Shapes Demo Documentation

Release 2.7.2

eProsima software team

Feb 13, 2023

INSTALLATION MANUAL

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.

It demonstrates the capabilities of eProsima *Fast DDS* or as a proof of interoperability with other RTPS-compliant implementations.

ONE

WINDOWS INSTALLATION FROM BINARIES

To install eProsima Shapes Demo from binaries just go to the eProsima website and download the desired version of eProsima Shapes Demo. Download links can be found by following the link below.

https://www.eprosima.com/index.php/products-all/eprosima-shapes-demo

After downloading, unzip the content in the directory of preference and execute the file *ShapesDemo.exe* located in the *bin* directory.

TWO

LINUX INSTALLATION FROM BINARIES

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
```

1. Allow root to use graphical interface:

```
xhost local:root
```

2. Load the Docker image:

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

3. Run the Docker image:

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

4. Launch the Shapes Demo:

```
ShapesDemo
```

For more information regarding the capabilities of eProsima's Fast DDS Suite Docker image, you can check the *Fast DDS Docker Image documentation*

LINUX INSTALLATION FROM SOURCES

For simplicity, the eProsima Shapes Demo installation manual follows the Colcon installation, since eProsima *Fast DDS* and *Fast CDR* dependencies are downloaded and installed at the same time that eProsima Shapes Demo is built. However, the user must assure that Qt5 is installed.

3.1 Colcon installation

To install eProsima Shapes Demo using colcon, please follow the steps below:

1. Install the eProsima Fast DDS dependencies and verify that the system meets the installation requirements. The complete list of requirements and dependencies can be found in the Fast DDS Linux Installation Manual. Specifically, follow the steps outlined in sections Requirements and Dependencies.

2. Install the ROS 2 development tools (Colcon and Vcstool) by executing the following command:

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

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

3. Create a *ShapesDemo* directory and download the repos file that will be used to install eProsima Shapes Demo and its dependencies:

```
mkdir -p ShapesDemo/src && cd ShapesDemo
wget https://raw.githubusercontent.com/eProsima/ShapesDemo/master/ShapesDemo.repos
vcs import src < shapes-demo.repos
```

3. Build the packages:

```
colcon build
```

4. Link the application executable to make it accessible from the current directory:

```
source install/setup.bash
```

5. Run eProsima Shapes Demo:

```
ShapesDemo
```

FOUR

USING EPROSIMA SHAPES DEMO

This section serves as a guide to the main menus of eProsima Shapes Demo application. After the executable is launched, a window similar to the one presented in the following image should be displayed.

4.1 Publishing a Topic (Shape)

The Publish button allow the users to define the Shape (topic) and Quality of Service (QoS) for their publication. The following image shows an example of the Publication menu.
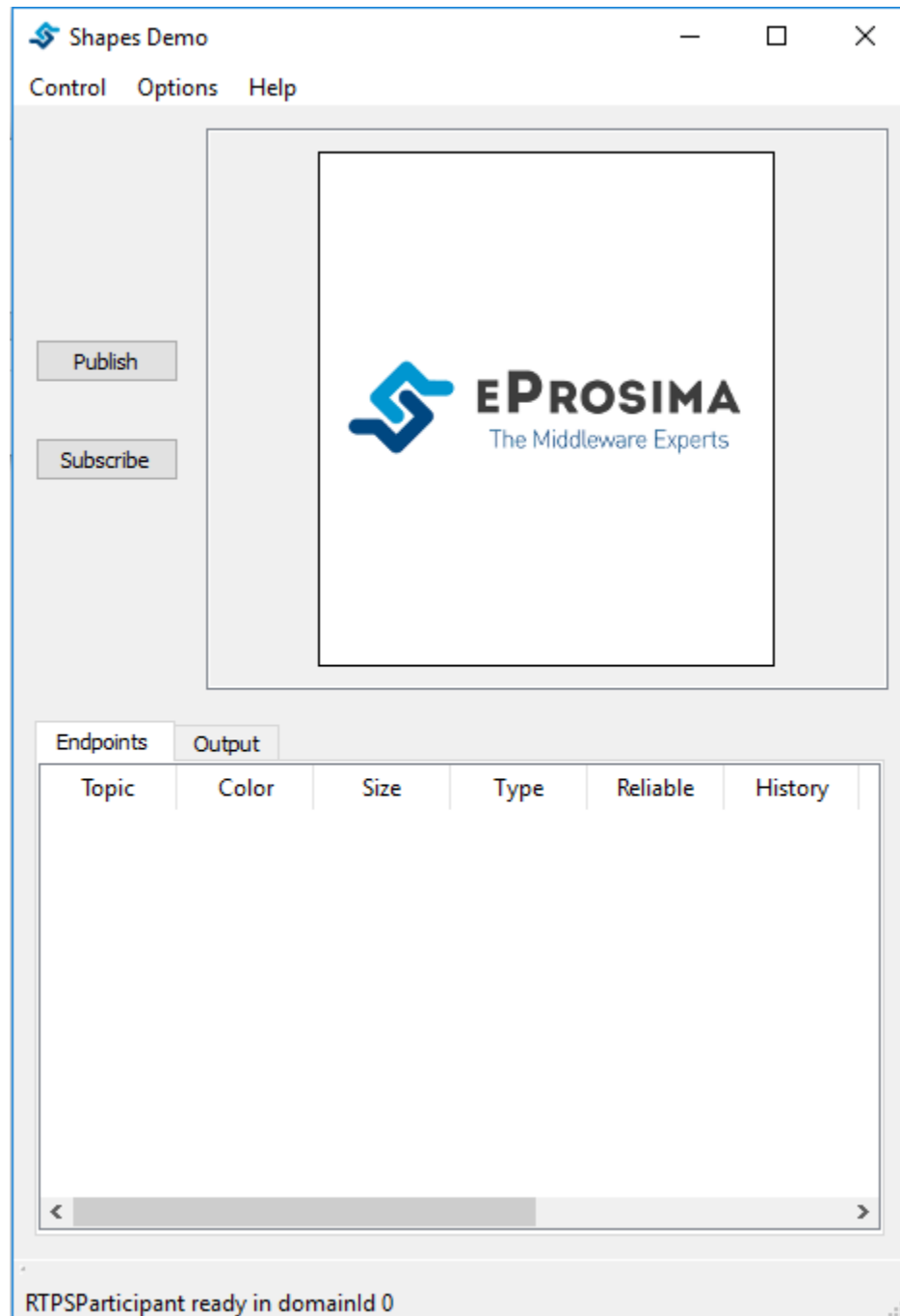
There are multiple parameters that the user can define in this menu:

- **Shape:** This parameter defines the topic where the publication is going to occur. Three different shapes can be published: `Square`, `Circle` and `Triangle` (see Fast DDS Topic Documentation).

- **Color:** The user can define the color of the shape. This parameter will be used as key; that is, a way to distinguish between multiple instances of the same shape (see Fast DDS Topics with key documentation).

- **Size:** This parameter allows to control the size of the shape. The size can vary between 1 and 99.

- **Partition:** The user can select different partitions to differentiate groups of publishers and subscribers. The user can select between four partitions (A, B, C and D). Additionally the user can select the * partition, that will be matched against all other partitions (see Fast DDS Partitions Documentation).

Note: Using the wildcard (*) partition is not the same as not using any partition. A publisher that uses the wildcard partition will not be matched with a subscriber that do not defines any partitions.

- **Reliable:** The user can select to disable the Reliable check-box to use a Best-Effort publisher (see Fast DDS ReliabilityQosPolicy Documentation).

- **History and Durability:** The publishers's History is set to KEEP_LAST. The user can select the number of samples that the publisher is going to save and whether this History is going to be VOLATILE or TRANSIENT_LOCAL. The latter will send that last stored values to subscribers joining after the publisher has been created. (see Fast DDS DurabilityQosPolicy Documentation).

- **Liveliness:** The user can select the Liveliness QoS of the publisher from three different values: AUTOMATIC, MANUAL_BY_PARTICIPANT and MANUAL_BY_TOPIC. The Lease Duration value and Announcement Period can also be configured. The latter only applies if Liveliness is set to AUTOMATIC or MANUAL_BY_PARTICIPANT (see Fast DDS LivelinessQosPolicy Documentation).

- **Ownership:** The Ownership QoS determines whether the key (color) of a Topic (Shape) is owned by a single publisher. If the selected ownership is EXCLUSIVE the publisher will use the Ownership strength value as the strength of its publication. Only the publisher with the highest strength can publish in the same Topic with the same Key (see Fast DDS OwnershipQosPolicy Documentation).

- **Deadline:** The Deadline QoS determines the maximum expected amount of time between samples. When the deadline is missed the application will be notified and a message will be printed on the console (see Fast DDS DeadlineQosPolicy Documentation).

- **Lifespan:** The Lifespan QoS determines the duration while the sample is still valid. When a sample's lifespan expires, it will be removed from publisher and subscriber histories. (see Fast DDS LifespanQosPolicy Documentation).
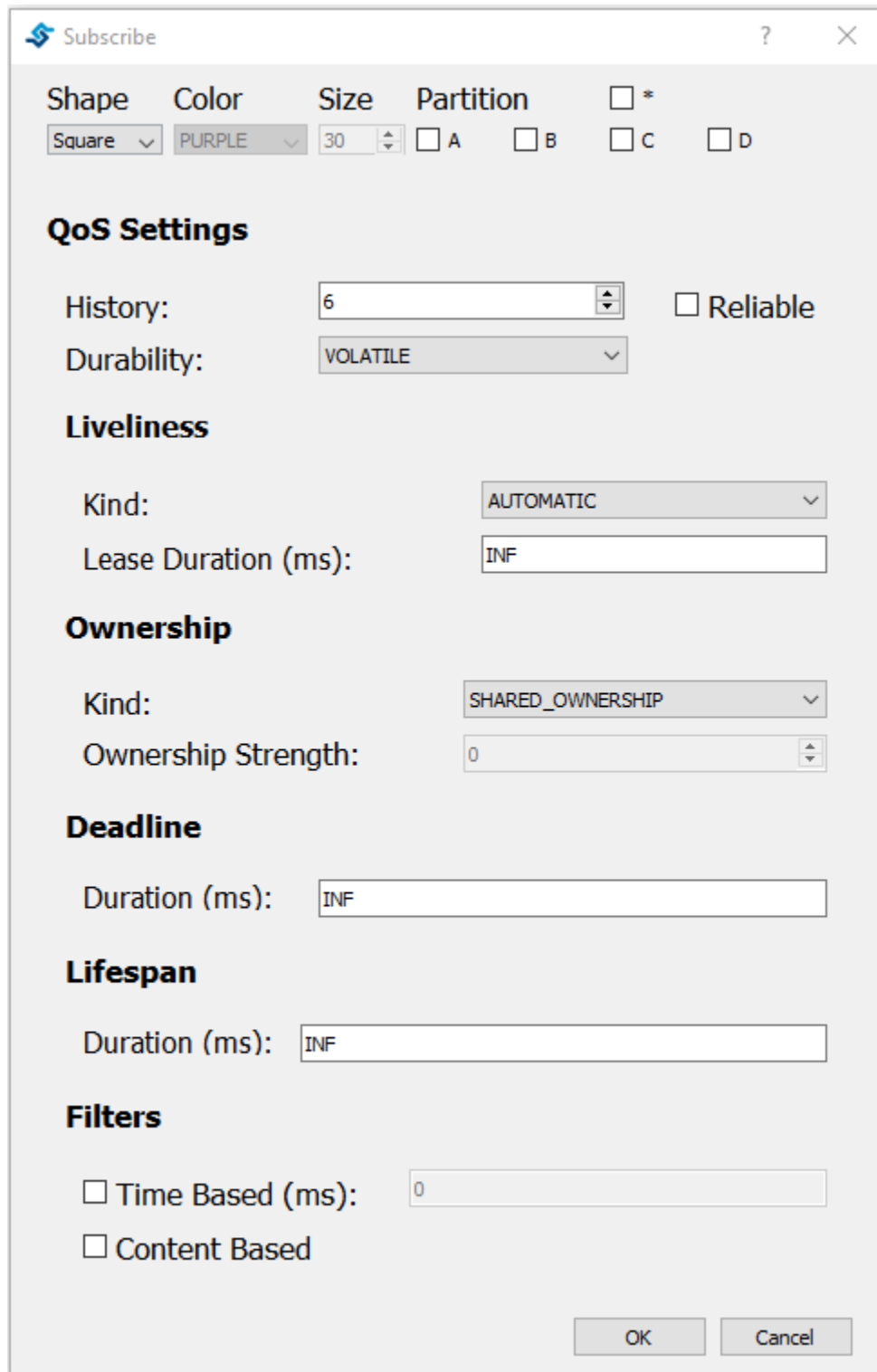

Note: Using Lifespan QoS will not have any visual effect.

4.2 Subscribing to a Topic (Shape)

When the Subscriber button is pressed, a new window appear to allow the user to define the Shape (topic) and Quality of Service (QoS) for its subscription. The following image shows an example of the Subscribe menu.

This menu is highly similar to the Publication menu but the user cannot change the color and size of the Shape, and it has additional elements:

- **Liveliness:** This QoS policy is applied in the same way as in the publisher except for the Announcement Period, which does not apply for the Subscriber (see Fast DDS LivelinessQosPolicy Documentation).

- **Time Based Filter:** This value can be used by the user to specify the minimum amount of time (in milliseconds) that the subscriber wants between updates. (see Fast DDS TimeBasedFilterQosPolicy Documentation).

- **Content Based Filter:** This filter draws a rectangle in the instances window. Only the shapes that are included in this rectangle are accepted while the rest of them are ignored. The user can dynamically resize and move this content filter.

Note: Using Lifespan QoS will not have any visual effect.

4.3 Participant configuration

The eProsima Shapes Demo application allows the user to define Participant policies. To see the Options window, please go to *Options->Participant Configuration* in the main bar. The following image shows the Options Menu.
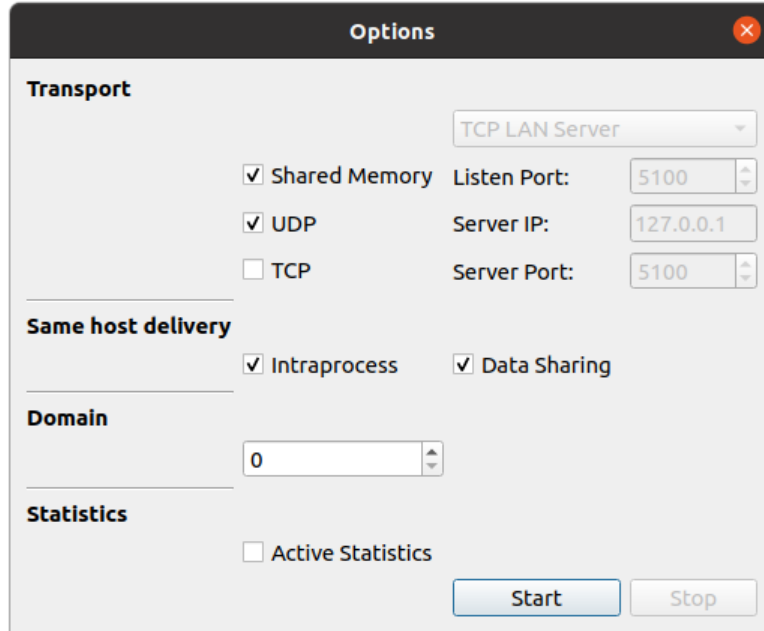
- **Transport Protocol:** You can select between UDP protocol, TCP LAN Server, TCP WAN Server or TCP Client, and Shared Memory protocol.

    - **UDP:** With UDP Protocol the application will work sending multicast packets to communicate with other apps. (See Fast DDS UDP Transport Documentation).

    - **TCP:** TCP protocol needs a minimal configuration to create the connection between the machines: (See Fast DDS TCP Transport Documentation).

        * If the application is going to run as a *LAN server*, it only needs to set the listening port where it is going to accept connections (note that firewall must be configured to allow inbound traffic).

        * If the application is going to run as a *WAN server*, it needs to set the listening port where it is going to accept connections and the server WAN address (note that firewall must be configured to allow inbound traffic and router must relay listening port traffic to server machine).

        * If the application is going to run as a *client* it needs to know the IP address of the server (or its WAN address if both instances don't share network) and the port where the server is listening for connections.

    - **Shared Memory (SHM):** Activating Shared Memory protocol will use the Shared Memory Transport, a *Fast DDS* feature that allows a faster and more efficient communication for Participants running in the same host. (See Fast DDS Shared Memory Transport Documentation).

    - **Default** In case no transport has been activated, *Fast DDS* default transports will be used (UDP + SHM) (See Fast DDS Transports Documentation).

- **Same host delivery:** *Fast DDS* has some features that allow Participants running in the same host or process to share resources in order to improve the communication:

    - **Intraprocess:** Allow using Intraprocess delivery when both Endpoints are running in the same process. (See Fast DDS Intraprocess Documentation).

    - **Data Sharing:** Allow using Data Sharing delivery when both Endpoints are running in the same host. (See Fast DDS Data Sharing Documentation).

- **Domain:** The user can select different Domain IDs. Shapes Demo instances using different Domain IDs will not communicate. To modify the Domain ID the user needs to stop the participant (thus removing all existing publishers and subscribers) and start a new one with the new Domain ID. (See Fast DDS Domain Documentation).

- **Statistics:** The user can activate *Fast DDS Statistics module* so different instrumentation data could be collected and analyzed by the *Fast DDS Statistics Backend*, or be represented by *Fast DDS Monitor*. This module requires to have compiled *Fast DDS* with Statistics Module ON. (See Fast DDS Statistics Module Documentation).

In case that the Participant is already running, it should be stopped in order to change its configuration. This will drop every endpoint already created.

4.4 Preferences

The eProsima Shapes Demo application allows the user to define additional options. To see the Options window, please go to *Options->Preferences* in the main bar. The following image shows the Options Menu.

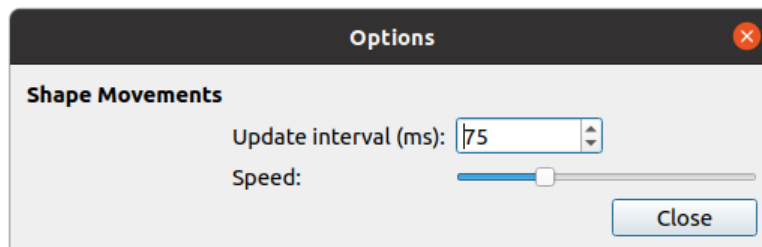The user can customize several aspects of Shapes Demo operation:

- **Update interval:** This value changes the publication period for all the publishers.

- **Speed:** This scroll bar allows the user to change how much the Shape moves between two write calls.

4.5 Endpoints and Output tabs

A table including all created endpoints is also provided. An example of this legend is shown in the following figure.

This table can be used to remove endpoints. Two methods are provided:

- Right click in an endpoint: An option to remove the endpoint is shown.

- Pressing the delete button when the endpoint is selected.

The output tab shows the output log messages. An example of the output tab is shown in the figure below.

	Topic	Color	Size	Type	Reliable	History	Partitions	Ownership	Durability	Liveliness
1	Square	RED	30	Pub	True	6	-	SHARED	TRANSIENT	AUTOMATIC
2	Triangle	CYAN	30	Pub	True	6	-	SHARED	TRANSIENT	AUTOMATIC
3	Circle	GRAY	30	Pub	True	6	-	SHARED	VOLATILE	AUTOMATIC
4	Circle	---	---	Sub	False	6	-	SHARED	VOLATILE	AUTOMATIC

Endpoints Output

```
[11:49:13.997]: RTPSParticipant ready in domainId 0
[11:49:46.266]: Publisher created in topic: RED Square
[11:50:38.268]: Publisher created in topic: CYAN Triangle
[11:50:57.571]: Publisher created in topic: GRAY Circle
[11:53:25.364]: Subscriber created in topic: Circle
```

FIVE

DISCOVERY AND BASIC CONNECTIVITY

In *Fast DDS*, the discovery task is automatic. *Fast DDS* performs the task of finding the relevant information and distributing it to its destination. It means that new nodes are automatically discovered by any other in the network. Please refer to the Fast DDS Discovery Documentation for more information on the various *Fast DDS* discovery mechanisms.

In this test, three Publishers and three Subscribers are launched. At the end, two additional squares will be displayed in each window, reflecting the movements of the original square in real time. That is, subscribers subscribing to the "Square" topics are matched with the publishers of the other instances.

5.1 Step-by-step example implementation

First, three publishers must be created.

1. Create a red square publisher:

   - Start eProsima Shapes Demo (this instance will be referred to as *Instance1*).

   - Click on Publish.

   - Select SQUARE option for Shape and RED for Color.

2. Create a blue square publisher:

   - Start eProsima Shapes Demo (this instance will be referred to as *Instance2*).

   - Click on Publish.

   - Select SQUARE option for Shape and BLUE for Color.

3. Create a black square publisher:

   - Start eProsima Shapes Demo (this instance will be referred to as *Instance3*).

   - Click on Publish.

   - Select SQUARE option for Shape and BLACK for Color.

The current setting should be similar to that shown in the figure below.

Then, three subscribers must be created.

1. Click Subscribe on *Instance1*.

   - Select SQUARE option for Shape.

   - Change the History field from 6 to 1.

2. Click Subscribe on *Instance2*.

   - Select SQUARE option for Shape.
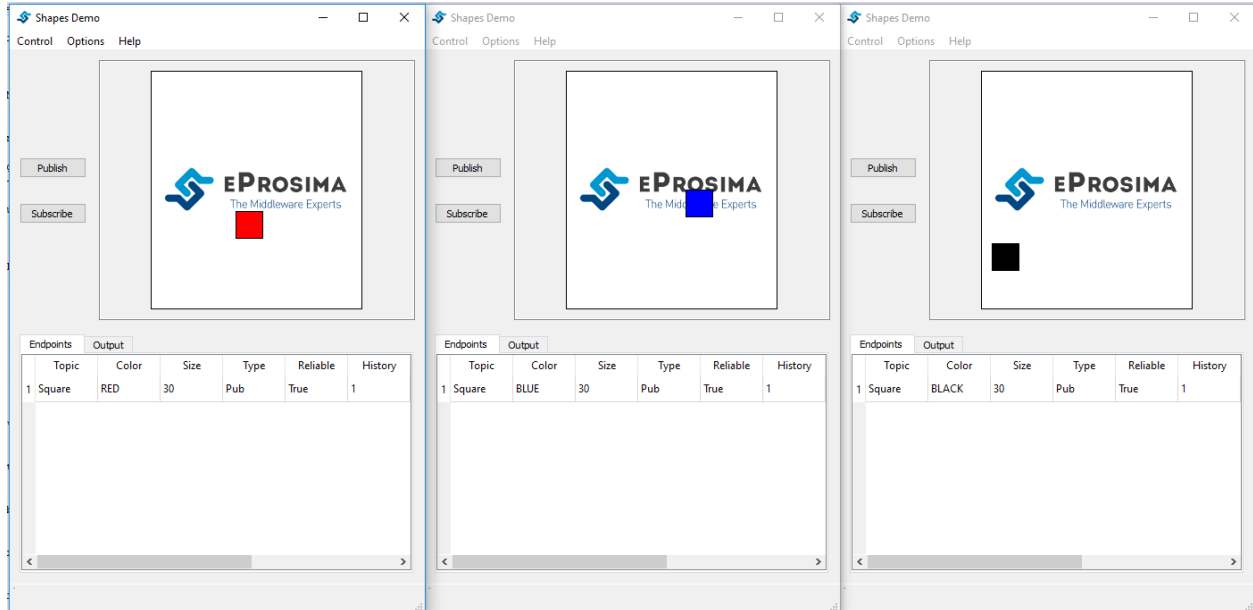
- Change the History field from 6 to 1.

3. Click Subscribe on *Instance3*.

  - Select SQUARE option for Shape.

  - Change the History field from 6 to 1.

The eProsima Shapes Demo windows should look similar to the following image.
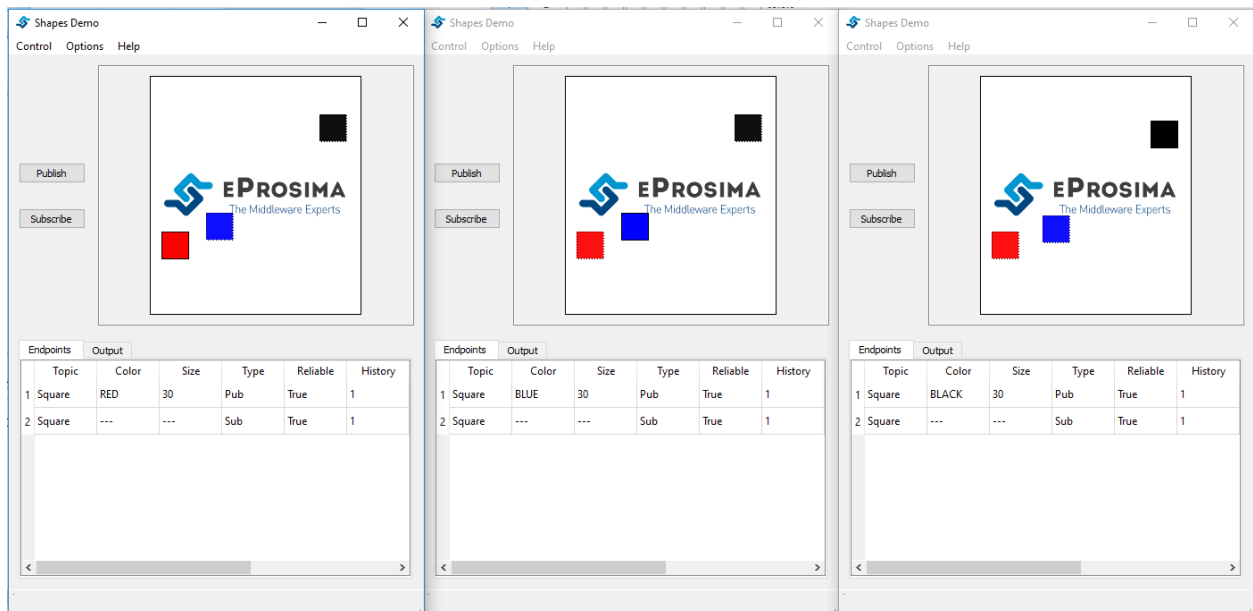
HISTORY AND DURABILITY

A publisher can send messages throughout a Topic even if there are no DataReaders on the network. Moreover, a DataReader that joins to the Topic after some data has been written could be interested in accessing that information. The durability defines how the system will behave regarding those samples that existed on the Topic before the subscriber joins. Please refer to Fast DDS DurabilityQosPolicy Documentation for more information on Durability QoS.

In the following example, the publishers' history is set to `KEEP_LAST`, and there are two options for the durability configuration which are `VOLATILE` and `TRANSIENT_LOCAL`. If `VOLATILE` is selected, the previous data samples will not be sent. However, if `TRANSIENT_LOCAL` is selected, the nth previous data samples will be sent to the late-joining subscriber.

In this example, one hundred red squares will be displayed in *Instance2* and *Instance3*, reflecting the movements of the red square of the publisher from *Instance1*. The leading square indicates the current position of the published square.

6.1 Step-by-step example implementation

First, three instances are launched and a publisher is created in each of them:

1 - Create a red square publisher:

- Start eProsima Shapes Demo (this instance will be referred to as *Instance1*).
- Click on Publish.
- Select SQUARE option for Shape and RED for Color.
- Change the History field from 6 to 100.
- Select TRANSIENT_LOCAL.

2 - Create an orange square publisher:

- Start eProsima Shapes Demo (this instance will be referred to as *Instance2*).
- Click on Publish.
- Select SQUARE option for Shape and ORANGE for Color.
- Change the History field from 6 to 100.
- Select TRANSIENT_LOCAL.

3 - Create a black square publisher:

- Start eProsima Shapes Demo (this instance will be referred to as *Instance3*).
- Click on Publish.

- Select SQUARE option for Shape and BLACK for Color.
- Change the History field from 6 to 100.
- Select TRANSIENT_LOCAL.

The eProsima Shapes Demo environment should look similar to the following figure.
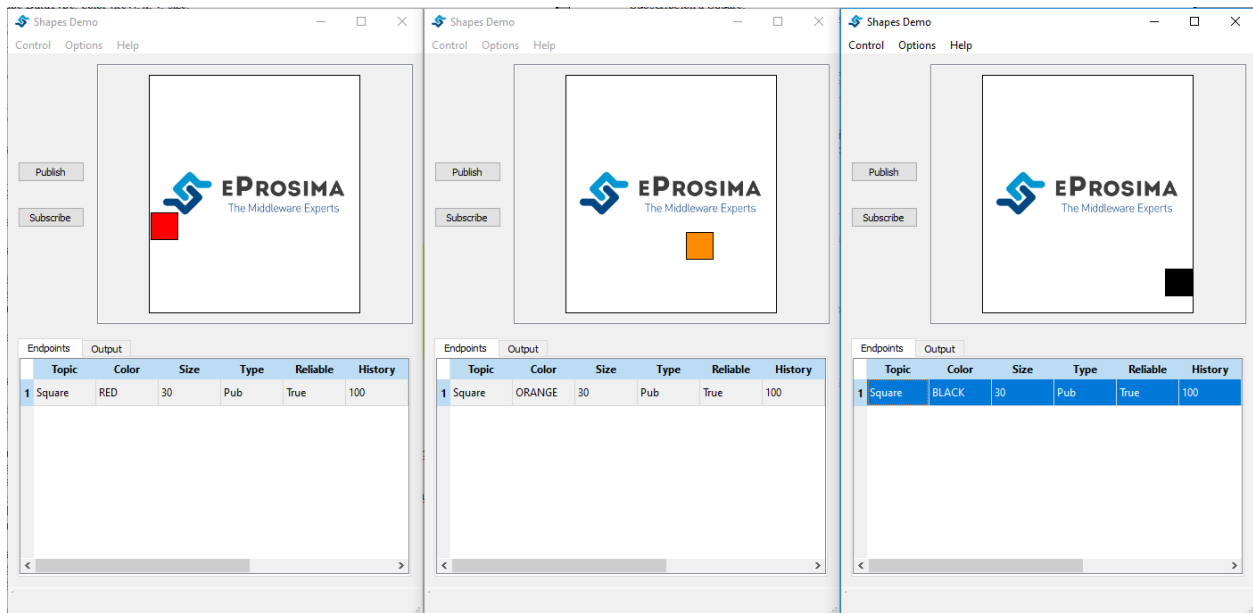
Then, subscriber in each instance is created.

4. Click Subscribe on *Instance1*.

    - Select SQUARE option for Shape.
    - Change the History field from 6 to 100.

5. Click Subscribe on *Instance2*.

    - Select SQUARE option for Shape.
    - Change the History field from 6 to 100.

6. Click Subscribe on *Instance3*.

    - Select SQUARE option for Shape.
    - Change the History field from 6 to 100.

The eProsima Shapes Demo environment should look similar to the following figure.
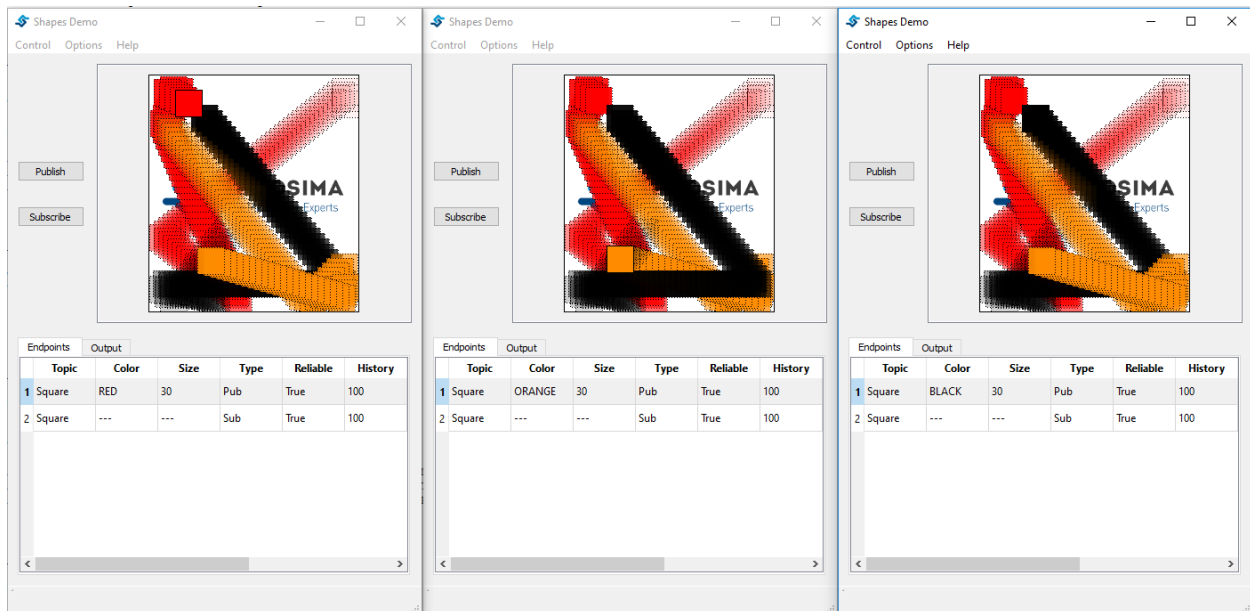
PARTITION

In *Fast DDS*, Partitions can be used to group subscribers and publishers. If a publisher with a partition is deployed, only the subscriber with the same partition will receive data from it. In this demo, there are four partitions (`A`, `B`, `C` and `D`). Additionally, you can select the `*` partition, which refers to all partitions. Please refer to Fast DDS Partitions Documentation for more information on partitions.

In this test, three publishers (Square in Partition A, Circle in Partition B, and Triangle in Partition *), and three subscribers per instance, all of them in Partition A, will be created. Finally, red squares and black triangles in Partition A are set. Therefore, all instances are able to find triangles and squares. However, orange circles are published in partition B and they are only visible in the Instance2.

7.1 Step-by-step example implementation

First, we must create three publishers.

1. Create a red square publisher:

   • Start eProsima Shapes Demo (this instance will be referred to as *Instance1*).

   • Click on Publish.

   • Select SQUARE option for Shape and RED for Color.

   • Change the History field from 6 to 1.

   • Check Partition A.

2. Create an orange circle publisher:

   • Start eProsima Shapes Demo (this instance will be referred to as *Instance2*).

   • Click on Publish.

   • Select CIRCLE option for Shape and ORANGE for Color.

   • Change the History field from 6 to 1.

   • Check Partition B.

3. Create a black triangle publisher:

   • Start eProsima Shapes Demo (this instance will be referred to as *Instance3*).

   • Click on Publish.

   • Select TRIANGLE option for Shape and BLACK for Color.

   • Change the History field from 6 to 1.

   • Check Partition *.

Instance1 will publish red squares in the partition A, Instance2 will publish orange circles in the partition B, and Instance3 will publish black triangles in all partitions.
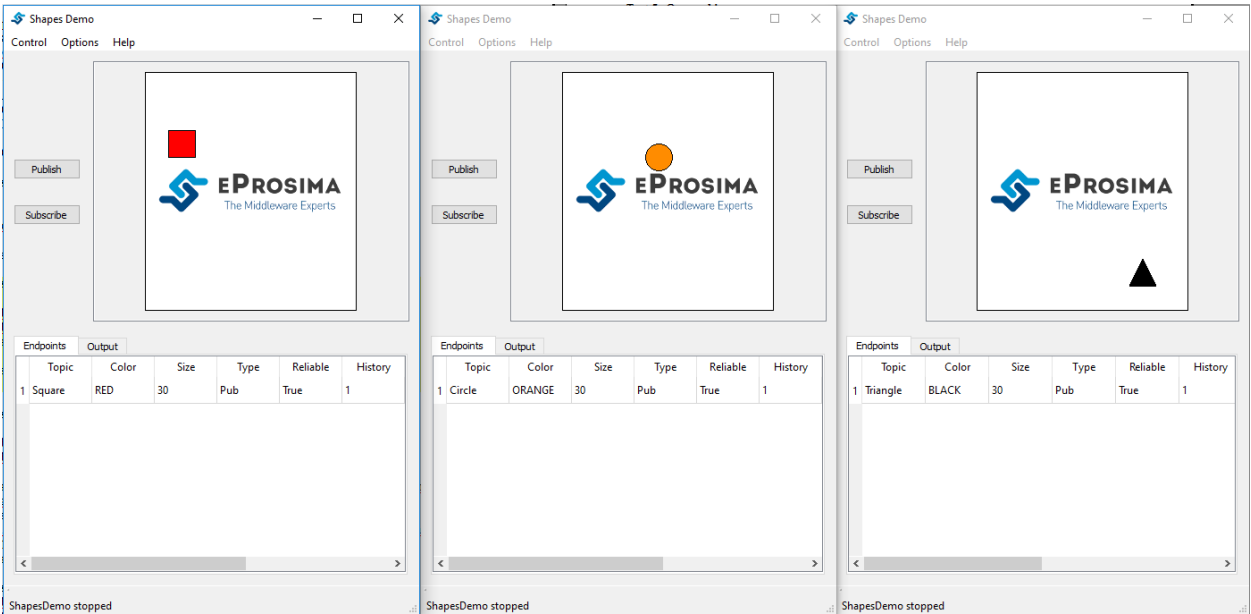
Then, create three subscribers per Instance with the following characteristics.

Shape	Partition	History (Reliable)	Durability	Ownership
Square	A	1 (ON)	VOLATILE	SHARED
Circle	A	1 (ON)	VOLATILE	SHARED
Triangle	A	1 (ON)	VOLATILE	SHARED

EIGHT
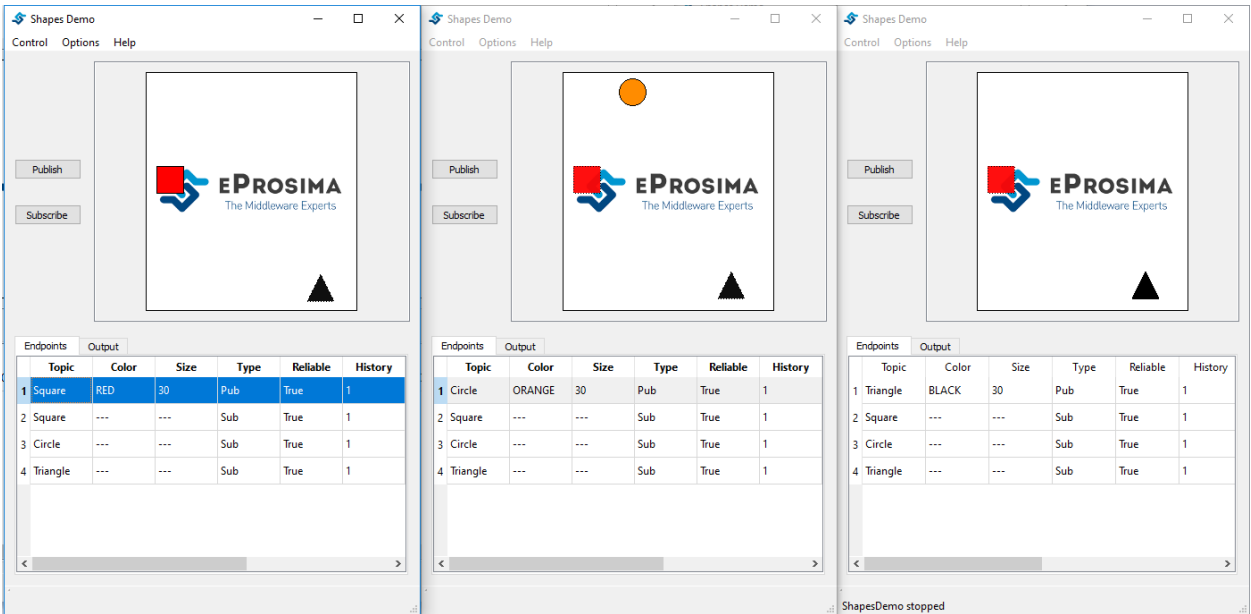
REDUNDANCY AND FAULT TOLERANCE

Fast DDS allows more than one publisher to write data on the same Topic. All publishers may have the same relevance, or one publisher can be set as the primary publisher and keep the rest as secondary publishers. In that case, only the main publisher can send data to the subscribers. Please refer to Fast DDS OwnershipQosPolicy Documentation for more information on Ownership QoS.

The Ownership QoS determines if the Topic (Shape) is owned by a single publisher or not. There are two ownership options: SHARE or EXLUSIVE ownership. The value of a publisher's strength can be set using the EXCLUSIVE configuration. Therefore, only the publisher with the highest strength can send data on this Topic. If SHARE is selected, all the publishers can write data at the same time.

In this test, two publishers with EXCLUSIVE ownership in SQUARE Shape, and one subscriber with EXCLUSIVE ownership at the same Shape, will be created.

8.1 Step-by-step example implementation

First, launch two instances and create a publisher in each of them:

1. Create a red square publisher:

   - Start eProsima Shapes Demo (this instance will be referred to as *Instance1*).

   - Click on Publish.

   - Select SQUARE option for Shape and RED for Color.

   - Change the History field from 6 to 1.

   - Select EXCLUSIVE.

   - Set Strength to 1.

   - Set Size to 15.

2. Create a red square publisher:

   - Start eProsima Shapes Demo (this instance will be referred to as *Instance2*).

   - Click on Publish.

   - Select SQUARE option for Shape and RED for Color.

   - Change the History field from 6 to 1.

   - Select EXCLUSIVE.

   - Set Strength 2.

   - Set Size to 30.

A small red square on Instance1 and a big red square on Instance2 should be displayed.

Note: The Instance3 shown in the image below creates a square subscriber. Its creation will be explained later.
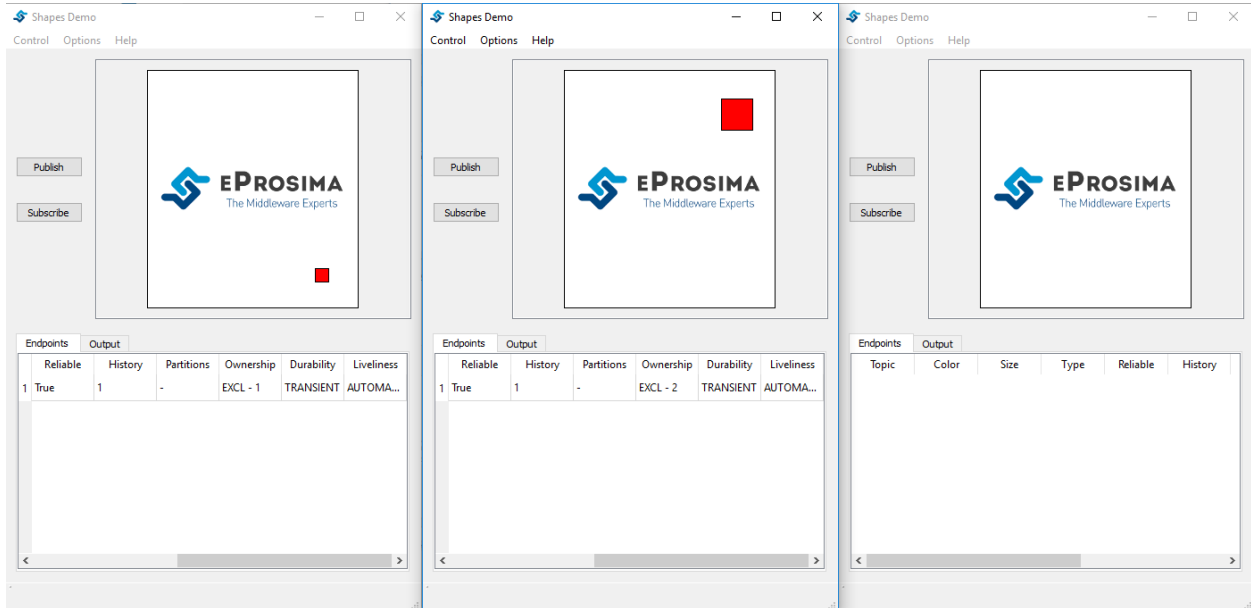
Then, create a subscriber.

3. Create a square subscriber:

   • Start eProsima Shapes Demo (this instance will be referred to as *Instance3*).

   • Click on Subscribe.

   • Select SQUARE option for Shape.

   • Select EXCLUSIVE.

A big square on Instance3 should be seen, since Instance2 has a higher strength than Instance1.

8.1.1 Response to a failure of the main publisher

To see how the roles of the publishers change, Instance2 will be stopped. Initially, Instance2 had higher strength and a big red square on Instance2 was observed. However, a small red square is displayed on Instance3 since Instance1 has higher strength now.

Repeating the creation of a publisher with higher strength (*2. Create a red square publisher*), a small red square replicating the big red square from Instance2 can be seen again.
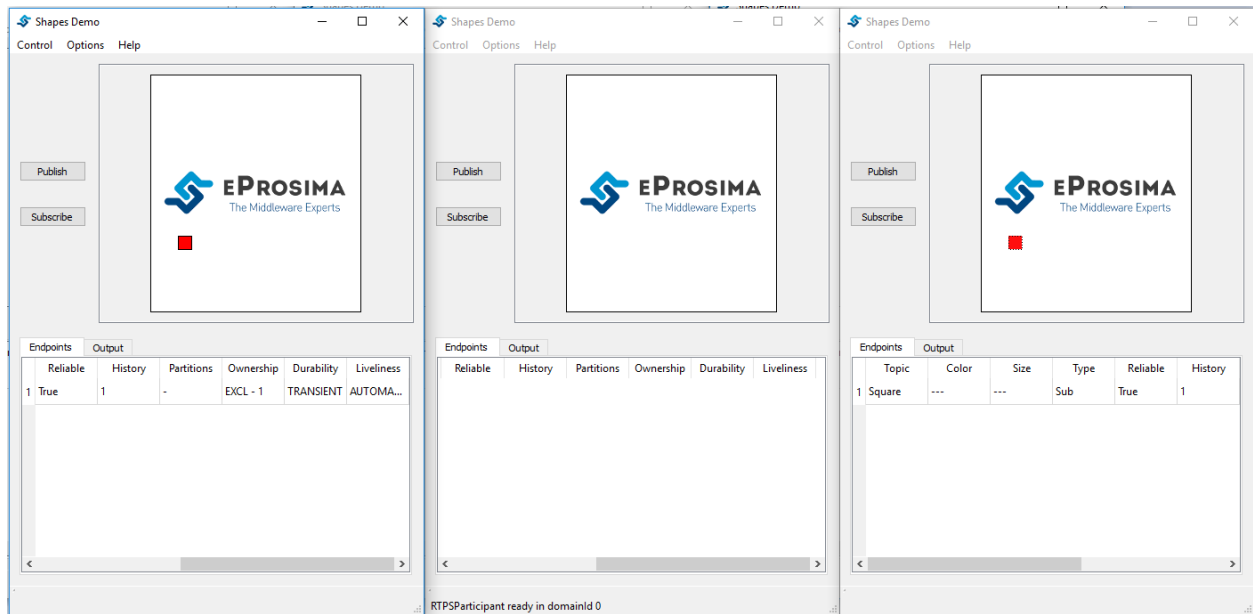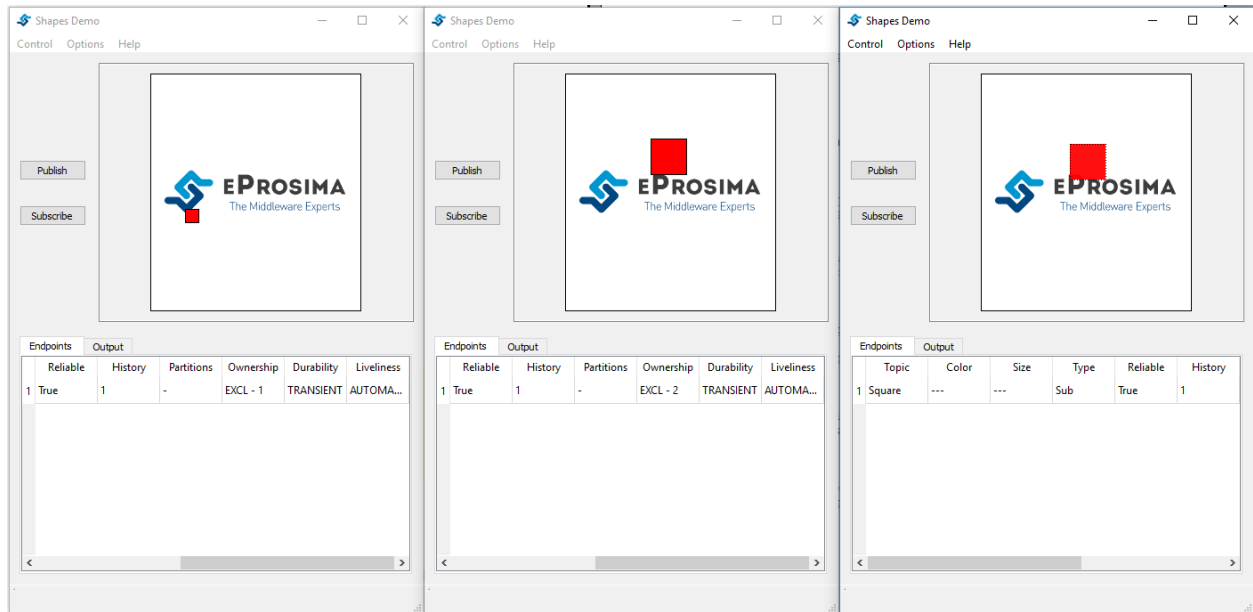
LIVELINESS

The Liveliness QoS can be used to ensure whether specific entities are alive or not. There are three values to specify the liveliness' kind: `AUTOMATIC`, `MANUAL_BY_PARTICIPANT` or `MANUAL_BY_TOPIC` liveliness. If any of the first two is selected, a value for the lease duration and announcement period can be set. However, if `MANUAL_BY_TOPIC` is selected, only the lease duration can be configured, as the announcement period is not used with this configuration. With the `AUTOMATIC` liveliness kind, the service takes the responsibility for renewing the timer associated to the lease duration, and as long as the remote participant keeps running and remains connected, all the entities within that participant will be considered alive. The other two kinds (`MANUAL_BY_PARTICIPANT` and `MANUAL_BY_TOPIC`) need a periodic assertion to consider the remote participants as alive. Please refer to Fast DDS LivelinessQosPolicy Documentation for more information on Liveliness QoS.

In this test, a publisher and subscriber using `AUTOMATIC` liveliness will be created, and a lease duration value higher than the write rate of the publisher will be set.

9.1 Step-by-step example implementation

First, launch two instances and create a publisher and a subscriber:

1. Create a red square publisher:

    - Start eProsima Shapes Demo. (We will refer to this instance as Instance1)

    - Click on Publish.

    - Select SQUARE option for Shape and RED for Color.

    - Select `AUTOMATIC` for liveliness kind.

    - Set Lease Duration to 150. (The default write rate is 75 ms)

2. Create a square subscriber:

    - Start eProsima Shapes Demo. (We will refer to this instance as Instance2)

    - Click on Subscribe.

    - Select SQUARE option for Shape.

    - Select `AUTOMATIC` for liveliness kind.

    - Set a value for the Lease Duration higher or equal to the one stated for the publisher. (If the value of subscriber lease duration is lower the entities do not match)

The *Output Tab* of Instance2 shows that the subscriber has recovered the liveliness once it matches with the publisher.

Then, kill the process corresponding to the publisher (Instance1). As a result, the subscriber reported that liveliness was lost, as the publisher did not terminate cleanly.
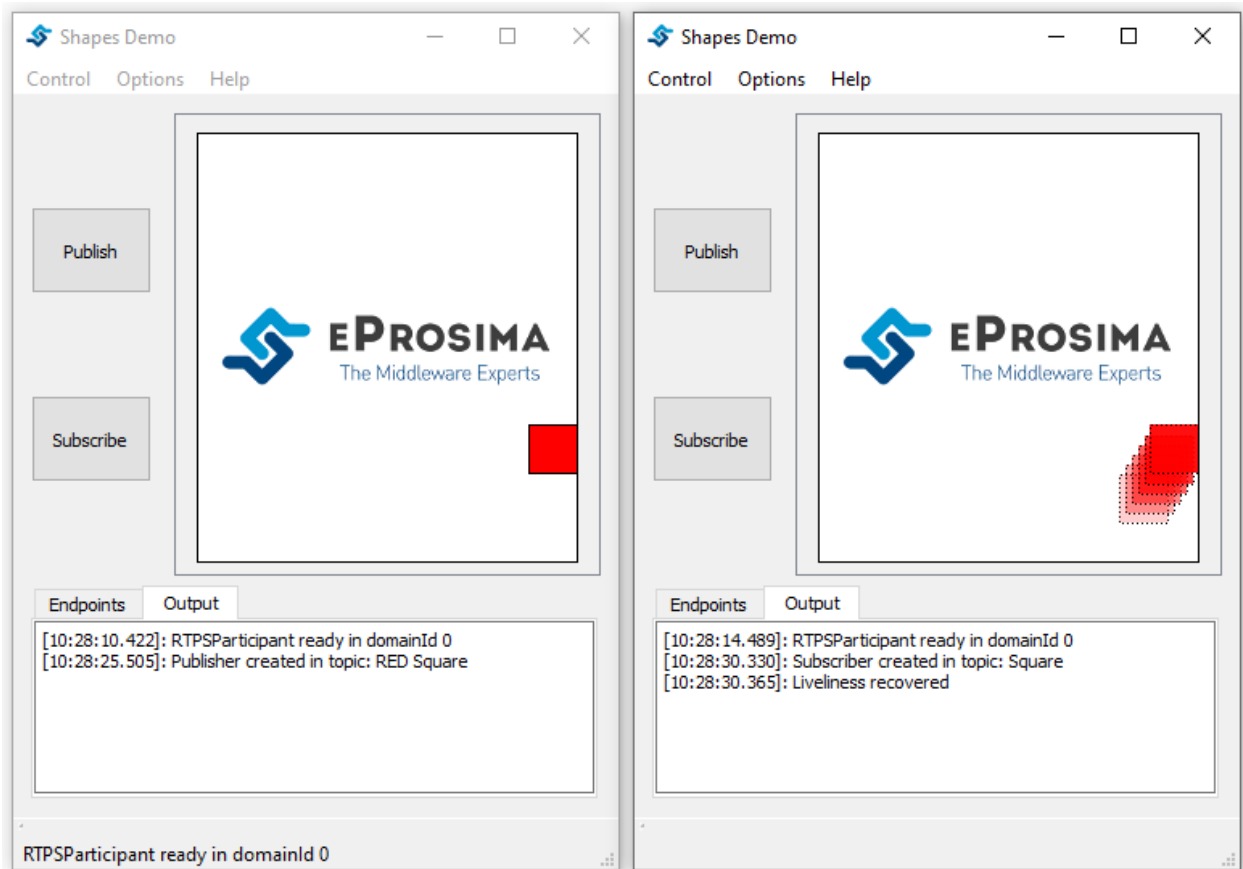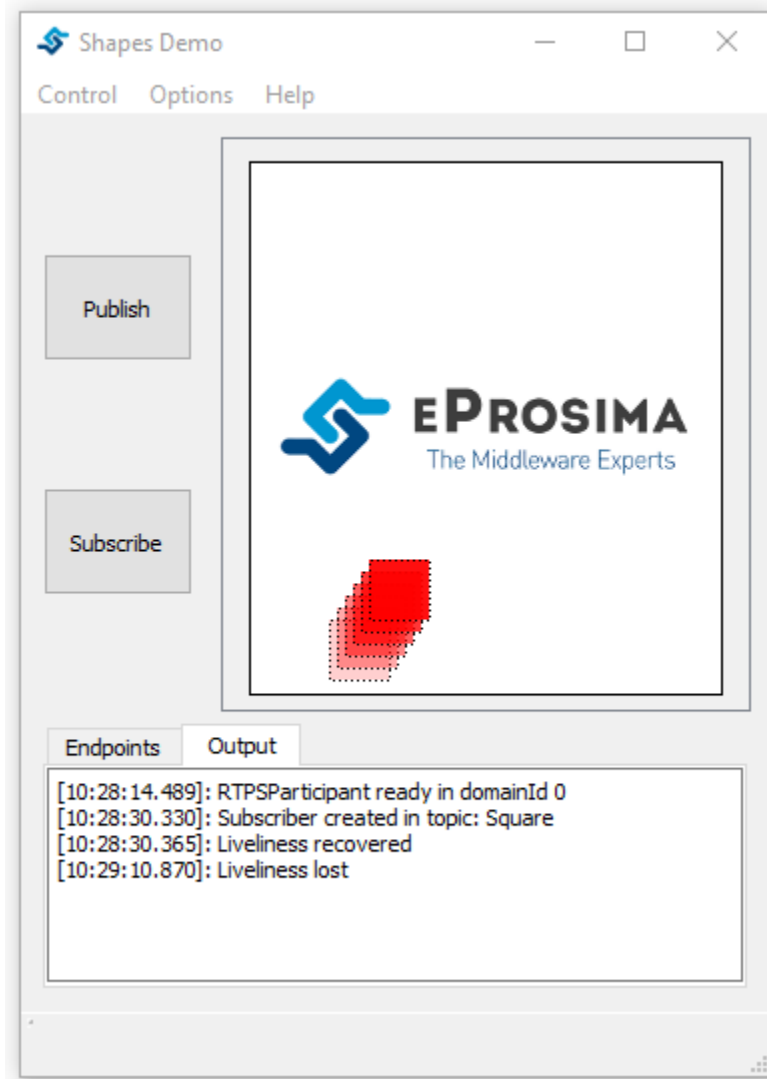
DEADLINE

The Deadline QoS raises an alarm when the frequency of new samples falls below a predefined threshold. This policy can be useful for those cases which need periodical updates. There exists a distinction between publisher and subscriber deadline period. On the publisher side, that period defines the maximum interval between writes, while on the subscriber's, it establishes the maximum interval in which the reader expects to receive a new sample. Please refer to Fast DDS LivelinessQosPolicy Documentation for more information on Liveliness QoS.

In this test, a publisher and a subscriber with a deadline period higher than the write rate of the publisher will be created. That shows the normal behavior of the system. Then the write rate will be increased to illustrate the effects of non-compliance with the deadline.

10.1 Step-by-step example implementation

First, launch two instances and create a publisher and a subscriber:

1. Create a red square publisher:

   - Start eProsima Shapes Demo (this instance will be referred to as *Instance1*).

   - Click on Publish.

   - Select SQUARE option for Shape.

   - Select RED for Color.

   - Set Deadline Duration to 100. (The default write rate is 75 ms)

2. Create a square subscriber:

   - Start eProsima Shapes Demo (this instance will be referred to as *Instance2*).

   - Click on Subscribe.

   - Select SQUARE option for Shape.

   - Set a value for the Deadline Duration higher or equal to the one stated for the publisher.

> **Warning:** If the value of the subscriber Deadline Duration is lower than the value stated for the publisher the entities will not match.

In the following figure, the normal behavior of the system can be seen, where the publisher is sending new samples and the subscriber is receiving them before the deadline expires.

Now, increase the write rate:

3. On Instance1:
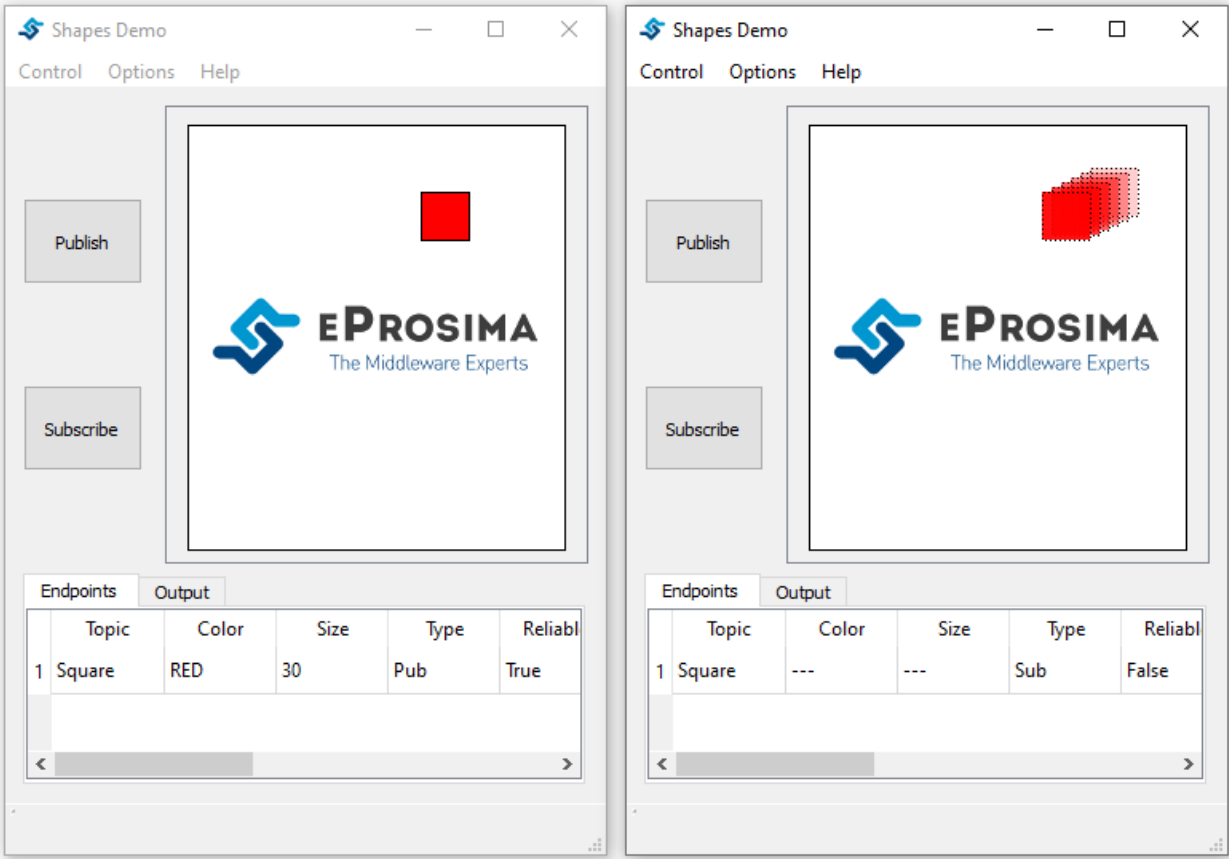
- Click on Options.

- Select Preferences.

- Set the update interval to 200.
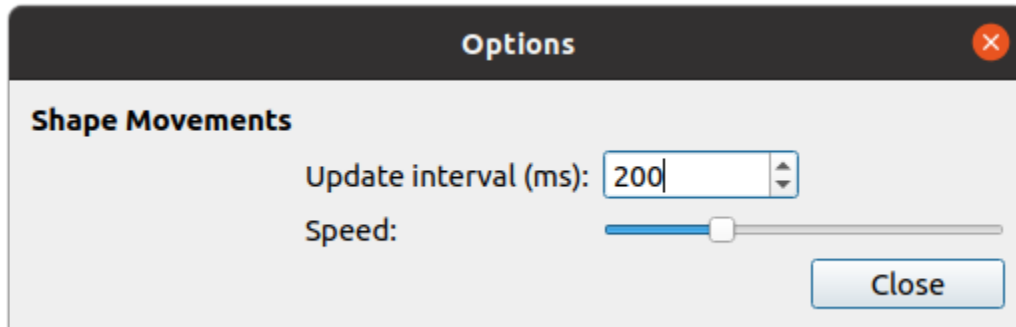
On the *Output Tab* can be observed that both publisher and subscriber are continuously missing the deadline, as the value established for the deadline period is lower than the publishing rate.

LIFESPAN

The Lifespan QoS establishes the maximum validity period of the samples saved on an entity's history. When the lifespan period elapses, the corresponding sample is automatically removed.
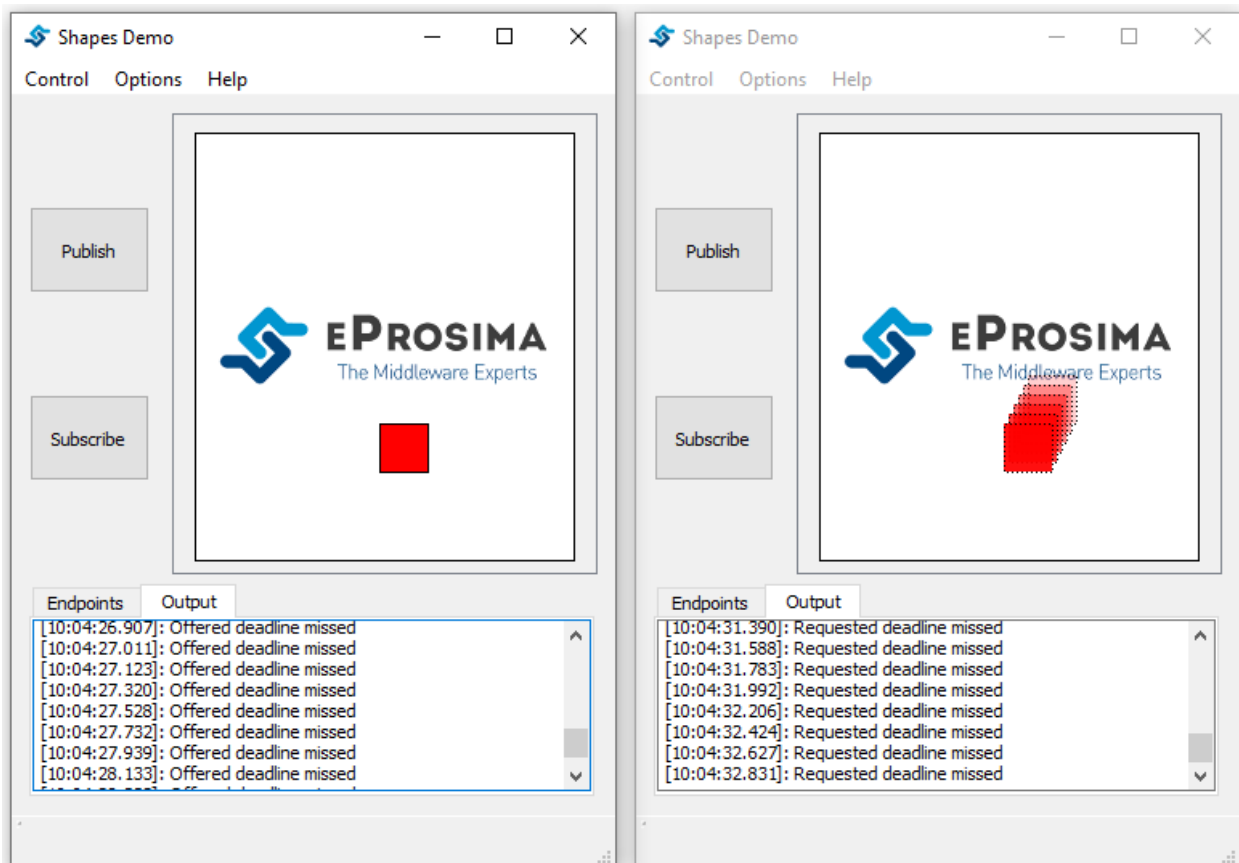
Unlike other QoS, such as *Deadline* or *Liveliness*, this test does not provide the means to inform the user that the sample is being removed, which makes this test more complicated to illustrate. For this reason, two publishers and two subscribers will be created, and making only one of them use Lifespan, the effect of using this QoS can be graphically seen.

11.1 Step-by-step example implementation

First, launch an instance and create the two publishers:

1. Create a red square publisher:

   - Start eProsima Shapes Demo (this instance will be referred to as *Instance1*).

   - Click on Publish.

   - Select SQUARE option for Shape and RED for Color.

   - Select `TRANSIENT_LOCAL` option for the Durability.

   - Make sure that RELIABLE checkbox is marked.

   - Set History to 100.

   - Leave the Lifespan Duration to infinite.

2. Create an orange triangle publisher on the previous Shapes Demo instance:

   - Click on Publish.

   - Select TRIANGLE option for Shape and ORANGE for Color.

   - Select `TRANSIENT_LOCAL` option for the Durability.

   - Make sure that RELIABLE checkbox is marked.

   - Set History to 100.

   - Set Lifespan Duration to 50.

After that, change the write rate to 1000:

3. On Instance1:

   - Click on Options.

   - Select Preferences.

- Set the update interval to 1000.
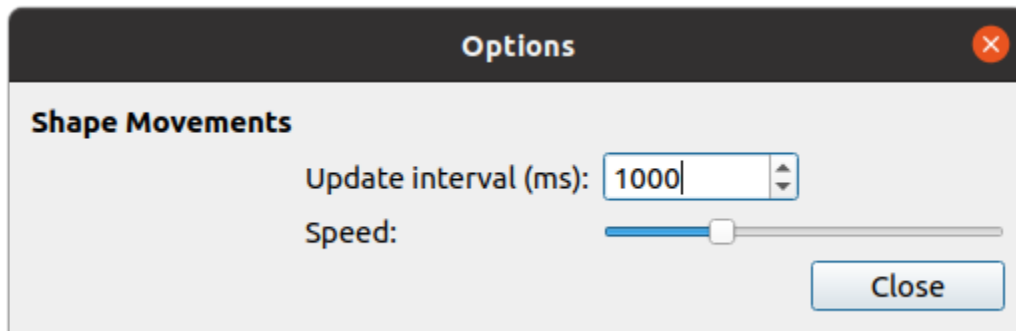
Now, create two subscribers:

4. Create a square subscriber:
   - Start eProsima Shapes Demo (this instance will be referred to as *Instance2*).
   - Click on Subscribe.
   - Select SQUARE option for Shape.
   - Select `TRANSIENT_LOCAL` option for the Durability.
   - Make sure that RELIABLE checkbox is marked.
   - Set History to 100.
   - Leave the Lifespan Duration to infinite.

5. Create a triangle subscriber:
   - Start eProsima Shapes Demo (this instance will be referred to as *Instance3*).
   - Click on Subscribe.
   - Select TRIANGLE option for Shape.
   - Select `TRANSIENT_LOCAL` option for the Durability.
   - Make sure that RELIABLE checkbox is marked.
   - Set History to 100.
   - Set Lifespan Duration to 50.

When a new subscriber matches with the publisher, due to the `TRANSIENT_LOCAL` durability, all the samples stored on the publisher history are sent automatically to the new subscriber.

Furthermore, on Instance2 and Instance3, the square subscriber history is filled rapidly, while the triangle subscriber is filled at the same speed as the orange triangle publisher sends new samples. This is because in the second case, samples in the orange triangle publisher history were removed by the QoS, and are no longer available to be sent to the subscriber, while the red square publisher history samples were kept.

TWELVE
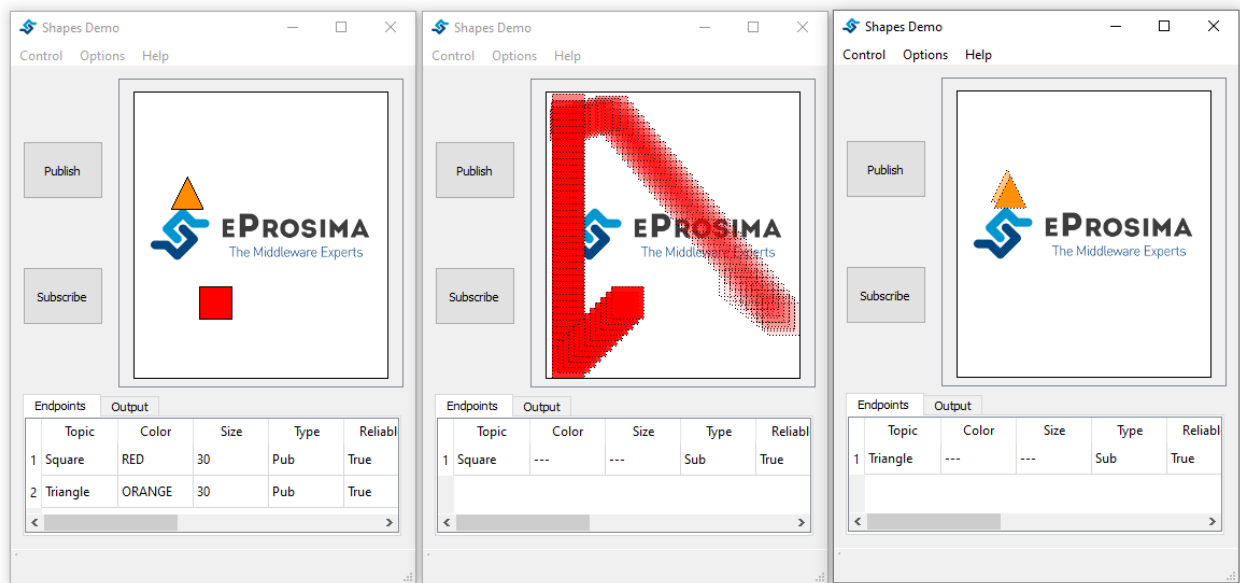
CONTENT BASED FILTER

In *Fast DDS*, the data available to the subscriber can be restricted to control network and CPU usage. The Content Based Filter can be checked when a new subscriber is deployed. This filter draws a shaded region in the instance windows. Only the samples that are covered by the shade will be available to the subscriber. This region can be resized and moved dynamically.

In this test, two Publishers and two subscriber will be created, one of the latter with Content Based.

12.1 Step-by-step example implementation

First, you have to launch two instances and create a Publisher in each of them:

1. Create a red square publisher:

   - Start eProsima Shapes Demo (this instance will be referred to as *Instance1*).

   - Click on Publish.

   - Select SQUARE option for Shape and RED for Color.

   - Change the History field from 6 to 1.

2. Create an orange circle publisher:

   - Start eProsima Shapes Demo (this instance will be referred to as *Instance2*).

   - Click on Publish.

   - Select CIRCLE option for Shape and ORANGE for Color.

   - Change the History field from 6 to 1.

Your windows should look similar to the following image.

Note: The Instance3 shown in the image below creates a circle subscriber. Its creation will be explained later.

Then, create two subscribers:

3. Create a circle subscriber:

   - Start eProsima Shapes Demo (this instance will be referred to as *Instance3*).

   - Click on Subscribe.

   - Select CIRCLE option for Shape.

   - Change the History field from 6 to 1.
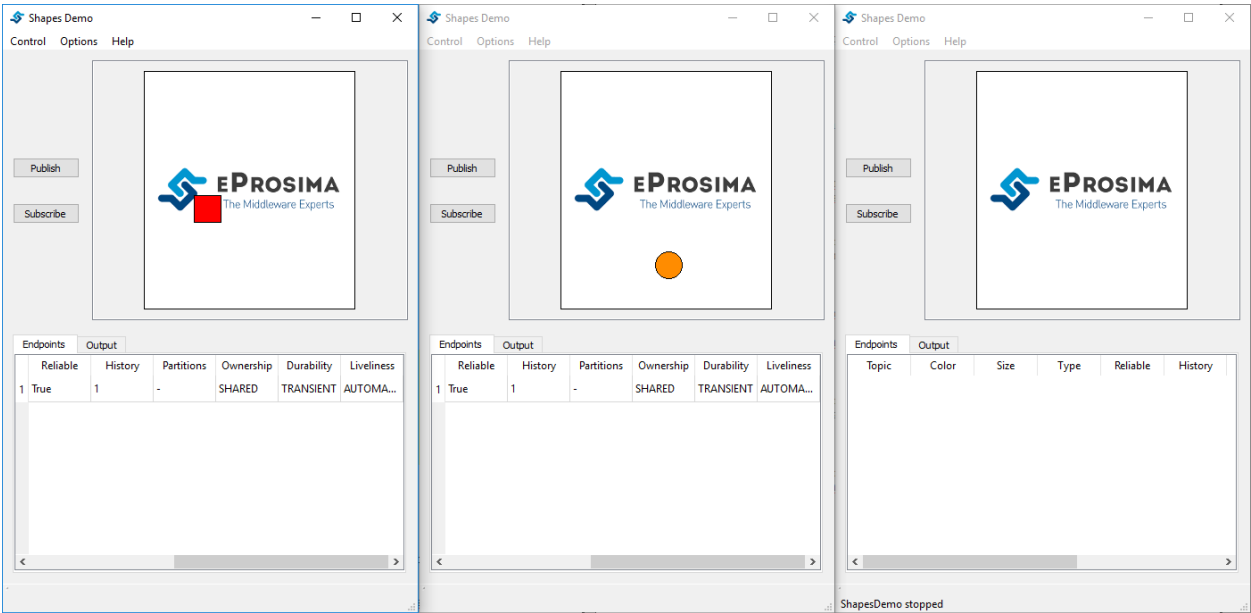
- Check Content Based.

4. Create a square subscriber:

  - Click on Subscribe in Instance3.

  - Select SQUARE option for Shape.

  - Change the History field from 6 to 1.

In the following figure, a shaded rectangle in Instance3 is shown. This is the filter for the samples of the Circle Shape. If the circle is out of the rectangle, it is not available for the subscriber.
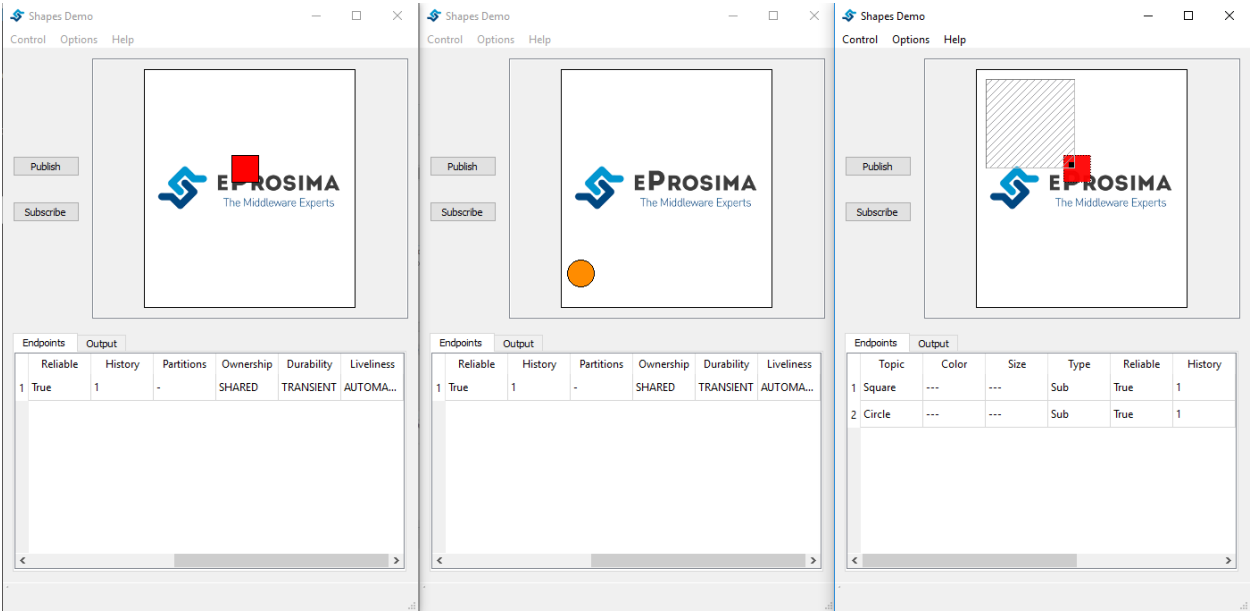
However, if the instance is in the rectangle, it is available for the subscriber..

The rectangle is configurable, i.e. it can be resized and moved dynamically. The following images show examples of
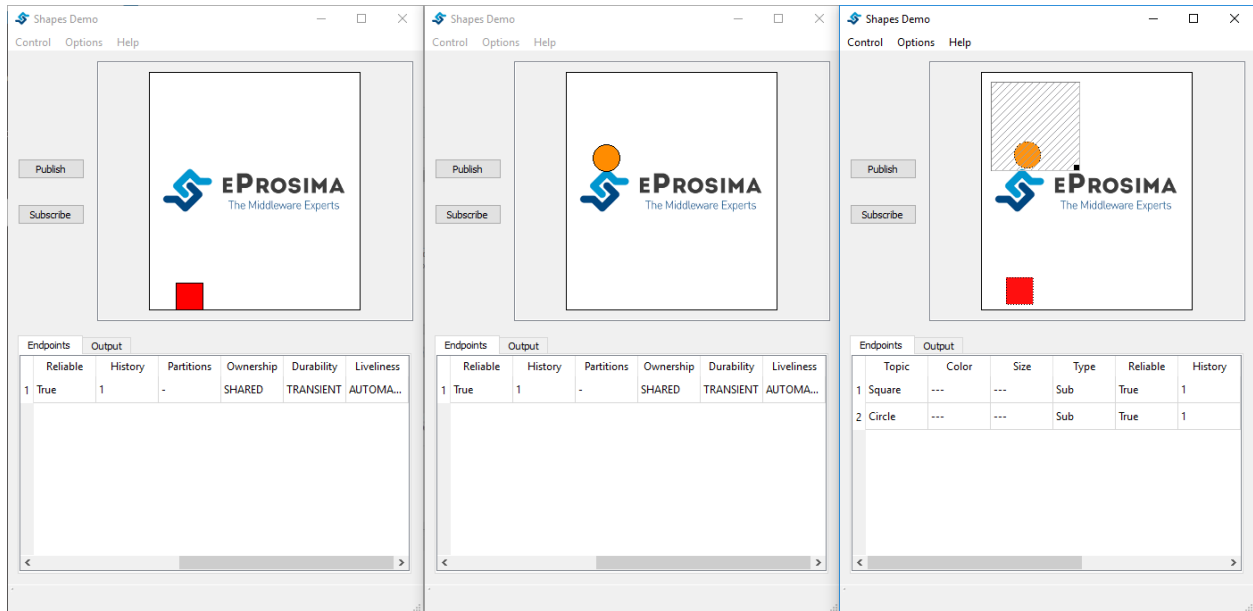
the content filter.
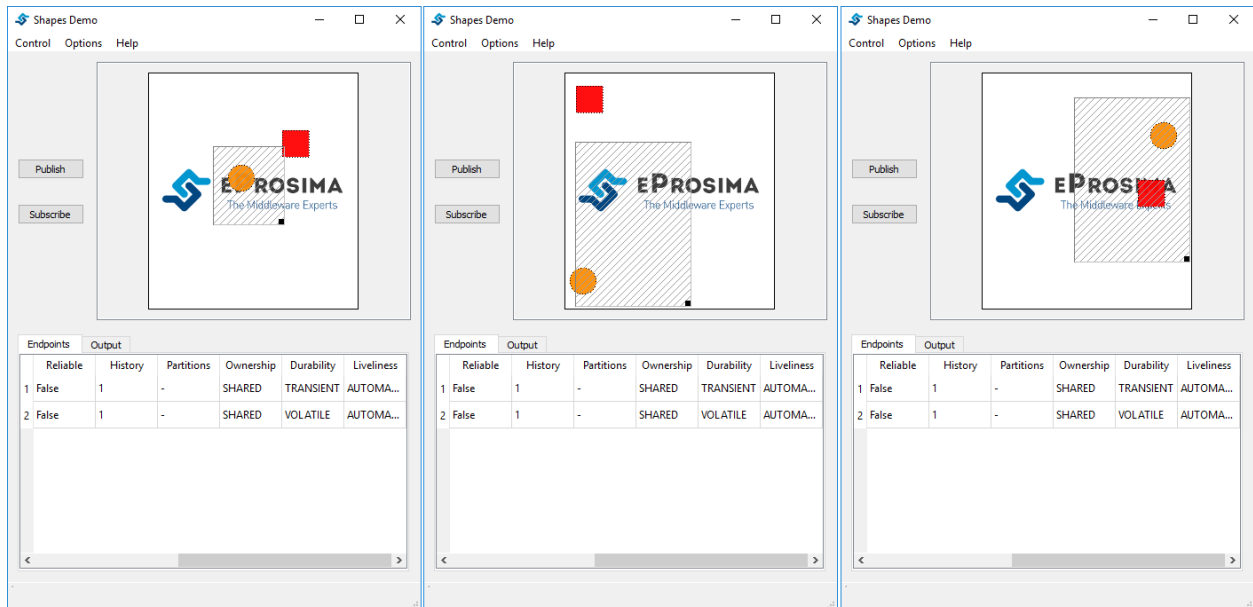
TIME BASED FILTER

Fast DDS allow the implementation of a time based filter for the subscribers. Thus, the number of data updates in the subscriber can be restricted specifying the minimum separation time (in milliseconds) between updates. Any data received during this interval will be discarded. Please refer to Fast DDS TimeBasedFilterQosPolicy Documentation for more information on Time Based Filter QoS.

In this test, two publishers and two subscribers, one with a time based filter of 2000ms, will be created. You will see that one subscriber updates its position continuously, but the other jumps every two seconds.

13.1 Step-by-step example implementation

First, launch two instances and create a Publisher in each of them:

1. Create a red square publisher:

    - Start eProsima Shapes Demo (this instance will be referred to as *Instance1*).

    - Click on Publish.

    - Select SQUARE option for Shape and RED for Color.

2. Create an orange circle publisher:

    - Start eProsima Shapes Demo (this instance will be referred to as *Instance2*).

    - Click on Publish.

    - Select CIRCLE option for Shape and ORANGE for Color.

    - Change the History field from 6 to 1.

Then, create two subscriber:

3. Create a circle subscriber:

    - Start eProsima Shapes Demo (this instance will be referred to as *Instance3*).

    - Click on Subscribe.

    - Select CIRCLE option for Shape.

    - Check Time Based an set 2000ms.

4. Create a square subscriber:

    - Click on Subscribe in Instance3.

    - Select SQUARE option for Shape.

The eProsima Shapes Demo output should look similar to the following image.
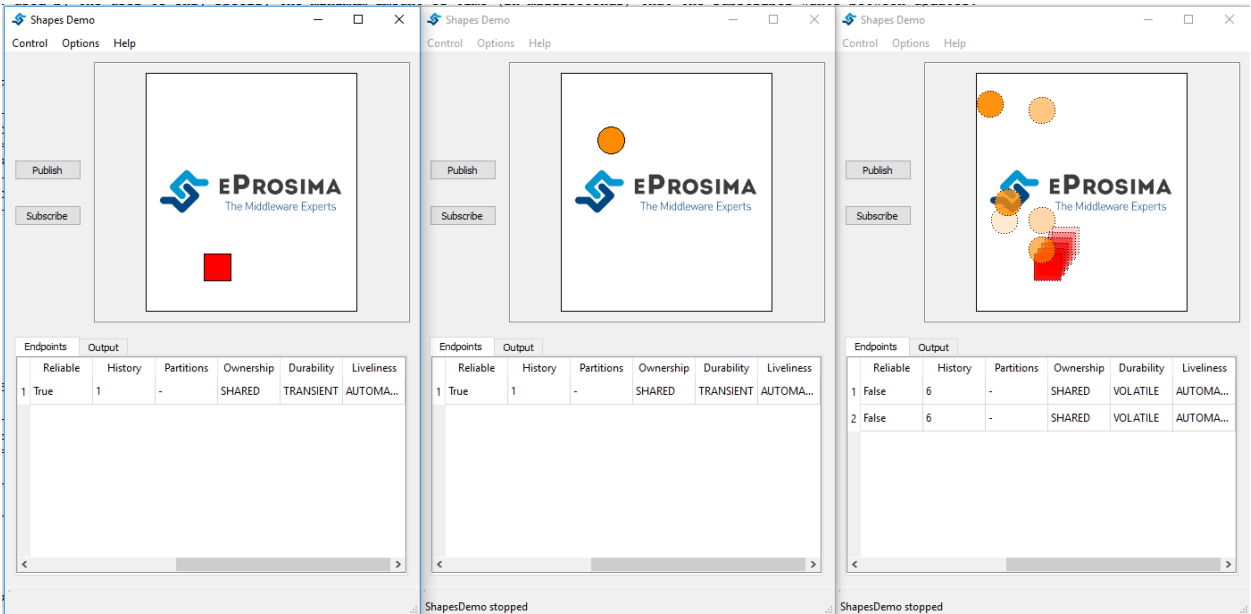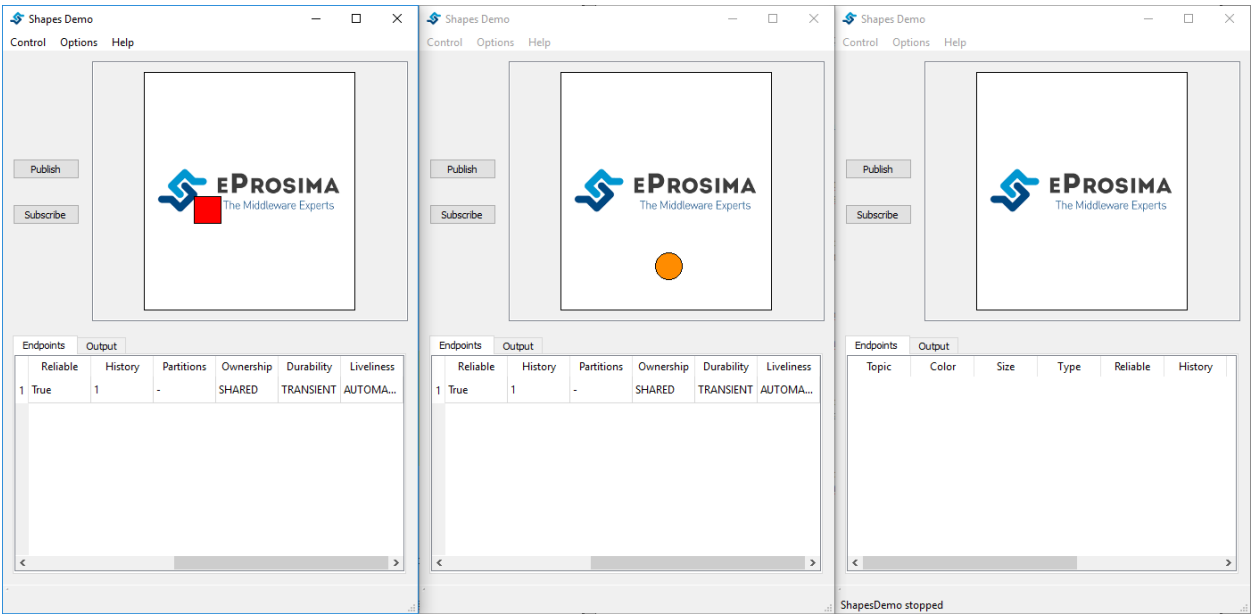
TCP TRANSPORT

Fast DDS transport layer could be adapted to specific network requirements. UDP transport is optimal for local network applications, but in order to interact on a Wide Area Network (WAN), where one or several NAT mappings may be enforced, a TCP transport is more suitable. Please refer to Fast DDS Transports Documentation for more information.

This section describes how Shapes Demo should be set up in order to fit some network specific layouts.

14.1 LAN configuration

TCP over LAN can be tested in a scenario where several machines share the same LAN network; nevertheless, UDP performs better and is the advisable choice in practical situations. For this configuration example, one of the machines set up eProsima Shapes Demo as a server and all the others as clients. Assume that the server LAN IP address is `192.168.1.75`, then all clients instances of eProsima Shapes Demo deployed on other machines must specify this IP Server address. In this case, the 5100 port is selected as TCP port, but any other available TCP port is valid.

> **Warning:** The server firewall must allow inbound traffic on the selected port.

Server TCP side:

Client TCP side:
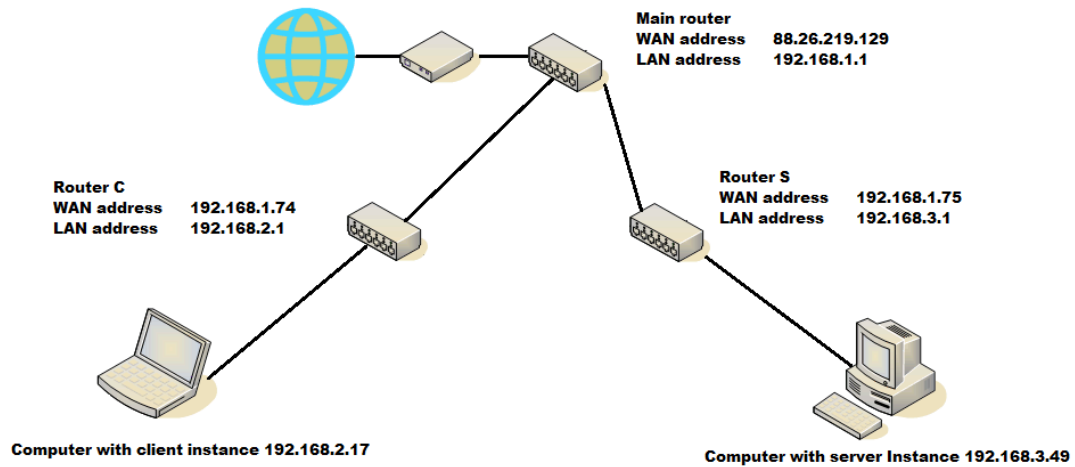
14.2 WAN configuration

This is the typical TCP scenario in which the server machine does not share network with its clients but its reachable through a WAN IP address. This may happen if the server and clients are in a different LANs of the same WAN. To test this scenario we used the network architecture shown in the figure below. It contains the following elements:

- A Main Router which simulates the WAN network. In this network Router C has address 192.168.1.74 and Router S has address 192.168.1.75.

- A client LAN network managed by Router C.

- A server LAN network managed by Router S. The Router S NAT settings relay any inbound TCP traffic to port 5100 towards Sever machine. The TCP port 5100 was arbitrarily chosen, any available port will do.

- An eProsima Shapes Demo client running in the machine with IPv4 address 192.168.2.17.

- An eProsima Shapes Demo server running in the machine with IPv4 address 192.168.3.49. The Server machine firewall settings allow inbound TCP traffic to port 5100.

The previous configuration (see *LAN configuration*) does not work in this network scenario because server and client are behind a NAT.

The following image shows server and client settings:

- On the client side, the *server IP* field contains the server's router IP address, i.e. the Router S IP address (192.168.1.75). The client's router can understand this address and properly lead the outbound traffic.

Main router
WAN address   88.26.219.129
LAN address   192.168.1.1

Router C
WAN address   192.168.1.74
LAN address   192.168.2.1

Router S
WAN address   192.168.1.75
LAN address   192.168.3.1

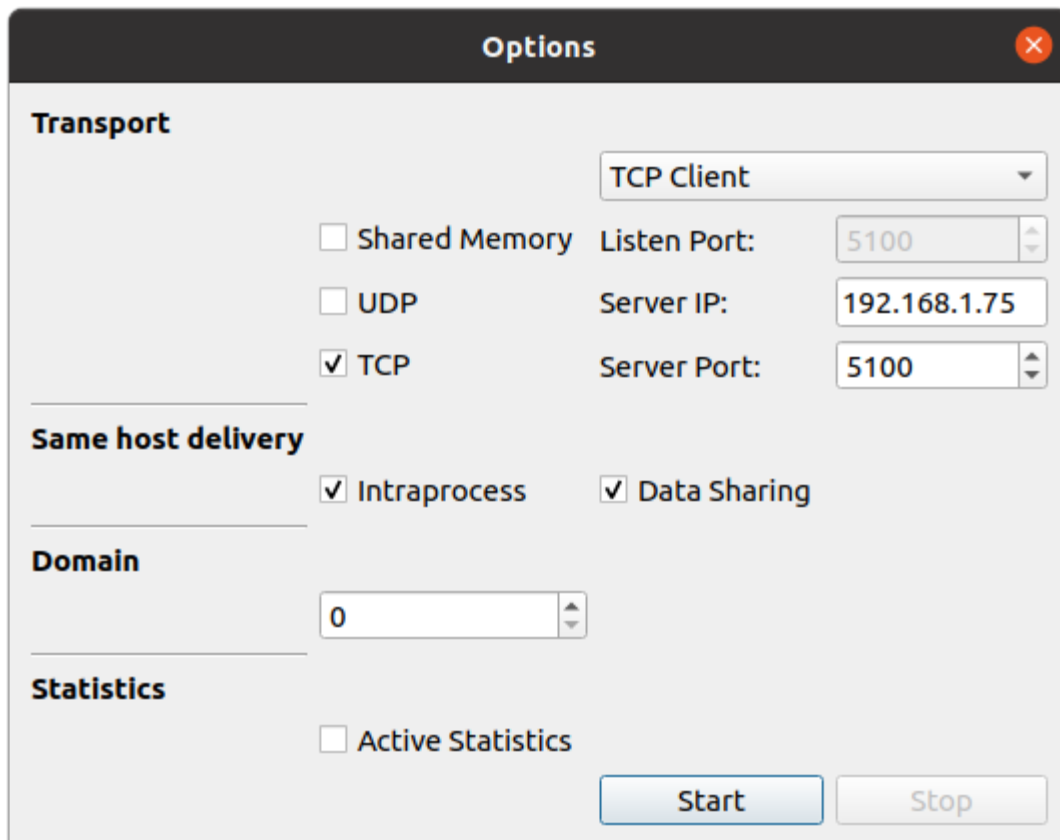Computer with client instance 192.168.2.17

Computer with server Instance 192.168.3.49

- On the server side, the *WAN IP* field contains the server's router IP address, i.e. the Router S IP address (192.168.1.75) since Router S NAT settings relay inbound traffic to server's TCP port towards Server computer.

Client TCP side:

Server TCP side:

TROUBLESHOOTING

This document compiles the known compatibility issues with other Shapes Demo implementations and the possible solutions.

15.1 Two instances in the same machine

When using eProsima Shapes Demo and RTI Shapes Demo in the same machine, the file

```
<Shapes Demo installation directory>/resource/xml/RTI_SHAPES_DEMO_QOS_PROFILES.xml
```

needs to be modified to force RTI to use the UDPv4 transport protocol only. This can be done writing the following lines inside the `<qos_profile>` element of the RTI_Shapes_Lib::Shapes_Default_Profile.

```xml
<qos_library name="RTI_Shapes_Lib">
    <qos_profile name="Shapes_Default_Profile" is_default_qos="true">
        <!-- The following lines are added to change the participant's transport protocol.
↪ -->
        <participant_qos>
            <transport_builtin><mask>UDPv4</mask></transport_builtin>
            <discovery_config>
                <builtin_discovery_plugins>SDP</builtin_discovery_plugins>
                <participant_liveliness_lease_duration>
                    <sec>30</sec>
                    <nanosec>0</nanosec>
                </participant_liveliness_lease_duration>
            </discovery_config>
        </participant_qos>
        <!-- ... -->
    </qos_profile>
    <!-- ... -->
</qos_library>
```

This setting avoids the use of Shared Memory Transport.

SIXTEEN

INTEROPERABILITY

You can see an example of the interoperability between eProsima Shapes Demo and other RTPS compliant implementations in the following video: https://www.youtube.com/watch?v=e9_oAJDh-tY

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
```

17.1 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.

- *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.).

  You can read more about this application on the Fast DDS Monitor 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.

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.

17.1.1 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
```

17.1.2 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.

17.1.3 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.

VERSION 2.7.2

This patch release adds the following **improvements**:

- Support for Fast DDS v2.7.2

PREVIOUS VERSIONS

19.1 Version 2.7.1

This patch release adds the following **improvements**:

- Support for Fast DDS v2.7.1

19.2 Version 2.7.0

This patch release adds the following **improvements**:

- Support for Fast DDS v2.7.0
- The ShapesDemo's `DomainParticipant` acts as `TypeLookup` server.

19.3 Version 2.6.1

This patch release adds the following **improvements**:

- Support for Fast DDS v2.6.1
- Option to use ROS 2 compatible topics to interact with ROS 2 application

19.4 Version 2.6.0

The minor release adds the following **improvements** to support Fast DDS v2.6.0:

- Use Fast DDS content filter API to perform content filtering
- Update type support with Fast DDS-Gen v2.1.2
- Avoid sending type object and type information

19.5 Version 2.5.1

This release includes the following **improvements**:

- Updated to support Fast DDS v2.5.1

19.6 Version 2.5.0

This release includes the following **improvements**:

- Updated to support Fast DDS v2.5.0

19.7 Version 2.4.1

This release includes the following **improvements**:

- Updated to support Fast DDS v2.4.1

19.8 Version 2.4.0

This release includes the following **improvements**:

- Updated to support Fast DDS v2.4.0
- Fixed link to troubleshooting documentation

19.9 Version 2.3.4

Change log:

- Updated to support Fast DDS v2.3.4
- Updated to use Fast DDS DDS API
- SHM can be selected as transport
- Intra-process and datasharing can be enabled/disabled for Shapes Demo DomainParticipants
- Fast DDS Statistics module can be enabled for Shapes Demo DomainParticipants
- Separate participant configuration from other options
- Change License to GNU GPLv3, to be compliant with Qt's license

19.10 Version 2.3.2

Change log:

- Updated to support Fast DDS 2.3.2.

19.11 Version 2.3.1

Change log:

- Updated to support Fast DDS 2.3.1.

19.12 Version 2.3.0

Change log:

- Updated to support Fast DDS 2.3.0.

19.13 Version 2.2.0

Change log:

- Updated to support Fast DDS 2.2.0.

19.14 Version 2.0.1

Change log:

- Updated to support Fast DDS 2.0.1

19.15 Version 2.0.0

Change log:

- Updated to support Fast DDS 2.0.0.

19.16 Version 1.9.0

Change log:

- Updated to support Fast RTPS 1.9.0.
- Updated to illustrate LivelinessQos.

19.17 Version 1.8.1

Change log:

- Updated to support Fast RTPS 1.8.1.

19.18 Version 1.7.1

Change log:

- Support Fast RTPS 1.7.1.
- Bug fixing.

19.19 Version 1.7.0

Change log:

- Updated to support Fast RTPS.
- Added support to configure TCP transports in the GUI.