## JCore - Technical Background

## Task:

The following document shall explain the major structure of the JCore and its data model within the database.

## **Realization:**

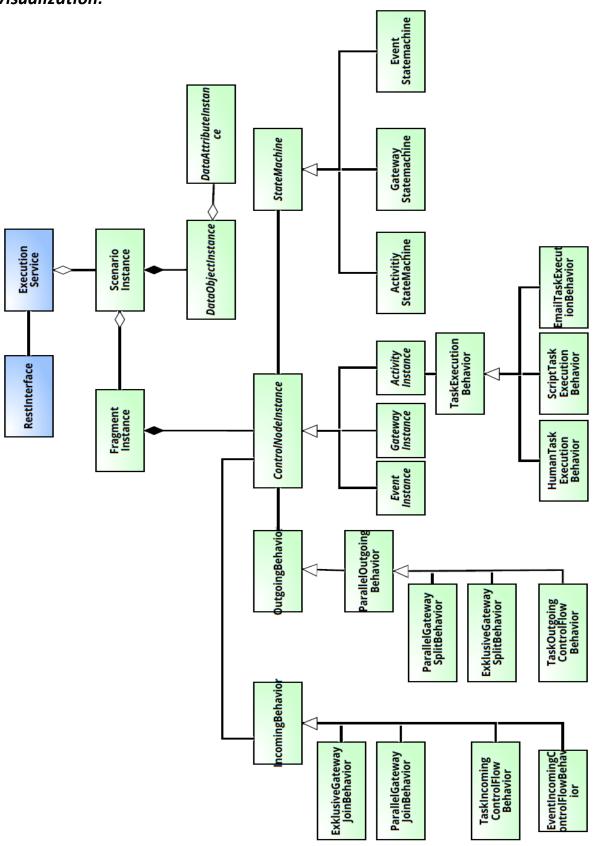
The *ExecutionService* is the class that handles the complete *JCore*. The *ExecutionService* manages all *ScenarioInstances* and provides methods to interact with them, for example executing an activity. The class *ScenarioInstance* represents an instance of a PCM scenario. It saves all its *FragmentInstances, ControlNodeInstances* and *DataObjectInstances*. If a *ScenarioInstance* gets initialized, it initializes all its *FragmentInstances* and *DataObjectInstances*. *DataObjectInstance* represents an instance of a PCM data object. It has a state and a list of all its *DataAttributeInstances*, instances of data attributes, which save the value and the type of the attribute.

The FragmentInstances are responsible for initializing all ControlNodeInstances belonging to the corresponding fragment. ControlNodeInstances might appear as ActivityInstances or GatewayInstance, as shown in the PCM model. All ControlNodeInstances have a StateMachine, an IncomingBehavior and an OutgoingBehavior. The StateMachine controls the state of the ControlNodeInstance and handles the state transitions. The IncomingBehavior controls the incoming control flow. This is important for gateways, because this behavior decides if an AND-gateway can be enabled. The OutgoingBehavior controls the outgoing control flow, including the data object output states for an activity, which are set here.

An *ActivityInstance* also has a *TaskExecutionBehavior* managing futher consequences of executing the activity. This may be the *HumanTaskExecutionBehavior* that sets the data attribute states or a service task execution behavior like the *EmailTaskExecutionBehavior* that automatically sends an email.

JEngine – JCore 1

## Visualization:



JEngine – JCore 2