC;RTPS_LOST_TOPIC;\
HEARTBEAT_COUNT_TOPIC;ACKNACK_COUNT_TOPIC;NACKFRAG_COUNT_TOPIC;\
GAP_COUNT_TOPIC;DATA_COUNT_TOPIC;RESENT_DATAS_TOPIC;SAMPLE_DATAS_TOPIC;\
PDP_PACKETS_TOPIC;EDP_PACKETS_TOPIC;DISCOVERY_TOPIC;PHYSICAL_DATA_TOPIC"

./build/fastrtps/examples/C++/DDS/HelloWorldExample/DDSHelloWorldExample subscriber
```

where `subscriber` argument creates a *DomainParticipant* with a *DataReader* in the topic `HelloWorldTopic` in *Domain* `0`.

2. Execute a *Fast DDS* `DDSHelloWorldExample` **publisher** with statistics data active.

```
export FASTDDS_STATISTICS="HISTORY_LATENCY_TOPIC;NETWORK_LATENCY_TOPIC;PUBLICATION_
↪THROUGHPUT_TOPIC;\
SUBSCRIPTION_THROUGHPUT_TOPIC;RTPS_SENT_TOPIC;RTPS_LOST_TOPIC;\
HEARTBEAT_COUNT_TOPIC;ACKNACK_COUNT_TOPIC;NACKFRAG_COUNT_TOPIC;\
GAP_COUNT_TOPIC;DATA_COUNT_TOPIC;RESENT_DATAS_TOPIC;SAMPLE_DATAS_TOPIC;\
PDP_PACKETS_TOPIC;EDP_PACKETS_TOPIC;DISCOVERY_TOPIC;PHYSICAL_DATA_TOPIC"

./build/fastrtps/examples/C++/DDS/HelloWorldExample/DDSHelloWorldExample publisher
↪0 500
```

where `publisher` argument creates a *DomainParticipant* with a *DataWriter* in the topic `HelloWorldTopic` in *Domain* `0`. The following arguments indicate this process to run until the user press `enter` (`0` samples) and to write a message every half second (`500` milliseconds period).

Statistics topics

The environment variable `FASTDDS_STATISTICS` activates the statistics writers for a *Fast DDS* application execution. This means that the *DomainParticipants* created within this variable will report the statistical data related to them and their sub-entities.

For this example, only some of the available topics has been activated. This means that each participant will only report this specific data and it will not store or send any other data. This is very useful in order to limit the network traffic, as sending all the data highly increase the amount of DDS data sent.

The topics that are going to be reported by this example are:

- **HISTORY_LATENCY_TOPIC**: Reports the latency of the messages between the two entities.

- **NETWORK_LATENCY_TOPIC**: Reports the network latency of the messages between the two entities.

- **PUBLICATION_THROUGHPUT_TOPIC**: Reports the publication throughput of the user's DataWriters.

- **SUBSCRIPTION_THROUGHPUT_TOPIC**: Reports the subscription throughput of the user's DataReaders.

- **RTPS_SENT_TOPIC**: Reports the number of RTPS packets and bytes being sent by each DDS entity.

- **RTPS_LOST_TOPIC**: Reports the number of RTPS packets and bytes that are being lost in the transport layer (dropped somewhere in between) in the communication between each DDS entity and locator.

- **HEARTBEAT_COUNT_TOPIC**: Reports the number of heartbeat messages sent by each user's DataWriter.

- **ACKNACK_COUNT_TOPIC**: Reports the number of ACKNACK messages sent by each user's DataReader.

- **NACKFRAG_COUNT_TOPIC**: Reports the number of NACKFRAG messages sent by each user's DataReader.

- **GAP_COUNT_TOPIC**: Reports the number of gap messages sent by each user's DataWriter.

- **DATA_COUNT_TOPIC**: the total number of user's data messages and data fragments (in case that the message size is large enough to require RTPS fragmentation) that have been sent by each user's DataWriter.

- **RESENT_DATAS_TOPIC**: Reports the total number of user's data messages and data fragments (in case that the message size is large enough to require RTPS fragmentation) that have been necessary to resend by each user's DataWriter.

- **SAMPLE_DATAS_TOPIC**: the number of user's data messages (or data fragments in case that the message size is large enough to require RTPS fragmentation) that have been sent by the user's DataWriter to completely deliver a single sample.

- **PDP_PACKETS_TOPIC**: Reports the number of PDP discovery traffic RTPS packets transmitted by each DDS DomainParticipant.

- **EDP_PACKETS_TOPIC**: Reports the number of EDP discovery traffic RTPS packets transmitted by each DDS DomainParticipant.

- **DISCOVERY_TOPIC**: Reports the time when each local DomainParticipant discovers any remote DDS entity.

- **PHYSICAL_DATA_TOPIC**: Reports the physical data of the participant. This will make possible to see in which hosts and context each entity is running.

Please refer to Fast DDS documentation for further information about these topics.

2.8.2 Fast DDS Monitor Execution

In the following section is presented a full *Fast DDS Monitor* execution monitoring a real DDS network.

Initial Window

The Monitor starts with non DDS entities running. First of all, the *Fast DDS Monitor* initial window is shown. Press `Start monitoring!` in order to enter the application and start the monitoring.
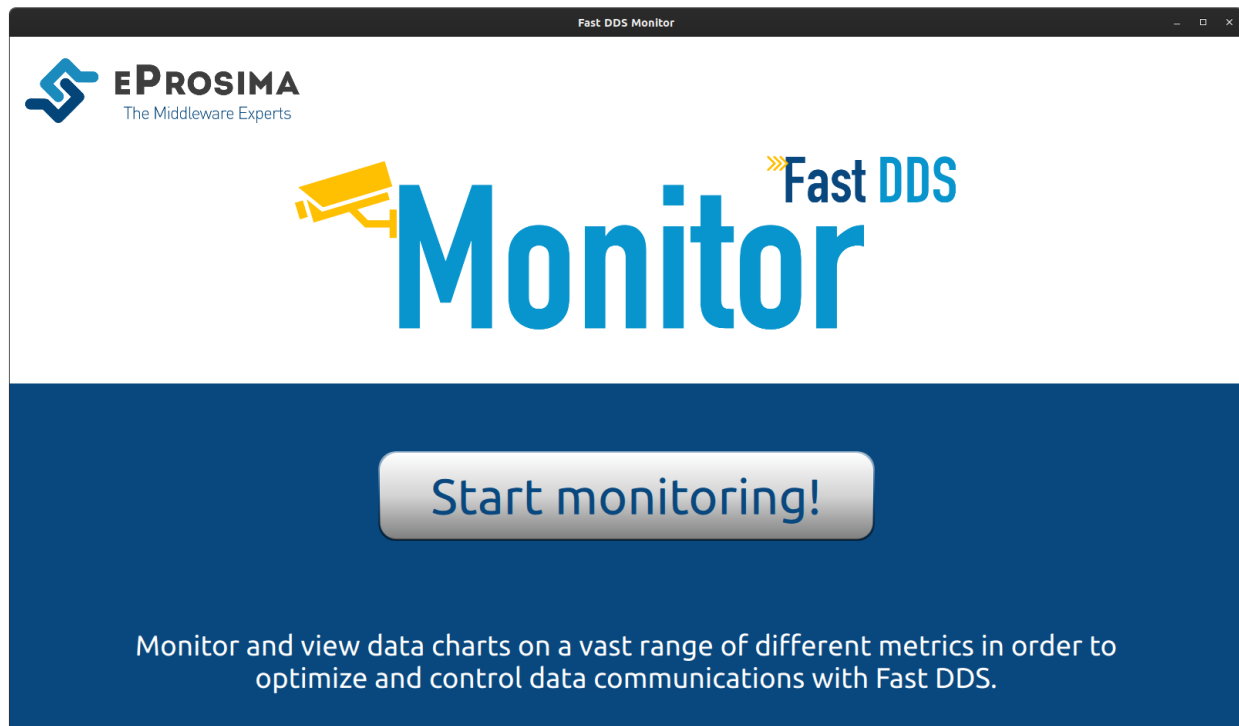

Initiate monitoring

Once in the application, the first Dialog that appear ask the user to insert a domain in order to start monitoring that domain. Monitoring a domain means to listen in that domain for DDS entities that are running and reporting statistical data. Please refer to section *Monitor Domain* for further information.

First, the `Cancel` button is pressed so the user can see around the monitor and check its configurations, but no entities or data will be shown as there are not domains being monitored. You can always return to the *Initialize Monitoring* dialog from *Edit* or *Shortcuts Bar*.

Let's initialize monitoring in **domain 0** and pressing `OK`.
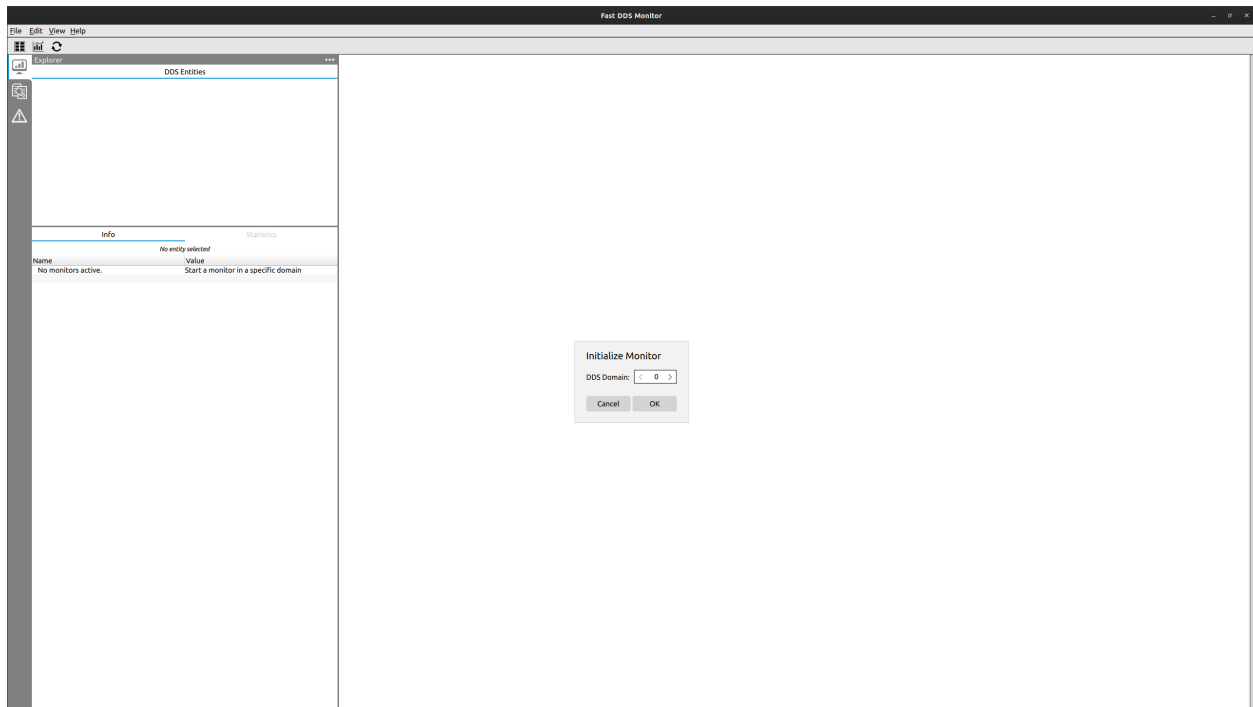
Add physical and logical panels

By default, the Monitor only displays the DDS panel which lists the DDS entities together with their configuration and available statistics information. In order to open the logical and the physical panels, click on the top right corner of the `Explorer` panel, in button ⋯ and add all the panels to visualize the whole information.

At this point, you are going to see the whole window of the application. You should be able to see how an unique entity is present in the application in the left sidebar. This is the domain that you have just initiated. Once a domain is initiated, it is set as *Selected Entity* and so its information is shown in the *Monitor Status Panel*.

For specific details on how the information is divided and where to find it, please refer to *User Manual*.

Execute subscriber

Now, execute the first DDS entity in our DDS network: a *DomainParticipant* with one *DataReader* in topic `HelloWorldTopic` in domain `0` following the steps given in *Hello World Example*. Once the subscriber is running our window will update and you could see new information in the left sidebar.

First of all, the number of entities discovered has increased. Now, you have a *DomainParticipant* called `Participant_sub`, that holds a *DataReader* called `DataReader_HelloWorldTopic_0.0.1.4`. This *DataReader* has a locator, which will be the *Shared Memory Transport* locator. That is because the Monitor and the *DomainParticipant* are running in the same host, and so they communicate using the Shared Memory Transport (SHM) protocol.

You should be able to see as well that now *Host* exists, with a *User* and a *Process* where `Participant_sub` is running. This information is retrieved by the *DomainParticipant* thanks to activate the `PHYSICAL_DATA_TOPIC`. There is also a new *Topic* `HelloWorldTopic` under *Domain* `0`.

Making double click in any entity name we could see its specific information, such as name, backend id, QoS, etc.
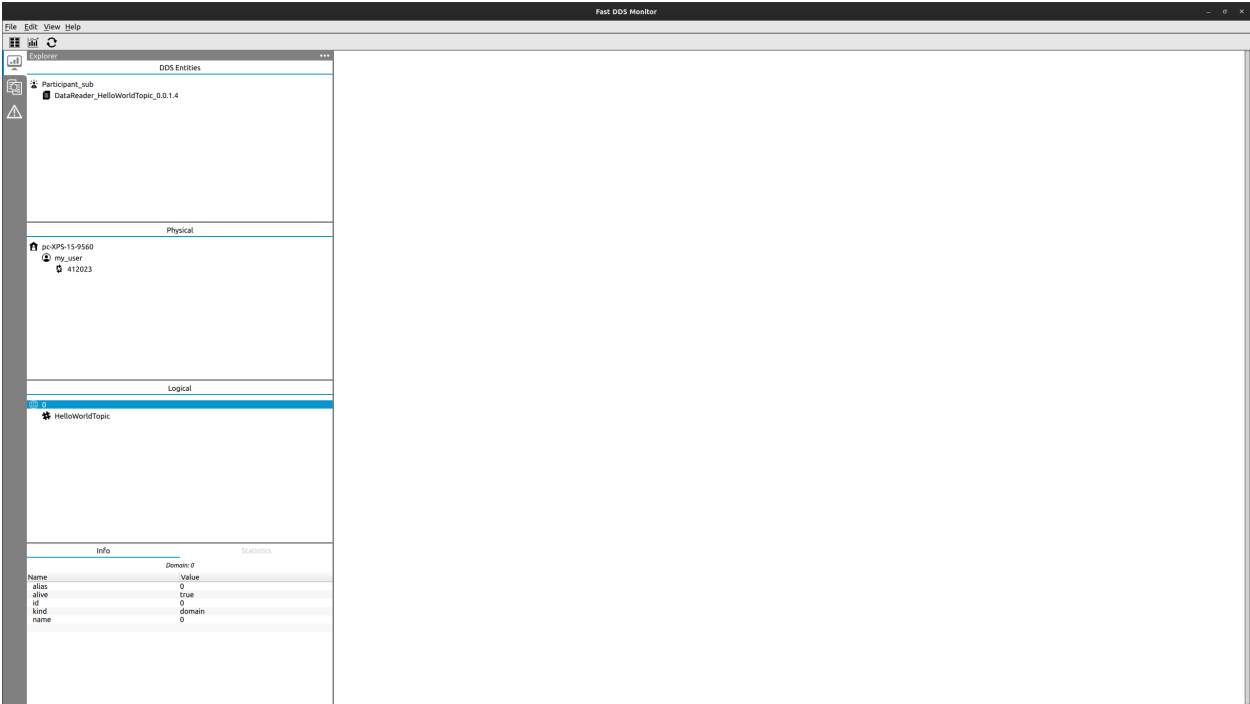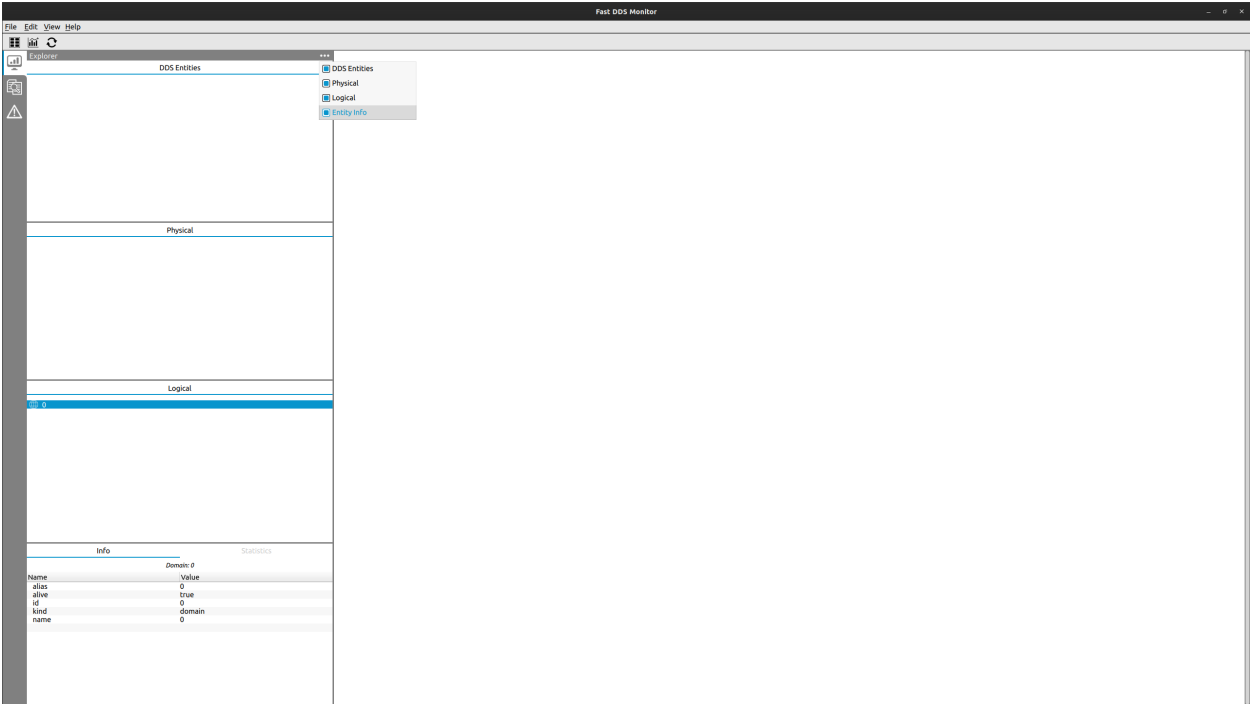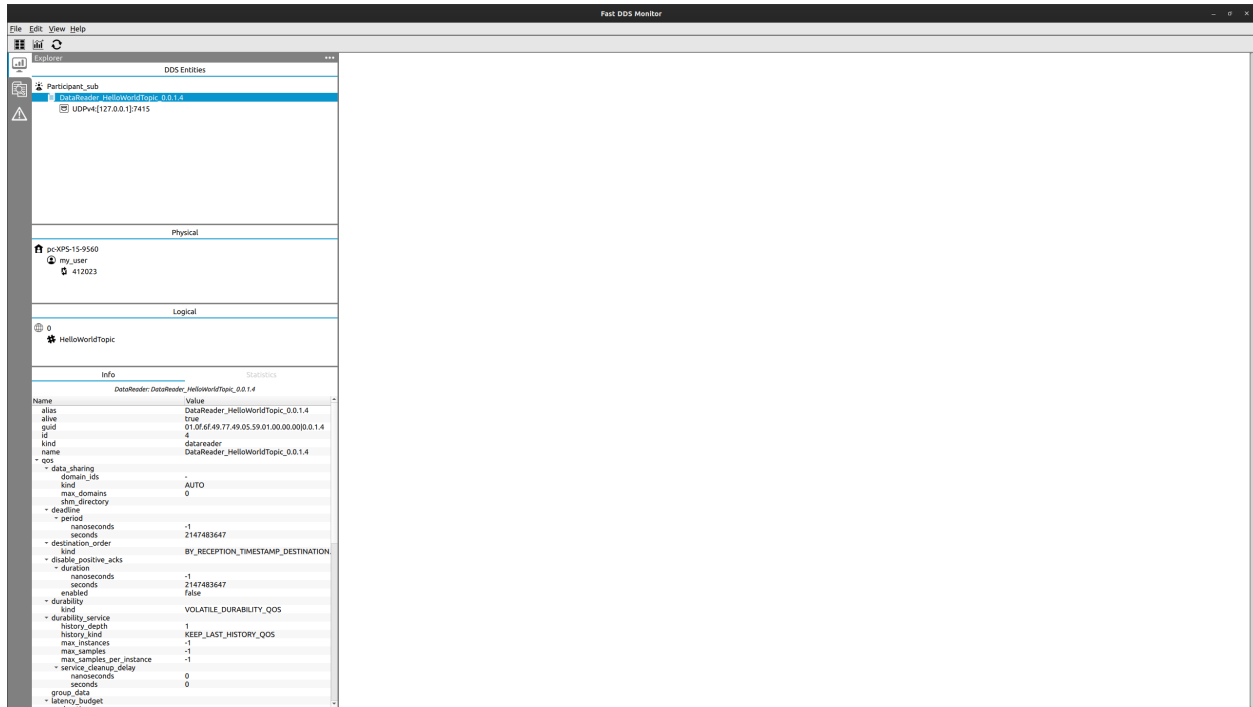
Execute publisher

The next step is to execute a publisher in topic `HelloWorldTopic` in domain `0`, following the steps given in *Hello World Example*. Once the publisher is running you will see that new entities has appeared. Specifically, a new *DomainParticipant* `Participant_pub` with a *DataWrtier* `DataWriter_HelloWorldTopic_0.0.1.3` and, in the case that this publisher has been executed from same *Host* and *User*, there will be a new *Process* that represents the process where this new `Participant_pub` is running.

Discovery Callbacks

Whenever a new entity is discovered by the Monitor there is an event registered in *Log Panel*. Checking this panel you could see the time when every entity that is being monitoring has been discovered. This panel is refreshed automatically whenever a new entity is discovered. Using the *Refresh* this information is cleared until the current moment.

Summary of Statistical Data

In *Statistics Panel* you can see the main information retrieved by each entity. This panel shows a summary of the data retrieved by the entity that is clicked. In this case, you could only see those data that the entities are publishing, and so the rest of the *DataKinds* that are related with the topics that we are not using will remain without data.
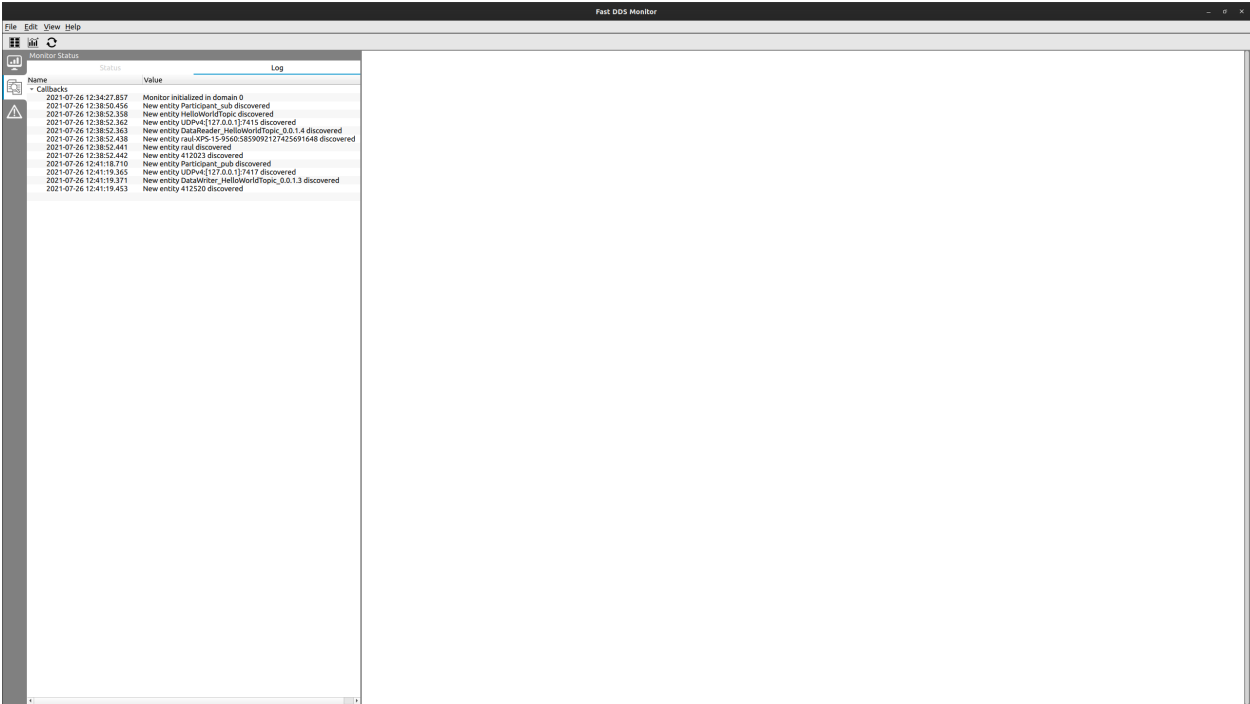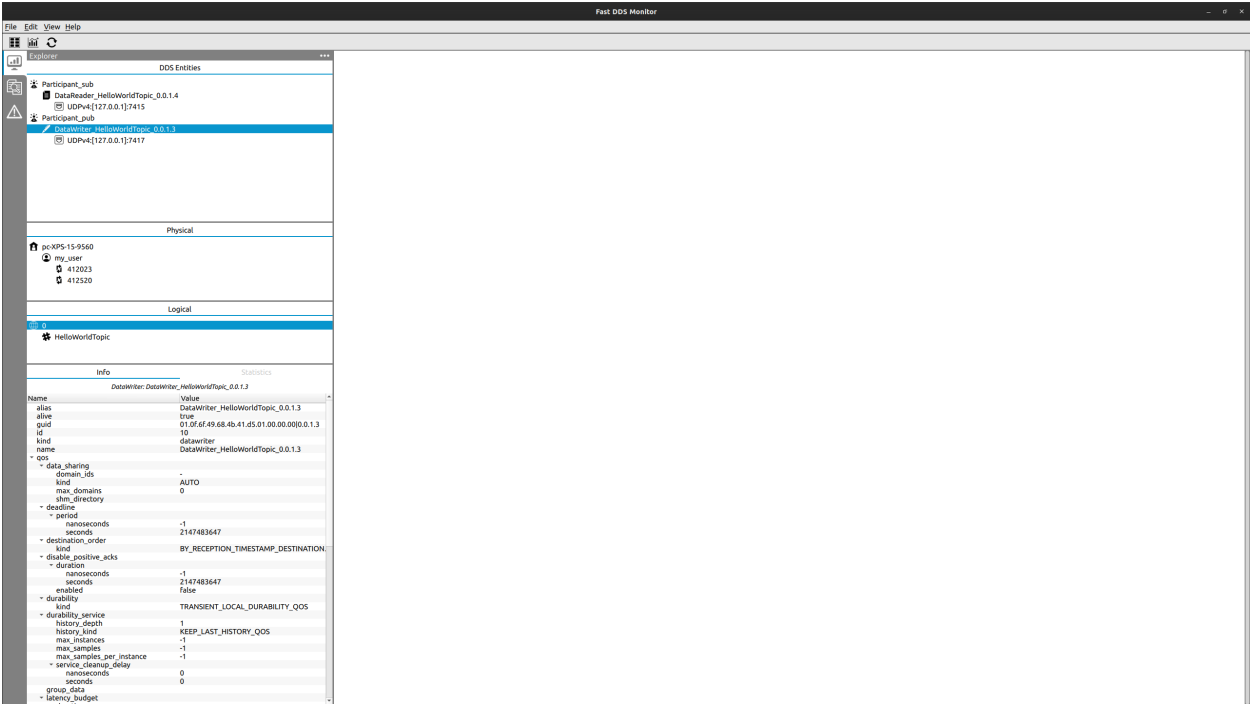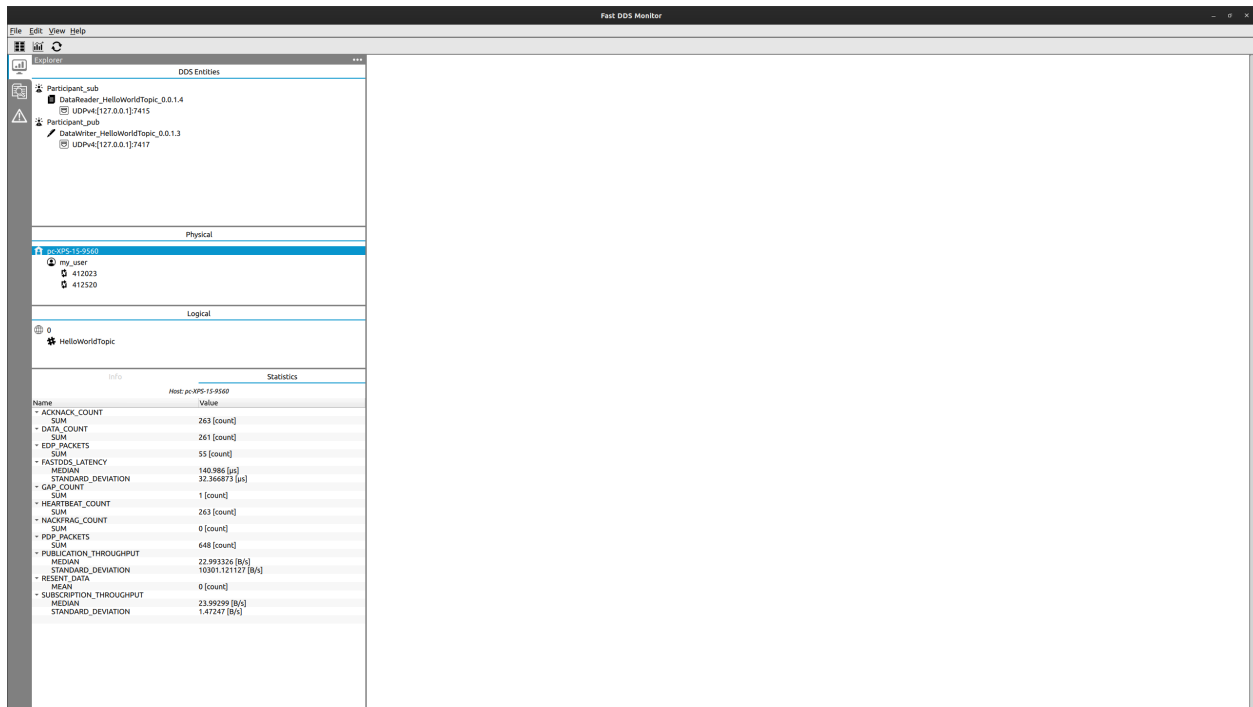
Change entity alias

In order to make the user experience easier, there is a possibility to change the name of an specific entity. We are going to change the name of our *Host* to make it easier to identify. For that, just do right-click over the entity name and menu with the available options for that entity will popup. Click on *Change alias* to re-name the entity.

Set the new alias that you want for this entity. From now on this name will be used all along the monitor.

Note: Be aware that this changes the alias of the entity inside the monitor, and does not affect to the real DDS network.


### Create Historic Series Chart

This section describes how to graphically represent data reported by a DDS network.

Data Count Plot

This section explains how to represent the data being monitored and retrieved by the DDS entities. First of all, go to *Edit->Display Historical Data*. This will open a Dialog where you should choose one of the topics in which you want to see the data collected. The `DATA_COUNT` has been chosen for this tutorial.

Once done this, a new Dialog will open asking you to configure the series that is going to be displayed. In the case of `DATA_COUNT`, the data belongs to the *DataWriter*, and so you should choose this entity in the `Source Entity Id:` checkbox. The `Number of bins` is the number of points in which our data is going to be stored. In our case, we are going to use `20` bins. Selecting the `Default initial timestamp` as the `Start time`, the initial timestamp shall be the time at which the monitor was executed. Using `Now` in option `End time` will get all the data available until the moment the chart is created. Now for the `Statistics kind` option, we are going to use `SUM` as we want to know the number of data sent in each time interval.
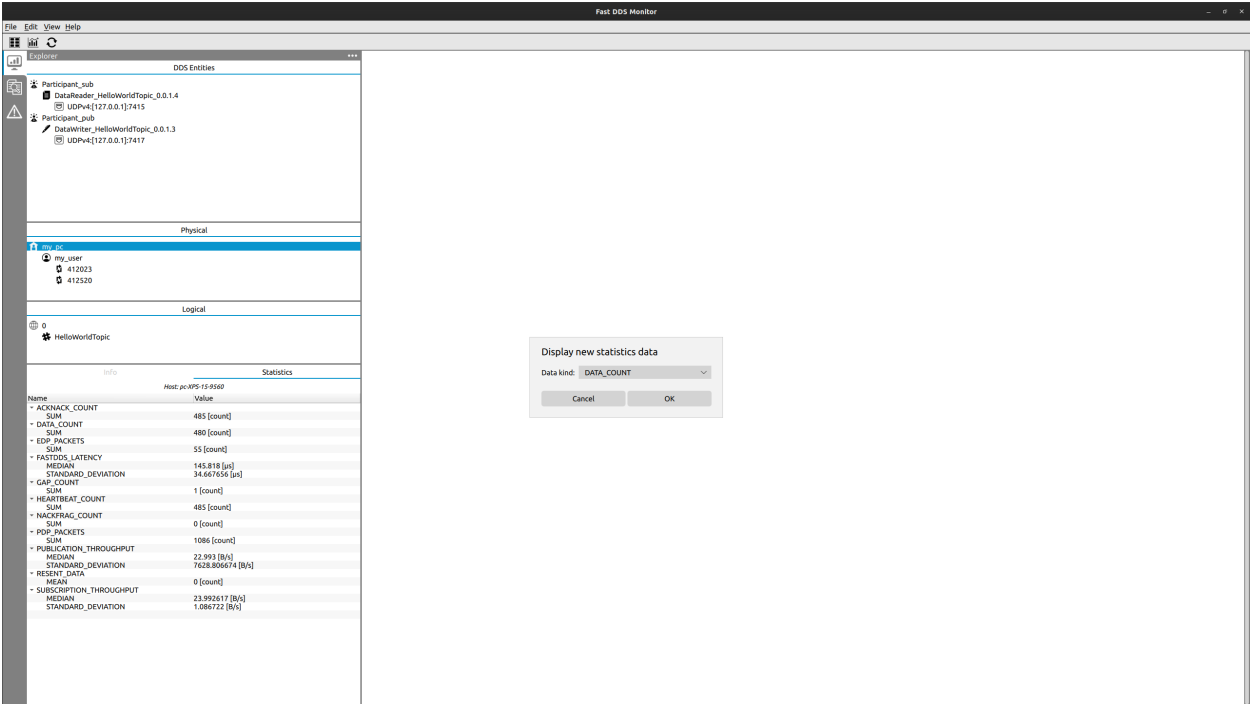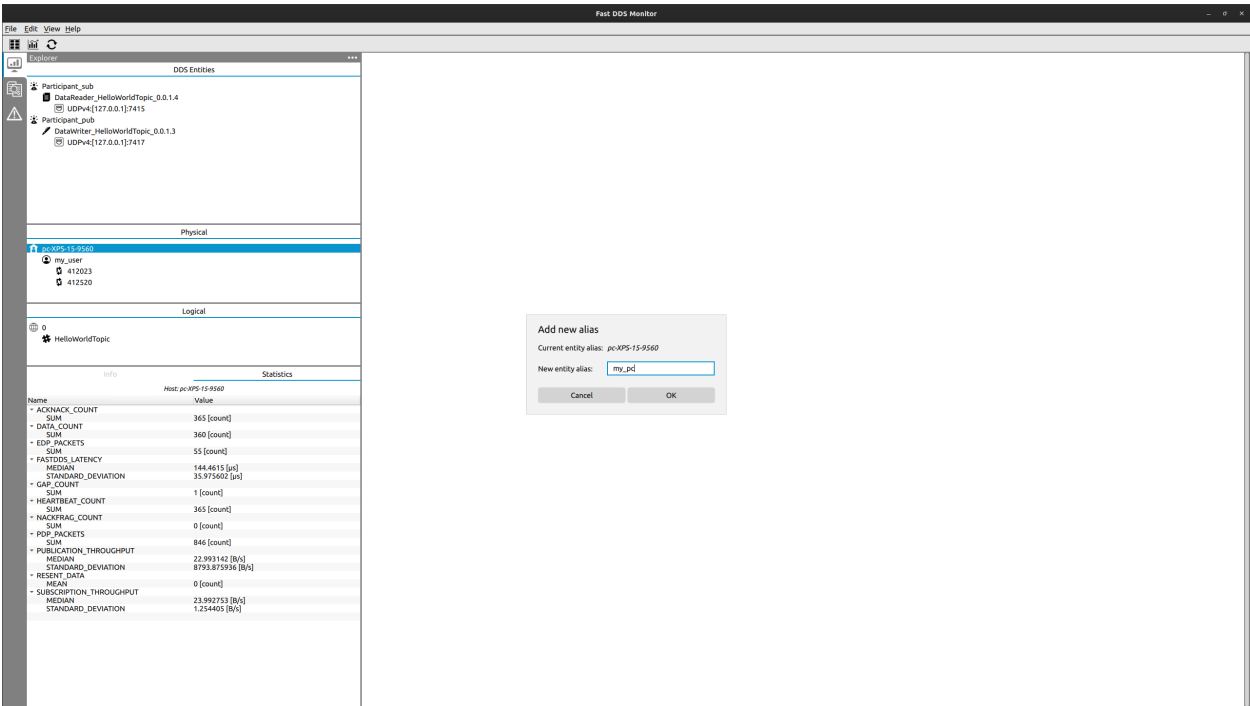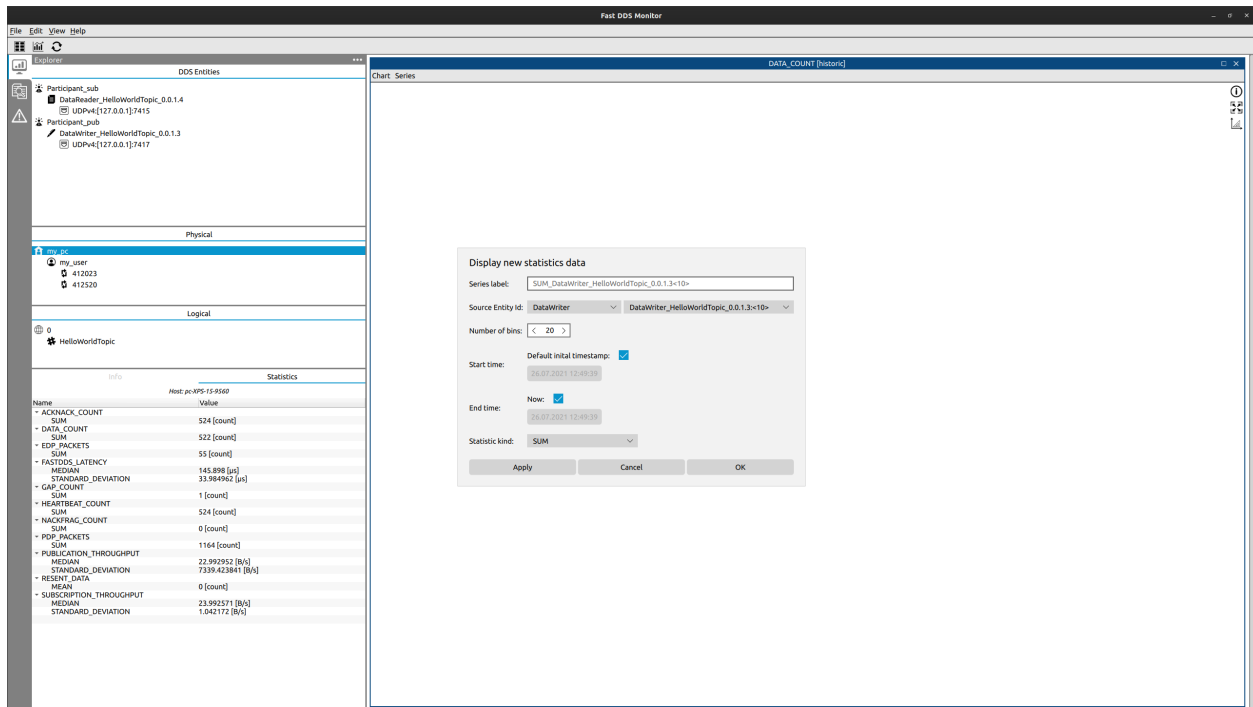
Clicking `Apply` the series will be created in the main window, but the Dialog will no close. This is very useful in order to create a new series similar to the one already created. Here we are going to reuse all the information but we are going to change the `Number of bins` to `0`. Using the value `0` means that we want to see all the different *datapoints* that the writer has stored. Be aware that option `Statistics kind` do not have effect when `Number of bins` is `0`. Then, click `OK` and now you should be able to see both series represented in the `DATA_COUNT` window.

In this new chart created, you could see in the blue series the total amount of data packages sent in each time interval. Each points means that from that timestamp until the timestamp of next point the publisher sent `10/11` messages. The green series reports that this data has been sent periodically by the publisher each time it had updated the number of data sent by 1.

Latency Plot

Next, you are going to see how to represent the latency between these *DomainParticipants*. First, go to *File->Display Historical Data*. This will open a Dialog where you should choose one of the topics in which you want to see the data collected. For this case, we will choose `FASTDDS_LATENCY`. This data is called like this because it represents the time elapsed between the user call `write` function and the reader in the other endpoint receives it in the user callback. For the network latency there is other topic named `NETWORK_LATENCY`. However our endpoints are not storing neither publishing this type of data, and so it can not be monitored.

Once done this, a new Dialog will open asking to configure the series that is going to be displayed. In the case of `FASTDDS_LATENCY` the data to show is related with two entities. In our example we are going choose both *Domain-Participants*, and this will give us all the latency between the *DataWriters* of the first participant and the *DataReaders* of the second one.

For simplicity, we will use the same bins, start time, and end time configuration parameters as in the previous example.

Now for the `Statistics kind` option, we are going to use some of them in order to see more than one series of statistical data. Change the `Statistics kind` and click `Apply` for each of them in order to create a series for each one. The statistic kinds that we are going to use for this example are:

- `MEDIAN` (blue series)
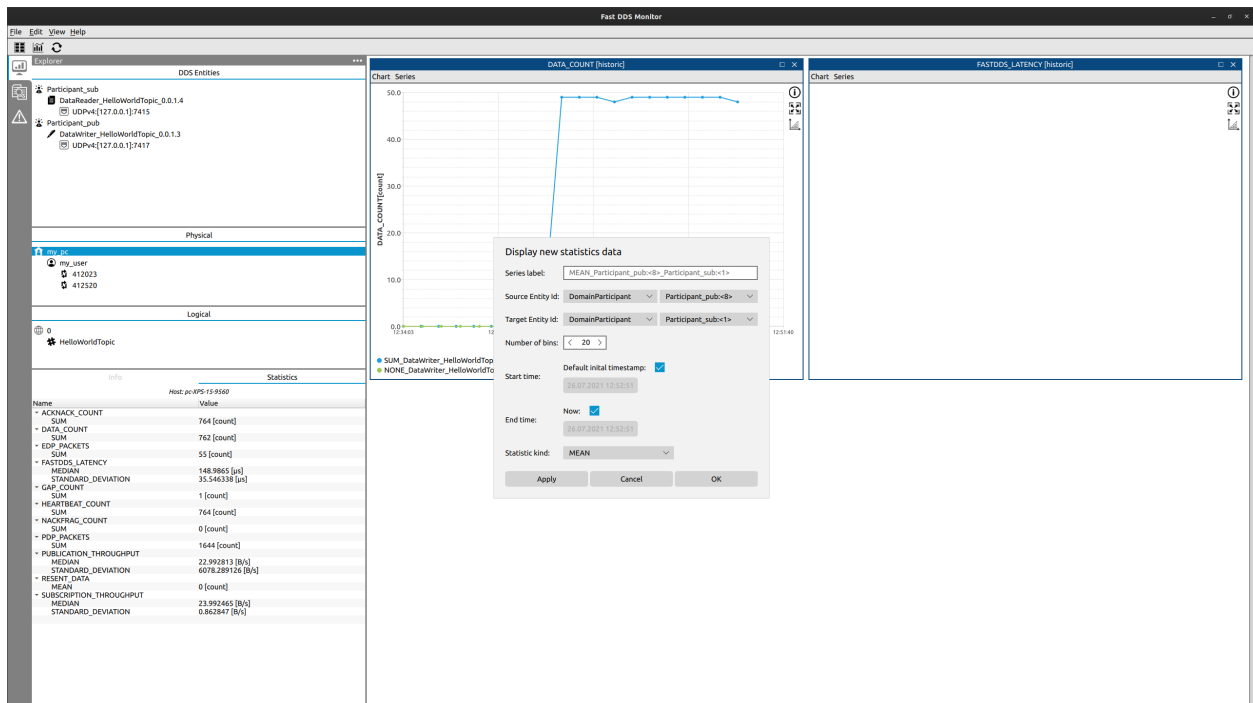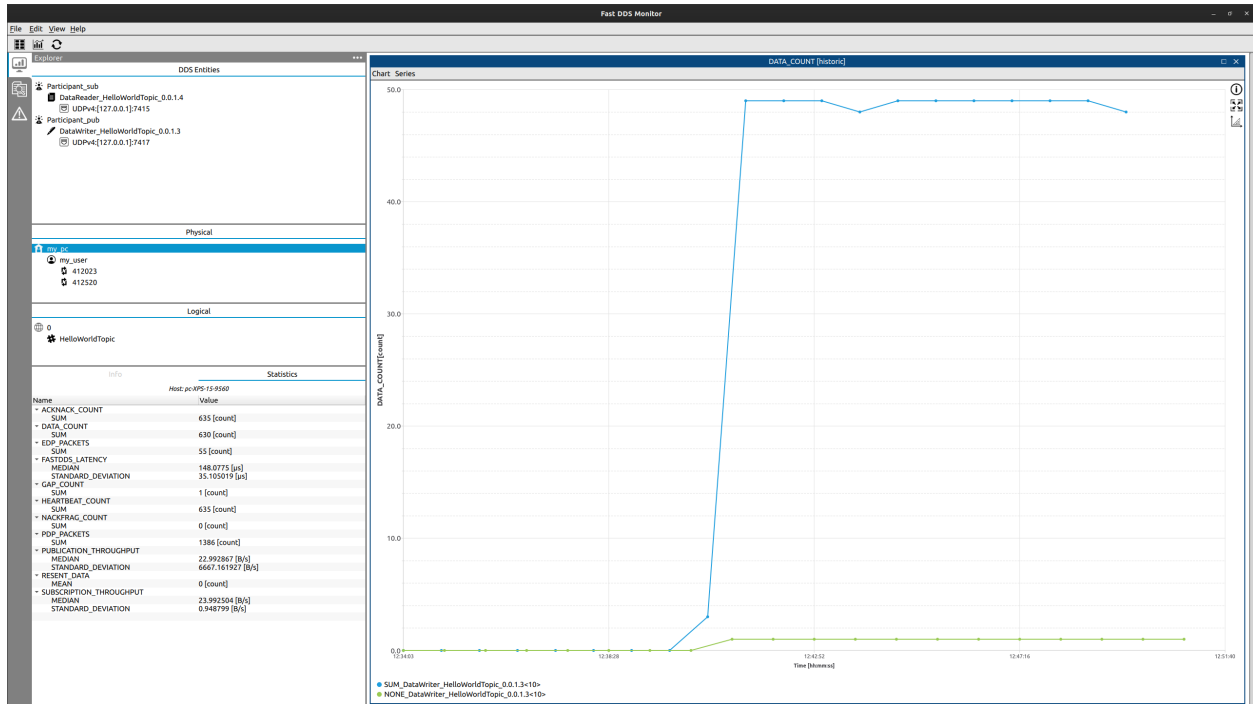
- `MAX` (green series)

- `MIN` (yellow series)

- `STANDARD_DEVIATION` (purple series)
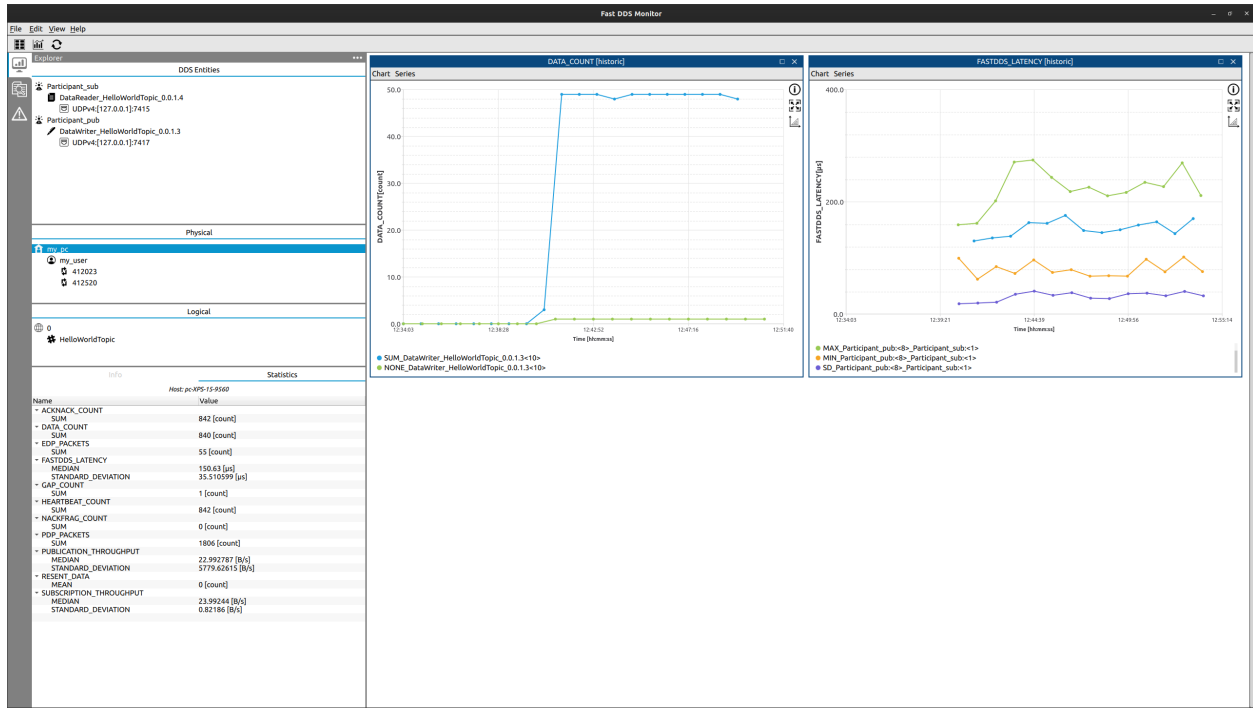
It is worth mentioning that the series name, its color, the axis, and some features of the chart box could be changed as mention in *Chartbox*.

Create Dynamic Series Chart

This section describes how to graphically represent data in real-time of a running DDS network.

Periodic Latency Plot

This section explains how to represent the FastDDS latency in real-time between the publisher and the subscriber. First of all, click in . This will open a Dialog where you should choose one of the topics in which you want to see the data collected. For this case, choose `FASTDDS_LATENCY`. Set a `Time window` of 1 minute. This means you will be able to see the data of the last minute of the network. Finally, set an `Update period` of 5 seconds. This will query for new data every 5 seconds and retrieve and display it in the chart.

After this, a new Dialog will open asking to configure the series that is going to be displayed. In the case of `FASTDDS_LATENCY` the data to show is related with two entities. In our example we are going choose our *Host*. This will retrieve the latency measured in the communication between the entities of this host to itself. For this case, it is going to be the latency between the two participants, but this trick is very useful when you want to filter latency between two specific hosts or even to collect all the latency in the same domain.

Now for the `Statistics kind` option, we are going to use some of them in order to see more than one series of statistical data. Change the `Statistics kind` and click `Apply` for each of them in order to create a series for each one. The statistic kinds that we are going to use for this example are:

- `MEAN` (blue series)
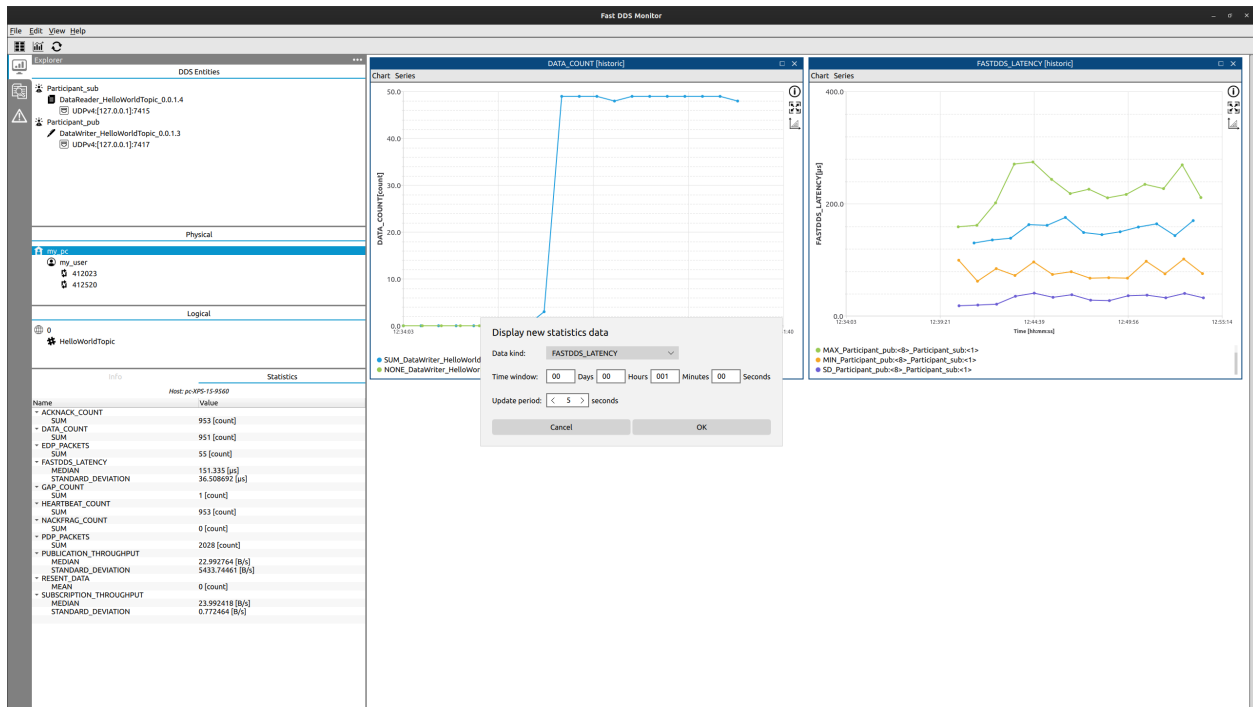
- `MAX` (green series)

- `MIN` (yellow series)

This chart will be updated each 5 seconds, displaying the data collected by the monitor within the last 5 seconds. The axis are updated periodically, and so the zoom and chart move is not available in this kind of charts while running. For this propose, the *play/pause* button stops the axis to update, and so you could zoom and move along the chart. Be aware that pausing the chart do not stop new points to appear, as every 5 seconds the update of the data will still happen.

Latency DataPoints

There is a special feature for real-time data display that allow to see every *DataPoint* received from the DDS entities monitored (similar to `bins 0` in historic series). In order to see this data in real-time, add a new series in this same chartbox in *Series->Add series*. Choose again the *Host* as source and target and choose `NONE` as `Statistics kind`.

Now you should be able to see a new series (by default purple, in the example image red) that represents each of the *DataPoints* sent by the DDS entities and collected by the monitor in the last 5 seconds. This is very helpful to understand the `Statistics kind`. As you can see, the `MEAN`, `MAX` and `MIN` in each interval are calculated with this *DataPoints*.

Note: The monitor retrieves each *DataPoint* with the time of the beginning of the interval it represents. This means that the real *DataPoints* used to calculate the `MEAN`, `MAX` and `MIN` are the ones ahead of it.

It is worth mentioning that dynamic series could be configurable as the historic series. The label and color of each series is mutable, and the chart could zoom in and out and move along the axis while the chart is paused.
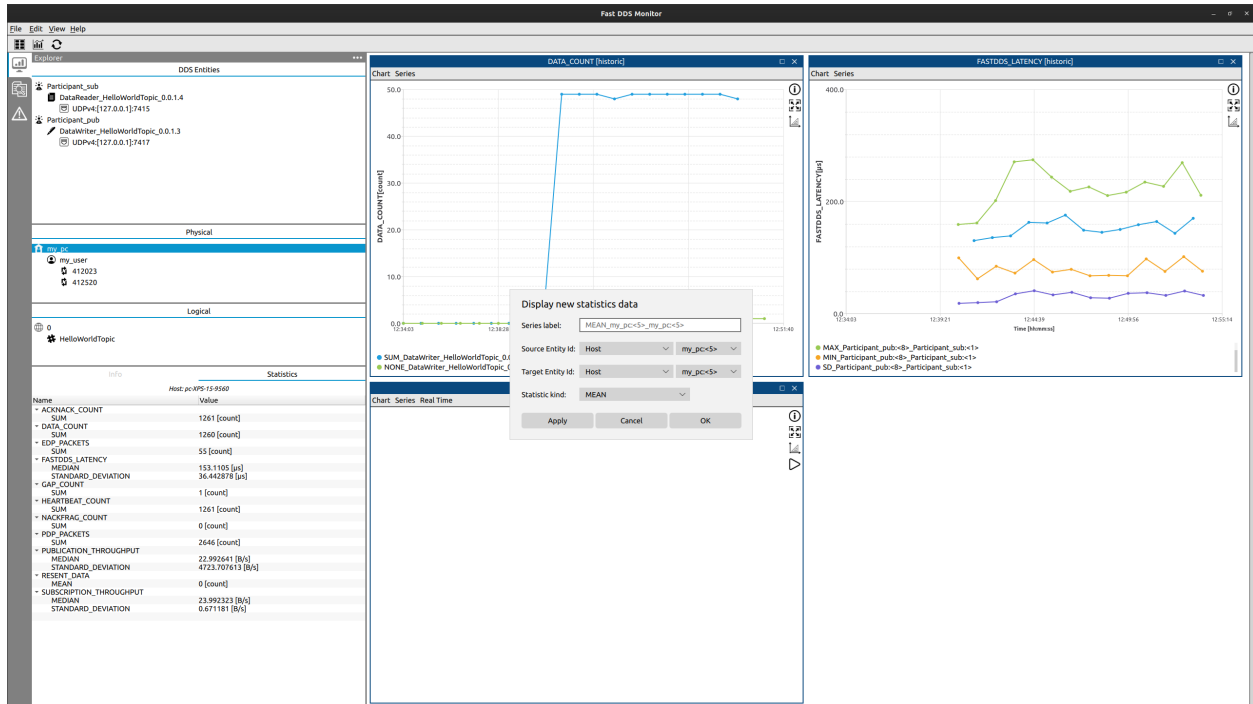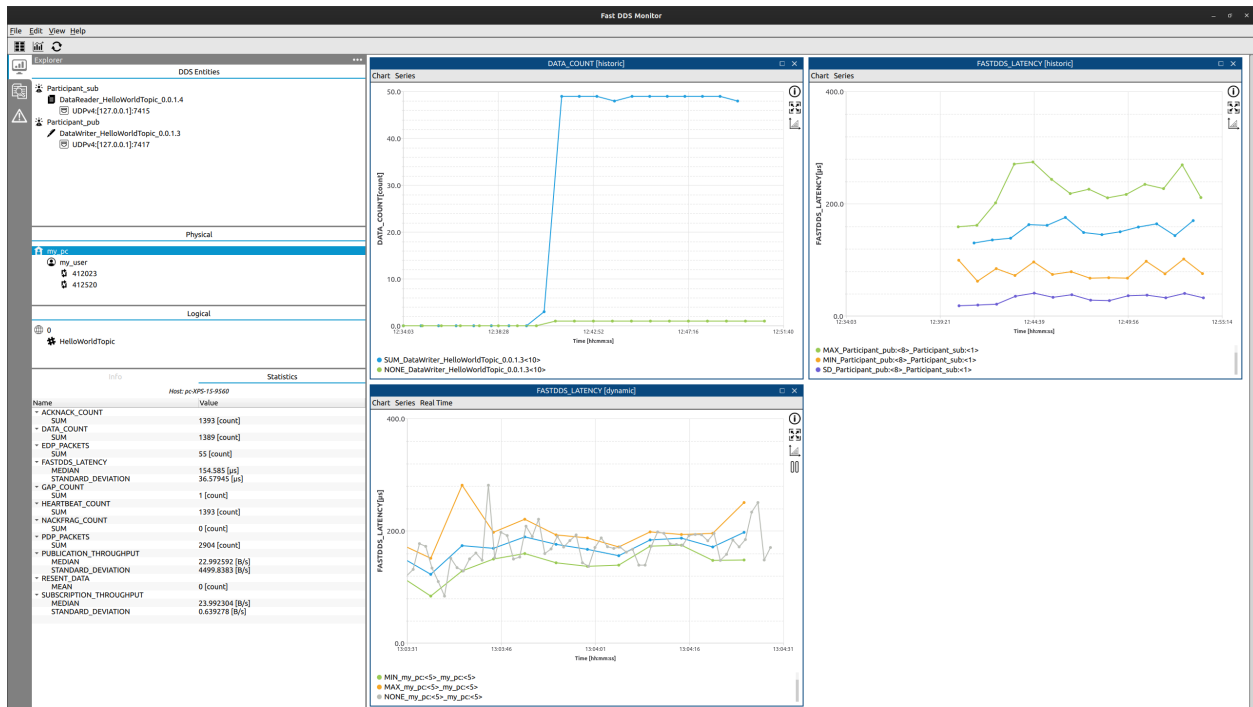
2.9 Initialize Monitoring

When the *eProsima Fast DDS Monitor* application is launched, the following screen is displayed. Click on the "Start monitoring!" button to start monitoring a Fast DDS application.

Next, it is possible to select the type of monitoring to be performed, as explained in Section *Monitor Domain*. The screenshot below shows the whole monitor interface and the dialog box that the user must fill in to start monitoring a DDS Domain or a Discovery Server network.

2.10 Layout

This section explains the Graphical User Interface (GUI) of the *Fast DDS Monitor* application. It will briefly explain the main menus and windows that could be seen, in order to familiarize the user with where to find the buttons and information. Below is a screenshot of the *Fast DDS Monitor* application in operation.

2.10.1 Application Menu

This general menu contains all the options available for this application divided in four groups depending on their area of operation:

- **File**: General propose buttons.

- **Edit**: Specific buttons with application functionality

- **View**: Window layout configuration.

- **Help**: Useful links for getting application information or support.

For the explanation of the functionality of this buttons refer to the section *Application Menu*.
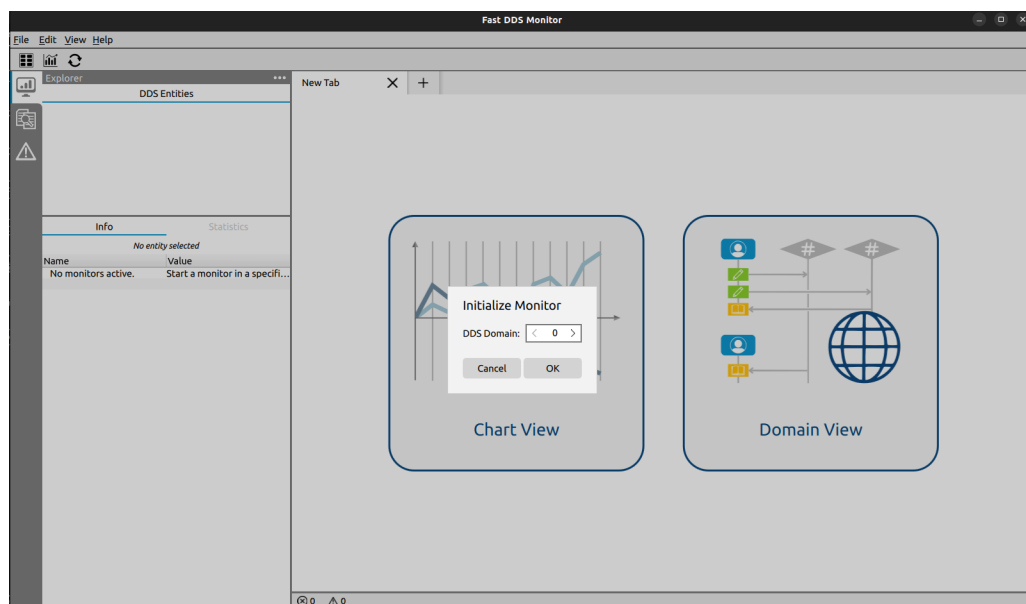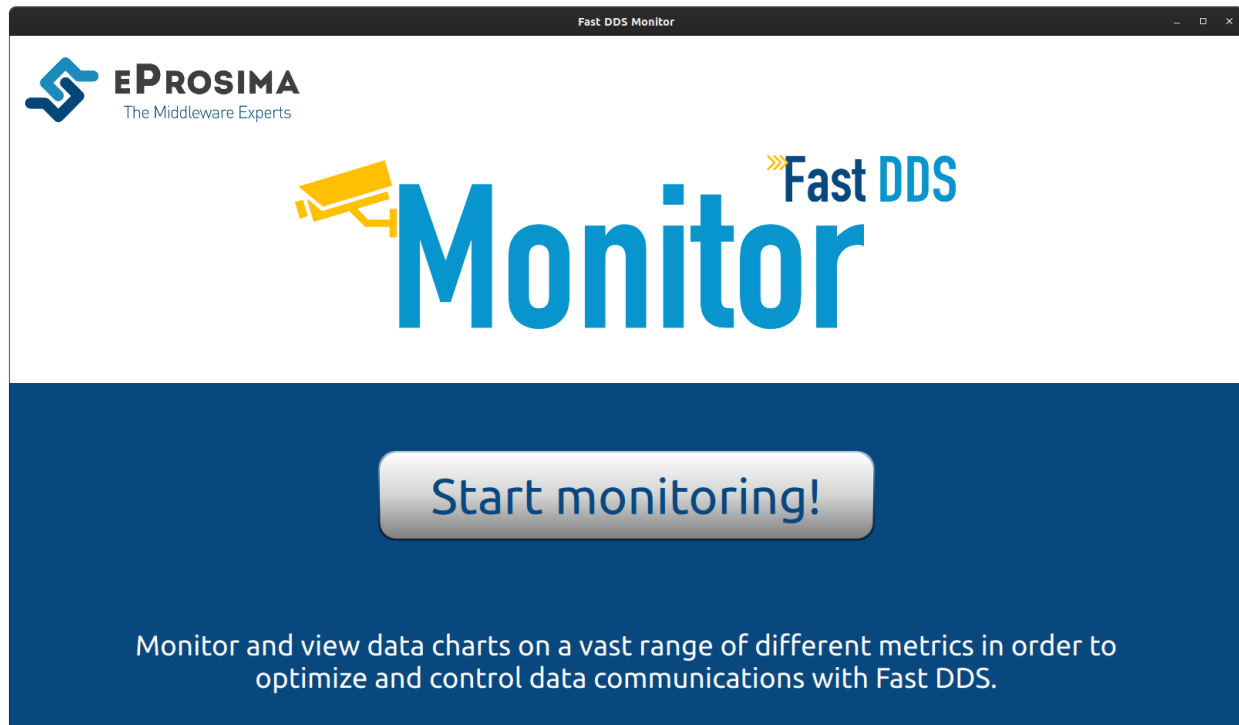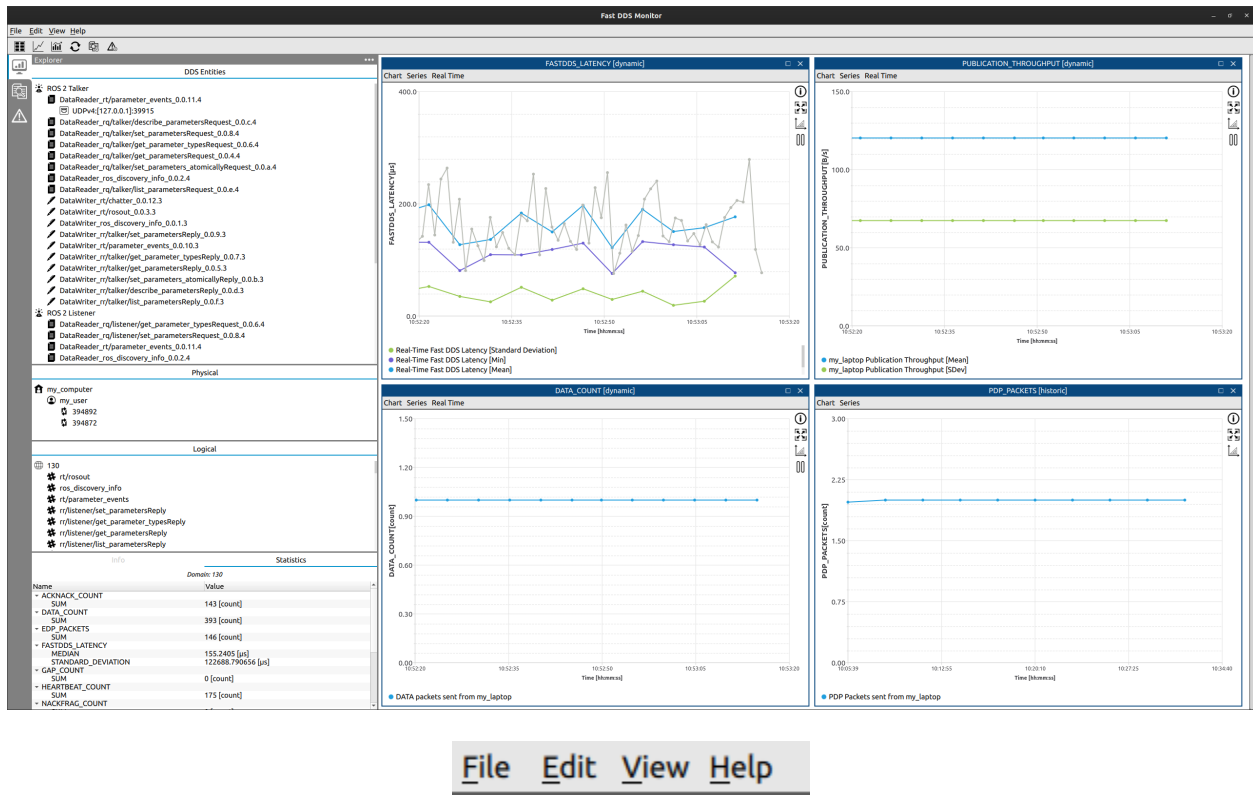
2.10.2 Shortcuts Bar

This horizontal bar contains shortcuts to the main operations supported by the application, so the user has a faster access to the main functionalities. This bar could be configured in the *View* tab of the application menu.
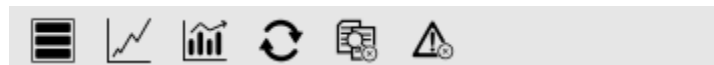
For the explanation of how to configure this bar refer to the section *Shortcuts Bar*.

2.10.3 Explorer Panel

In this panel it will be shown the different entities that has been discovered by the monitor. They will be displayed in interactive lists that the user could expand or collapse. These entities are clickable as well, in order to inspect their information shown in this same panel

This panel contains a mutable number of subpanels. The different panels that could be shown are the *DDS Panel*, the *Physical Panel*, the *Logical Panel* and the *Monitor Status Panel*. In these subpanels, the entities discovered are going to be displayed depending on their kind. In order to know which kind of entities the application has and how are they divided in categories, please refer to *Entities*.

To add a new panel to display other relevant information, use the **. . .** button in the upper bar of the panel and select those subpanels that want to be show or hide. To increase or decrease this sidebar size, grab the border of it and move it to the desired size. In order to hide the whole left sidebar, click in the < button in the upper panel or click *Hide Left sidebar* in the *View* menu..

For more information about what is an entity and how they are organized refer to *Entities*. For more information about what means to select an entity refer to *Selected Entity*.

Explorer ...

DDS Entities

- Participant_pub
  - DataWriter_HelloWorldTopic_0.0.1.3
    - UDPv4:[127.0.0.1]:7415
- Participant_sub
  - DataReader_HelloWorldTopic_0.0.1.4
    - UDPv4:[127.0.0.1]:7413
- Participant_sub
  - DataReader_HelloWorldTopic_0.0.1.4

Physical

- my_laptop
  - my_user
    - 373850
    - 373869
    - 378222

Logical

- 0
  - HelloWorldTopic

Info	Statistics

Domain: 0

Name	Value
alias	0
alive	true
id	0
kind	domain
name	0

DDS Panel

This subpanel shows the *DDS Entities* of the monitor. This entities are the DDS *DomainParticipant*, the DDS *DataReader* and *DataWriter*, and the transport *Locators* that each entity is using. This subpanel will be filled with the DDS entities that are related with the entity currently selected, so it could happen that not all the DDS entities discovered by the monitor appears in this subpanel at a certain point (see *Selected Entity* for further details).
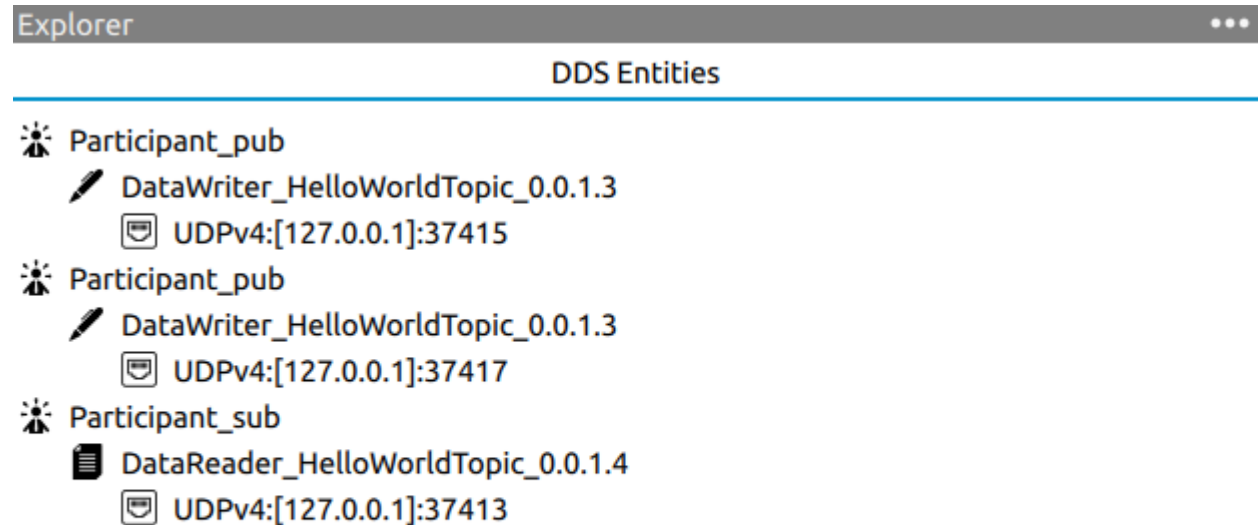
For the explanation of this entities and the interaction with them refer to the section *DDS Panel*.

Physical Panel

In this subpanel it will appear the Entities discovered by the monitor that refers to physical entities. There are three different kind of physical entities, *Host*, *User* and *Process*. These entities refer to the main information of the machine and the context where an application using *Fast DDS* is running. For further explanation of this entities and the interaction with them refer to the section *Physical Panel*.
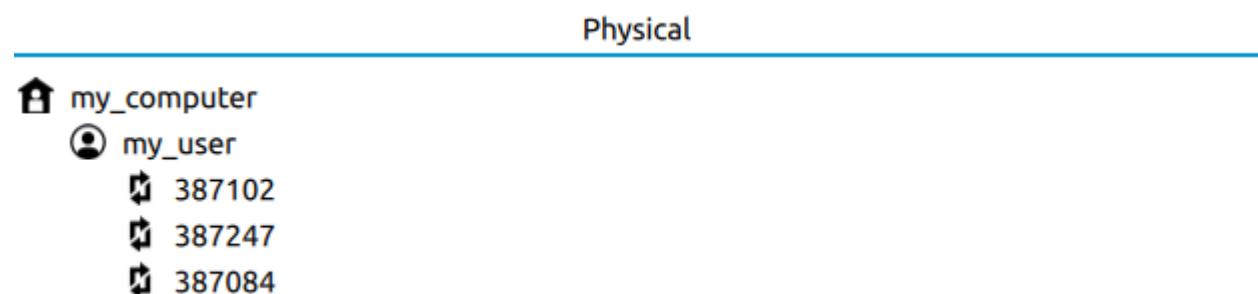
Logical Panel

In this subpanel it will appear the Entities discovered by the monitor that refers to abstract entities in a DDS communication network. These entities are *Domain* and *Topic*. These entities refer to abstract partitions in a DDS network. Only entities in the same *Domain* could communicate to each other by publishing or subscribing in the same *Topic*. For further explanation of this entities and the interaction with them refer to the section *Logical Panel*.
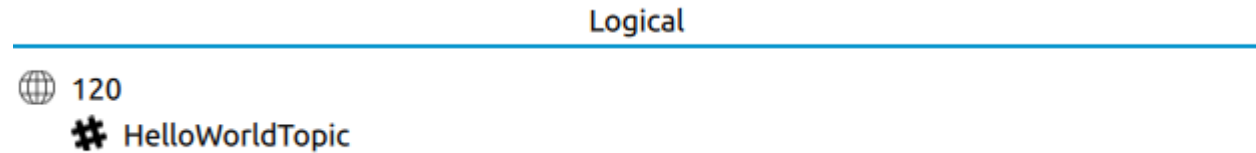
Monitor Status Panel

This subpanel displays information regarding the last entity clicked. This information is divided in two categories and are shown in the different subpanel tabs. First, the `info` tab contains the general information of the last entity clicked. Secondly, the `Statistics` tab contains a summary of the main statistical data regarding the last entity clicked.

Info Panel

This panel shows the main information of the last entity clicked. This information differs depending on the kind of the entity, i.e. for a *DDS Entity*, the *QoS* information would be shown, while for a *Process*, its *process id* will be displayed.

For the explanation of this information refer to the section *Info Panel*.

Statistics Panel

This panel shows a summary of the main statistic data related with the last entity clicked.

For the explanation of this information refer to the section *Statistics Panel*.

2.10.4 Monitor Status Panel

In this side bar it will be shown different data related with the entities that are being monitored or related with the actual state of the application. This side bar is divided in two different side bars. The upper one contains *Monitor Status Panel* and *Statistics Panel*. To change from one another press on the name of the tab an choose the panel desired to be displayed. The lower side bar contains *Monitor Status Panel*, *Log Panel* and *Issues Panel*. To change from one to another press on the name of the tab an choose the panel desired to be displayed.

To increase or decrease this sidebar size, grab the border of it and move it to the desired size. In order to hide the whole left sidebar, click in the < button in the upper panel or click *Hide Left sidebar* in the *View* menu.

Info		Statistics

DomainParticipant: Participant_pub

Name	Value
alias	Participant_pub
alive	true
guid	01.0f.7c.dd.1e.e8.9c.e5.01.00.00.00|0.0.1.c1
id	1
kind	participant
▾ locators	
0	UDPv4:[127.0.0.1]:37415
name	Participant_pub
▾ qos	
available_builtin_endpoints	3135
▾ lease_duration	
nanoseconds	0
seconds	20
▾ properties	
▾ 0	
name	PARTICIPANT_TYPE
value	SIMPLE
user_data	
▾ vendor_id	
0	1
1	15

Info	Statistics

DataReader: DataReader_HelloWorldTopic_0.0.1.4

Name	Value
▾ ACKNACK_COUNT	
SUM	1129 [count]
▾ DATA_COUNT	
SUM	1125 [count]
▾ EDP_PACKETS	
SUM	156 [count]
▾ FASTDDS_LATENCY	
MEDIAN	713.014 [µs]
STANDARD_DEVIATION	166.137972 [µs]
▾ GAP_COUNT	
SUM	2 [count]
▾ HEARTBEAT_COUNT	
SUM	1129 [count]
▾ NACKFRAG_COUNT	
SUM	0 [count]
▾ PDP_PACKETS	
SUM	407 [count]
▾ PUBLICATION_THROUGHPUT	
MEDIAN	45.899754 [B/s]
STANDARD_DEVIATION	0.023539 [B/s]
▾ RESENT_DATA	
MEAN	0 [count]
▾ SUBSCRIPTION_THROUGHPUT	
MEDIAN	103.599106 [B/s]
STANDARD_DEVIATION	29.468267 [B/s]

Status Panel

This panel shows a collection of data related with the actual state of the application:

- Entities refer to the number of entities being monitored in the user application.

- Domains is a collection of the *Domains* that has been initialized in the Monitor.

For the explanation of this information refer to the section *Status Panel*.

Log Panel

This panel shows the callbacks that the application has obtained. These callbacks refer to different events in the DDS network that is being monitored. These callbacks could be cleared by using the *Refresh*. A callback may refer to:

- The discovery of a new Entity in the DDS network.

- The reception of new data related to any of the entities that are being monitored.

For the explanation of this information refer to the section *Log SubPanel*.

2.10.5 Issues Panel

This panel lists the error events of the application. The events that the application reacts to in the current version are:

- Attempt to start monitoring a DDS Domain or a Discovery Server network that has been previously initialized.

For the explanation of this information refer to the section *Log SubPanel*.

2.10.6 Main Panel

The central window allows multiple tabs for different views. It is also displayed a collapsed menu with the possible problems that have been detected on the DDS entities. It will display the data charts, so-called *Chartbox*, that the user has configured. It will display a domain graph where the physical, logical and DDS entities from a domain are represented, focusing on the connection between endpoints through the topics, and the entities physical inheritance.
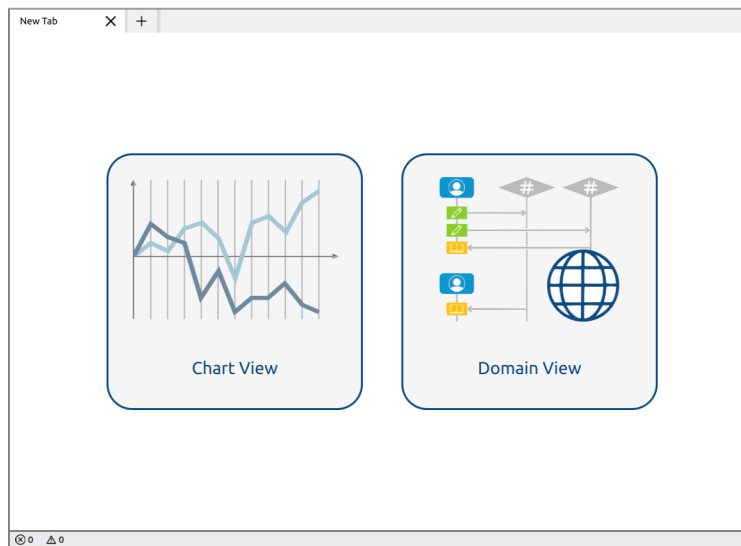
For further information about how to create a chart refer to the section *Charts Panel*.

Chartbox

These windows in the main panel store some *series* or *data configurations* to show a specific data type for one or several entities in a specified time interval and with different accumulative operations on the data.

To create a new *Chartbox* go to *Chart View* in the Main Panel default tab, and click on *Create new chart* button. This button will create a new *Chartbox* where new series could be added, removed or modified.

These Chartbox could be moved along the *Chart View* tab. In order to move one of this charts, press in the *Chartbox* title and grab the object to its new location inside the main panel. The rest of the *Chartboxes* will automatically rearrange when one of them is moved to a different position.

For further information about how to create a chart refer to the section *Charts Panel*.
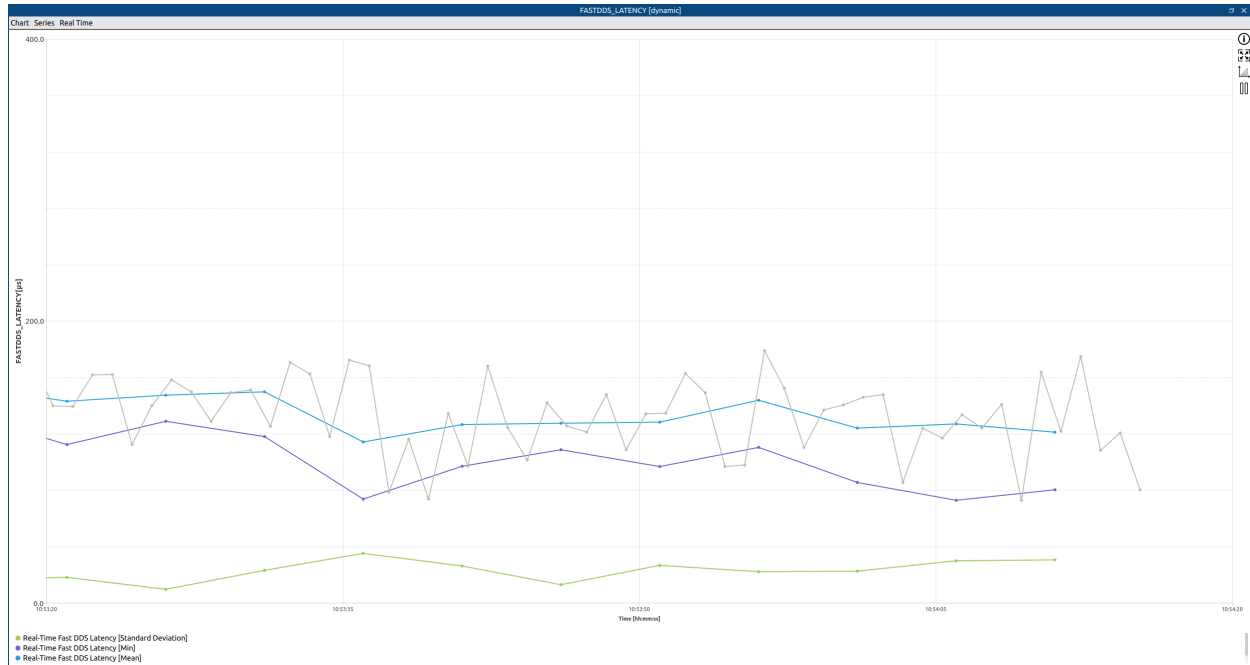
Create Series Dialog

This Dialog will appear every time a new Chartbox is created, or adding a new series by the button in the Chartbox *Series->AddSeries*.

For further details on how to configure a new series refer to *Historic Data* for historic data or *Real-Time Data* for dynamic data.

Domain View

This view in the main panel shows the connections between DataWriters and DataReaders that belong to the same DDS Domain. They are represented encapsulated inside their physical entities (see *Entities* relationship), and with a connection to the topic they are published and subscribed, respectively.

By clicking on any entity, its detailed information is displayed in the *Info Panel*. Right click allows changing the alias of the entity, filter the problems to display only the selected entity's problems and, in the case of topics, filtering the domain graph by topic so that only the entities related to the selected topic are displayed.

Problem summary

This section that can be expanded and collapsed displays all the collected problems per entity. Those problems are related to DataReader samples lost, incompatible QoS between endpoints, or DataWriter deadline missed counter, for instance.

The entities that have reported a problem would display a warning or an error icon close to the entity name, based on the severity of the problem. The entity representation in the domain graph may also display that icon.

Fig. 2.1: Create historical series dialog

Fig. 2.2: Create real-time series dialog
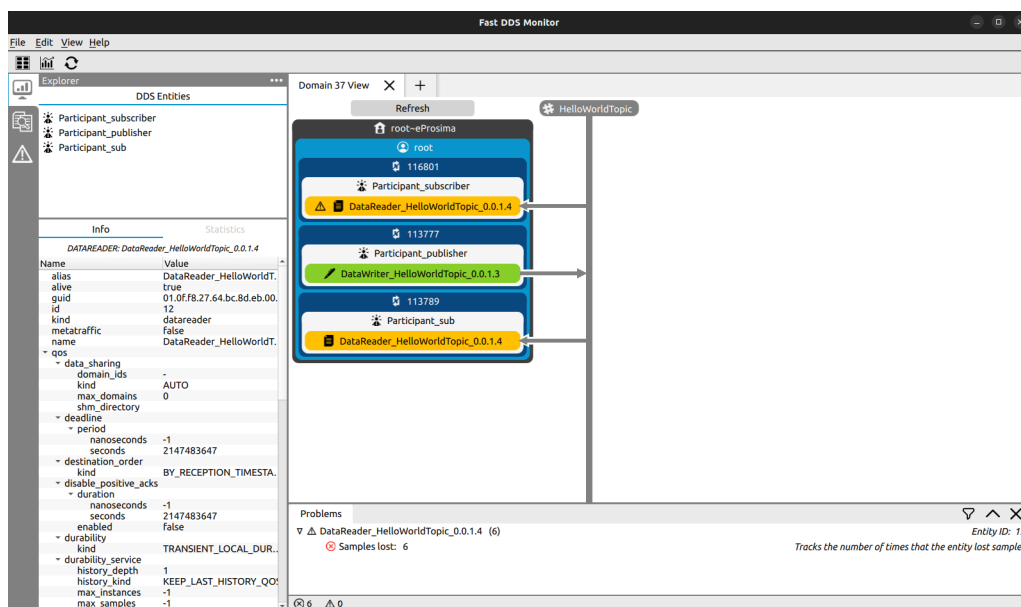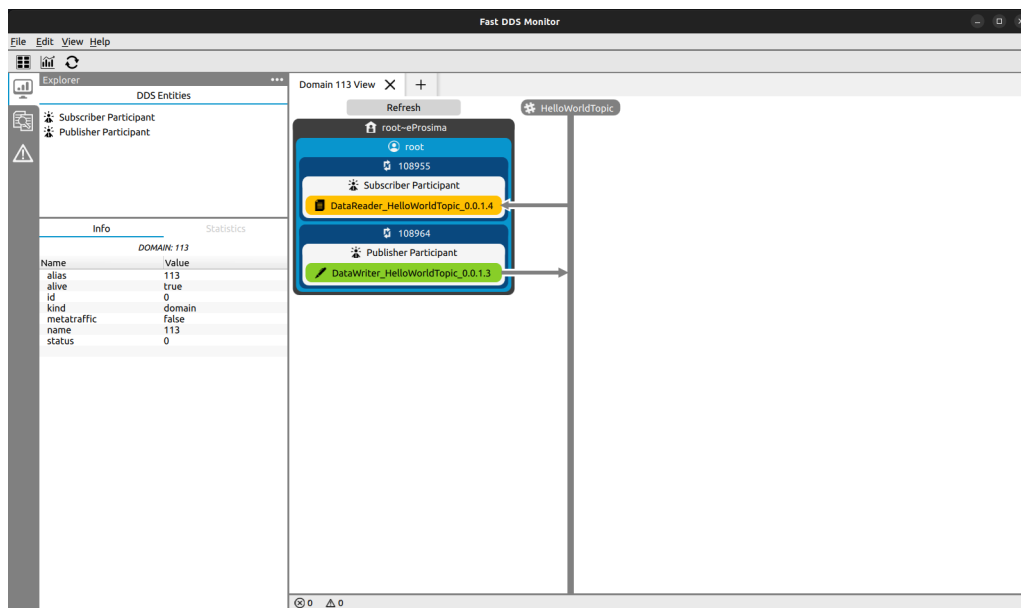
2.11 Application Menu

This section describes the operations that can be performed in *Fast DDS Monitor* through the application menu.

2.11.1 File

Initialize Monitor

Button to start monitoring a new DDS network. The entities of this network will be automatically discovered. Moreover, their connections, their configuration and statistical data reported by them will be built and displayed on the monitor for further user queries.

Section *Monitor Domain* contains all the definitions and explanation about what monitor a domain means in the context of the application.

Pressing this button, a new Dialog with two tabs will be displayed. Each tab allows to start a monitor in a DDS network deployed using the Simple Discovery Protocol (SDP) or the Fast DDS Discovery Server (DS).

> **Warning:** Starting a monitor in a domain or *Discovery Server* already initialize will duplicate the entities in them and may lead to an undefined behavior.

Simple Discovery

This dialog requests the user to introduce a number between 0 and 200 which will be the DDS Domain number. This will start a monitor in a specific DDS domain. The entities in this domain will be automatically discovered.

Discovery Server

This dialog requests the user to introduce a list of network addresses in a specific string format to connect with one or more *Fast DDS Discovery Servers*. This string must contain the IP-port pair where the Discovery Server is listening in the format `ip:port` separated by `;`.

This will connect the *Fast DDS Monitor* to the Discovery Servers listening on the addresses set, and it will get all the discovery information of the entities connecting through them.

Export to CSV

Export all the data displayed in the current Fast DDS Monitor session to a CSV file. Please refer to section *Export data* for more information on the format of the generated CSV file.

Dump

Dump the information from the database to a JSON file. Please refer to section *Export data* for more information on the format of the generated JSON file.

Dump and clear

Same as the previous one but clearing the statistics data of all the entities.

Quit

Close the application.

2.11.2 Edit

Display Historical Data

Create a new historic *Chartbox* in the central panel. To know how to configure a historic *Chartbox*, please refer to the section *Historic Data*.

Display Dynamic Data

Create a new dynamic *Chartbox* in the central panel. To know how to configure a dynamic *Chartbox*, please refer to the section *Real-Time Data*.

Delete inactive entities

This button remove all the inactive entities from the database.

Delete statistics data

This button clear the statistics data of all the entities.

Scheduler Configuration

Creates a schedule to dump the database to a file, remove old data and/or remove inactive entities at a specified interval.

Refresh

This button resets the entity clicked and the entities models in case there is some entity missing in the display.

Clear Log

This button clears the callbacks log.

Clear Issues

This button clears the issues log.

2.11.3 View

Hide/Show Inactive entities

The user of the application can decide whether to display the currently inactive entities. In case they are shown, it will be possible to access the data related to them, while if they are hidden from the monitor, they will no longer be available in the whole application framework. The latter means that it will not be possible to plot charts with data relating to these entities. This button hides/reveals the currently inactive entities detected by the monitor.

Hide/Show Metatraffic

Entities utilized for sharing metatraffic data are not shown by default. These include Fast-DDS Statistics module topics and the ones used by ROS for metatraffic data exchange, as well as the endpoints to which these topics are bound. Same as with the hide/show inactive entities feature, when metatraffic entities are hidden they are no longer available in the whole application framework. This button displays/hides the metatraffic entities detected by the monitor.

Dashboard Layout

Changes the size of the chart boxes displayed in the main panel of the application. Three mutually exclusive layout options are defined and explained below.

- **Large**: A single full-screen chart is displayed.
- **Medium**: Adjusts the chart size to show two chart boxes per row.
- **Small**: Adjusts the chart size to show three chart boxes per row.

Hide/Show Shortcuts Toolbar

Hide the shortcuts superior toolbar if visible, or reveal it otherwise.

Customize Shortcuts Toolbar

Allow to show or hide independently the shortcut buttons in the shortcut toolbar.

Hide/Show Left sidebar

Hide the left sidebar if visible, or reveal it otherwise.

2.11.4 Help

Documentation

Link to this documentation.

Release Notes

Link to the Releases section of the GitHub Fast DDS Monitor repository.

Join Us on Twitter

Link to eProsima Twitter account.

Search Feature Requests

Link to the Issues section of the GitHub Fast DDS Monitor repository.

Report Issue

Link to the Issues section of the GitHub Fast DDS Monitor repository.

About

General information of the currently running *Fast DDS Monitor* application.

2.12 Shortcuts Bar

This toolbar expose shortcuts to the buttons with the main functionality to interact with the *Fast DDS Monitor* application. This buttons functionality could be seen in *Edit*.

The meaning of each of the icons available in the shortcut bar is explained below:

- - Change the main dashboard layout.
- - Display historical data.
- - Display real-time data.
- - Refresh Fast DDS Monitor.
- - Clear the list of logs.
- - Clear the issues panel.

This bar can be hidden (or revealed in case it is already hidden) from the menu *View->Hide/Show Shortcuts Toolbar*. It is also possible to configure the shortcuts displayed from *View->Customize Shortcuts Toolbar*.

2.13 Explorer Panel

The left sidebar shows the various entities that the application has knowledge about and their available information. It is recommended to check the section *Entities* in order to get a better approach on the kind of entities that are displayed and the connection between them.

2.13.1 DDS Panel

In this panel are displayed all the *DDS Entities* that has been discovered by the monitor so far under every DDS domain or Discovery Server monitored. This panel displays specific DDS entities related with the Fast DDS Monitor entity currently selected (see *Selected Entity*). For example, it is possible to track the DDS entities created from an application running on a specific *Host*, *User*, or *Process*, as well as the DDS entities that are working on a specific DDS domain or are publishing or subscribed to a given topic. Every entity in this panel is interactive:

- Clicking in the Participant name or the Participant icon will expand or collapse the list of DataWriters/DataReaders of that Participant.

- Clicking in the DataReader/DataWriter name or the DataReader/DataWriter icon will expand or collapse the list of Locators of that DataReader/DataWriter.

- Double clicking in an entity will set this entity as *selected*. Please refer to *Selected Entity* for more information on what it means for an entity to be selected.

2.13.2 Physical Panel

This panel displays all the *Physical Entities* that the monitor has discovered so far. Similar to the *DDS Panel*, every entity in this panel is interactive:

- Clicking in the Host name or the Host icon will expand or collapse the list of Users of the Host.

- Clicking in the User name or the User icon will expand or collapse the list of Processes of the User.

- Double clicking in an entity will set this entity as *selected*. Please refer to *Selected Entity* for more information on what it means for an entity to be selected.

2.13.3 Logical Panel

This panel displays all the *Logical Entities* that are being monitored. he DDS domains that Fast DDS Monitor is monitoring are the ones set by the user (see *Monitor Domain*) and no new domains will or could be discovered apart from them since the Domains could not be discovered, but known beforehand. Therefore, this panel will only update the information. For example, having enabled the monitoring of Domain X, if an application using Fast DDS creates a new DomainParticipant in that domain with a DataWriter publishing in Topic Y, the information of that topic will appear in this view listed under Domain X to which the DomainParticipant discovered by the monitor belongs.

Similar to the *DDS Panel*, every entity in this panel is interactive:

- Clicking in the Domain name or the Domain icon will expand or collapse the list of Topics of the Domain.

- Double clicking in an entity will set this entity as *selected*. Please refer to *Selected Entity* for more information on what it means for an entity to be selected.

2.13.4 Info Panel

In this panel it is displayed the specific information of the entity that is currently **selected** (see *Selected Entity*). This information has some fields that are general for all the entity kinds, and some others that depends on the specific entity kind:

- **General fields**
    - **name**: internal name of the entity
    - **id**: internal unique id for each entity
- **Process**
    - **pid**: Process Id in its host
- **Topic**
    - **type_name**: name of the data type of the topic
- **Participant**
    - **GUID**: DDS GUID
    - **QoS**: DDS QoS information
- **DataWriter**
    - **GUID**: DDS GUID
    - **QoS**: DDS QoS information
- **DataReader**
    - **GUID**: DDS GUID
    - **QoS**: DDS QoS information

2.13.5 Statistics Panel

In this panel it is displayed a summary of some data types of the entity that is currently **selected** (see *Selected Entity*).
Regarding the selected entity, the data will be fulfilled collecting all the data of all the entities related to the this one.
The data is calculated by accumulating the data of this entity (using a specific *StatisticKind* in each case) in one bin
from the first to the last data available. In case there is no selected entity, the information displayed is the group of all
the entities that exist in the application. The data displayed is the following:

Data Kind	Statistic kind	Description
NETWORK_LATENCY	*MEDIAN*	Median value of Network Latency
NETWORK_LATENCY	*STAN-DARD_DEVIATION*	Standard deviation of Network Latency
FASTDDS_LATENCY	*MEDIAN*	Median value of Application Latency
FASTDDS_LATENCY	*STAN-DARD_DEVIATION*	Standard deviation value of Application Latency
PUBLICA-TION_THROUGHPUT	*MEDIAN*	Median value of Publication Throughput
PUBLICA-TION_THROUGHPUT	*STAN-DARD_DEVIATION*	Standard deviation value of Publication Throughput
SUBSCRIP-TION_THROUGHPUT	*MEDIAN*	Median value of Subscription Throughput
SUBSCRIP-TION_THROUGHPUT	*STAN-DARD_DEVIATION*	Standard deviation value of Subscription Throughput
RTPS_BYTES_SENT	*MEDIAN*	Median value of Total Bytes sent in RTPS packages
RTPS_BYTES_LOST	*MEDIAN*	Median value of Total Bytes lost in RTPS packages
RESENT_DATA	*MEAN*	Mean value of Data packages that had to be resent
HEARTBEAT_COUNT	*SUM*	Total number of *Heartbeat* messages
ACKNACK_COUNT	*SUM*	Total number of *Acknack* messages
NACKFRAG_COUNT	*SUM*	Total number of *Nackfrag* messages
GAP_COUNT	*SUM*	Total number of *Gap* messages
DATA_COUNT	*SUM*	Total number of *Data* messages
PDP_PACKETS	*SUM*	Total number of PDP packets sent
EDP_PACKETS	*SUM*	Total number of EDP packets sent

2.14 Status Panel

The left sidebar's status panel shows settings about the entities monitored by the application and some general information about the state and the events of the monitor.

2.14.1 Status SubPanel

In this panel it is displayed a brief information of the actual state of the *Fast DDS Monitor*,

- **Entities**

    - *Domains*: A list of the Domains that has been initialized in the Monitor so far.

    - *Entities*: Total number of entities that are being tracked.

2.14.2 Log SubPanel

This panel displays the events that the application has received. These events arise as *callbacks* that are generated because of new entities has arrived to the network or has been discovered, or because it has been any change in the DDS network state. Each callback contains the entities discovered by the Monitor and the time it has happened. This list could be erased using *Refresh*.

2.15 Issues Panel

This panel lists the error events of the application. The events that the application reacts to in the current version are:

- Attempt to start monitoring a DDS Domain or a Discovery Server network that has been previously initialized.

Note: Issues section is in progress and will display more information soon.

2.16 Main Panel

In the central panel, there is a tab section that allows multiple views, including a collapsed menu that reports the possible problems that have been detected on the DDS entities.

The main feature of the *Fast DDS Monitor* application is to graphically display the data that is being monitored in the *Chart View*. DDS entities have associated different types of data (so-called *DataKind*) that could be visualize by configuring a chart. For example, it can be displayed the mean, median and standard deviation latency between two machines (*Hosts*) running *Fast DDS* applications for the period of two hours in intervals of ten minutes.

The *Fast DDS Monitor* can also show the detected entities in a graph. The *Domain view* would filter all entities that belong to the same DDS Domain, and represent the inheritance of the physical and DDS entities (the DataWriters or DataReaders that belong to a DomainParticipant, the DomainParticipants that run on the same process, the processes that a user is running, and the users that are on a host). Those relations are represented in different boxes that contain the sub-category of entities. Also, the connections between different endpoints that are publishing or subscribed to a Topic are represented with arrows. Those arrows would start from the DataWriter and point to the Topic, or start from the Topic and point to the DataReader (in publication and subscription cases, respectively).

If filtering that graph by Topic, only the entities whose endpoints are publishing in, or subscribed to the selected Topic would be represented in the view.

If there are problems reported by a DDS entity, they are condensed by entity in the bottom layout problem section. Among the problem counter, the problem is described and, in some cases, followed by a link to the documentation.

2.16.1 Charts Panel

In this panel will be displayed the different chartbox with the different series the user creates.
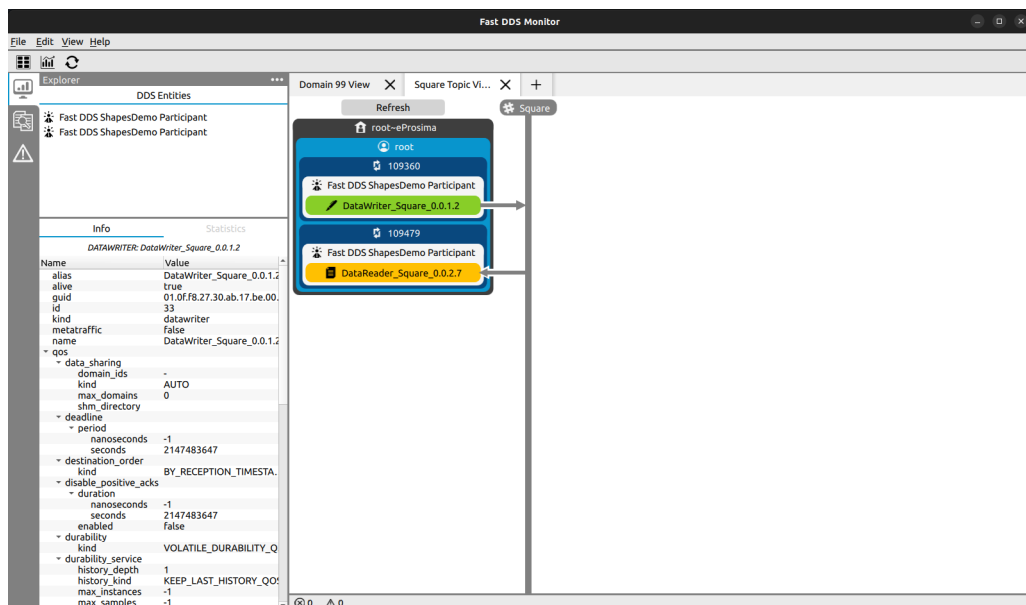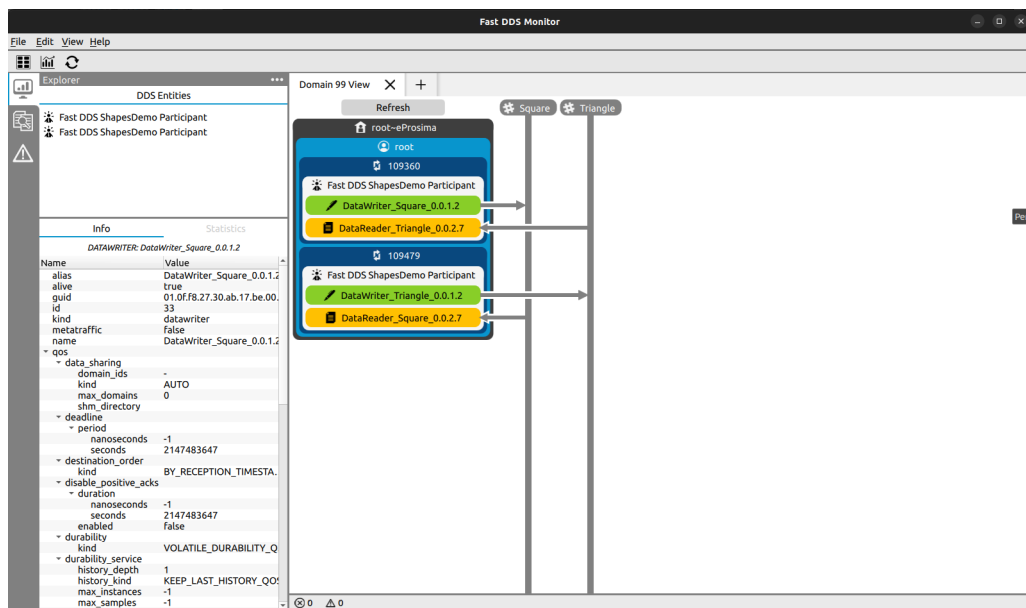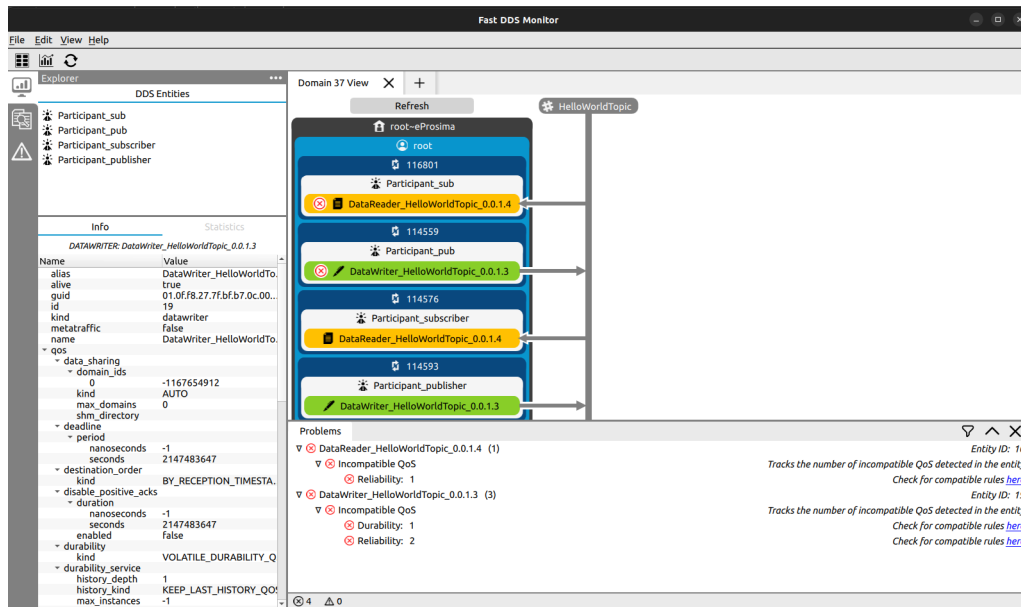
Chart series

A *DataPoint* in a chart series in the *Fast DDS Monitor* refers to a specific data type of the DDS network monitoring. Each *DataPoint* consists in a timestamp and a real value. The *timestamp* is the moment the data has been created, reported, received, etc. depending on the *DataKind*. The *value* refers to the real value of this *DataPoint* for this *DataKind* at the moment of the *timestamp* value.

For example, each *DataPoint* of the *DataKind* `DATA_COUNT` has a timestamp and an integer value. This value is the number of `Data packets` a *DataWriter* sent since the last time that same data was reported.

Chart view

Every *DataKind* is represented similarly inside a *Chartbox*. None, one or multiple lines of different colors will represent the different data series that are being displayed in the chart with the following format:

- The X axis is the time value of the data.

- The Y axis is the value that is store in the data.

This could be integers, doubles or times, but for the same *DataKind* will always be the same value type. Every point represents an accumulated value of *DataPoints* of a data kind in a time range.

Chartbox kinds

There are two kinds of Chartbox that could be created, historic and dynamic.

- The **historic** chartbox represents static data that has been accumulated along the monitor execution. This data is not updated with new data or refreshed in case somethings has changed. See the details of this chartbox kind in section *Historic Data*.

- The **dynamic** or **real-time** chartbox represents the new data that is being collected in real time by the monitor. These series will update over time with the new data received from a DDS network, and so the new data will be displayed in pseudo-real-time. See the details of this chartbox kind in section *Real-Time Data*.

Common Series parameters

Some of the parameters of the chartbox creations remains on the kind of chartbox to create. The ones mentioned here are the common parameters used by the two kinds of chartbox.

Data kind

The *DataKind* refers to the data that this chartbox will refer to. There are several *DataKinds* that represent each of the data kinds that a Fast DDS network can report. (Be aware that by default a DDS network will not report most of this data, and in case of Fast DDS it must be configured beforehand in order to report it periodically).

Series label

Name the new series in the Chartbox. If not set, the default series name is formed according to the following rule: `<cumulative_function>_<source_entity_kind>-<target_entity_id>_<target_entity_kind>-<target_entity_id>`.

Source Entity Id

This is the name and *entity Id* of the entity from which the data will be collected in the format `<name>:\<<id>\>`. This field has an entity kind to encapsulate the ids of the entities with the same kind, and make it easier for the user to search for the id required.

Each *DataKind* is related with one entity kind, normally a *DataWriter* or a *DataReader*, which are the producers of that *DataKind*. However, every entity kind is available to choose the data to represent. When an entity without this type of data is selected, the monitor will look for the entities contained in the specified entity that do report this specific *DataKind*. For example, in case the user wants to represent the number of packets (`DATA_COUNT`) transmitted by a *Host*, the monitor will look for all the *DataWriters* contained in that *Host* and report the total number of packets.

Continuing with the `DATA_COUNT` *DataKind* example, the monitor will find every *DataWriter* in this *Host* by finding every *User* in this *Host*, then finding every *Process* in these *Users*, finding every *DomainParticipant* in these *Processes* and finding every *DataWriter* in these *DomainParticipants*. Hence, the data displayed will be all the data that all those *DataWriters* have stored, accumulated by the cumulative function. Please refer to *Entities* section for more information on the monitor entities connections.

It is recommended to check some examples (see *Example of usage*) in order to better understand this functionality.

Target Entity Id

Note: Not every *DataKind* has a target entity, so the dialog will only show this field when the *DataKind* selected requires it.

This is the entity Id of the entity to which the data refers. This field works similar to the *Source Entity Id*. Some *DataKind* has a target entity to which the data refers, and this target must be of an specific entity kind. Choosing an entity of a different kind than the one this data requires will be solved by using the same mechanism explained in *Source Entity Id*. That is, searching for the correct entity kind by following the connections between entities.

i.e. To show the *DataKind* `FASTDDS_LATENCY`, that is reported by a *DataWriter*, and targeted to a *DataReader*, from an entity `Host_1` to an entity `Host_2`, both of kind *Host*, the data displayed would be collected following the steps described below:

- Get all the *DataWriters* of `Host_1` by searching for its *Users*, their *Processes*, their *DomainParticipants* and their respective *DataWriters*.

- Get all the *DataReaders* of `Host_2` by searching for its *Users*, their *Processes*, their *DomainParticipants* and their respective *DataReaders*.

- Get all the data from the *DataWriters* found that refer to the *DataReaders* found inside the interval set.

- Accumulate the data using the *cumulative function*.

It is recommended to check some examples (*Example of usage*) in order to better understand this functionality.

Statistics kind

The cumulative function is used to accumulate the data points that fulfilled the conditions set in this configuration. When there are several data points to show in a single time frame these data points are transformed into one by a *cumulative function*. This function is the one set in *Statistics kind*.

The available methods to accumulate some data points into one are:

- *NONE*: returns the *DataPoint* with the lower time.

- *MEAN*: calculate the mean value of all the data points.

- *STANDARD_DEVIATION*: calculate the standard deviation of all the data points.

- *MAX*: returns the *DataPoint* with the maximum value.

- *MIN*: returns the *DataPoint* with the minimum value.

- *MEDIAN*: calculate the median of all the data points.

- *COUNT*: returns the number of *DataPoint* in this time frame.

- *SUM*: calculate the sum of all the data points.

In case the *Number of bins* is `0` the *Statistics kind* is not used as the data is not going to be accumulated.

2.16.2 Historic Data

Create Historic Series Chartbox

An static chartbox could only contain series referring the same *DataKind*. In order to create a new chartbox in the central panel, use the button *Display Historical Data* in the *Edit* or in *Shortcuts Bar*.

Data Kind

Check common parameters explanation *Data kind*.

Clicking *OK* will create a new chartbox referring the *DataKind* chosen that will hold historic series.

Create Historic Series Dialog

The Dialog (*Create Series Dialog*) allows to create a new data series in a Chartbox. The fields in the dialog configure the data that will be displayed When all the data has been set in the *Create Historic Series Dialog*, press *Apply* to create the series and continue with the same parameters set in order to create a new series. Pressing *OK* the actual parameters are used to create a new *series* unless it has not been any changed since *Apply* has been pressed. Press *Cancel* close the window without creating any series.

Series label

Check common parameters explanation *Series label*.

Source Entity Id

Check common parameters explanation *Source Entity Id*.

Target Entity Id

Check common parameters explanation *Target Entity Id*.

Number of bins

Number of *DataPoints* that will be displayed for this chart series. The data is collected in individual points in every entity, without a regular time interval or pattern. Therefore, to show the data in a better understandable manner, each of these points will be merged in a single point inside a fraction of the time interval. The *Number of bins* determines how many fractions on which this time interval will be split, and thus, how many points will be displayed in the chart. To see all the individual data points without accumulate them, set the *Number of bins* to 0.

> **Warning:** Some of the data queried could not exist. In these cases the point will not be plotted, and the number of points in the series will not match with the number of bins.

Start time

The time that represents the minimum limit for the data points that will be displayed. That is, every data that refers to a time before this value will not be displayed. *Default initial timestamp* means the time when the monitor has been initialized.

End time

The time that represents the maximum limit for the data points that will be displayed. That is, every data that refers to a time after this value will not be displayed. *Now* means the current time. Be aware that the current time is not the maximum time that could be set up to see the data, but setting a time bigger than *Now* could (obviously) not have data points in the higher time frames.

Statistics kind

Check common parameters explanation *Statistics kind*.

Quick explanation of the data displayed

First, the application will create a complete list of data points by merging the data of all entities referred to in this query. (see the example in *Source Entity Id*).

The current represented chart consists in a number of frames or bins, set by the *Number of bins* variable from the value *Start time* to the value *End time*. Inside one of this fractions of time, referred to as frames, there could be none, one or several data points. If there are no data points in this frame, the value will be 0 or will not be displayed in the chart. If there are data points in the frame, only one point will be displayed in the chart for this time frame. This point will be calculated by accumulating all these points inside the frame by a *cumulative funcion*, specified by the value *Statistics kind*.

In case the *Number of bins* is set to 0, there is no accumulation or time frame, and so the data displayed will be all the data points for this configuration, each in its time value.

> **Warning:** Some of the data queried could not exist in the database for many reasons, i.e. the entity did not exist in the time where the query requires data, the entity does not report such data, or simply some data is reported with lower frequency than the one asked. In these cases, the graph could not be connected and the points where the data is not retrieved will not be shown.

2.16.3 Real-Time Data

A **Dynamic** or **Real-Time** series displays the data that is being receiving by the monitor in the current moment. This is a pseudo-real-time display of a running DDS network. Pseudo because it is not an exact representation of the DDS network activity in the exact moment it is represented, but it represents a periodical update of the last seconds of the network.

Every data displayed is delayed by 5 seconds in order to accurately represent all the data reported by the network. This is because the data is not instantly reported by Fast DDS, an thus it could be that the data used to update the chart is not complete.

Create Dynamic Series Chartbox

An static chartbox could only contain series referring the same *DataKind*. In order to create a new chartbox in the central panel, use the button *Display Dynamic Data* in the *Edit* or in *Shortcuts Bar*.

A new dialog will appear asking to choose some parameters that will be share for all the new series created in this new chartbox.

Data Kind

Check common parameters explanation *Data kind*.

Time window

This parameter is the default size and value of the X axis. The X axis will be set by default to have the current moment to the rightmost point. The X axis leftmost point will be the current time minus the size of the *Time window*.

Update period

This parameter refers to the time elapsed between two updates of the chartbox series. Each of the series inside the chartbox will be updated with the new data collected every *Update period* seconds.

Maximum data points

This parameter set the default for the maximum amount of data points that the series in this chartbox will have. Check series *Maximum data points* section for more information.

Clicking *OK* will create a new chartbox referring the *DataKind* chosen that will hold dynamic series.

Create Dynamic Series Dialog

This Dialog (*Create Series Dialog*) allows to create a new data series in a Chartbox. The fields in the dialog configure the data that will be displayed When all the data has been set in the *Create Dynamic Series Chartbox*, press *Apply* to create the series and continue with the same parameters set in order to create a new series. Pressing *OK* the actual parameters are used to create a new *series* unless it has not been any changed since *Apply* has been pressed. Press *Cancel* close the window without creating a new series with the current parameters.

Series label

Check common parameters explanation *Series label*.

Source Entity Id

Check common parameters explanation *Source Entity Id*.

Target Entity Id

Check common parameters explanation *Target Entity Id*.

Statistics kind

This parameter behaves as it is explained in *Statistics kind* except for *NONE* kind. Selecting *NONE* as Statistics kind will display every data available in the interval of time given by *Update period* with no accumulation.

Note: The data will not be displayed at the time it arrives in any case. It will always appear displayed after each *Update period*.

Maximum data points

This parameter limits the number of data points that will be showed for this specific series. Data points will be added dynamically to the series. This can, at some point, generate an efficiency problem due to memory exhaustion. In order to avoid this, this parameter will limit the number of data points, removing the old data as new ones are being added. Use value `0` for not limited series.

This value can be changed by series at any time in the series menu.

Cumulative data

This option allows the user to define the time interval for which the statistic selected in "Statistic kind" is to be calculated. That is, in case the user selects a time interval to obtain accumulated statistics data, the statistics will be applied to all the data collected by the monitor in that defined time interval. This allows the update interval of the chart and the time interval for calculating the statistics to be independent.

Let's look into a simple example of monitoring the latency of a publisher and a subscriber in the Shapes Demo application launched with statistics enabled (a detailed example of Fast DDS Monitor monitoring a Shapes Demo application is shown in the following video tutorial). First, a chart is created to monitor the application latency with a time window of 5 minutes and an update period of 5 seconds. Then the dialog box for creating series in the graph is opened. Select your host as source and target entities and the DataWriters and DataReaders running on your machine will be automatically detected. Select the mean as the type of statistical measurement to be applied. Finally, create three series with three different types of accumulation:

- **No accumulation**. The average latency between publisher and subscriber in the last update period, 5 seconds in this particular case, is calculated.

- **With accumulation from the first available data point**. This calculates the average latency between publisher and subscriber from the time the monitor has the first available data until the current time. This current instant is updated every update period, adding new points to the calculated statistic.

- **With accumulation by setting a time interval**. The average latency between publisher and subscriber is calculated from the current time minus the cumulative time interval set until the current time.

Below is an image of the three series created. It can be seen that the series with a longer accumulation period tends to have a steady latency value, being less susceptible to strong but momentary latency variations. On the contrary, the series that calculates the average latency with the available data of the last 5 seconds shows a large variation due to the smaller number of data points available for the calculation of the statistic.
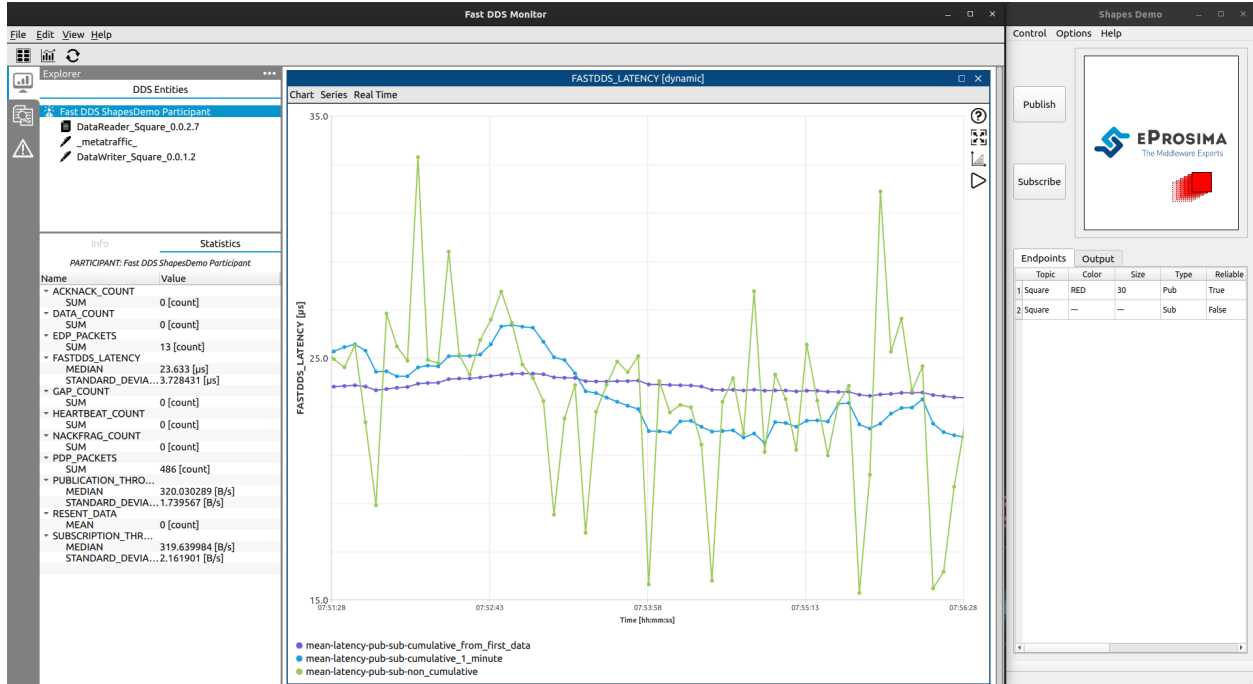
Quick explanation of the data displayed

First, the application will create an empty chart where X axis represents the time between the current moment and that time minus the *Time window* size. This window is permanently moving in order to always represent the current time. This X axis movement could be paused in order to move and resize the chart at any moment, and new data will still appear.

Every *Update period* new data will be displayed in the right side of the chart. The new data displayed references the accumulative value of the data that has been stored in that amount of time.

> **Warning:** Some of the data queried could not exist in the database for many reasons, i.e. the entity did not report anything in the time where the query requires data. In these cases, after an *Update period* there will not appear any point and the chart will be connected with the next interval with data.

2.16.4 Chartbox

In this section it is explained the main functionalities and interactions available within a *Chartbox*, a window contained in the *Main Panel* that displays entities data with different configurations.

To start a new Chartbox, press *Display Historical Data* or *Display Dynamic Data* in *Edit* or in *Shortcuts Bar*. These chartbox will be displayed in the central panel with the title of the *DataKind* they refer to. In these charts it will be displayed the Series of data that the user initialize. For how to set a new series please refer to *Create Historic Series Dialog* or *Create Dynamic Series Chartbox*.

Chart Menu
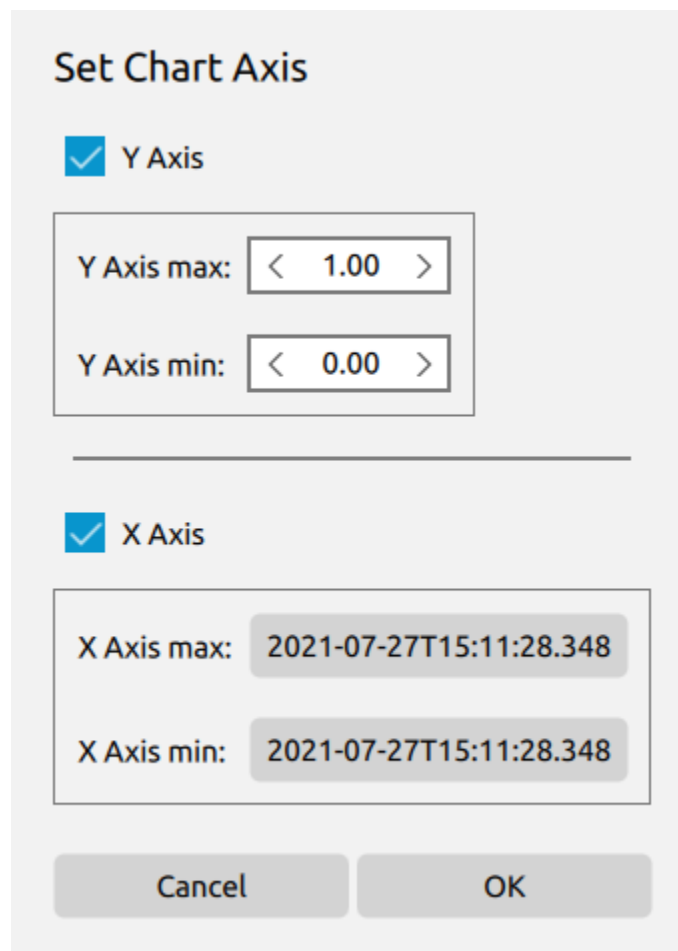
In the top bar of each Chartbox there is a Menu tab *Chart* with the following buttons.

Reset zoom

Reset the zoom of the Chartbox to the standard one. The standard zoom is calculated to fit all the data that is being currently displayed. It is also possible to take this action by clicking on the button displayed in the same chart box.

Set axes

This menu opens a dialog pop-up for the user to set the axes of the chart. Changing the X-axis (time axis) is disabled by default, thus maintaining the time values currently displayed on the chart. This allows dynamic charts to continue to update the value of the X-axis while the Y-axis remains fixed. It is also possible to access this dialog by clicking on the button displayed in the same chart box.

To return to the original time axis, and allow a dynamic update of the Y-axis, simply click on the *Reset zoom* button located to the right of the chart.

Clear chart

Eliminate every data configuration displayed in the Chartbox

Rename chart box

Change the name of the chart box.

Close chart box

Eliminate the Chartbox and every configuration in it. It could be removed as well pressing the **x** button in the top of the chart.

Export to CSV

Export the data of all series in the chart box to a CSV file. Please refer to section *Export data* for more information on the format of the generated CSV file.

Chart Controls

Displays an informative dialog with the key combination to execute actions on the chart. It is also possible to access this dialog by clicking on the button displayed in the same chart box.

Chart Interactive Controls

Click in chart point:	Show point value
Ctrl + click and drag:	Scroll axex
Ctrl + wheel:	Zoom in / out
Mayus + grab area:	Zoom in over the area

Only available while stopped

OK
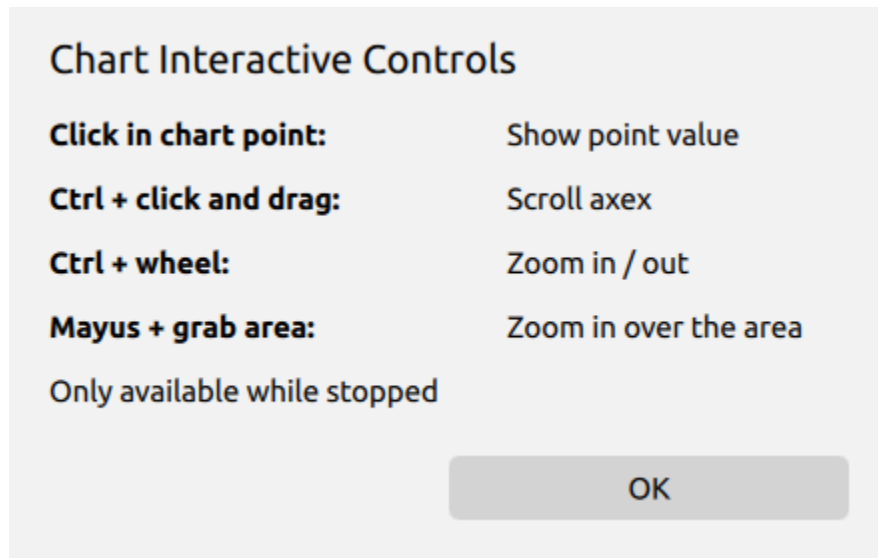
Series Menu

In the top bar of each Chartbox there is a Menu tab *Series* with these buttons.

Add series

Add a new series in this Chartbox. This will open a new *Create Historic Series Dialog* or *Create Dynamic Series Chartbox*.

Hide all series

Hide all series in this Chartbox.

Display all series

Reveal all series in this Chartbox.

Chart Interaction

The user could interact with the Chartbox and the data in it by resizing and moving the view.

Point description

Clicking in any point of a displayed series, an info box will be prompt with the x and y label values of the data point.

Moving the view

Press and hold `Ctrl` key and left click in one point of the Chartbox. Move the mouse to scroll over the data.

Zoom in/out

Press and hold `Ctrl` key and scroll up to zoom in to the center of the Chartbox. Press and hold `Ctrl` key and scroll down to zoom out from the center of the Chartbox.

Series Configuration

Right clicking in the name of a series in the *Legend* will open a dialog with the available configurations for a series.

Remove series

Remove this series.

Rename series

Change the name of this series.

Change color

A Dialog will open to choose a new color for the series and the points displayed. Also available with left click on the color of the series in the *Legend*.

Hide/Show series

Hide a series if it is displayed, or it reveal it if it is hide. Also available with left click on the name of the series in the *Legend*.

Export to CSV

Export the series data to a CSV file. Please refer to section *Export data* for more information on the format of the generated CSV file.

Dynamic chartbox

The chartbox that holds dynamic series have some extra functionality. These chartbox could be stopped at any time (or played in case they are already stopped). This allows to stop the updating the axis, so zoom and move along the chart is available. The data presented in the chartbox will keep updating with the same time interval independently of the play status.

To pause the real time update of the time axis, click on the Real-Time menu or on the / button placed on the right side of the chart, or set the axes to a specific value with the *Set axes* button.

2.17 Selected Entity

The application stores one entity as **last entity clicked** in order to manage what information is displayed. In the context of the Fast DDS Monitor, an entity is every element that can be tracked by the monitor (see *Entities*). To set one entity as *clicked* double click in any entity in any of the *Explorer Panel* panels, and this entity will be set as *Selected Entity* for the whole application from now on. Selecting an entity has the following effects in the application view:

- The entity clicked will remain marked with blue background while no other entity is clicked or the Selected Entity is restore.

- In the *Monitor Status Panel*, the information related to this entity is displayed, such as *QoS* or specific entity settings.

- In the *Statistics Panel*, a general summary of the data stored for this entity is listed.

- If the selected entity is a Fast DDS Monitor entity belonging to the Physical or Logical Entities group, the entities displayed in *DDS Panel* are only those *related* to this entity. Therefore, clicking one of the *DDS Entities* do not update the *DDS Panel*. To check the relation between please refer to *Entities* section.

2.17.1 Deselect Entity

To change the actual *Selected Entity* just click in a another entity in the *Explorer Panel*. Moreover, it is possible to deselect any entity. Use the *Refresh* button (*Refresh*) in order to preform that action. Not having any entity selected has the following effects in the application view:

- The *DDS Panel* will list all DDS entities present in any Domain monitored by Fast DDS Monitor, i.e. it will be possible to view any DomainParticipant, DataWriter and DataReader regardless of the physical entity or the DDS logical entity under which they are located.

- The information shown in the *Monitor Status Panel* is a brief summary of the current state of the application.

2.18 Export data

One of the main functionalities of the *Fast DDS Monitor* is the possibility of exporting the data generated in each monitoring session. The different monitor capabilities for exporting user data are explained below.

2.18.1 Export charts in a CSV file

The monitor offers the possibility to export to a CSV file all the data that have been or are being represented in each of the charts boxes, both historical and real-time charts.

Thus the monitor offers the user three possibilities:

- Export the data of a single series. This can be done from the series menu explained in section *Chart Menu*.

- Export the data of all the series belonging to a chart box. This can be done from the *Chart* menu available in each chart box. This menu is explained in section *Series Configuration*.

- Export all the data of all the series of all the chart boxes. This is done from the application *File* menu, explained in section *File*.

Format of the CSV file

This section presents a table with the format of the CSV file containing the exported data.

	‹DataKind›
	‹Chart box name›
ms	‹DataKind units›
UnixTime	‹Series name›
‹unix_time›	‹data_value›

2.18.2 Export database in a JSON file

The monitor gives you the option to dump the data from the database to a JSON file.

Thus the monitor provides the user two options:

- Dump. Explained in section *Dump*.

- Dump and clear. Explained in section *Dump and clear*.

Format of the JSON file

This section presents a JSON with the format of the JSON file containing the exported data.

```json
{
    "datareaders":{},
    "datawriters":{},
    "domains": {
        "0": {
            "alias":"0",
            "alive":true,
            "metatraffic":false
            "name":"0",
            "participants":[],
            "topics":[]
        }
    },
    "hosts":{},
    "locators":{},
    "participants":{},
    "processes":{},
    "topics":{},
    "users":{},
    "users":{},
}
```

2.19 Linux installation from sources

The instructions for installing the *eProsima Fast DDS Monitor application* from sources and the required *Qt* installation are provided in this page. It is organized as follows:

- *Dependencies installation*
    - *Requirements*
    - *Dependencies*
- *Colcon installation*
- *CMake installation*
    - *Local installation*
    - *Global installation*

> • *Run an application*

2.19.1 Dependencies installation

Fast DDS Monitor depends on *eProsima Fast DDS* library, *eProsima Fast DDS Statistics Backend* library, Qt and certain Debian packages. This section describes the instructions for installing *eProsima Fast DDS* dependencies and requirements in a Linux environment from sources. The following packages will be installed:

- `foonathan_memory_vendor`, an STL compatible C++ memory allocator library.
- `fastcdr`, a C++ library that serializes according to the standard CDR serialization mechanism.
- `fastrtps`, the core library of eProsima Fast DDS library.
- `fastdds_statistics_backend`, a C++ library that provides a simple and easy-to-use API for interacting with data from *Fast DDS* statistics module

First of all, the *Requirements* and *Dependencies* detailed below need to be met. Afterwards, the user can choose whether to follow either the *colcon* or the *CMake* installation instructions.

Requirements

The installation of *eProsima Fast DDS* in a Linux environment from binaries requires the following tools to be installed in the system:

- *CMake, g++, pip, wget and git*
- *Colcon* [optional]
- *Gtest* [for test only]

CMake, g++, pip, wget and git

These packages provide the tools required to install *eProsima Fast DDS* and its dependencies from command line. Install CMake, g++, pip, wget and git using the package manager of the appropriate Linux distribution. For example, on Ubuntu use the command:

```
sudo apt install cmake g++ pip wget git
```

Colcon

colcon is a command line tool based on CMake aimed at building sets of software packages. Install the ROS 2 development tools (colcon and vcstool) by executing the following command:

```
pip3 install -U colcon-common-extensions vcstool
```

Note: If this fails due to an Environment Error, add the `--user` flag to the `pip3` installation command.

Gtest

Gtest is a unit testing library for C++. By default, *eProsima Fast DDS Monitor* does not compile tests. It is possible to activate them with the opportune CMake options when calling colcon or CMake. For more details, please refer to the *CMake options* section. For a detailed description of the Gtest installation process, please refer to the Gtest Installation Guide.

Dependencies

eProsima Fast DDS has the following dependencies, when installed from sources in a Linux environment:

- *Asio and TinyXML2 libraries*
- *OpenSSL*
- *eProsima dependencies*
- *Qt 5.15*

Asio and TinyXML2 libraries

Asio is a cross-platform C++ library for network and low-level I/O programming, which provides a consistent asynchronous model. TinyXML2 is a simple, small and efficient C++ XML parser. Install these libraries using the package manager of the appropriate Linux distribution. For example, on Ubuntu use the command:

```
sudo apt install libasio-dev libtinyxml2-dev
```

OpenSSL

OpenSSL is a robust toolkit for the TLS and SSL protocols and a general-purpose cryptography library. Install OpenSSL using the package manager of the appropriate Linux distribution. For example, on Ubuntu use the command:

```
sudo apt install libssl-dev
```

eProsima dependencies

If it already exists in the system an installation of *Fast DDS* library with version greater than *2.3.0* and an installation of *Fast DDS Statistics Backend*, just source these libraries when building the *Fast DDS Monitor* by using the command:

```
source <fastdds-installation-path>/install/setup.bash
```

In other case, just download *Fast DDS* project from sources and build it together with *Fast DDS Monitor* using colcon as it is explained in section *Colcon installation*.

Qt 5.15

Qt 5.15 is needed in order to build *Fast DDS Monitor*. To install this Qt version, refer to Qt Downloads website.

Note: When going through the installation steps, make sure the box of component *Qt Charts* is checked.

2.19.2 Colcon installation

1. Create a `Fast-DDS-Monitor` directory and download the `.repos` file that will be used to install *eProsima Fast DDS Monitor* and its dependencies:

```
mkdir -p ~/Fast-DDS-Monitor/src
cd ~/Fast-DDS-Monitor
wget https://raw.githubusercontent.com/eProsima/Fast-DDS-monitor/main/fastdds_
→monitor.repos
vcs import src < fastdds_monitor.repos
```

Note: In case there is an already *Fast DDS* installation in the system it is not required to download and build every dependency in the `.repos` file. It is just needed to download and build the *Fast DDS Monitor* project having sourced its dependencies. Refer to section *eProsima dependencies* in order to check how to source *Fast DDS* and *Fast DDS Statistics Backend* libraries.

In order to build the project, it must be specified the path to the Qt 5.15 `gcc_64` installation path. Using the standard Qt installation, this path should be similar to `/home/<user>/Qt/5.15.2/gcc_64`.

2. Build the packages:

```
colcon build --cmake-args -DQT_PATH=<qt-installation-path>
```

Note: Being based on CMake, it is possible to pass the CMake configuration options to the `colcon build` command. For more information on the specific syntax, please refer to the CMake specific arguments page of the colcon manual.

2.19.3 CMake installation

This section explains how to compile *eProsima Fast DDS Monitor* with CMake, either *locally* or *globally*.

Local installation

1. Create a `Fast-DDS` directory where to download and build *eProsima Fast DDS Monitor* and its dependencies:

```
mkdir ~/Fast-DDS-Monitor
```

2. Clone the following dependencies and compile them using CMake.

   • Foonathan memory

```
cd ~/Fast-DDS-Monitor
git clone https://github.com/eProsima/foonathan_memory_vendor.git
mkdir foonathan_memory_vendor/build
cd foonathan_memory_vendor/build
cmake .. -DCMAKE_INSTALL_PREFIX=~/Fast-DDS-Monitor/install -DBUILD_
↪SHARED_LIBS=ON
cmake --build . --target install
```

- Fast CDR

```
cd ~/Fast-DDS-Monitor
git clone https://github.com/eProsima/Fast-CDR.git
mkdir Fast-CDR/build
cd Fast-CDR/build
cmake .. -DCMAKE_INSTALL_PREFIX=~/Fast-DDS-Monitor/install
cmake --build . --target install
```

- Fast DDS

```
cd ~/Fast-DDS-Monitor
git clone https://github.com/eProsima/Fast-DDS.git
mkdir Fast-DDS/build
cd Fast-DDS/build
cmake .. -DCMAKE_INSTALL_PREFIX=~/Fast-DDS-Monitor/install -DCMAKE_
↪PREFIX_PATH=~/Fast-DDS-Monitor/install
cmake --build . --target install
```

- Fast DDS Statistics Backend

```
cd ~/Fast-DDS-Monitor
git clone https://github.com/eProsima/Fast-DDS-statistics-backend.git
mkdir Fast-DDS-statistics-backend/build
cd Fast-DDS-statistics-backend/build
cmake .. -DCMAKE_INSTALL_PREFIX=~/Fast-DDS-Monitor/install -DCMAKE_
↪PREFIX_PATH=~/Fast-DDS-Monitor/install
cmake --build . --target install
```

3. Once all dependencies are installed, install *eProsima Fast DDS Monitor*:

```
cd ~/Fast-DDS-Monitor
git clone https://github.com/eProsima/Fast-DDS-monitor.git
mkdir Fast-DDS-monitor/build
cd Fast-DDS-monitor/build
cmake .. \
    -DCMAKE_INSTALL_PREFIX=~/Fast-DDS-Monitor/install \
    -DCMAKE_PREFIX_PATH=~/Fast-DDS-Monitor/install \
    -DQT_PATH=<qt-installation-path>
cmake --build . --target install
```

Note: By default, *eProsima Fast DDS Monitor* does not compile tests. However, they can be activated by downloading and installing Gtest and building with CMake option -DBUILD_TESTS=ON.

Global installation

To install *eProsima Fast DDS* system-wide instead of locally, remove all the flags that appear in the configuration steps of `Fast-CDR`, `Fast-DDS`, `Fast-DDS-Statistics-Backend`, and `Fast-DDS-Monitor`, and change the first in the configuration step of `foonathan_memory_vendor` to the following:

```
-DCMAKE_INSTALL_PREFIX=/usr/local/ -DBUILD_SHARED_LIBS=ON
```

2.19.4 Run an application

To run the *eProsima Fast DDS Monitor* application, source the *Fast DDS* and *Fast DDS Statistics Backend* libraries and execute the executable file that has been installed in `<install-path>/fastdds_monitor/bin/fastdds_monitor`:

```
# If built has been done using colcon, all projects could be sourced as follows
source install/setup.bash
./<install-path>/fastdds_monitor/bin/fastdds_monitor
```

Be sure that this executable has execute permissions.

2.20 CMake options

eProsima Fast DDS Monitor provides numerous CMake options for changing the behavior and configuration of *Fast DDS Monitor*. These options allow the developer to enable/disable certain *Fast DDS Monitor* settings by defining these options to `ON`/`OFF` at the CMake execution, or set the required path to certain dependencies.

> **Warning:** These options are only for developers who installed the *Fast DDS Monitor* following the compilation steps described in *Linux installation from sources*.

Option	Description	Possible values	Default
QT_PATH	Path to the directory where Qt has been installed. This argument is required to find the Qt binaries. An example of the path to the Qt 5.15 installation directory is: `/opt/Qt/5.15.2/gcc_64`.	•	•
BUILD_TESTS	Build the *Fast DDS Monitor* application and documentation tests. Setting `BUILD_TESTS` to `ON` sets `BUILD_DOCUMENTATION_TESTS` to `ON`.	OFF ON	OFF
BUILD_APP_TESTS	Build the *Fast DDS Monitor* application tests. It is set to `ON` if `BUILD_TESTS` is set to `ON`.	OFF ON	OFF
BUILD_DOCUMENTATION_TESTS	Build the *Fast DDS Monitor* documentation tests. It is set to `ON` if `BUILD_TESTS` is set to `ON`.	OFF ON	OFF
BUILD_DOCUMENTATION	Build the *Fast DDS Monitor* documentation. It is set to `ON` if `BUILD_TESTS_DOCUMENTATION` is set to `ON`.	OFF ON	OFF
BUILD_MOCK	Build the *Fast DDS Statistics Backend* mocks, which is a simulator of a *Fast DDS* application that generates random statistics data for testing the *Fast DDS Monitor*. It is set to `ON` if `BUILD_TESTS` is set to `ON`, `STATIC_MOCK` is set to `ON`, or `COMPLEX_MOCK` is set to `ON`.	OFF ON	OFF
STATIC_MOCK	Build the *Fast DDS Statistics Backend* static mock and link the *Fast DDS Monitor* application against it. The static mock is a simulator of a *Fast DDS* application that generates deterministic statistics data for testing the *Fast DDS Monitor*.	OFF ON	OFF
COMPLEX_MOCK	Build the *Fast DDS Statistics Backend* complex mock and link the *Fast DDS Monitor* application against it. The complex mock is a simulator of a *Fast DDS* application that random statistics data	OFF ON	OFF

2.21 Fast DDS Monitor with ROS 2

Fast DDS Monitor is a useful tool for monitoring and studying a ROS 2 network. The automatic discovery of entities in a local network allows to easily see the different Participants that are running, as its Endpoints, the Topics that each one is using, and even the network interfaces they are using to communicate with each other. Additionally, one could receive statistical data from every endpoint in the network. This data is very useful to analyze the performance and seek any possible communication problem in the network.

ROS 2 relies on DDS communication protocol to communicate different nodes. There are different RMWs (ROS MiddleWare) available in the latest ROS 2 versions. Make sure you are using Fast DDS as your ROS 2 middleware in order to receive statistical data and monitor your network.

In the following sections, instructions for using Fast DDS with Statistics in different ROS2 versions are provided, as well as a brief tutorial demonstrating the integration of Fast DDS Monitor with ROS2 in a simple scenario.

2.21.1 Galactic

This section shows how to install and deploy some ROS 2 Galactic nodes in order to monitor them with Fast DDS Monitor.

Installation

First of all, follow the *Fast DDS Monitor on Linux* on this documentation to install Fast DDS Monitor.

Fast DDS is not installed by default in the ROS 2 Galactic release. Thus, first required step is to download Fast DDS and compile it with statistics.

Installation from sources

Follow the ROS 2 galactic installation from sources documentation Fast DDS is downloaded within the rest of the packages. The only consideration here is to compile `fastrtps` library with the Statistics Module activated. When compiling with colcon, the following arguments must be provided:

```
colcon build --symlink-install --cmake-args -DFASTDDS_STATISTICS=ON
```

Execution

We are going to recreate a simple DDS network with one `talker` and one `listener` from ROS 2 demo nodes.

Execute Fast DDS Monitor

Initiate Fast DDS Monitor by running the executable file created in the installation process. Once Fast DDS Monitor is launched, start a monitor in domain `0` (default domain).
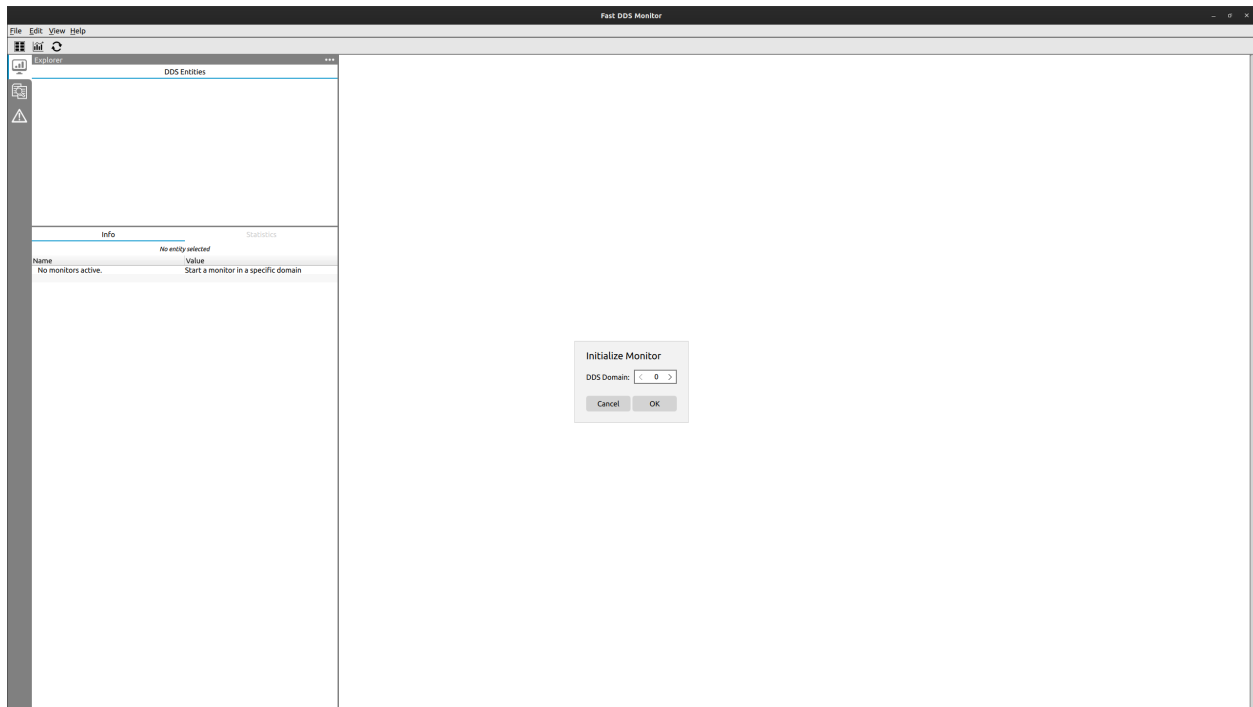
Execute ROS 2 demo nodes with statistics

To execute ROS 2 nodes with statistics two aspects of the configuration must be taken into account:

- The middleware used must be Fast DDS, set through an environment variable.

- In order to activate the publication of statistical data, Fast DDS requires an environment variable specifying those kinds of statistical data to be reported.

To execute each of the nodes, run the following commands in different terminals:

```
export RMW_IMPLEMENTATION=rmw_fastrtps_cpp

export FASTDDS_STATISTICS="HISTORY_LATENCY_TOPIC;NETWORK_LATENCY_TOPIC;PUBLICATION_
↪THROUGHPUT_TOPIC;\
SUBSCRIPTION_THROUGHPUT_TOPIC;RTPS_SENT_TOPIC;RTPS_LOST_TOPIC;\
HEARTBEAT_COUNT_TOPIC;ACKNACK_COUNT_TOPIC;NACKFRAG_COUNT_TOPIC;\
GAP_COUNT_TOPIC;DATA_COUNT_TOPIC;RESENT_DATAS_TOPIC;SAMPLE_DATAS_TOPIC;\
PDP_PACKETS_TOPIC;EDP_PACKETS_TOPIC;DISCOVERY_TOPIC;PHYSICAL_DATA_TOPIC"

ros2 run demo_nodes_cpp listener
```

```
export RMW_IMPLEMENTATION=rmw_fastrtps_cpp

export FASTDDS_STATISTICS="HISTORY_LATENCY_TOPIC;NETWORK_LATENCY_TOPIC;PUBLICATION_
↪THROUGHPUT_TOPIC;\
SUBSCRIPTION_THROUGHPUT_TOPIC;RTPS_SENT_TOPIC;RTPS_LOST_TOPIC;\
HEARTBEAT_COUNT_TOPIC;ACKNACK_COUNT_TOPIC;NACKFRAG_COUNT_TOPIC;\
GAP_COUNT_TOPIC;DATA_COUNT_TOPIC;RESENT_DATAS_TOPIC;SAMPLE_DATAS_TOPIC;\
PDP_PACKETS_TOPIC;EDP_PACKETS_TOPIC;DISCOVERY_TOPIC;PHYSICAL_DATA_TOPIC"
```

```
ros2 run demo_nodes_cpp talker
```
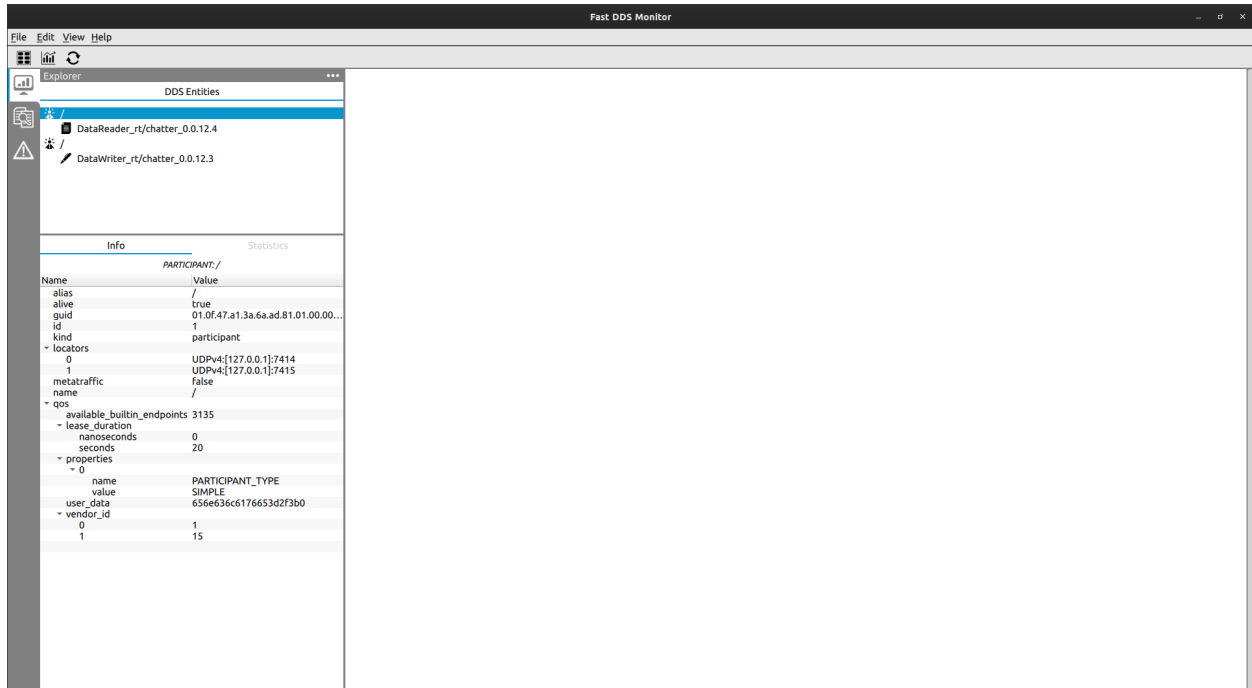
Remember to source your ROS 2 installation before every ros2 command.

Monitoring network

Now one should see in the *DDS Panel* two new Participants.

Alias

Participants in ROS 2 are named / by default. In order to differentiate them one could change the alias of the Participant (see *Change entity alias*). The `talker` would be the one with one writer, and the `listener` the one with a reader.

Physical data

In order to see the information of the host and the physical context where every node is running, go to the *Explorer Panel* and activate the *Physical Panel*. There, the host, user and process of each node are displayed.
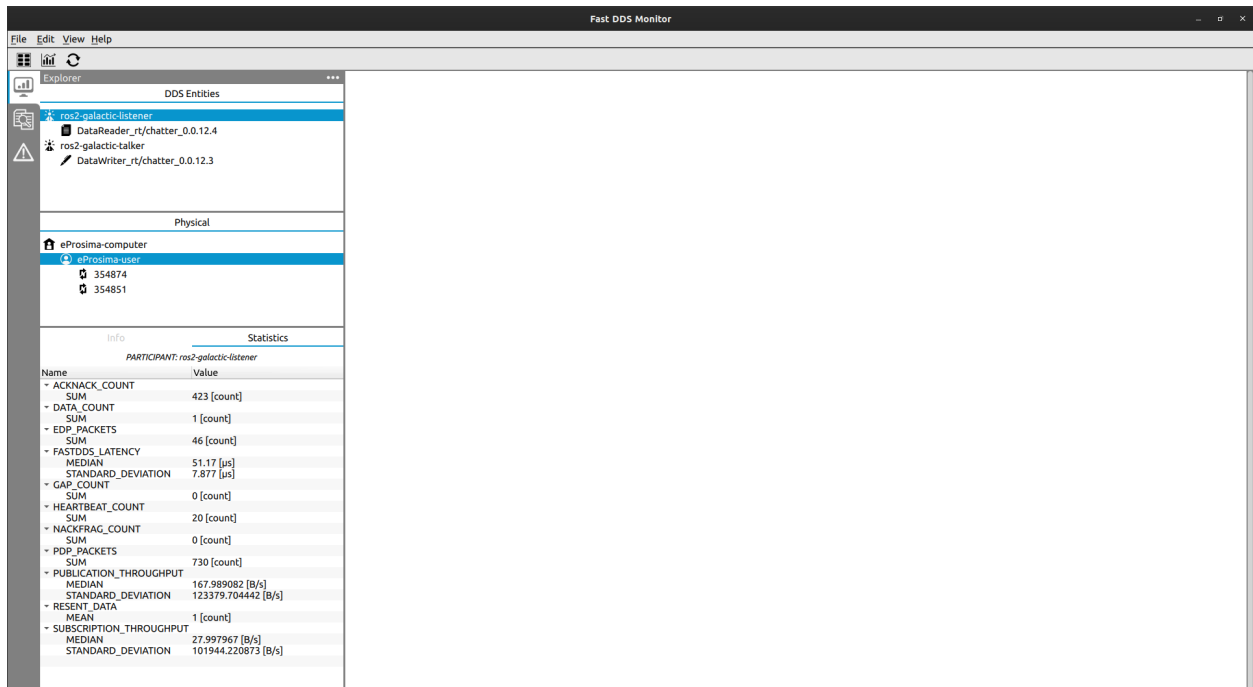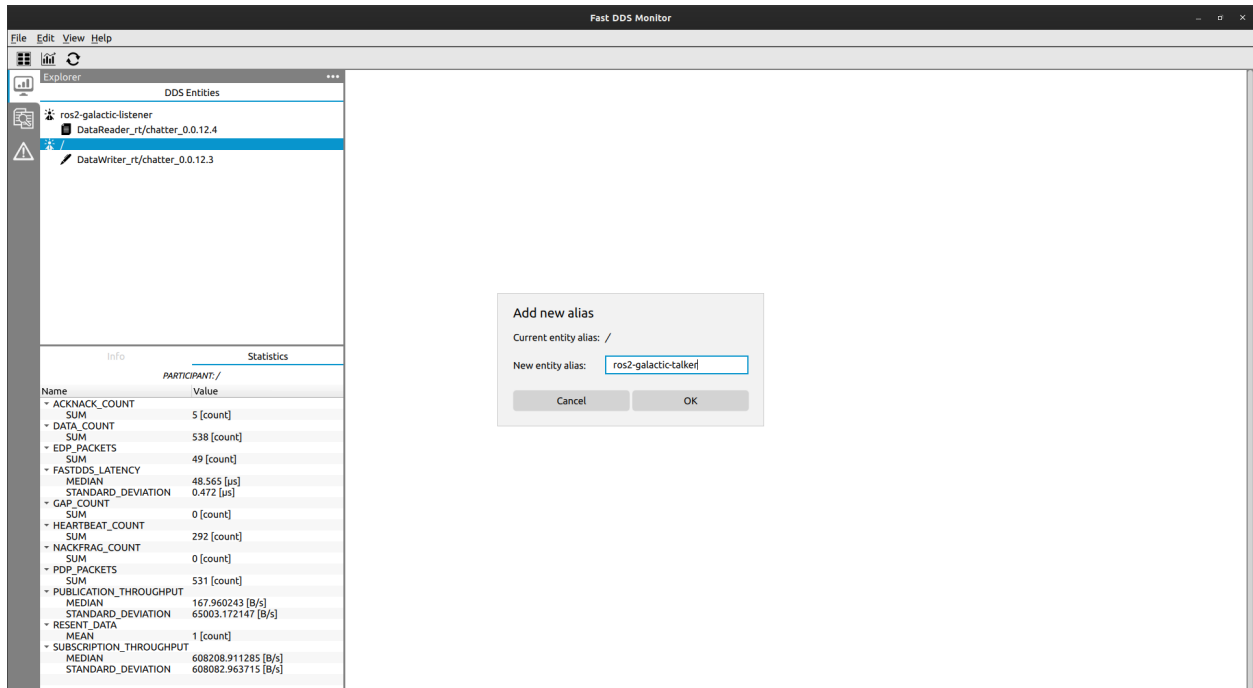
Statistical data

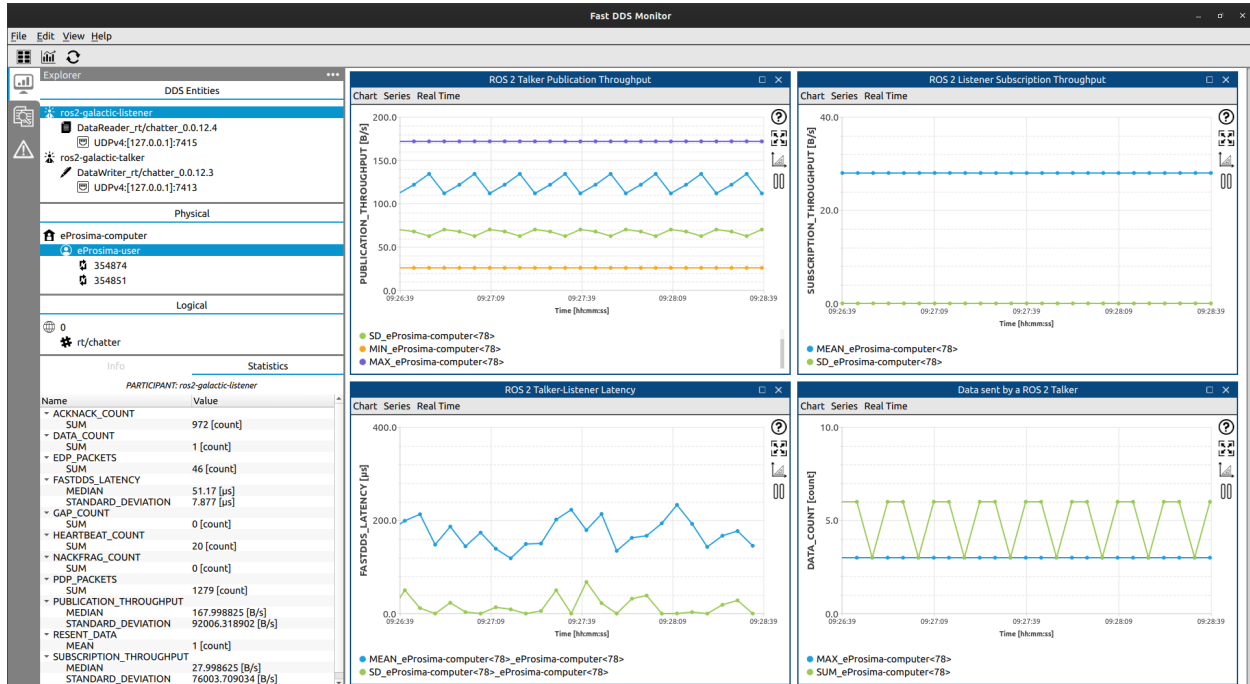To show statistical data about the communication between the `talker` and the `listener`, follow the steps to *Create Dynamic Series Chart* and plot this statistical data in a real time chart.

Introspect metatraffic topics

Fast DDS Monitor filters by default the topics used for sharing metatraffic and the endpoints related to them so the user can inspect their network easily. These topics are the ones that ROS 2 uses for discovery and configuration purposes, such as `ros_discovery_info`, as well as those used by Fast DDS to report statistical data.

In order to see these topics in the monitor, click *View->Show Metatraffic* menu button (see *Hide/Show Metatraffic*). Now, these topics are shown in the logical panel, and also the Readers and Writers associated to them under their respective Participants.

Video Tutorial

There is a video tutorial going through the steps described in this section.

2.22 eProsima Docker Image

eProsima provides the eProsima Fast DDS Suite Docker image for those who want a quick demonstration of Fast DDS running on an Ubuntu platform. It can be downloaded from eProsima's downloads page.

This Docker image was built for Ubuntu 20.04 (Focal Fossa).

To run this container you need Docker installed. From a terminal run
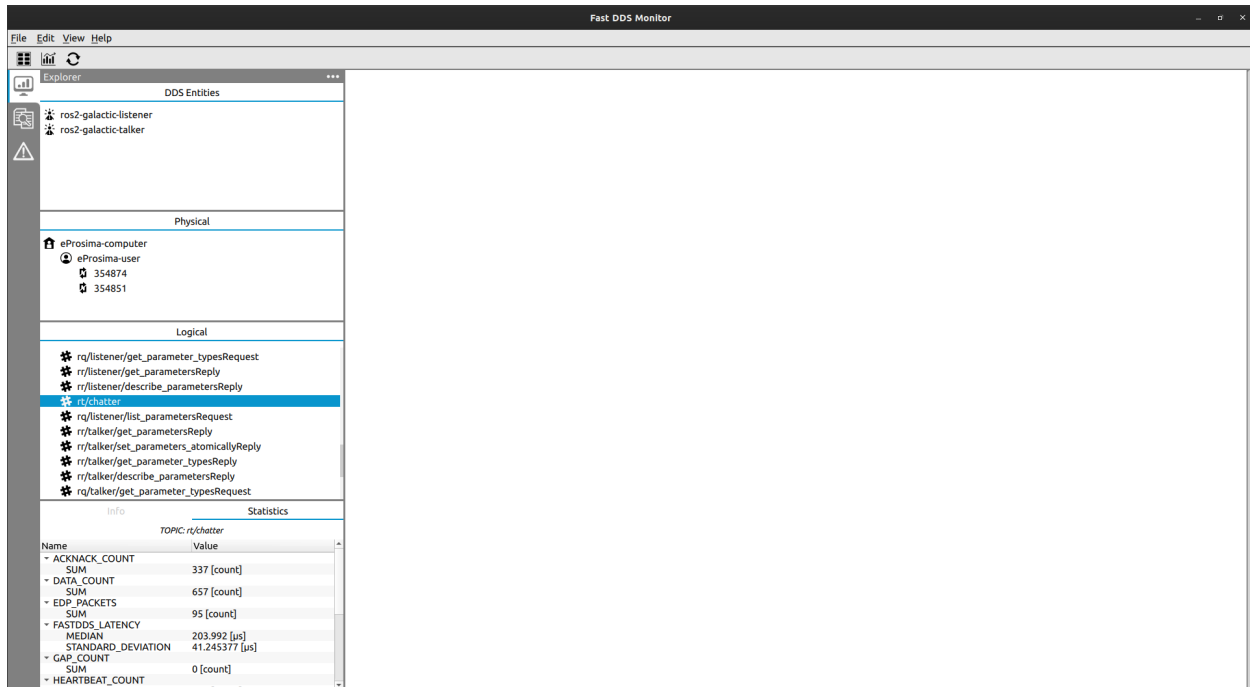
```
$ sudo apt install docker.io
```

2.22.1 Fast DDS Suite

This Docker image contains the complete Fast DDS suite. This includes:

- *eProsima Fast DDS libraries and examples*: *eProsima Fast DDS* is a C++ implementation of the DDS (Data Distribution Service) Specification, a protocol defined by the Object Management Group (OMG). The *eProsima Fast DDS* library provides both an Application Programming Interface (API) and a communication protocol that deploy a Data-Centric Publisher-Subscriber (DCPS) model, with the purpose of establishing efficient and reliable information distribution among Real-Time Systems. *eProsima Fast DDS* is predictable, scalable, flexible, and efficient in resource handling.

  This Docker Image contains the Fast DDS libraries bundled with several examples that showcase a variety of capabilities of eProsima's Fast DDS implementation.

  You can read more about Fast DDS on the Fast DDS documentation page.

- *Shapes Demo*: eProsima Shapes Demo is an application in which Publishers and Subscribers are shapes of different colors and sizes moving on a board. Each shape refers to its own topic: Square, Triangle or Circle. A single instance of the eProsima Shapes Demo can publish on or subscribe to several topics at a time.

  You can read more about this application on the Shapes Demo documentation page.

- *Fast DDS Monitor*: eProsima Fast DDS Monitor is a graphical desktop application aimed at monitoring DDS environments deployed using the *eProsima Fast DDS* library. Thus, the user can monitor in real time the status of publication/subscription communications between DDS entities. They can also choose from a wide variety of communication parameters to be measured (latency, throughput, packet loss, etc.), as well as record and compute in real time statistical measurements on these parameters (mean, variance, standard deviation, etc.).

To load this image into your Docker repository, from a terminal run

```
$ docker load -i ubuntu-fastdds-suite:<FastDDS-Version>.tar
```

You can run this Docker container as follows

```
$ xhost local:root
$ docker run -it --privileged -e DISPLAY=$DISPLAY -v /tmp/.X11-unix:/tmp/.X11-unix \
ubuntu-fastdds-suite:<FastDDS-Version>
```

From the resulting Bash Shell you can run each feature.

Fast DDS Examples

Included in this Docker container is a set of binary examples that showcase several functionalities of the Fast DDS libraries. These examples' path can be accessed from a terminal by typing

```
$ goToExamples
```

From this folder you can access all examples, both for DDS and RTPS. We detail the steps to launch two such examples below.

Hello World Example

This is a minimal example that will perform a Publisher/Subscriber match and start sending samples.

```
$ goToExamples
$ cd HelloWorldExample/bin
$ tmux new-session "./HelloWorldExample publisher 0 1000" \; \
split-window "./HelloWorldExample subscriber" \; \
select-layout even-vertical
```

This example is not constrained to the current instance. It is possible to run several instances of this container to check the communication between them by running the following from each container.

```
$ goToExamples
$ cd HelloWorldExample/bin
$ ./HelloWorldExample publisher
```

or

```
$ goToExamples
$ cd HelloWorldExample/bin
$ ./HelloWorldExample subscriber
```

Benchmark Example

This example creates either a Publisher or a Subscriber and on a successful match starts sending samples. After a few seconds the process that launched the Publisher will show a report with the number of samples transmitted.

On the subscriber side, run:

```
$ goToExamples
$ cd Benchmark/bin
$ ./Benchmark subscriber udp
```

On the publisher side, run:

```
$ goToExamples
$ cd Benchmark/bin
$ ./Benchmark publisher udp
```

Shapes Demo

To launch the Shapes Demo, from a terminal run

```
$ ShapesDemo
```

eProsima Shapes Demo usage information can be found on the Shapes Demo First Steps.

Fast DDS Monitor

To launch the Fast DDS Monitor, from a terminal run

```
$ fastdds_monitor
```

eProsima Fast DDS Monitor usage information can be located on the Fast DDS Monitor User Manual.

2.23 Version v2.0.0

This release includes the following **features**:

- Split main view into tabs to open different views simultaneously
- Display a *domain graph* view with the entities that belongs the domain
- Display connections between physical entities in nested groups
- Display connections between endpoints through topics in the domain graph
- Filter domain graph view per topic
- Display a *Problem summary* view in a new footer section.
- Filter problem summary per DDS entity
- Display eProsima products icon as part of the DDS entity information

This release includes the following **dependencies update**:

	Repository	Old Version	New Version
Fast CDR	eProsima/Fast-CDR	v1.1.0	v2.1.2
Fast DDS	eProsima/Fast-DDS	v2.11.0	v2.13.0
Fast DDS Statistics Backend	eProsima/Fast-DDS-statistics-backend	v0.11.0	v1.0.0

2.24 Previous versions

2.24.1 Version v1.5.0

This release includes the following **features**:

- Schedule the dump of the database information to a JSON file.

2.24.2 Version v1.4.0

This release includes the following **features**:

- Button to dump the information from the database to a JSON file.

- Button to remove inactive entities from the database.

- Button to clear the statistical data of all the entities.

- Button to schedule the removal of old data every **x** seconds.

- Support for limiting data points in dynamic data series.

2.24.3 Version v1.3.0

This release includes the following **bug fixes**:

- Fix error in y-axis resize that went to infinite.

This release includes the following new **internal changes**:

- Upgrade Fast DDS v2.8.0 to Fast DDS v2.9.0.

- Upgrade Fast DDS Statistics Backend v0.7.1. to Fast DDS Statistics Backend v0.8.0.

2.24.4 Version v1.2.0

This release includes the following new **documentation sections**:

- Guide to solve the non-reception of statistics data after enabling them.

This release includes the following new **internal changes**:

- Upgrade Fast DDS v2.7.0 to Fast DDS v2.8.0.

- Upgrade Fast DDS Statistics Backend v0.7.0 to Fast DDS Statistics Backend v0.7.1.

2.24.5 Version v1.2.0

This release includes the following new **documentation sections**:

- Windows installation guide.

This release includes the following new **minor features**:

- Windows app icon.
- Support GTest new version.

This release includes the following **bugfixes**:

- Fixes included in Fast DDS Statistics Backend v1.2.0.

2.24.6 Version v1.1.0

This release includes the following new **features**:

- Add Hide/Show metatraffic entities capabilities.
- Support cumulative data in Dynamic Charts.

This release includes the following **improvements**:

- Application background
- Move last clicked logic from QML to C++.
- Blank text when open "*change alias*" dialog.
- Documentation improvement:
  - Fast DDS Suite documentation.
  - Tutorial for ROS 2 galactic.

This release includes the following **bugfixes**:

- Fix Y axis rendering when all values were 0.
- Fix timestamps in Historical charts.
- Fix deselect entity minor bugs.
- Fix width QML warning.
- Fix mu(micro) symbol display in Windows

2.24.7 Version v1.0.0

This is the first stable release of *eProsima Fast DDS Monitor*.

2.24.8 Version v0.2.0

This release includes the following **improvements**:

- Add ToolTips for dialog parameters

- Add Real-Time charts

- Refactor sidebars * New left sidebar with all the DDS, physical and logical entities together with its corresponding information. * Separated panel for Fast DDS Monitor status. * New panel to display Issues information.

- Possibility to change the Entity name setting an alias for it.

- Entities not alive are marked in gray color.

- Possibility to display/hide the not alive entities.

- Entity menu displayed right-clicking on them.

- Support for Windows platforms.

- Resizable charts in three different sizes: full screen, two chart boxes per row and three chart boxes per row.

- Display two chart boxes in a row by default.

- Re-nameable chart boxes.

- Export charts data to CSV. It is possible to export a single series, a whole ChartBox or all ChartBoxes of the monitor to a single CSV file.

- Change the cursor shape when hovering over the legend and when action buttons are pressed over the selected chart.

- Add information button on each ChartBox.

- Normalize Y axis.

- Button to set axes in the ChartView.

- Button to reset zoom.

- Maximize and minimize chart boxes.

- Initialize monitor in a Discovery Server network.

This release includes the following **bugfixes**:

- Get correct Entity Id on the add statistics series dialog.

- Set correct Entity Id on the add statistics series dialog.

- Icons colorization in Windows platforms.

- Avoid launch a command prompt when the application is executed in Windows.

- TreeModels refactor * Convert all arrays to dictionaries with the array index as the element key. * Fix item duplication when collapsed.

- When receiving data from a new entity, it is added to its corresponding entity model as an additional model item instead of destroying and rebuilding the entire model.

- Avoid destroy ChartBoxes when they are out of the view.

- Increase Y axis in historic ChartBoxes so all the data is visible.

- Refactor Historic series so all the internal data is deleted when graphics are removed.

- Increased the number of characters for series labels.

- Change displayed chart units for some DataKinds.

- Add units to the statistics entity summary.

- Charts dashboard layout scroll bar always visible.

- Sort data from backend.

- When receiving data of DataKinds that are measured in *counts*, the data points that are NaN are replaced by 0.

2.24.9 Version v0.1.0

This is the first release of *eProsima Fast DDS Monitor*.

This release includes several **features** regarding the monitoring of DDS network and the visualization of a real time working network, as instrumentation data related with this network:

This release includes the following **visual features**

- Visual Interface to monitor a DDS network.

- Visualize *alive* entities in a DDS network in real time.

- Represent instrumentation data of several kinds of a DDS network:

    – Latency

    – Throughput

    – Data sent

    – Metadata sent

    – Discovery time

- Represent instrumentation data regarding different cumulative functions:

    – Mean

    – Median

    – Standard deviation

    – Maximum

    – Minimum

    – Count

    – Sum

- Real time events:

    – New entities discovered.

    – New issues detected.

This release includes the following **interactive features**

- Initiate monitoring a DDS network:

    – Monitoring a DDS domain.

    – Monitoring a Discovery Server network.

- Represent any data kind with any cumulative function regarding any one or two entities in the network.

- Select an Entity in order to retrieve information about the entity configuration and its data summary.

- User friendly modifiable layout:

  - Resize windows and panels.

  - Create, delete and move charts and panels.