Blueriq Case Engine

The Case Engine manages all operations on cases, case data and tasks. It communicates with the Blueriq Runtime through asynchronous messages and a REST API.

The Case Engine is essential to use the event driven DCM architecture and depends on multiple other components to function properly.

This section explains how to install the Case Engine, how to configure it and which other components are needed to be installed.

Installation

The Case Engine currently is a Runtime with specific configuration. It is installed by deploying the Runtime WAR to a supported application server, see Installing Runtime. Alternatively you can use the Blueriq DCM Development Installation which includes the case engine with configuration.

Dependencies

The Case Engine depends on the following components:

- Audit component
- DCM Lists Components
- Timeline SQL Store component
- Trace SQL Store component
- DCM queueing
- Document database MongoDB
- Customerdata service
- Process SQL store component
- Case Engine Scheduler
- Case Engine Scheduler Quartz

The `bootstrap.properties` file needs to have the following content to enable these profiles:

bootstrap.propeties

```
spring.profiles.active=native,\
    case-engine,\
    customerdata-client,\
    dcm-lists-publisher,\
    externaldatasources,\
    process-sql-store,\
    timeline-event-publisher-amqp,\
    trace-event-publisher-amqp,\
    audit
```

Configuration

Specific for the case engine there are multiple properties that need to be set.

These properties are divided into three files:

- `application.properties`: common properties that concern security, logging and the connection to the customer data
- `application-case-engine.properties`: properties for several queues, MongoDB and Quartz
- `application-externaldatasources.properties`: properties that enable the Case Engine to communicate with the Process SQL Store

Create the files in the additional config location of the case engine and copy and paste the corresponding properties into each file.

When using the Blueriq DCM Development Installation this configuration is included

application.properties

```
### Users ###
blueriq.security.auth-providers.local01.type=in-memory
blueriq.security.auth-providers.local01.users.location=users.properties
blueriq.security.auth-providers-chain=local01

### Customerdata ###
blueriq.customerdata-client.url=http://localhost:30002/customerdata/api/v1/
blueriq.customerdata-client.username=blueriq
blueriq.customerdata-client.password=welcome

### Exports ###
blueriq.exports.description=Exports
blueriq.exports.prefix=export
blueriq.exports.folder=exports
blueriq.exports.enabled=true

### Security settings ###
blueriq.security.csrf-protection.enabled=false

### Blueriq logging ###
#logging.level.com.aquima=DEBUG
#logging.level.com.blueriq=DEBUG
logging.file.name=logs/case-engine.log

### Default Queue configuration, can be ommitted when overruled at specific configuration in application-case-
engine.properties ###
blueriq.default.rabbitmq.host=localhost
blueriq.default.rabbitmq.port=30010
blueriq.default.rabbitmq.username=guest
blueriq.default.rabbitmq.password=guest
blueriq.default.rabbitmq.virtualHost=/
blueriq.default.rabbitmq.ssl.enabled=false
```

application-case-engine.properties

```
blueriq.case.engine.concurrency.concurrent-consumers=1
blueriq.case.engine.concurrency.max-concurrent-consumers=1

blueriq.case.engine.data.mongodb.host=localhost
blueriq.case.engine.data.mongodb.port=30012
blueriq.case.engine.data.mongodb.database=caseEngine

blueriq.locking.mongodb.host=localhost
blueriq.locking.mongodb.port=30012
blueriq.locking.mongodb.database=locks

spring.quartz.job-store-type=memory

spring.quartz.properties.org.quartz.threadPool.class=org.quartz.simpl.SimpleThreadPool
spring.quartz.properties.org.quartz.threadPool.threadCount=2
```

For more information on concurrent consumers see Configuring RabbitMQ.

application-external-datasources.properties

```
#### Datasources  PostgreSQL ###
blueriq.datasource.process-sql-store.url=jdbc:postgresql://bq-postgres:5432/blueriq
blueriq.datasource.process-sql-store.username=blueriq
blueriq.datasource.process-sql-store.password=welcome
blueriq.datasource.process-sql-store.driverClassName=org.postgresql.Driver
hibernate.process-sql-store.hbm2ddl.auto=validate
```

hbm2ddl.auto

When using a production database, please use 'validate' instead of 'update' in combination with the supplied database scripts.

application-dcm-lists-publisher.properties

```
# queue configuration from blueriq.default.rabbitmq can be overruled
blueriq.dcm.lists-publisher.rabbitmq.exchangeName=dcmListsEvents
```

Authentication

The synchronous operations are protected with basic authentication. To specify the credentials required to log in to the Case Engine, create a user in `user.properties` with the `case-engine` role.

users.properties

```
#format: USERNAME=PASSWORD,PRIVILEGE1,PRIVILEGE2,...
#example: admin=welcome,agent,customer,underwriter

# User for Case Engine
caseengine={noop}caseengine,case-engine
```

If you are using a different authentication provider, like LDAP, you can create a user in that provider instead of `users.properties`, as long as it has the proper role.

Event logging

It is possible to enable logging for events sent to the Case Engine. A line is logged when the Case Engine starts processing the event and when it is finished.

Events list

- AbortTaskService
- AbortTaskWithoutUnlocking
- AssignTask
- CompleteTask
- CreateCase
- DemuxedThrowMessageEvent
- GetCaseInfo
- GetTaskInfo
- RemoveCase
- ScheduledEvent
- StartTask
- ThrowMessage

How to enable

By default, the events are logged on INFO level. To specifically enable logging for the events only, configure the following package on INFO level:

application(-case-engine).properties

```
logging.level.com.blueriq.dcm.caseengine.service=INFO
```

Example output

StartTask event logline:

```
2024-04-25 13:43:09.100 INFO        c.b.d.c.c.CaseEngineRestController httpSessionId="
43C5CF48C812F97E301843659A09A04A" runtimeSessionId="" userId="caseengine" projectName="" projectVersion=""
currentPageName="" tenantName="" [trace=662a41cd99970f5e778c9835294b1413,span=be8f45537809ead6] - [startTask]
Starting transaction - task ID: '6' case ID: '662a41c4fcab766683cab7c5'
```

Finishing a case

Whenever the process for a case finishes, the case will be closed. This means that the following persistent parts of a case will be removed:

- The process and its tasks and data in the process-sql-store
- The case and its manual tasks in the DCM Lists Service's MongoDB database
- The case document in MongoDB
- Any remaining entries in the scheduler database
- The case aggregate and the case data aggregate

We are aware that it may not be desirable to remove the aggregates, for example if your process doesn't have steps for archival. Therefore, we introduced a property to control this behavior:

application-case-engine.properties

```
blueriq.case.engine.data.general.remove-aggregates-on-case-close = false
```

The default value for this property is `true`.