­



PAC Conference Webapplikation – Service Architecture  
27.11.2013

Auor: Martin Schwietzke



[Projektinformation 3](#_Toc373320228)

[Build und Installation der Anwendung 3](#_Toc373320229)

[Code Repository 3](#_Toc373320230)

[Technologie Stack 3](#_Toc373320231)

[Projektstruktur 3](#_Toc373320232)

[Übersicht der Applikation 4](#_Toc373320233)

[Domain Model 5](#_Toc373320234)

[Datenbankschema 6](#_Toc373320235)

[Komponenten Diagramme 6](#_Toc373320236)

[Entwicklungsvorgaben 7](#_Toc373320237)

[Code Konventionen 7](#_Toc373320238)

[Code Formatierung 7](#_Toc373320239)

[Packagenamen 7](#_Toc373320240)

[Projekt Basis Package 7](#_Toc373320241)

[Modul/Komponenten Basis Package 7](#_Toc373320242)

[Modul/Komponenten Package bei Trennung Interface und Implementierung 8](#_Toc373320243)

[Komponenten Benamungen 8](#_Toc373320244)

[Methodenbenamungen 8](#_Toc373320245)

[CRUD Methoden 8](#_Toc373320246)

[Logging Statements 8](#_Toc373320247)

[Exceptions 8](#_Toc373320248)

[Exceptions in Services 8](#_Toc373320249)

[Validierung 9](#_Toc373320250)

[Unit Tests 9](#_Toc373320251)

[Globale Test Resourcen und Utilities 9](#_Toc373320252)

[EJB Session Bean Tests 9](#_Toc373320253)

[Entwicklungsumgebung 9](#_Toc373320254)

[IDE 9](#_Toc373320255)

[Testing 9](#_Toc373320256)

[Build Tools 9](#_Toc373320257)

[Abbildungsverzeichnis 10](#_Toc373320258)

# Projektinformation

Die Web Applikation „Conference“ wurde im Rahmen der PRODYNA Architecture Certification entwickelt.

# Build und Installation der Anwendung

Hierzu bitte die separate Installationsanleitung verwenden:

PAC-Conference-Installation.txt

# Code Repository

Der Source Code der Applikation wird in einem Git Repository verwaltet und ist unter folgender URL öffentlich erreichbar:

<https://github.com/martinschwietzke/pac-conference.git>

# Technologie Stack

|  |  |  |
| --- | --- | --- |
| **Technologie** | **Version** | **Kategorie/Beschreibung** |
| JBoss | 7.1.3 | Laufzeitumgebung |
| MySQL | 5.5.32 | Datenbank |
| Java Servlets | 3.0 | JEE Standard API |
| JAX-RS | 1.1 | JEE Standard API |
| JMS | 1.1 | JEE Standard API |
| CDI | 1.0 | JEE Standard API |
| JMX | 2.0 | JEE Standard API |
| EJB | 3.1 | JEE Standard API |
| JPA | 2.0 | JEE Standard API |
| JSF | 2.1 | JEE Standard API |
| CDI API | 1.0-SP4 | JSF UI Bibliothek |
| IceFaces | 3.3 | JPA Framework |
| Hibernate | 1.0.1.Final | Test Framework |
| Arquillian | 1.1.0.Final | Test Freamwork |
| Arquillian Persistence Extension | 1.0.0.Alpha6 | Test Framework Extension für Persistenz Test |
| Shrinkwrap | 2.0.0-alpha-3 | Test Framework |
| JUnit | 4.11 |  |

# Projektstruktur

|  |  |
| --- | --- |
| **Projekt** | **Beschreibung** |
| conference | Maven Parent Project (Multimodule). Gibt Dependencies, Plugins und Profile vor. |
| pac-conference-conference-api | Conference Service Implementierung |
| pac-conference-conference-impl | Conference Service API |
| pac-conference-speaker-api | Speaker Service API |
| pac-conference-speaker-impl | Speaker Servive Implementierung |
| pac-conference-room-api | Room Service API |
| pac-conference-room-impl | Room Service Implementierung |
| pac-conference-talk-api | Talk Service API |
| pac-conference-talk-impl | Talk Service Implementierung |
| pac-conference-web | User Frontend Web Applikation |
| pac-conference-rest | Web Modul für REST Services |
| pac-conference-common | Enthält Exceptions, Utils und querschnittlich betreffende Aspekte (Monitoring etc.) |
| pac-conference-ear |  |
| pac-conference-documentation | Enthält Dokumentation und Installationsanleitungen. |
| pac-conference-test-common | Enthält Test Resource und gemensame Tes t Klassen. |
| pac-conference-dist | Projekt für die Erzeugung des Distributionspakets (Assembly) |

# Übersicht der Applikation

RDBMS (MySQL)

JEE Container (Jboss)

JPA (Hibernate)

EJB Session Beans

JSF

CDI

JMS

JAX-RS

JTA

Logging

Bean Validation

# Domain Model

Talk

Speaker

Room

Conference

TalkSpeaker

1

\*

1

\*

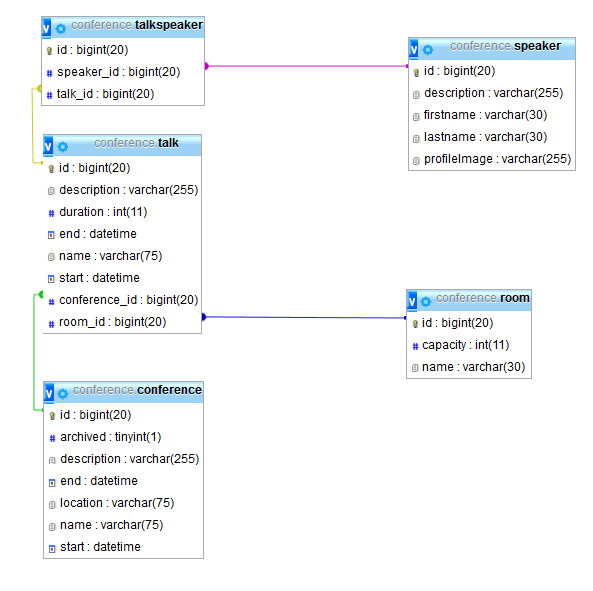
1

\*

\*

1

# Datenbankschema



# Komponenten Diagramme

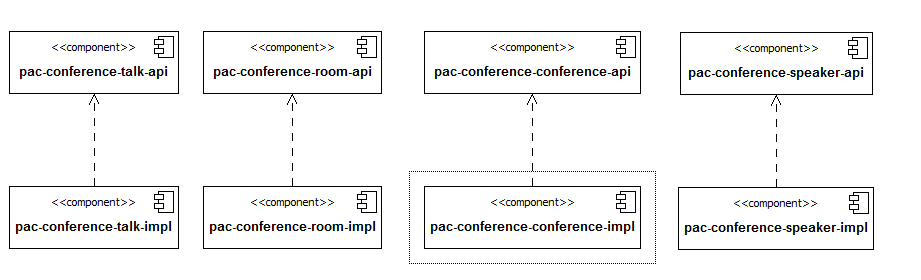


Abbildung Interface Realisierungen

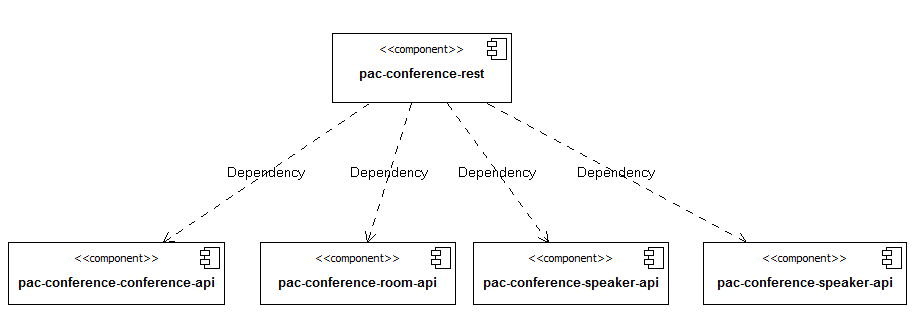


Abbildung Service Dependencies REST

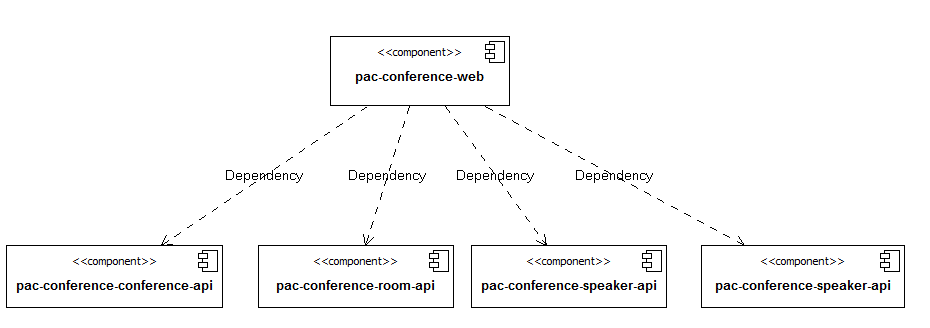


Abbildung 3 Service Dependencies Web Frontend

# Entwicklungsvorgaben

## Code Konventionen

### Code Formatierung

Es wird der Standard eclipse Codeformatter der im Projekt eingesetzten eclipse Version verwendet.

Vor dem Code-Checkin sind grundsätzlich alle Sourcen zu formatieren.   
Im weiteren Verlauf des Projektes wird die Formatierung automatisiert auf Seite des Code Repositories durchgeführt. Bei Abweichungen zwischen Checkin Code und formatiertem Code wird der Commit zurückgewiesen.

### Packagenamen

#### Projekt Basis Package

Alle dem Projekt zugehörigen Klassen befinden sich unterhalb des Basis Packages:

com.prodyna.pac.conference

#### Modul/Komponenten Basis Package

Jede Komponente bzw. jedes Modul hat ein eindeutige Basis Package. Der Aufbau ist:

<Projekt Basis Package>. <Fachlicher Komponenten Key>

Für die fachliche Komponente „Speaker“ ist die Package Bezeichnung z. B.:

com.prodyna.pac.conference.speaker

#### Modul/Komponenten Package bei Trennung Interface und Implementierung

Bei der Trennung einer Komponente in API- und Implementierung ist folgender Package Aufbau zu verwenden:

<Komponenten Basis Package>.api

<Komponenten Basis Package>.impl

Für die fachliche Komponente „Speaker“ sind die Package Bezeichnungen z. B.:

com.prodyna.pac.conference.speaker.api

com.prodyna.pac.conference.speaker.impl

#### Komponenten Benamungen

|  |  |  |
| --- | --- | --- |
|  | Suffix |  |
| JEE Interceptor Klassen | Interceptor |  |
| JEE Decorator Klassen | Decorator |  |
| JEE Injection Producer Klassen | Producer |  |
| MBeans | MBean |  |
| EJB Session Beans | ServiceBean |  |
| REST Resources | RESTService |  |

## Methodenbenamungen

### CRUD Methoden

create<Type>()  
update<Type>()  
delete<Type>()  
get<Type>()

## Logging Statements

Logging Statements mit Ausgabe von Variablen Werten werden immer in folgender Form erstellt:

log.info(„Updated Talk with id [„+talkId+“]“)

# Exceptions

## Exceptions in Services

Exceptions die in Service Methoden deklariert bzw. geworfen warden, müssen alle direkt oder indirekt von folgender Exception ableiten:

com.prodyna.pac.conference.common.exception.ConferenceServiceException

# Validierung

Für die Validierung soweit es geht Bean Validation verwendet.

# Unit Tests

## Globale Test Resourcen und Utilities

Alle Projekt haben eine test-scoped Dependency zum Module “pac-conference-test-common”. Die dort enthaltenen Utilities, Testklassen und Test Resourcen sind zu verwenden.

Für Persistenz Tests sind die Testdaten aus folgender Datei zu verwende:

com.prodyna.pac.conference.testcommon.TestConstants.TEST\_DATA\_SET\_JSON\_FILE

Die Datenbankinitialisierung erfolgt über das Annotieren der Unit Test Methoden:

@UsingDataSet(TestConstants.TEST\_DATA\_SET\_JSON\_FILE)

## EJB Session Bean Tests

Alle Session Bean Test leiten von folgender abstrakten Test Klasse ab:

com.prodyna.pac.conference.common.exception.ConferenceServiceException

# Entwicklungsumgebung

## IDE

Bei der Entwicklung ist eclipse Keplerr einzusetzen:

<http://www.eclipse.org/downloads/packages/eclipse-ide-java-ee-developers/keplerr>

## Testing

Für das Testen von Software Komponenten werden folgenden Technologien eingesetzt:

* JUnit
* Arquillian
* Shrinkwrap

## Build Tools

Als Build und Dependency Tool word Maven eingesetzt.

# Abbildungsverzeichnis

[Abbildung 1 Interface Realisierungen 4](#_Toc373318825)

[Abbildung 2 Service Dependencies REST 5](#_Toc373318826)

[Abbildung 3 Service Dependencies Web Frontend 5](#_Toc373318827)