Minutes of the Architecture Committee

Place:	WebEx	
Date/Time:	2016-07-01 – 11:00-12:00 CEST	
Minutes:	Andreas Benzing, Canoo	
Participants:	Andreas Benzing	Canoo (Daimler) (chair)
	Stefan Ebeling	BMW
	Gerwin Mathwig	PL MDM@WEB
	Stefan Wartini	MBBM
	Jan Wiegelmann	NorCom
	- missing -	AUDI
	- missing -	PL MDM|BL
Guests:	Sebastian Dirsch, Dennis Schröder	

Participants are referred to by their initials, i.e. GM refers to Gerwin Mathwig.

1 Definition of a collaboration concept and a repository structure

This issue [1] has been directed to the AC for clarification.

The way of collaborating in openMDM 4 is revisited. Given the change from Subversion to Git along with the new environment at Eclipse, this mode is no longer adequate. At the moment, there are multiple repositories, grouped by interfaces, deployment artifacts, and other parts of openMDM 5. While this allows for individual tracking of the history of each of these repositories, the basic requirement of having individual versioning is not explicitly addressed.

In order to allow for parallel development of multiple artifacts, the separation of code into multiple repositories is not required. Instead of working on an isolated part of the code, Git provides the means to work on a separate branch [2] of the code base. This mechanism can be used to develop multiple versions of the software, for example to implement new features on one branch and bug fixes for an existing release on another [3]. The collaboration between teams then can be implemented using so-called merge requests, where the maintainer of the main branch is basically asked to integrate (merge) the new code into the branch [4].

Given this collaboration model, the versioning can be done very flexibly. Since the resulting artifacts rather than the code base itself should be versioned, these artifacts are made available to non-developers once a version is released. The release process itself is covered by the Eclipse Development Process [5]. A release can of course reference a specific status (commit) of the source code repository.

The combination of community and proprietary elements can be accomplished by adding released artifacts as project dependencies. This way, no manual copying of source files into another source directory is required unless working on an unreleased version of the code base. This exception can still be addressed using automated nightly builds in addition to regular releases [5].

However, to get the described approach working, the details must be further clarified. The initial release of the web client has to be finalized, including the IP clearance currently in progress. The responsibilities in the the Eclipse projects participating in the openMDM WG in terms of opening, reviewing, and approving merge requests must be clarified by the QC.

2 Components in openMDM 4 vs Modules in openMDM 5

Since the meaning of the term component has changed significantly from openMDM 4 to openMDM 5, the participants agree to refer to openMDM 5 modules to avoid confusion.

3 Architecture Conformity Check

AB suggests a service package to check the code base for conformity to the architecture specification. The participants agree that conformity has to be ensured. However, this check must be integrated into the release process (see above), rather than having a separate service.

4 Next Meeting

The next AC meeting will be scheduled when needed.

References

[1] https://openmdm.atlassian.net/browse/ORGA-137
[2] https://git-scm.com/book/en/v1/Git-Branching
[3] http://nvie.com/posts/a-successful-git-branching-model/
[4] http://doc.gitlab.com/ee/workflow/gitlab_flow.html
[5] https://eclipse.org/projects/dev_process/development_process.php#6_4_Releases