# Verwendete Technologien

- Es werden entsprechend dem Kundenwunsch JBoss 7.1.1, JSF 2.2 (TODO version) und JPA verwendet.

- Als JPA Implementierung wird Hibernate 2.0 als etablierter Standard gewählt

TODO arquillian

# Aufbau der Quellen im Repository

Das Repository "PAC" gliedert sich in drei Unterordner:

## conference

Dies ist der Wurzelknoten des Java Workspace

Die einzelnen Java-Projekte sind als Maven Multi Module Projekte organisiert:

conference Oberprojekt

conference-base Querschnittliche Funktionen

conference-room Oberprojekt der Room-Komponente

conference-room-api RoomService API und Entität

conference-room-impl Implementierung des RoomService

conference-speaker Oberprojekt der Speaker-Komponente

conference-speaker-api SpeakerService API und Entität

conference-speaker-impl Implementierung des RoomService

conference-conference Oberprojekt der Conference-Komponente

conference-conference-api Conference API und Entitäten

conference-conference-impl Implementierung Conference

conference-talk Oberprojekt der Talk-Komponente

conference-talk-api TalkService API und Talk Entität

conference-talk-impl Implementierung TalkService und TalkSpeakerAssignment Entität

conference-web JSF GUI

conference-rest REST Services

## conference-base

Dieses Projekt beinhaltet querschnittliche Funktionen.

* persistence.xml
  + Da alle Serviceprojekte letztlich auf die gleiche DB zugreifen und die Tabellen teilweise per FK miteinander verbunden sind, wird nur eine querschnittliche persistence.xml verwendet.
* Producer für Logger, QueueConnection, InitialContext und EntityManager
* PerformanceLoggingInterceptor
* ServiceLoggingInterceptor

## conference-room

Implementierung des RoomService und dessen Entität Room.

Keine Beziehungen zu anderen Services. So wird sichergestellt, dass Rooms im Objektgeflecht ein „Blatt“ bleiben und keine unnötigen Zirkulärbeziehungen entstehen

## conference-speaker

Implementierung des SpeakerService und dessen Entität Speaker.

Keine Beziehungen zu anderen Services. So wird sichergestellt, dass Speaker im Objektgeflecht ein „Blatt“ bleiben und keine unnötigen Zirkulärbeziehungen entstehen

## conference-conference

Beinhaltet die Implementierungen von ConferenceService sowie der Conference Entität.

Keine Beziehungen zu anderen Services. So wird sichergestellt, dass Rooms im Objektgeflecht ein „Blatt“ bleiben und keine unnötigen Zirkulärbeziehungen entstehen

**conference-talk**

## doc

Dokumentation. Enthält dieses Dokument

## soapui

Enthält ein zugehöriges SoapUi Projekt zum Testen der Rest Services

# Standard für Coding

Codeformatting:

Es wird der Standard-Codeformatter von Eclipse Kepler WTP verwendet.

# Namenstandards für Projekte, Packages, Interfaces, Services, Klassen, Exceptions

# Exceptions

Es werden unchecked Exceptions verwendet.

Für technische Probleme stehen bereits verschiedene RuntimeExceptions zur Verfügung; falls nötig wird in RuntimeException gewrappt und weitergeworfen.

Für fachlich gültige Fehlersituationen gibt es die unchecked Exception „BusinessException“, bei Bedarf auch als Oberklasse. Durch die Verwendung dieser Exception oder einer Ableitung davon werden fachliche Exceptions für das Monitoring kenntlich gemacht.

Namensstandard: Der Name von Exceptions endet mit „…Exception“

# Interfaces

## Klassen:

Namenstandard: Es wird ein gewöhnlicher fachlicher Klassenname verwendet, kein „I-Präfix“

Z.B. „RoomService“

## Projekte:

Projekte mit Interfacedefinitionen und Entitäten erhalten das Postfix „-api“ und einen eigenen vollständig beschreibenden Namen.

Z.B. „conference-speaker-api“

# Umgebung