Transformation AML-Modell zu Code

Umsetzung für Besprechung am 09.08.2017.

Für AMLmodel\_v4.

Inhalt

[ArchitectureRoleClassLib/Service 2](#_Toc490517708)

[ArchitectureRoleClassLib/Component 3](#_Toc490517709)

[CommunicationRoleClassLib/Ports 4](#_Toc490517710)

[CommunicationRoleClassLib/Port 5](#_Toc490517711)

[CommunicationRoleClassLib/PortStyle 6](#_Toc490517712)

[CommunicationRoleClassLib/PortParameters 7](#_Toc490517713)

[CommunicationRoleClassLib/PortType 8](#_Toc490517714)

[DomainModelRoleClassLib/MessageModel 9](#_Toc490517715)

[AML-Elemente ohne Auswirkung auf den generierten Code 10](#_Toc490517716)

[Festlegungen, bekannte Einschränkungen und mögliche Verbesserungen 11](#_Toc490517717)

[Festlegungen 11](#_Toc490517718)

[Einschränkungen 11](#_Toc490517719)

[mögliche Verbesserungen 11](#_Toc490517720)

# ArchitectureRoleClassLib/Service

|  |  |
| --- | --- |
| * Für jedes direkte IE mit Role ArchitectureRoleClassLib/Service in der Instance-Hierarchy:   + neues Maven-Projekt im Codegenerator-Output-Directory in Subfolder <IE-Name> anlegen:     - pom.xml anlegen 🡪 ‚groupId’ und ‚artifactId’ wie in den Attributen des IE spezifiziert setzen, initial keine ‚modules’     - README.md anlegen 🡪 Readme mit fixfertigem Inhalt, Service-Name auf <IE-Name> setzen | |
|  |  |

# ArchitectureRoleClassLib/Component

|  |  |
| --- | --- |
| * Für jedes direkte IE mit Role ArchitectureRoleClassLib/Component unterhalb eines IE mit Role ArchitectureRoleClassLib/Service:   + neues Maven-Module mit Namen <Component-artifactId> anlegen:     - Main.java anlegen 🡪 Klasse mit leerer Methode ‚main’     - log4j2.xml anlegen 🡪 Standard-Logging-Konfiguration     - pom.xml anlegen 🡪 ‚parent’: den Service setzen, ‚groupId’: <Service-groupId>.<Service-artifactId>‚artifactId: <Component-artifactId>, initial keine ‚dependencies’, ‚profile’ anlegen, welches die Main.java ausführt   + pom.xml von Service anpassen 🡪 module <Component-artifactId> hinzufügen | |
|  |  |

# CommunicationRoleClassLib/Ports

|  |  |
| --- | --- |
| * Für jedes direkte IE mit Role CommunicationRoleClassLib/Ports unterhalb eines IE mit Role ArchitectureRoleClassLib/Component:   + wenn Maven-Module ‚communication’ noch nicht angelegt ist:     - neues Maven-Module mit Namen ‚communication’ anlegen und die fixfertigen Komponenten darin hinzufügen     - pom.xml von Service anpassen 🡪 module ‚communication’ hinzufügen   + Maven-Module der Component anpassen:     - CommunicationService.java anlegen 🡪 hat initial noch keine Ports zugewiesen     - Main.java anpassen 🡪 Example-Usage des CommunicationService der Methode ‚main’ hinzufügen     - pom.xml anpassen 🡪 module ‚communication’ als ‚dependency’ hinzufügen | |
|  |  |

# CommunicationRoleClassLib/Port

|  |  |
| --- | --- |
| * Für jedes direkte IE mit Role CommunicationRoleClassLib/Port unterhalb eines IE mit Role CommunicationRoleClassLib /Ports:   + Maven-Module der Component anpassen:     - CommunicationService.java anpassen 🡪 Port zu Methode ‚init’ hinzufügen, Parameter ‚portName’ = IE-Name | |
|  |  |

# CommunicationRoleClassLib/PortStyle

|  |  |
| --- | --- |
| * Für jedes direkte IE mit Role CommunicationRoleClassLib/PortStyle unterhalb eines IE mit Role CommunicationRoleClassLib /Port:   + Maven-Module der Component anpassen:     - CommunicationService.java anpassen 🡪 Style zum Port in Methode ‚init’ hinzufügen | |
|  |  |

# CommunicationRoleClassLib/PortParameters

|  |  |
| --- | --- |
| * Für jedes direkte IE mit Role CommunicationRoleClassLib/PortParameters unterhalb eines IE mit Role CommunicationRoleClassLib /Port:   + Maven-Module der Component anpassen:     - CommunicationService.java anpassen 🡪 Parameters zum Port in Methode ‚init’ hinzufügen | |
|  |  |

# CommunicationRoleClassLib/PortType

|  |  |
| --- | --- |
| * Für jedes direkte IE mit Role CommunicationRoleClassLib/PortType unterhalb eines IE mit Role CommunicationRoleClassLib /Port:   + Maven-Module der Component anpassen:     - CommunicationService.java anpassen 🡪 Type zu Enum(s) und Port in Methode ‚init’ hinzufügen | |
|  |  |

# DomainModelRoleClassLib/MessageModel

|  |  |
| --- | --- |
| * Wenn das IE mit Role CommunicationRoleClassLib/Port ein zugewiesenes IE mit Role CommunicationRoleClassLib /MessageModel hat:   + wenn Maven-Modul ‚messagemodel’ noch nicht angelegt ist:     - neues Maven-Module mit Namen ‚messagemodel’ anlegen und fixfertige MessageModel.java darin hinzufügen     - pom.xml von Service anpassen 🡪 ‚module’ ‚messagemodel’ hinzufügen   + wenn pom.xml von Component noch keine ‚dependency’ namens ‚messagemodel’ hat:     - pom.xml von Component anpassen 🡪 ‚dependency’ ‚messagemodel’ hinzufügen   + wenn spezifiziertes Message-Model in Maven-Module ‚messagemodel’ noch nicht angelegt ist:     - spezifiziertes Message-Model in Maven-Module ‚messagemodel’ anlegen | |
|  |  |

# AML-Elemente ohne Auswirkung auf den generierten Code

Folgende AML-Elemente haben derzeit keine Auswirkung auf den generierten Code:

* alle RCs in
  + EnvironmentRoleClassLib
  + ProgrammingLanguageRoleClassLib
  + SourcecodeRepositoryRoleClassLib
* External Interfaces und Internal Links zwischen den Ports

# Festlegungen, bekannte Einschränkungen und mögliche Verbesserungen

## Festlegungen

* tbd

## Einschränkungen

* tbd

## mögliche Verbesserungen

* Derzeit werden alle Port-Factory- und Port-Implementierungen in das Maven-Modul ‚communication’ kopiert. Hier würde es ausreichen, nur die Implementierungen (genauer: deren Package) zu kopieren, die auch tatsächlich im AML-Modell referenziert werden. Performancetechnisch ist es aber kein Problem, da nur die benötigten Port-Factories und nur bei Bedarf geladen werden.
* Derzeit erzeugt die RC ‚MessageModel’ nur Message-Models mit einem bestimmten Namen. Eventuell sollte auch der dazugehörige Port (genauer: der hier gesetzte Receiver bzw. Sender) nur das Senden bzw. Empfangen dieses einen Message-Models zulassen.