Architecture of OpenMDM 5 Explorer & Navigator

22. September 2015

Franz-Josef Basler
Sibylle Peter
MDM Component according to OpenMDM 5 Architecture

With headless MDM Components there is no client part.
Actual Architecture of Explorer MDM Component

Client
- Rich
- PresentationModel

Server
- PresentationModel
- Logic
- Component API

Explorer Component

Server
- Logic
- Component API

DataProvider Component

Libraries
- de.rechner.openatfx

ATFX-Datei

[delivering end-user happiness]
Technologies used

Client
Rich
PresentationModel

Server
Explorer Component
Logic
Component API
Java 8, Google Guice

Server
DataProvider Component
Logic
Component API
Java 8, Google Guice

Libraries
de.rechner.openatfx

ATFX-Datei

Data Access Mock
OpenMDM API

[delivering end-user happiness]
Technologies used II

- **Java 8** (general programming language)
- **Google Guice** (dependency injection)
- **Griffon** (client-MVC-framework)
- **JavaFX** (UI technology)
- **OpenDolphin** (remote presentation model as client-server communication between MDM Component server and client part)
- **Gradle** (build-automation)

Test environment only

- **JaCoCo** (test coverage)
- **TestFX** (JavaFX integrationstests)
- **Spock** (unit and non ui integration testing)
Distribution of Project modules

Client
- Rich
  - PresentationModel
- Explorer Component
- Libraries
  - de.rechner.openatfx
  - ATFX-Datei

Server
- Explorer Component
- DataProvider Component
  - Logic
  - Data Access
    - Mock

Logic
- Component API
  - Explorer component
  - Explorer component-guice
  - Explorer component-api
  - Data provider component
  - Data provider component-guice
  - Data provider component-api

Model
- Server
- OpenMDM API

[delivering end-user happiness]
Rich Client

Jetty used for server mock.
The rich client provides an application window in which all components can be integrated.
For this Griffon is used again (MVC framework)
Client-Server Communication between MDM Component server and client parts is made using the remote presentation model pattern, implemented through Open Dolphin.
There will be a JSR filed for the remote presentation model.
There are .bat scripts to start client and server.
More informations available in the README.adoc files in the root directories of the projects.
Usage of Eclipse infrastructure

- **Hudson** (automated builds)
- **Sonar** (static source code analysis)
- **Release Repository** (artefacts repository)
- **Snapshot Repository** (artefacts repository)