



Conference-System

Patrick Robinson

Bastian Mager



Inhaltsverzeichnis

- Einleitung / Motivation
- Kontextsicht des Systems
 - Use-Cases
 - Kontextabgrenzung
- Bausteinsicht des Systems
 - Komponentenschichten
 - Verteilungssicht des Systems
 - Komponentenentwurf
 - Klassendiagramm der Entities
- Konzept (Page-Flow)
- Entwurfsentscheidungen
- Demo / Quellcode-Vorstellung

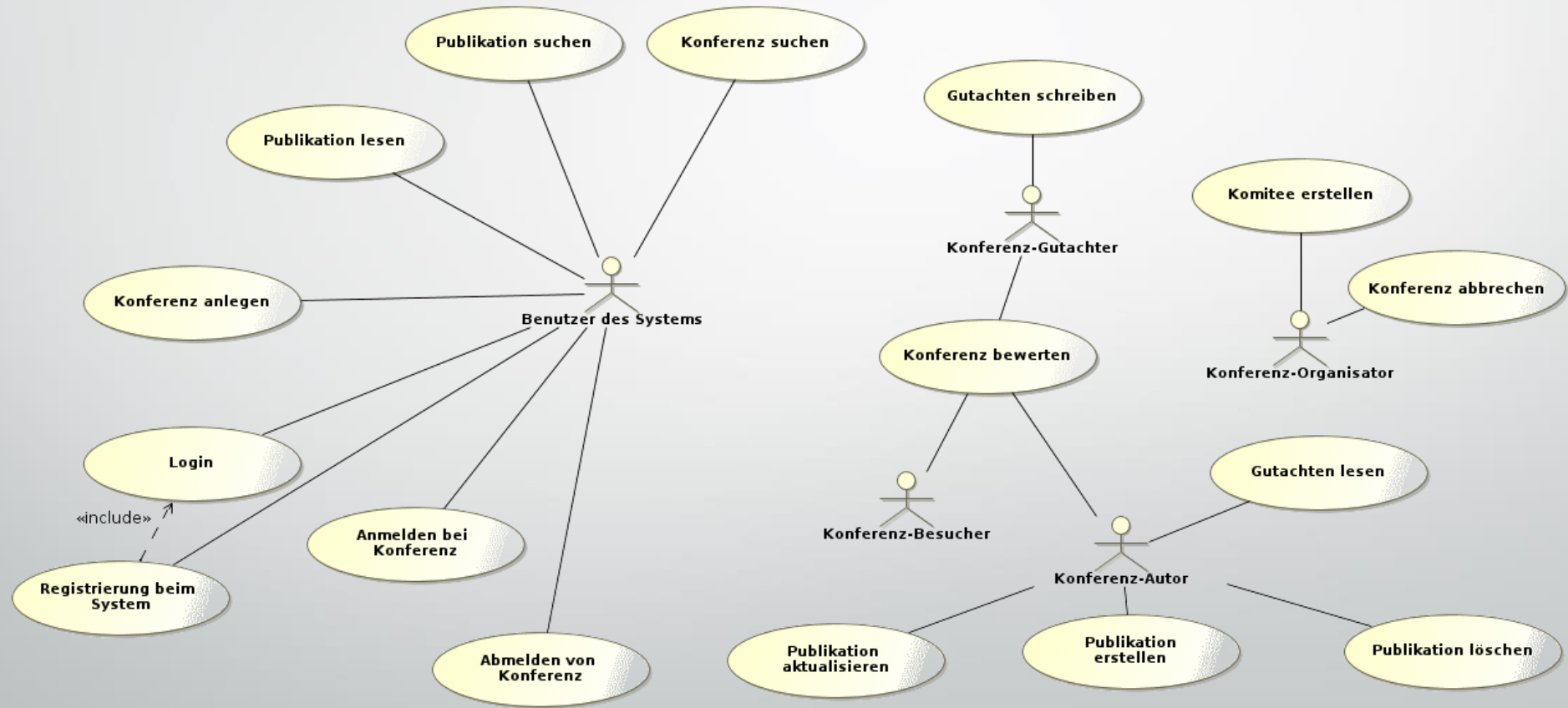


Einleitung / Motivation

- Aufgabenstellung
 - Entwicklung einer Konferenzverwaltungssoftware
- Rahmenbedingungen
 - Dokumentation mit arc42
 - Modellierung nach 4-Sichten
 - Entwicklung auf Basis gelernter Techniken und Methoden
- Technische Anforderungen
 - Java EE 7 (EJB, CDI)
 - Web-Technologien (JSF, Servlet)
 - Deployment auf Glassfish

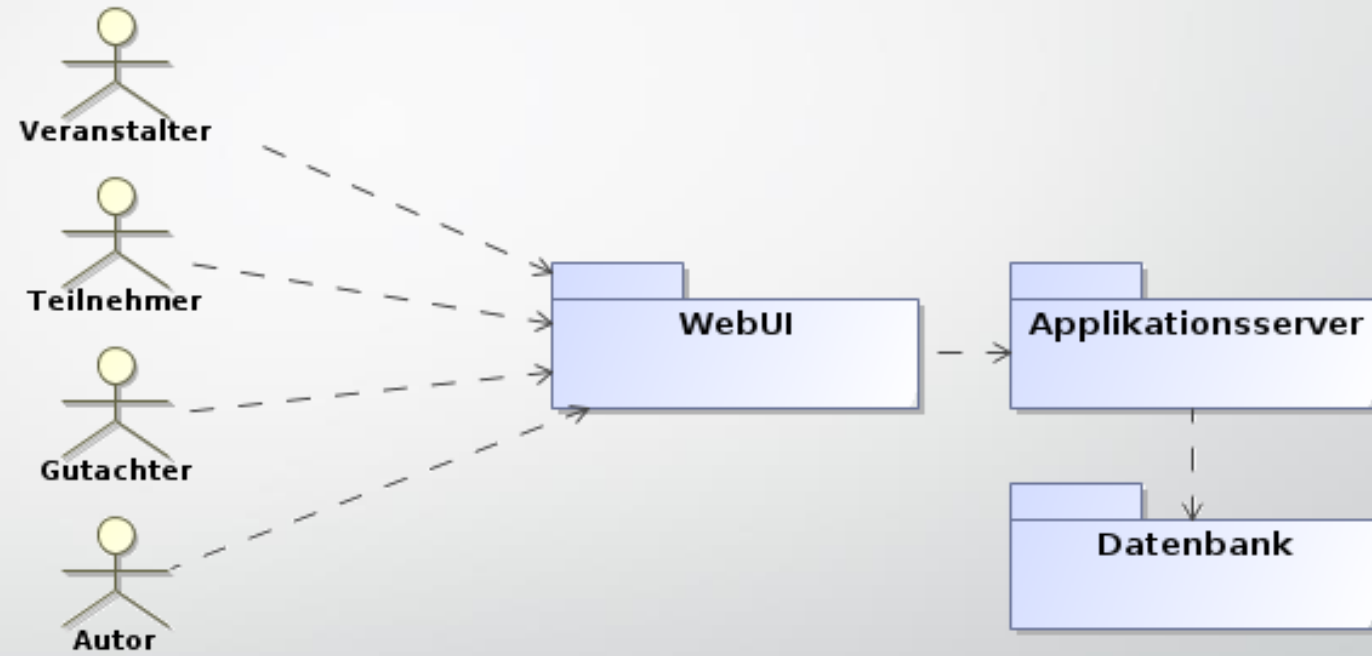


Use-Cases





Kontextabgrenzung



Komponentenschichten

Glassfish

View (JSF xhtml-Pages)

Controller (JSF Managed Beans)

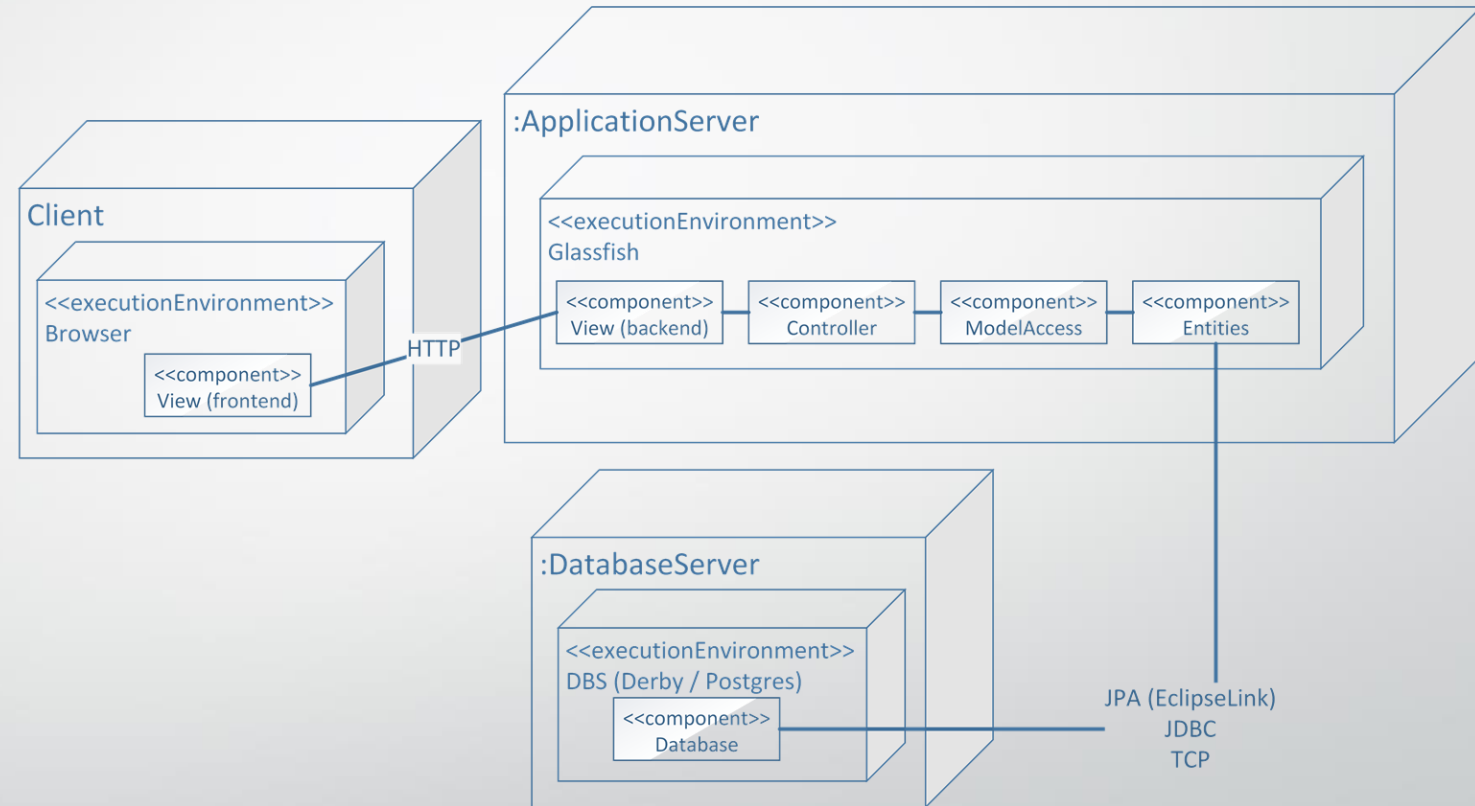
Model Access (EJBs)

Entities (JPA)

Persistence DB (Derby / Postgres)

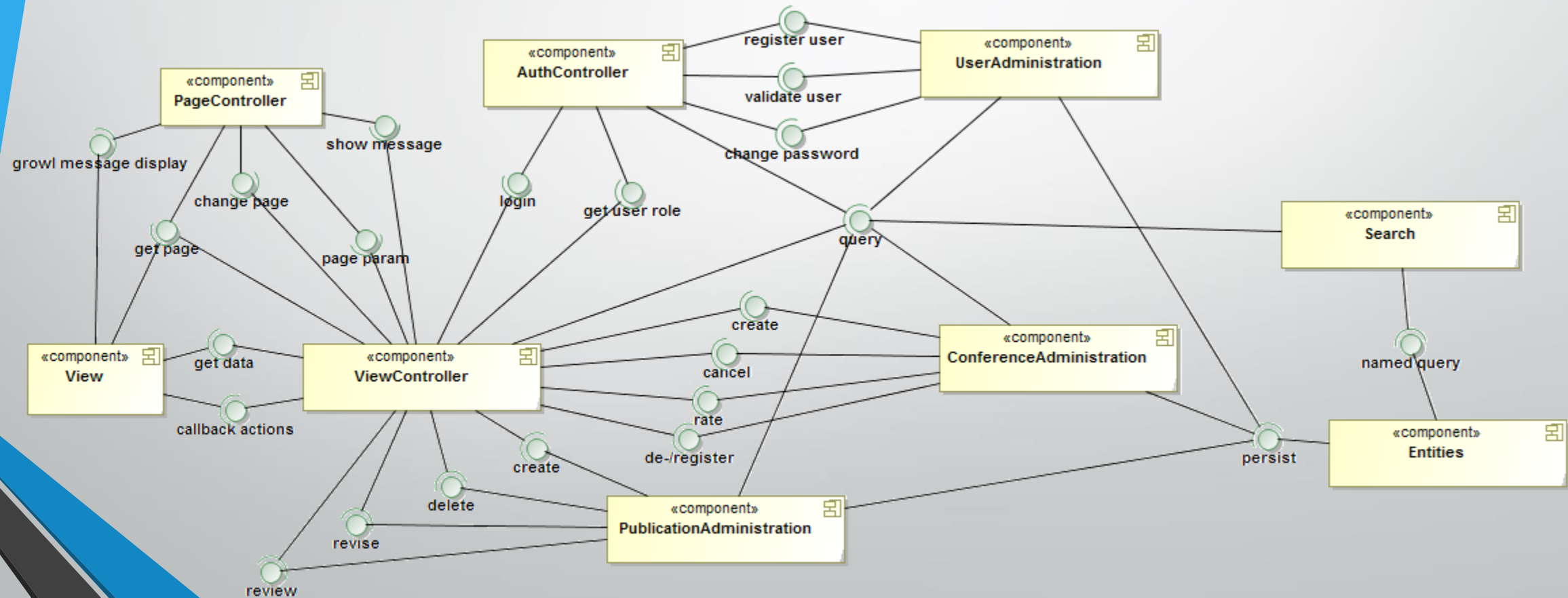


Verteilungsdiagramm



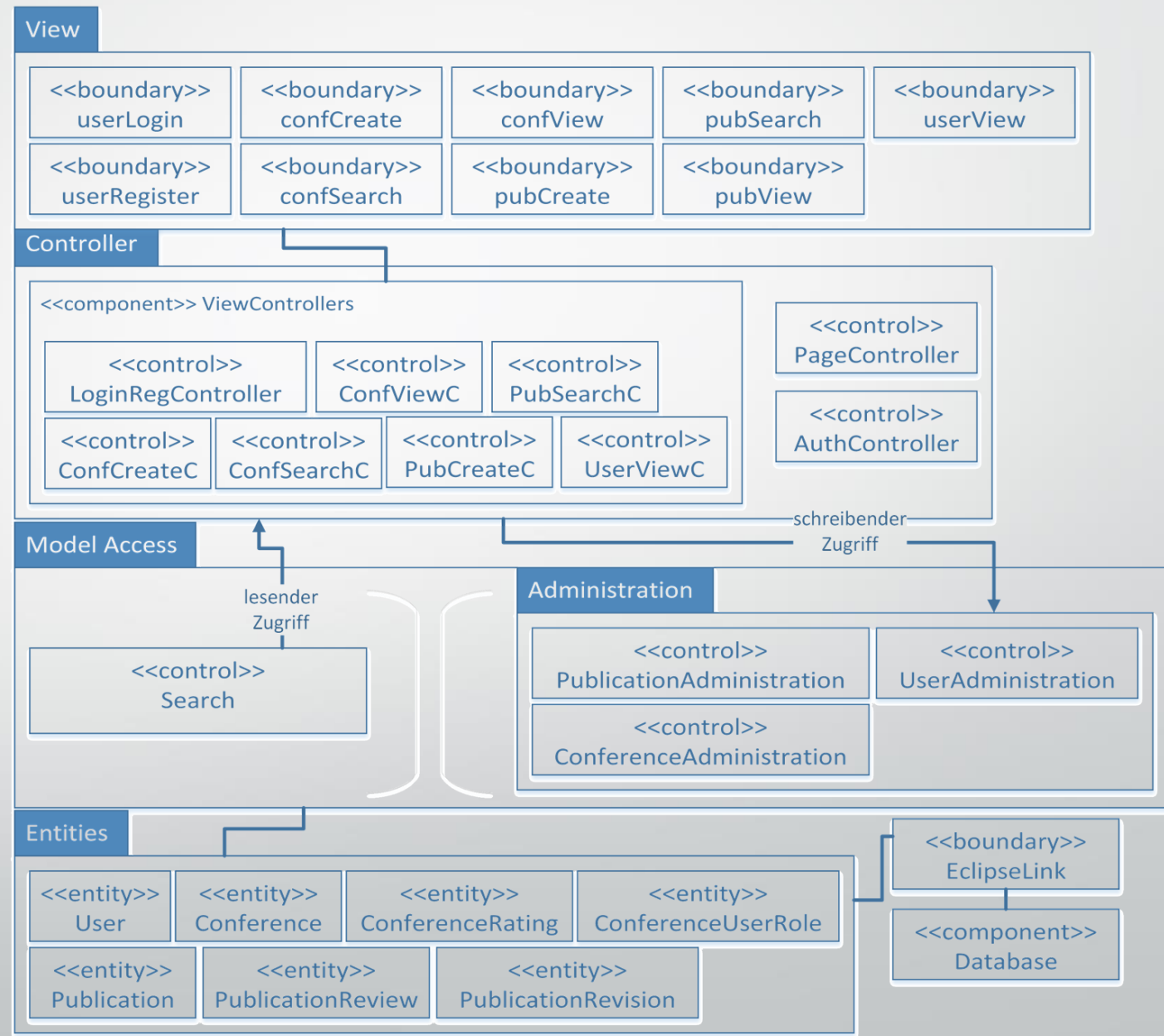


Komponentenentwurf



Komponentenschichten

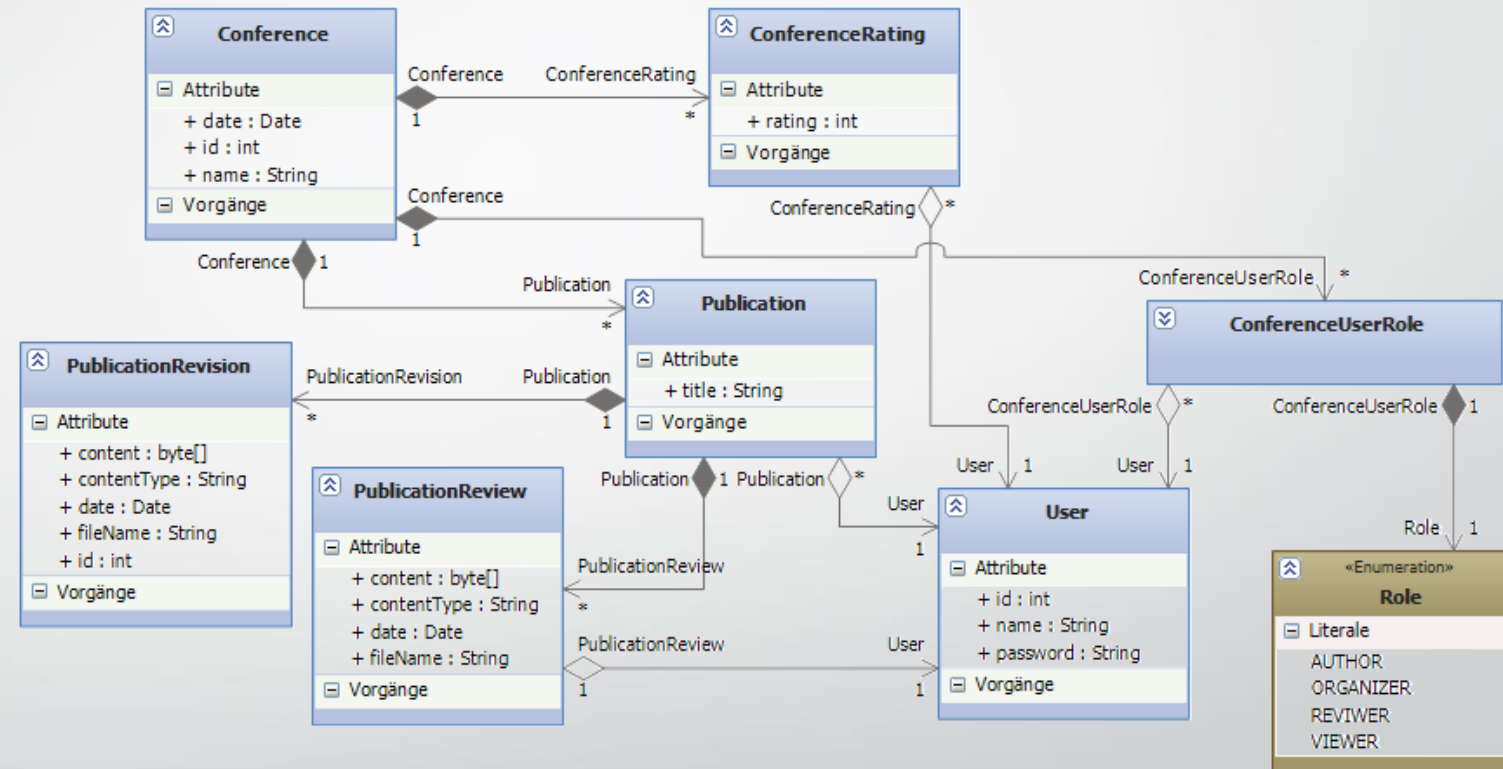
- Lesender und schreibender Zugriff streng getrennt (CQRS)
- Relevante Daten von Entities in DTOs kopieren





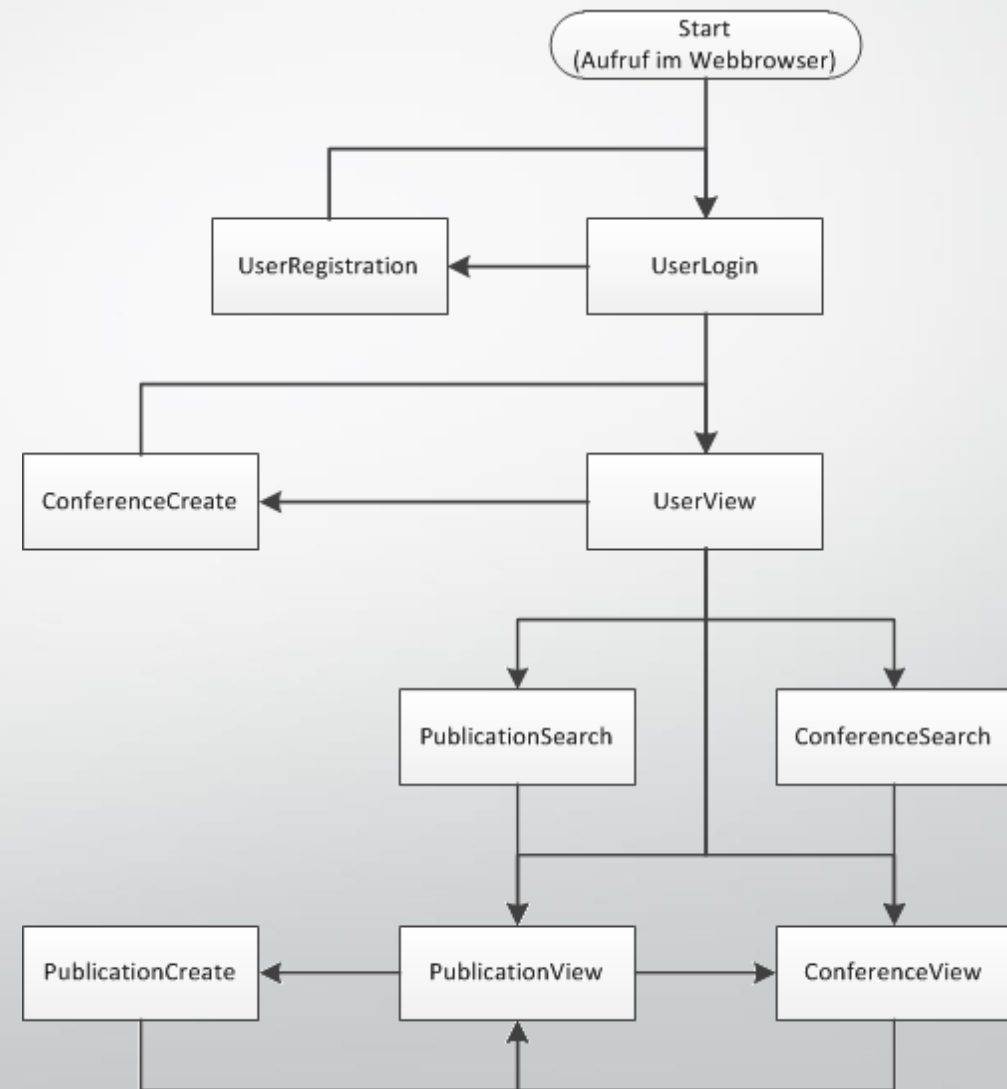
Klassendiagramm der Entities

- Anemic Domain Model
 - Keine Methoden (außer Getter und Setter)
 - Verarbeitende Methoden befinden sich im Model Access





Page-Flow





Entwurfsentscheidungen

- Rollenmodell
 - Rolle für jeden Nutzer **pro Konferenz**
 - Keine feste Rolle für gesamte Applikation (Admin, User, ...)
 - Jeder Nutzer (im Prinzip) gleich berechtigt
- Authentifizierung / Autorisierung
 - Rollenverwaltung von Jave EE / Glassfish vergibt jedem Nutzer eine Rolle
 - Glassfish-Security daher als ungeeignet betrachtet und nicht verwendet
 - Eigene Authentifizierung mit Session-Scope-Bean (CDI) und entsprechendem Datenmodell umgesetzt



Demo / Quellcode-Vorstellung

