Conference Applikation - Service Architektur

# Versionshistorie

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Datum | Author | Beschreibung |
| 1.0 | 14.11.2013 | Christian Predikant | Initiale Erstellung |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Inhaltsverzeichnis

[Versionshistorie 2](#_Toc372233174)

[Übersicht und Ziel 4](#_Toc372233175)

[Architektur der Applikation 4](#_Toc372233176)

[Schicht 1 – Persistenz 4](#_Toc372233177)

[Schicht 2 - Business-Logik 4](#_Toc372233178)

[Schicht 3 – Web 4](#_Toc372233179)

[Entwicklungsleitfaden 4](#_Toc372233180)

[Projektaufbau 4](#_Toc372233181)

[Conference-common 5](#_Toc372233182)

[Coneference-ear 5](#_Toc372233183)

[Conference-jsf-web 5](#_Toc372233184)

[Conference-rest-api 5](#_Toc372233185)

[Conference-rest-web 5](#_Toc372233186)

[Conference-service-api 5](#_Toc372233187)

[Conference-service-impl 5](#_Toc372233188)

[Conference-assembly 6](#_Toc372233189)

[Namenskonventionen 6](#_Toc372233190)

[Codingstandards 6](#_Toc372233191)

[Technologiestack 6](#_Toc372233192)

# Übersicht und Ziel

Dieses Projekt hat zum Ziel eine Conference-Applikation zur erstellen. Diese soll Coference-Besuchern ermöglichen sich über die Conference, die Talks, die Rooms und die Speaker zu informieren. Weiter soll es den Administratoren möglich sein diese Einheiten zu Pflegen.

# Architektur der Applikation

Die Applikation ist in drei Schichten unterteilt, welche im Folgenden beschrieben werden. Für alle Schichten gilt das CDI verwendet wird, um die Komponenten miteinander und Schichten zu verknüpfen

## Schicht 1 – Persistenz

Die Persistenz wird durch eine MySQL-Datenbank bereitgestellt. Damit man von Java aus einfacher auf die relationalen Daten zugreifen kann wird JPA2 in Kombination mit Hibernate 4 als ORM-Provider verwendet. Die Datenobjekte werden via JPA-Entities abgebildet. Diese sind mit Named Queries versehen. Auf Relationen, welche über JPA-Mechanismen bereigestellt werden ist zu verzichten. Dies gilt jedoch nur für Relationen, die ein Mapping auf Collections verwenden. Diese würden in Schicht 3- Web Probleme bereiten.

## Schicht 2 - Business-Logik

Die Businesslogik wird via Enterprise-Java-Beans (EJB) und CDI bereitgestellt. Die Business-Logik ist für eine Domäne gekapselt. Im Fall der Applikation ist dies die Domäne Conference.

## Schicht 3 – Web

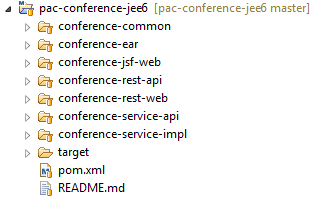
In dieser Schicht wird die Business-Logik verwendet und zur Anzeigen gebracht bzw. für den Zugriff via HTTP bereitgestellt. Dies geschieht einerseits per JSF-2.1 und auf der anderen Seite per JAX-RS 1.1 und JSON.

# Entwicklungsleitfaden

In den folgenden Abschnitten wird beschrieben, welche Namenskonventionen und welche Codingstandards gelten. Weiter wird der Aufbau des Projekts erklärt und welcher Technologie verwendet wird.

## Projektaufbau

Grundsätzlich ist zu erwähnen, dass als Build und Dependency Management Tool Maven 3 verwendet wird und es sich beim Projekt um ein Maven-Multiprojekt handelt. Der Projektaufbau wird durch den Folgenden Screenshot dargestellt:



Das Root-Projekt ist „pac-conference-jee6“ und beinhaltet alle Globalen Einstellungen für das Projekt und alle Module innerhalb dieses Projekts. Im Folgenden werden die einzelnen Module beschrieben.

### Conference-common

In diesem Modul werden Modulübergreifende querschnittliche Klassen und Utilities platziert. Dieses Modul ist ein einfaches JAR.

### Coneference-ear

Dieses Modul ist, wie der Name andeutet das Enterprise-Archive, welches alle anderen Module zusammen fasst und für das Deployment auf einen JEE –Server bereitstellt. Es bildet zugleich das Release-Unit des Projekts. Wird später ein zentrales Pesistence-Unit benötigt, würde sie hier platzieren werden.

### Conference-jsf-web

In diesem Modul ist die JSF-Webanwendung untergebracht. Es beinhaltet alle für die JSF-Webanwendung benötigten XHTML-Seiten, Konfigurationen, Controller-Klassen und Ressourcen. Er wird als WAR-File bereitgestellt.

### Conference-rest-api

Dieses JAR-Modul beinhaltet die Interfaces für die REST-Schnittstelle der Applikation. Die Schnittstelle ist in „Secure“ und „Unsecure“ Teile zerlegt worden. Damit ist es möglich die Teile der Schnittstelle abzusichern, welche nicht öffentlich sein sollen. Dies betrifft all Create, Delete und Update Funktionen.

### Conference-rest-web

Dieses Modul enthält die Implementierung der REST-Services ist wie die API in „Secure“ und „Unsecure“ unterteilt. Es wird als WAR-bereitgestellt und läuft völlig eigenständig neben der JSF-Webanwendung.

### Conference-service-api

Dieses Modul ist die API für eine Fachliche Domäne. Im konkreten Fall ist es die Domäne Conference. Die API enthält die benötigten Entities, Interfaces und Exceptions der Domäne. Dieses Modul wird als JAR zur Verfügung gestellt.

### Conference-service-impl

Dieses Modul ist die Implementierung der fachlichen Domäne und enthält neben den Service-Implementierungen die Interceptoren, Decorator, Monitoring und Messaging Komponenten. Ebenso besitzt das Modul das verwendete Persistence-Unit.

### Conference-assembly

Das Conference-Assembly Modul erstellt das Assembly für die Applikation und Enthält:

* Das Ear-File der Applikation
* Die Sourcen des Releases
* Die JavaDoc des Releases
* SQL für CREATE und UPDATE

## Namenskonventionen

Für die Entwicklung gelten folgende Namenskonventionen und Regeln:

* Das Basis-Package der Applikation ist „com.prodyna.pac.conference“
* Innerhalb der Module werden die Klassen technisch gruppiert. Und mit dem Infix des Moduls versehen. Dies führt zu folgenden Package-strukturen

|  |  |
| --- | --- |
| Komponente | Package |
| Interceptor + Binding | com.prodyna.pac.conference.{infix}.interceptor.CamelCaseClasses |
| Decorator | com.prodyna.pac.conference.{infix}.decorator.CamelCaseClasses |
| JMX | com.prodyna.pac.conference.{infix}.jmx.CamelCaseClasses |
| JMS | com.prodyna.pac.conference.{infix}.jmx.CamelCaseClasses |
| Domäne-Impl | com.prodyna.pac.conference.{infix}.CamelCaseClasses |
| Domäne-API | com.prodyna.pac.conference.{infix}.api.CamelCaseClasses |
| Exceptions | com.prodyna.pac.conference.{infix}.exception.CamelCaseClasses |
| Entities | com.prodyna.pac.conference.{infix}.model.CamelCaseClasses |
| Util | com.prodyna.pac.conference.{infix}.util.CamelCaseClasses |

## Codingstandards

Als Entwicklungsumgebung ist Eclipse zuverwenden, welche die JBOSS-Tools enthält oder das vorgefertigte Eclipse JBOSS-Tools von JBOSS. Dieses korrespondiert am besten mit dem JBOSS-AS-7 und Maven.

Für die Implementierung ist zwingend erforderlich, dass das Encoding der Entwicklungsumgebung auf UTF-8 eingestellt ist. Für die Formatierung der Quellcodes wird der Eclipse-Standard verwendet.

Das mindeste an Code-Dokumentation, was vorhanden sein soll ist JavaDoc der Klasse und JavaDoc aller Interface-Methoden. Klassen-Member müssen nicht unbedingt dokumentiert werden.

## Technologiestack

|  |  |
| --- | --- |
| Technologie | Version |
| Maven | 3.1.1 |
| JBoss AS | 7.1.1Flnal |
| MYSQL | 5.5.x |
| Java | 1.6 |
| PrimeFaces | 4.0 |
| JBoss.BOM | 1.0.7.CR8 |
| Davon abweichend: |  |
| Javax.faces | 2.1.26 |
| Myfaces-COODI | 1.0.5 |