OpenMDM Architecture Committee

Meeting Notes: 2018-10-12
Author: Stefan Wartini

Attendees

The following members were present
Matthias Ebeling (Daimler AG)
Stefan Wartini (Müller-BBM VibroAkustik Systeme GmbH)
Angelika Wittek (openMDM Toolkit Manager)
Guest: Matthias Koller, (Peak Solution GmbH)

The following members were absent:
Jan Blockx (Siemens AG)
Stefan Ebeling (BMW AG) (ill)
Christian Krenner (AUDI AG)

Topics

Status Milestone 5 for release version 5.0

Milestone 5 is in progress. Currently the developer team puts the source code under EPL 2.0, which is mainly a change in header of all files. In addition, Angelika Wittek prepares for the final review. It is quite extensive paperwork to prepare all relevant forms for this process.

Internationalization, I18N

The SC discussed this item on the last meeting. Tata asked for a quick internationalization of the GUI, since todays implementation displays a GUI in German language. Bugzilla Bug 537224 provides related information (https://bugs.eclipse.org/bugs/show_bug.cgi?id=537224).
The developer team identified two areas where internationalization is required:

a) Labels in the GUI – this can be solved easily by using mechanisms of the Angular framework.
b) Translation of elements or attributes provided by the ODS data sources. These translations must be available in the data sources. A solution of this topic requires more effort since different data sources might provide different translations.

It is not clear yet whether the team should implement a general translation logic in the kernel of openMDM 5 or whether separate implementations for web frontend (Angular) and the attribute names from the data sources (RESTful API) are reasonable.
The SC prefers a quick solution for the Web-GUI to enable Tata to work with openMDM 5.
After a short discussion, the AC decided concordantly to plan the implementation of the internationalization in the WebGUI as a single milestone for the next version - openMDM 5.1 - and to discuss the further proceeding then.

Handling of data sources supporting different data models

Stefan Wartini presented a request for discussion on how openMDM 5 should handle data sources containing data with different application models that deviate from the openMDM 5 application model. All AC members received the request already by Email (see attachment). Focus in the request was set on the different number of levels in the organization of data: project, structure level, test and test step. There are data sources with data organized in only two or three levels. In this case, the data source or the GUI duplicate levels to provide missing levels, to satisfy the openMDM 5 application model. These duplicated levels are very confusing for users browsing through the data.
Matthias Koller explained that in general ODS servers support different application models, while the application model of openMDM 5 has a fixed structure. Currently openMDM 5 is not supported by

ODS servers that do not have an openMDM 5 application model. The openMDM5 API relies on the application model and that data sources support this model. Otherwise, the openMDM 5 core would have to handle all exceptions. If the application model of a data source does not provide all required values for the openMDM 5 application model, the source or an adapter to the source should provide the missing information as dummy values.

In general, it is necessary to separate between the internal application model of openMDM 5 and the way a client application will present the data to a user.
Therefore, openMDM 5 has implemented a mechanism called NodeProvider to modify the view on the data in a client application. Currently only one NodeProvider is defined for the preference service of the openMDM backend that is valid for all data sources. The preference service stores information on e.g. tree hierarchies or attributes in a PostgreSQL database. Any client could query information using the RESTful interface of the preference service.
If the implementation of the NodeProvider would support different data sources with different preference settings, the clients could even display the results from a full text search in multiple sources all at once.

The result of this discussion is a new requirement – the support of different NodeProviders for different data sources – that should be presented to the SC for further judgement.

Flexible deployment if openMDM 5 to application servers

Matthias Koller described a problem concerning the deployment of openMDM5 in the BMW environment. Currently the application server GlassFish reads it's configuration from fixed directories. This is not very flexible when supporting e.g. different staging levels. Matthias Koller suggested providing the configuration directories for GlassFish by a system property. If the system property is available it is used, otherwise, the system defaults to the current implementation using the fixed directories.
After a short discussion, the AC agreed concordantly.

Changes and enhancements for Atfx-Import and -Export functionalities

BMW assigned Peak Solution to implement an enhancement of openMDM5 to import and export ATFX data to and from the openMDM5 data sources. In his presentation, Matthias Koller explained the project (see presentation hereto attached). For a prerequisite, the ATFX data conform to the openMDM5 data model so the importer does not have to map the data. The importer consists of two components, an importer component (ATFXAdapter), transporting the data through the openMDM5 Java API and a simple copy component (API Copy) copying the data from one data source to another without any mapping.
For the export, data from the openMDM shopping basket are exported to ATFX files.
In his presentation, Matthias Koller discussed the component model and the import and export sequence for ATFX-files.
For import the user uploads ATFX-Data via the REST-Interface, the ATFXAdapter is initialized with these data and the copy command from the REST-Interface triggers the API Copy component. The information from the ATFXAdapter is read, templates are resolved and added and the data are transferred to the ODS Adapter.
The export works in a similar sequence. The client sends a shopping basket describing the required ATFX-data in the ODS data source and the ATFXAdapter provides the corresponding empty ATFX headers. API Copy copies the requested data into the ATFX headers and the ATFXAdapter transfers the data back to the client.
Based on this concept, additional extensions are conceivable. Support of other file formats, mapping mechanisms or a file system based import of data are possible but not part of the current implementation.

In the first step, the extension shall transfer small amounts of data through the REST-Interface. Alternatively, storage of the data on a temporary file storage is possible.
Since the new components will be part of the openMDM5 backend, another advantage of this approach is that a client can address any data source adapter supporting the openMDM5 Java API to import or export ATFX Data. Even the ATFXAdapter can act like a simple backend supporting exactly one AFTX file.
Peak Solution started to implement the presented concept for BMW on base of the openATFX-library and BMW announced to contribute this extension as open source to the openMDM repositories.
Peak Solution will document the contribution a Jira ticket.
The attending members of the AC welcome the initiative of BMW and agree concordantly.

Next telephone conference

The next telephone conference according to the schedule will be on November 9 2018 at 10:00.
Since Stefan Wartini cannot participate definitely, Stefan Ebeling might have to find an auxiliary date.

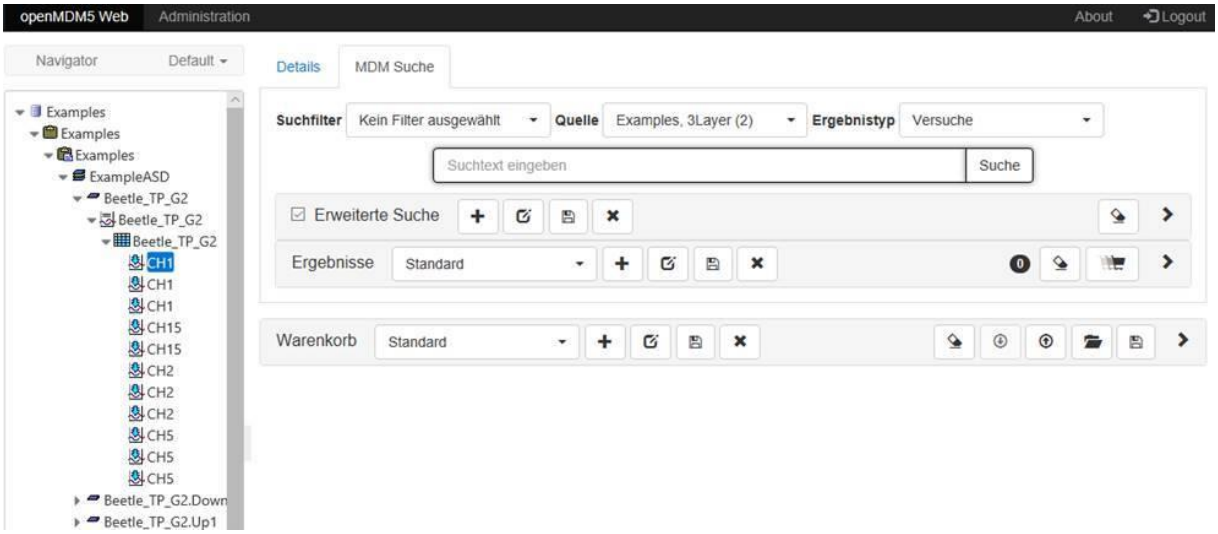
Attachments

Request, Stefan Wartini (Müller BBM VAS):

Hello all,

I have a topic I'd like to discuss on the next AC telco since I expect it might result in a request to modify the API.
When connecting to different data sources via openMDM we expect that these sources may be different organized e.g. looking at the different levels when browsing in the data. We even expect that within one company older and newer data may be structured differently and a migration of the data to keep them always up to date is too costly.
The PAK cloud e.g. can manage older measurement data that are organized in only two structure levels (test and subtitle) while newer measurement data contain 3 levels (project, test, subtitle) and the openMDM model even consists of four levels (project, structure level, test and test step).
Currently the openMDM-Browser always uses four levels to display measurement data and so for data sources with older data the additional levels are created by duplicating entries.
We believe that these duplicates are very inconvenient for the user and will decrease the acceptance of the openMDM-tools.

As an example, I include a screenshot of the current mdm-Browser displaying measurement data from a source with only two levels.

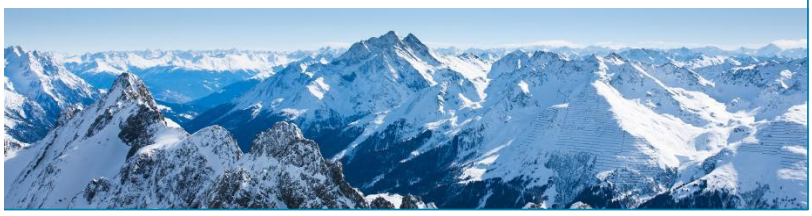


Do we want to preserve these duplicates in future?
I would like to discuss whether we provide a way to handle different data sources efficiently. One idea could be to allow the browser to ask for the number of levels when opening the data source or to deposit this information when configuring the data source in openMDM.

Best regards
Stefan

Presentation, Matthias Koller (Peak Solution)

ATFX Import and Export in openMDM 5



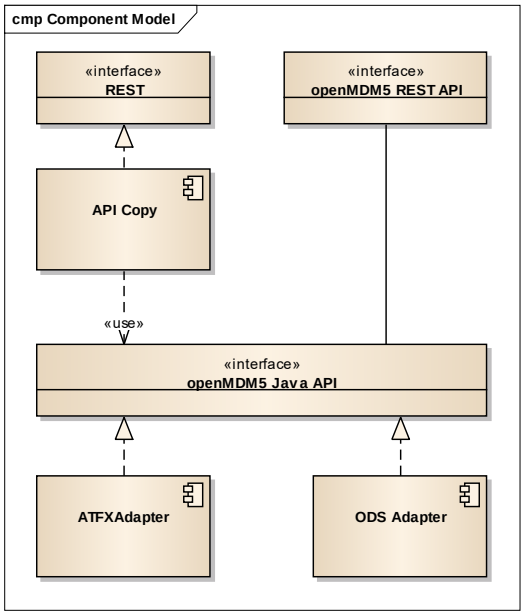
Peak Solution GmbH

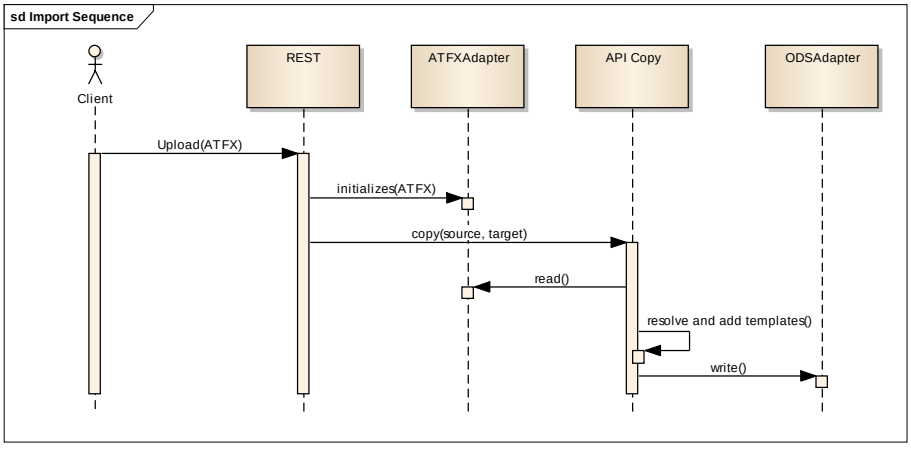
Goal

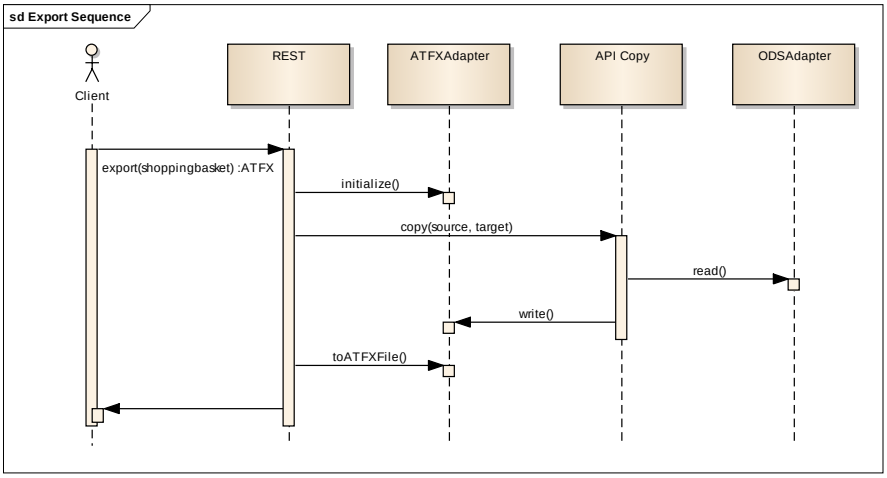
- Read an ATFX file and import the data into an openMDM5 datasource
- ATFX file already has an openMDM model
- Export data items selected in shopping basket into an ATFX file

Basic Idea

- Split the importer into two components
  - An adapter providing the data to import through the openMDM 5 Java API
  - A copy component responsible for coping data from one datasource to another


cmp Component Model
«interface»
REST
«interface»
openMDM5 REST API
API Copy
«use»
«interface»
openMDM5 Java API
ATFXAdapter
ODS Adapter


sd Import Sequence
Client
REST
ATFXAdapter
API Copy
ODSAdapter
Upload(ATFX)
initializes(ATFX)
copy(source, target)
read()
resolve and add templates()
write()



Possible future extensions

- Provide adapter for other file formats
- Provide mapping mechanisms similar to external systems in openMDM4
- Implement filesystem based imports