Creating custom transport channel

To create your own channel using the Kaa SDK you have to implement the KaaDataChannel interface and register the implementation using the Kaa Channel Manager.

Implement the KaaDataChannel interface (Java client)

The implementation of the KaaDataChannel interface will contain methods that you will use to transfer data between endpoints and Servers using the protocol of your choice.

For a definition of the KaaDataChannel interface please refer to the javadoc.

You can find examples of the interface implementation in the following java classes for the default transport channels:

1. DefaultOperationsChannel - implementation of the Operation HTTP Long poll channel;
2. DefaultOperationHttpChannel - implementation of the Operation HTTP channel;
3. DefaultBootstrapChannel - implementation of the Bootstrap HTTP channel;
4. DefaultOperationTcpChannel - implementation of the Operation Kaatcp channel.

Register your own channel using the Kaa Channel Manager

When the implementation of your channel is ready you should add the channel to Channel Manager by invoking the addChannel() method as follows:

Now Kaa SDK knows about your channel and will use it to send service data according to the channel settings.

Notice that in java sdk, after adding new data channel multiplexer and demultiplexer will be setted regarding to channel server type.

To send a request to the server and get a response please follow the steps described blow.

Step 1 - Get an instance of KaaDataMultiplexer

To prepare a request to the server, you have to use a data multiplexer that combines and serializes requests from different Kaa services.

Kaa provides two data multiplexers. One should be used for communication with the Operations server and the other for communication with the Bootstrap server.

To get an instance of KaaDataMultiplexer for communication with the Operation server, use the getOperationMultiplexer() method:

To get an instance of KaaDataMultiplexer for communication with the Bootstrap server, use the getBootstrapMultiplexer() method:

Step 2 - Prepare a request collecting information from the services

In order to create a request to be sent to the server you have to collect data from Kaa services. Collecting data is performed using the KaaDataMultiplexer interface obtained in the previous step.

The KaaDataMultiplexer interface has only one method: compileRequest ()

where **types** is a map of Kaa services and their data exchange directions that are supported by your channel.

For example, if you have implemented a channel with the following settings:

- The channel is able to send events, but cannot receive events
- The channel is able to receive notifications, but cannot send notification requests
- The channel supports the Configuration service in both directions.

then your **types** map will look as follows:

The method scans the services and collects data from those that have prepared data to be sent to the server. The method uses the **types** map to filter requests from the services. (For example, if for a Transport Type the "DOWN" direction is indicated in the type map, the request data from the respective service will be filtered out and will not be sent to the server.)

The data collected from the services is combined into the SyncRequest and serialized. As a result, the method returns a byte array with serialized data.

Step 3 - Send the prepared request to the server and receive a response

Insert the data returned by compileRequest into your transfer protocol and send it to the server. This step is performed using the methods that are in the implemented KaaDataChannel interface.

The response is received as a byte array, and it contains serialized responses for all the services from which requests were sent.

Step 4 - Get an instance of KaaDataDemultiplexer

To de-serialize the received response and provide a response to each service, you have to use a data demultiplexer.

Kaa provides two data demultiplexers. One should be used for communication with the Operations server and the other for communication with the Bootstrap server.

To get an instance of KaaDataDemultiplexer for communication with the Operation server, use the getOperationDemultiplexer() method:

To get an instance of KaaDataDemultiplexer for communication with the Bootstrap server, use the getBootstrapDemultiplexer() method:

Step 5 - Push a response to the Kaa SDK

The data demultiplexer contains only one method

void processResponse(byte [] response)

The method deserializes the response and decodes the raw data into SyncResponse which consists of subresponses for all services. Then the subresponses are delivered to each service for subsequent processing.