Documentation of the Assistant module "Core Service"

FKE

November 6, 2018

Contents

1 Description

Core Service is a micro service within the Assistant Service Fabric Application. Its function is to accept incoming messages from the Blackboard which are used for processing within the Assistant.

Model Updates These are messages sent by the KPUs about changed states of their internal data models. For example, a KPU could be used to monitor a temperature sensor. It might store the latest received temperature value and store it within its internal data model. Upon change of the value on the sensor, the internal state of the KPU would also change. Caused by that, the KPU could send out the Model Update which would then be routet to the Core Service by the Blackboard.

KPU registration KPUs need to be "known" within the Assistant in order for it to handle the data from the KPU and its control. For this purpose, KPUs send out registration messages upon their installation. That message gets send to the Blackboard which then routes it to the Core Service. Following that, the message causes (among other things) the generation of a dataset within the database about the "new" KPU. Additionally the KPU registration contains a hierarchically structured set of permissions that should be registered for that KPU. Those permissions can then be granted to specific user groups within the system. A user group with such a permission can then in effect request to perform actions on that KPU that are associated with that permission

KPU packages Since the client for the Assistant is designed in such a way that almost all the Business Logic remains within the Assistant, it follows that the client also doesn't statically include a model of the known KPUs. Instead, the client can request display and control of a KPU at a certain point in time. To enable the client to display / control a KPU, a prepared package is sent out by the KPU. The package takes the route of KPU - Blackboard - Core Service - External Communication - Client. It consists of all the data necessary for the client to display a view specifically designed for that KPU, including its data model in order for the client to be able to correctly handle Model Updates for that KPU and - where applicable - update the display.

These three types of messages make up the information transmission from blackboard to the Assistant / Client. Vice versa, from Client / Assistant to the Blackboard and the KPUs there exits another set of message types

Package Request The Package Request is the initial request for a KPU package. Being sent out by a specific client it contains the id of the KPU as meta information.

Request Execute A Request Execute message contains a query for a specific action to be executed on a KU. For example, one might request to start a certain workflow on a certain KPU.

2 Installation

Core Service is installed by deploying it within the Service Fabric Cluster Application "Assistant".

3 Configuration

To configure the service's logging, one can alter the Nlog.xml to switch to a different target / layout.[1] The service also contains an App.config file which stores connection information to the ActiveMq server used as communication technology for the Blackboard. For that purpose, one can configure two routes for the blackboard, one for messages from it and one for messages sent towards the Blackboard for routing it to some other component.

```
<Configuration>
    <Connection>
      <Endpoint>activemq:tcp://localhost:61616</Endpoint>
      <User>admin</User>
      <Password>admin</Password>
  </Connection>
  <Routes>
      <FromBlackboard>queue://CoreQ</FromBlackboard>
      <ToBlackboard>queue://BBQ</ToBlackboard>
  </Routes>
</Configuration>
```

Listing 1: Core Service App.config example

Listing 1 shows an example for the App.config that uses a locally installed ActiveMq server with a non-secure set of login credentials and two defined queues for messaging.

4 Dependencies

Also, Core Service requires the following Nuget packages not available from the official Nuget server:

[1] https://github.com/nlog/NLog/wiki/Configuration-file

- BreanosConnectors.ActiveMQConnector
- BreanosConnectors.Kpu.Communication.Common
- BreanosConnectors.SerializationHelper or BreanosConnectors.SerializationHelper.Standard

5 Requirements

Core Service requires

- an accessible External Communication service to send KPU packages to.
- an accessible Presenter Service to send model updates to.
- a running instance of an ActiveMq server for communication with the Blackboard[2].

[2] http://activemq.apache.org/download.html