JavaEE Grundlagen

FWP Aktuelle Technologien zur Entwicklung verteilter Java-Anwendungen

EINFÜHRUNG IN DIE JAVA EE-PLATTFORM

Die Java EE Plattform

- Java EE Spezifikation definiert
  - ein Programmiermodell für Applikationen
  - die Eigenschaften einer Laufzeitumgebung für Applikationen (Application Server)
- Hersteller liefern konkrete Implementierungen
- Fokus auf server-seitige Applikationen mit web-basierter Benutzeroberfläche
- Aktuelle Version: JavaEE 7

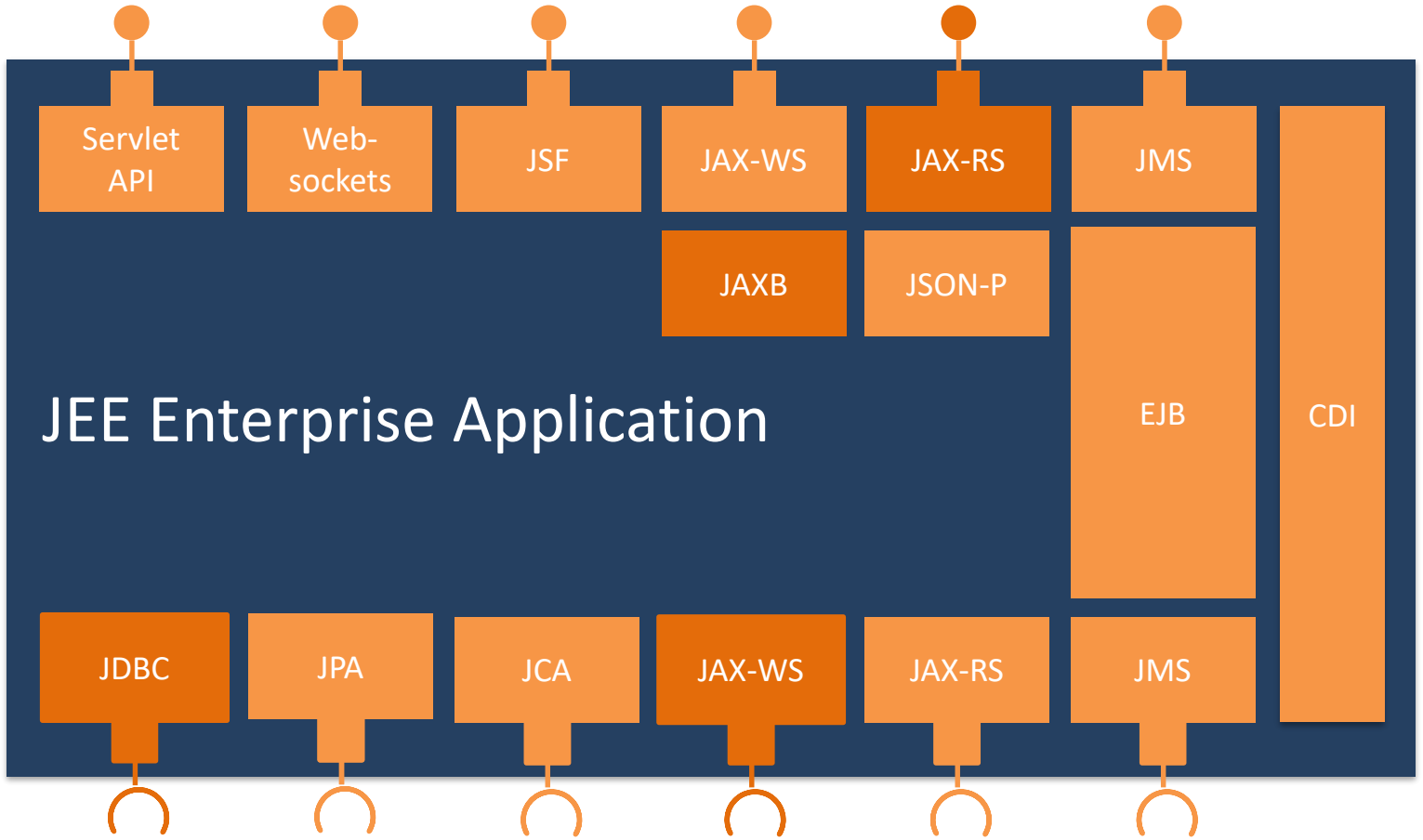
Ziele der Java EE Plattform

- Standardisierter Applikationscontainer mit essentiellen Infrastrukturdiensten
  - Remoting, Transaktionen, Security, Persistenz
- Leichte und schnelle Entwicklung von portablen, serverseitigen Applikationen
  - POJO-basiertes Modell, Annotations, Convention over Configuration, Dependency Injection, AOP
- Flexibler Technologiestack (Profiles)
- Ausgelegt auf Erweiterbarkeit

Profiles – Tailoring der Java EE Plattform

- Profile ermöglichen flexiblen Technologiestack
- Java EE kommt mit zwei Profilen
  - Full Profile = komplette Laufzeitumgebung
  - Web Profile = reduzierte Laufzeitumgebung für Web-Applikationen („Tomcat mit EJBs“)

Übersicht JEE Technologien



Servlet API: Support für (HTTP-)Endpunkte

Websockets: Support für Websockets

JSF: Java Server Faces; webbasierte Uis

JAX-WS: Java API for XML Web Services (SOAP)

JAXB: Java Architecture for XML Binding; schon aus Java SE

JAX-RS: Java API for RESTful Web Services (REST)

JSON-P: Java API for JSON Processing

JMS: Java Message Service; Integration von Messaging Systemen

EJB: Enterprise Java Beans; Support für serverseitige transaktionale Komponenten

CDI: Contexts and Dependency Injection

JDBC: Java Database Connectivity; Integration von relationalen Datenbanken aus Java SE

JPA: Java Persistence API; Standard für persistente Java-Objekte

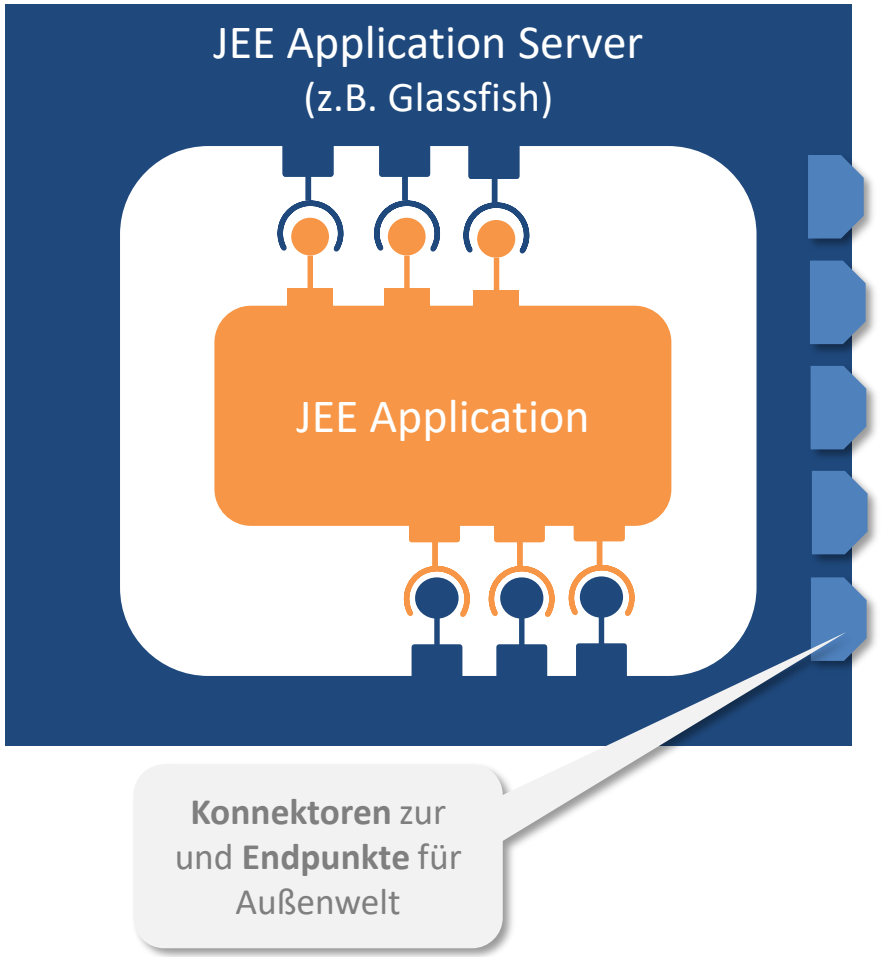
JCA: Java EE Connector Architecture; Integration von transaktionalen Enterprise Information Systemen

Rolle eines JEE Application Servers und dessen Zusammenspiel mit JEE Applikationen

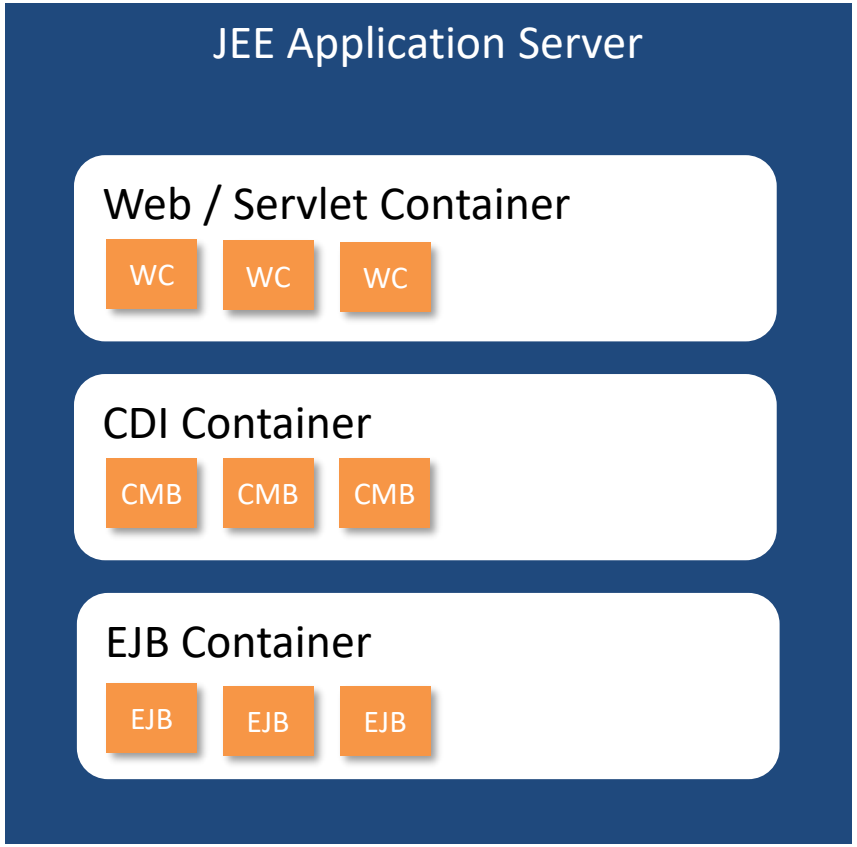
GRUNDLAGEN ZU JEE APPLICATION SERVERN

Container für JEE Applikationen

- JEE Application Server stellt Laufzeitumgebung für JEE Applikationen dar
- JEE Application Server bietet standardisierte Infrastrukturdienste und Ressourcen für Applikationen
- JEE Applikation muss bestimmte Schnittstellen unterstützen, damit der JEE Application Server sie ausführen kann



Container aus Containern



- JEE Applikation bestehen aus Komponenten von bestimmten Typen
- Pro Komponententyp gibt es eigene Container im JEE Application Server:
  - Web-Komponenten (WC) laufen im Servlet Container
  - CDI-Container verwaltet alle CDI managed Beans (CMB)
  - EJB-Container kontrolliert alle Enterprise Java Beans (EJB)
- Container kennen sich untereinander

ANATOMIE EINER JAVA EE-APPLIKATION

Gepackt in Module

Einheitliche Modulstruktur

Modul: austauschbares, komplexes Element innerhalb eines Gesamtsystems [..], das eine geschlossene funktionale Einheit bildet.
(Quelle: Duden)

Funktionseinheit (Functional Unit)

- Geschlossene fachliche Funktionseinheit
- Eigener Namensraum (Package)

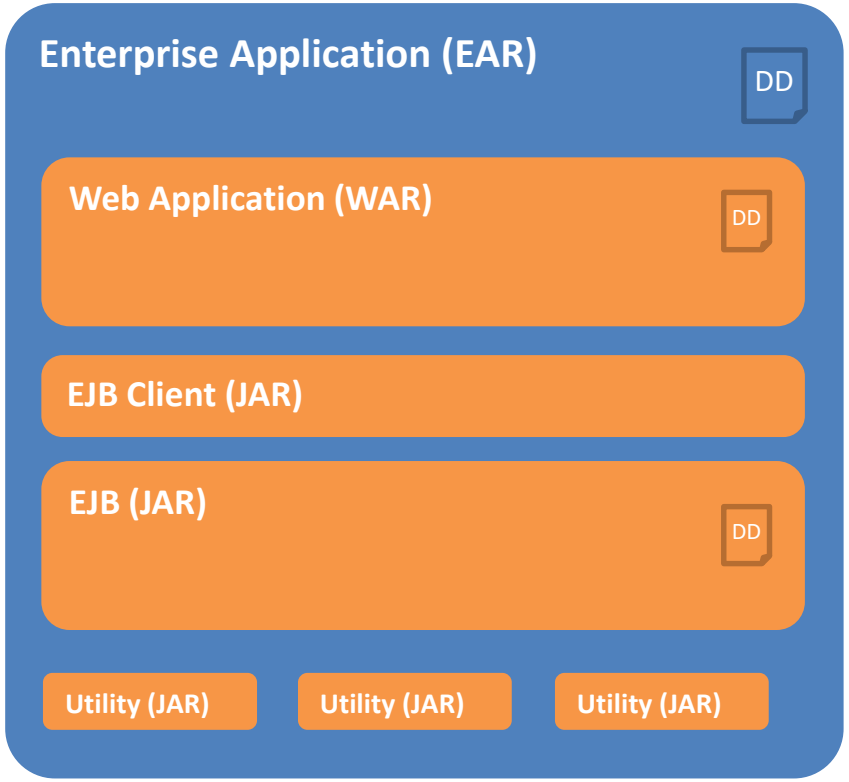
Baueinheit (Build Unit)

- Zusammenfassung von Funktions-einheiten
- Wird durch ein Build-Tool gebaut

Deploymenteinheit (Deployment Unit)

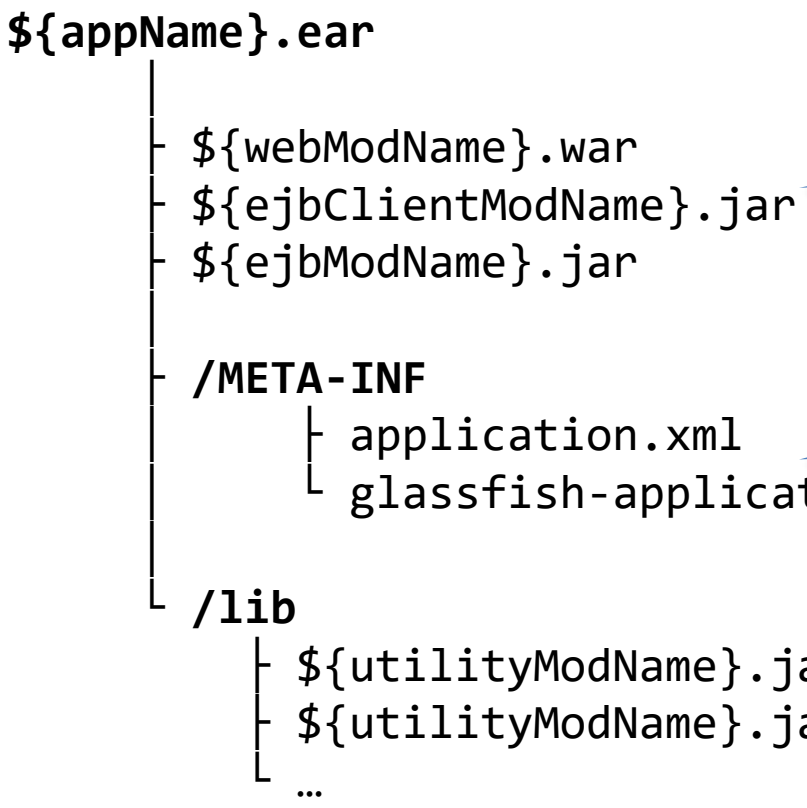
- Zusammenfassung von Baueinheiten
- Kann einzeln auf einer Laufzeit-umgebung bereit-gestellt und aus-geführt werden

Module einer Full Profile Applikation



- Eine typische Java EE Applikation wird durch ein Enterprise Application Archive (EAR) repräsentiert
  - ZIP-Datei mit standardisiertem Inhalt
  - In sich vollständige Installationseinheit
- Ein EAR besteht im allgemeinen
  - aus einer Webapplikation (WAR), die die Benutzeroberfläche repräsentiert
  - aus einem EJB JAR mit Enterprise Java Beans, die die Businesslogik der Applikation repräsentieren
  - aus einem EJB Client JAR, welches die Interfaces zu den EJBs zur Verfügung stellt
  - aus mehreren Utility JARs, die Querschnittsfunktionalität für alle Module zur Verfügung stellt (Frameworks, Security, Logging…) (/lib)

Physische Struktur eines EARs

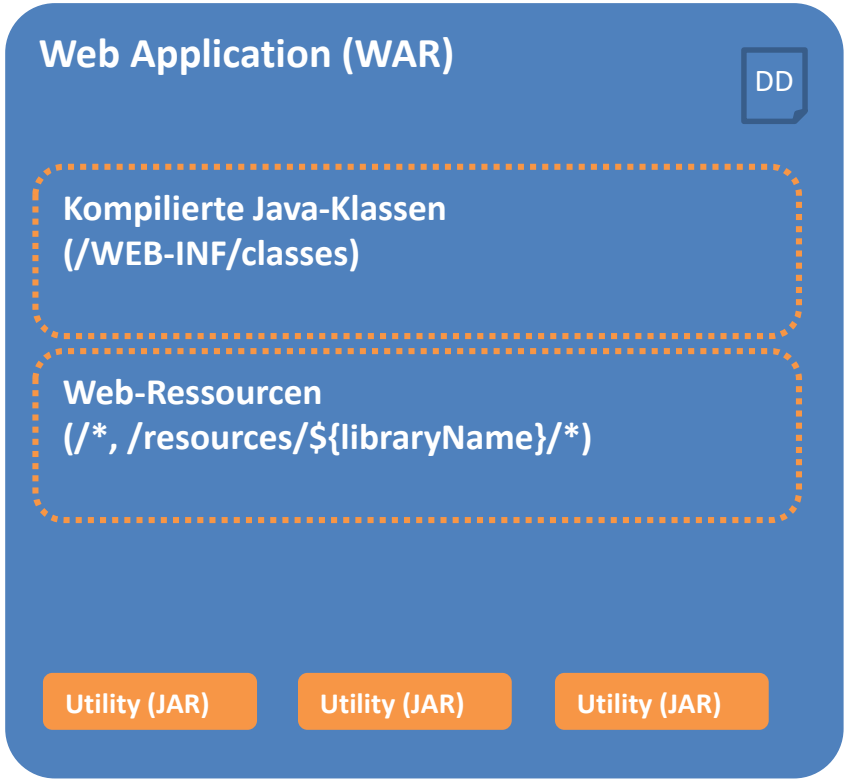


Applikationsmodule: Web Application Module (WAR), EJB Client Module (JAR), EJB Module (JAR)

Deployment Deskriptoren (optional):
application.xml = Standard-Deskriptor
glassfish-application.xml = Hersteller-spezifischer Deskriptor

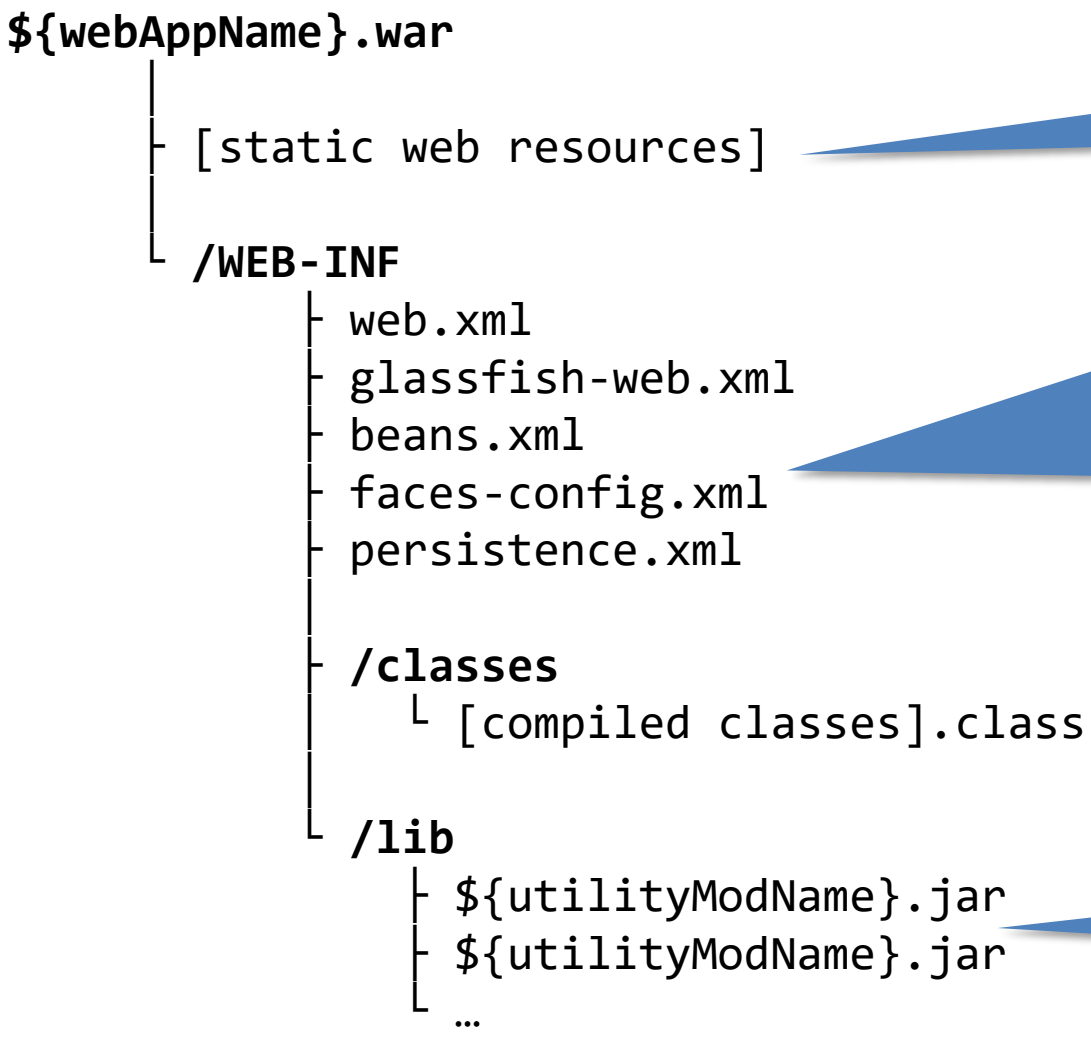
Alle Utility-JARs, die keine Web Application Module, keine EJB (Client) Module sind

Module einer Web Profile Applikation



- Eine typische Java EE Web-Applikation wird durch ein Web Application Archive (WAR) repräsentiert
  - ZIP-Datei mit standardisiertem Inhalt
  - In sich vollständige Installationseinheit
- Ein WAR besteht im allgemeinen
  - aus kompilierten Java-Klassen (/WEB-INF/classes)
  - aus Web-Ressourcen wie (X)HTML, Grafiken, CSS, JavaScript (/*)
  - aus JSF-Ressource-Bibliotheken (/resources/${libraryName}/*
  - aus mehreren Utility-JARs (/WEB-INF/lib)
- Enterprise Java Beans (EJB) werden auch unterstützt (EJB Lite = EJB ohne Remote Interfaces)

Physische Struktur eines WARs

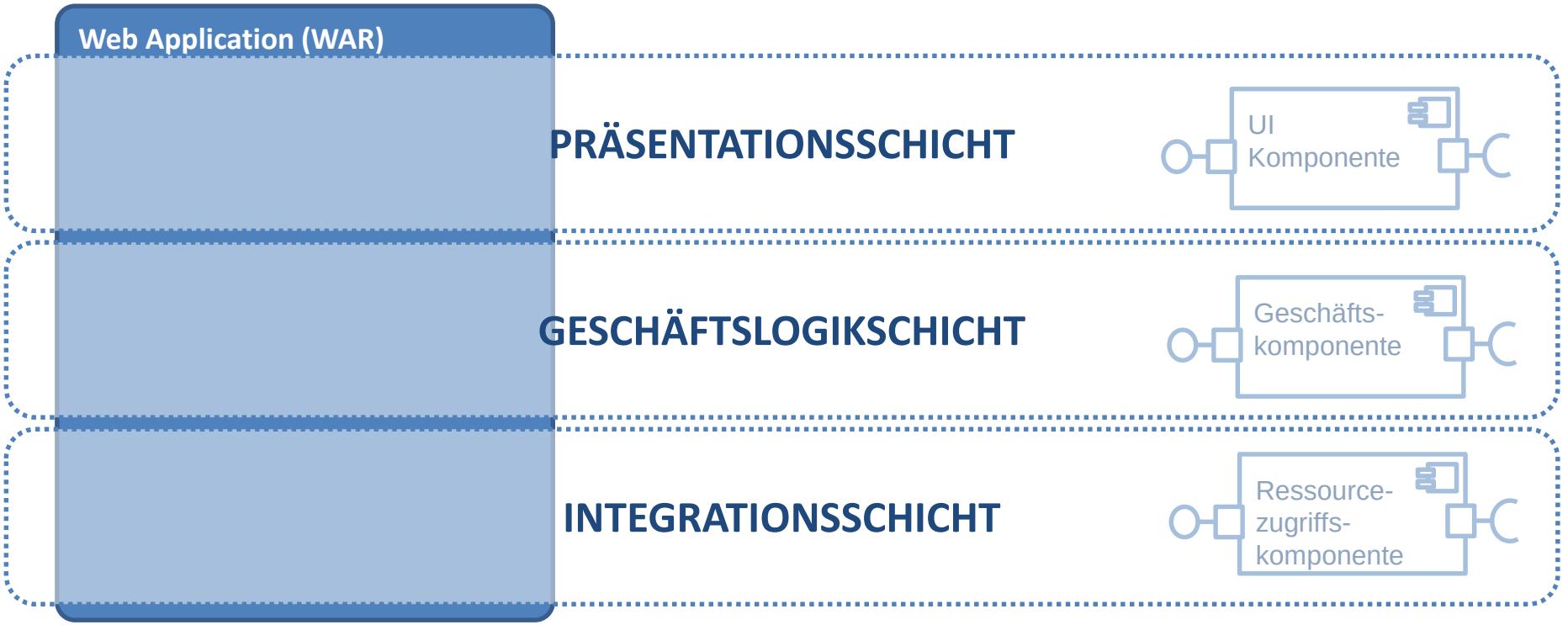


Statische Web-Ressourcen (HTML, CSS, JS, Bilder) mit beliebiger Verzeichnisstruktur

Deployment Deskriptoren und Konfigurationsdateien (optional):
web.xml = Standard-Deskriptor
glassfish-web.xml = Hersteller-spezifischer Deskriptor
beans.xml = CDI-Konfigurationsdatei
faces-config.xml = JSF-Konfigurationsdatei
persistence.xml = JPA-Konfigurationsdatei

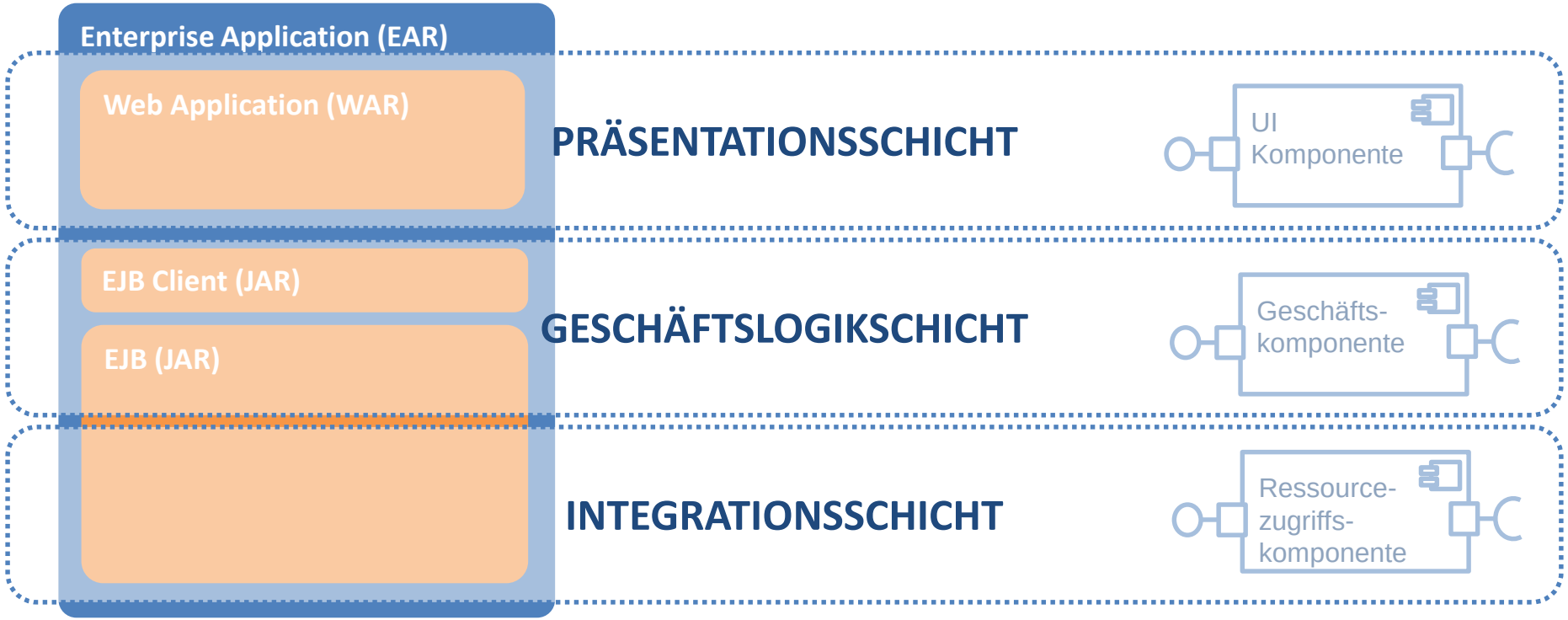
Alle Utility-JARs

Aufbau von Leichtgewichten



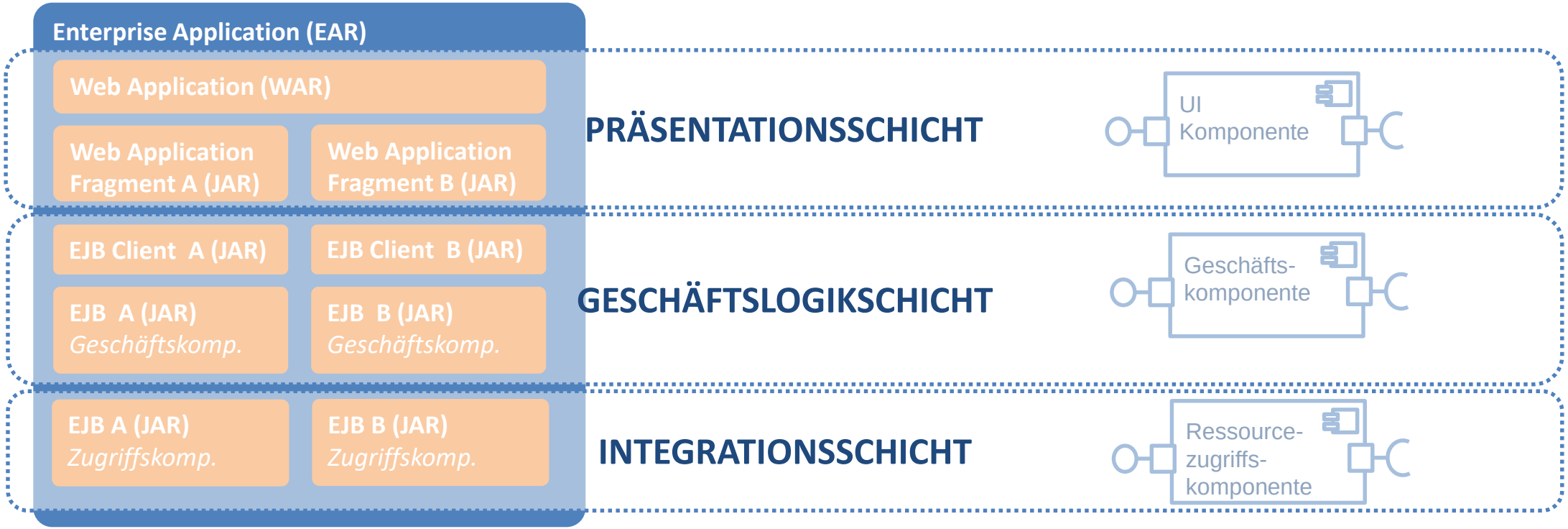
- Alle Komponenten befinden sich im Web-Modul
- Schichtentrennung ausschließlich über getrennte Packages
- Web-Profile Application-Server ausreichend

Aufbau von Mittelgewichten



- Alle UI-Komponenten aus der Präsentationsschicht befinden sich im Web-Modul
- Die öffentlichen Interfaces der Geschäftskomponenten und Domänenobjekte befinden sich im EJB-Client-Modul
- Alle Geschäftskomponenten aus der Geschäftslogikschicht befinden sich im EJB-Modul
- Die Ressourcezugriffskomponenten aus der Integrationsschicht befinden sich im EJB-Modul

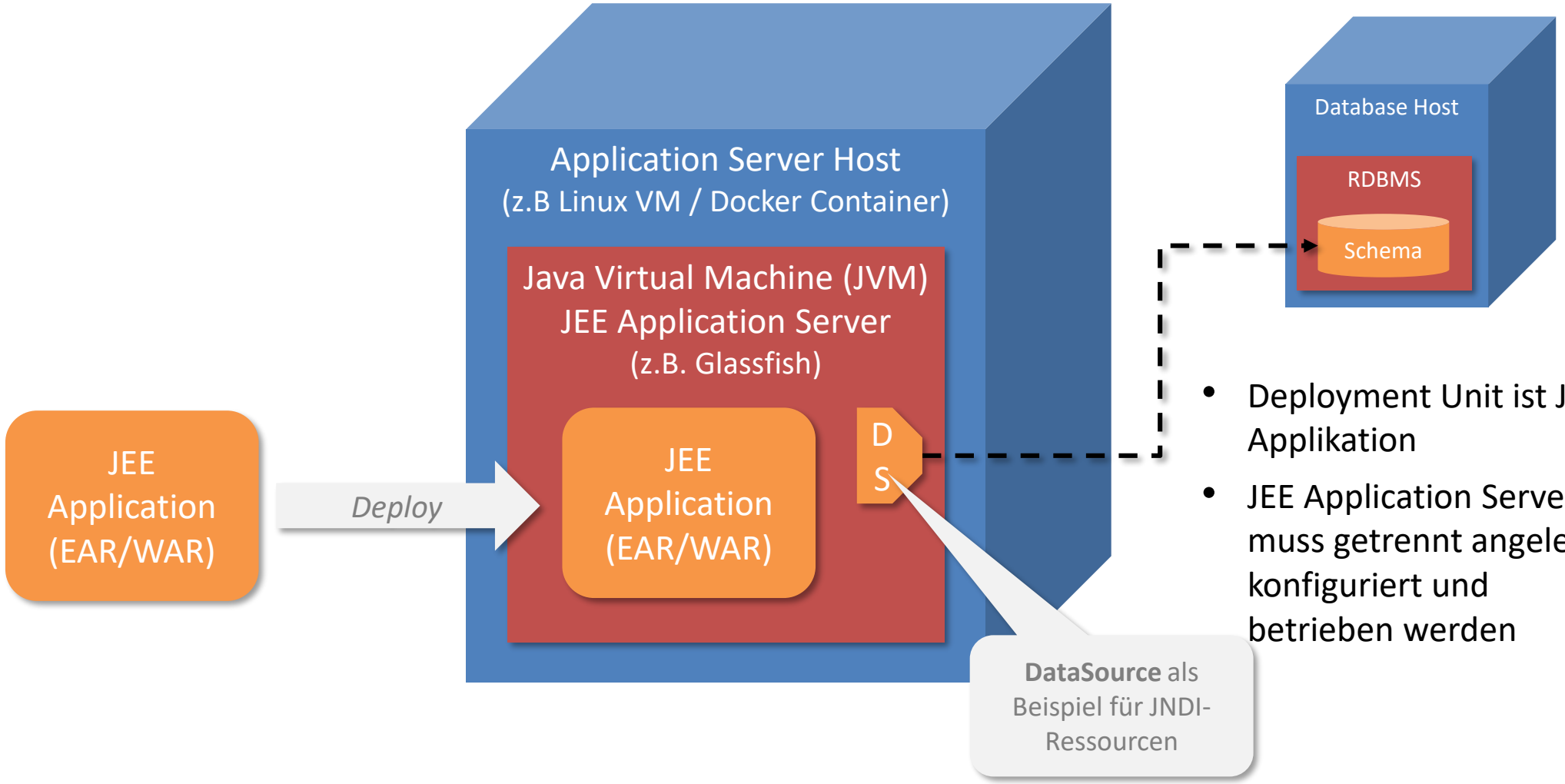
Aufbau von Schwergewichten



- Wie Variante Mittelgewicht, aber alle Datenzugriffskomponenten in eigenem EJB-Modul
- Mehrere Module pro Schicht möglich (getrennte vertikale fachliche Module über alle Schichten)
- Fachlich getrennte Benutzeroberflächen als physisch getrennte Web Application Fragmente

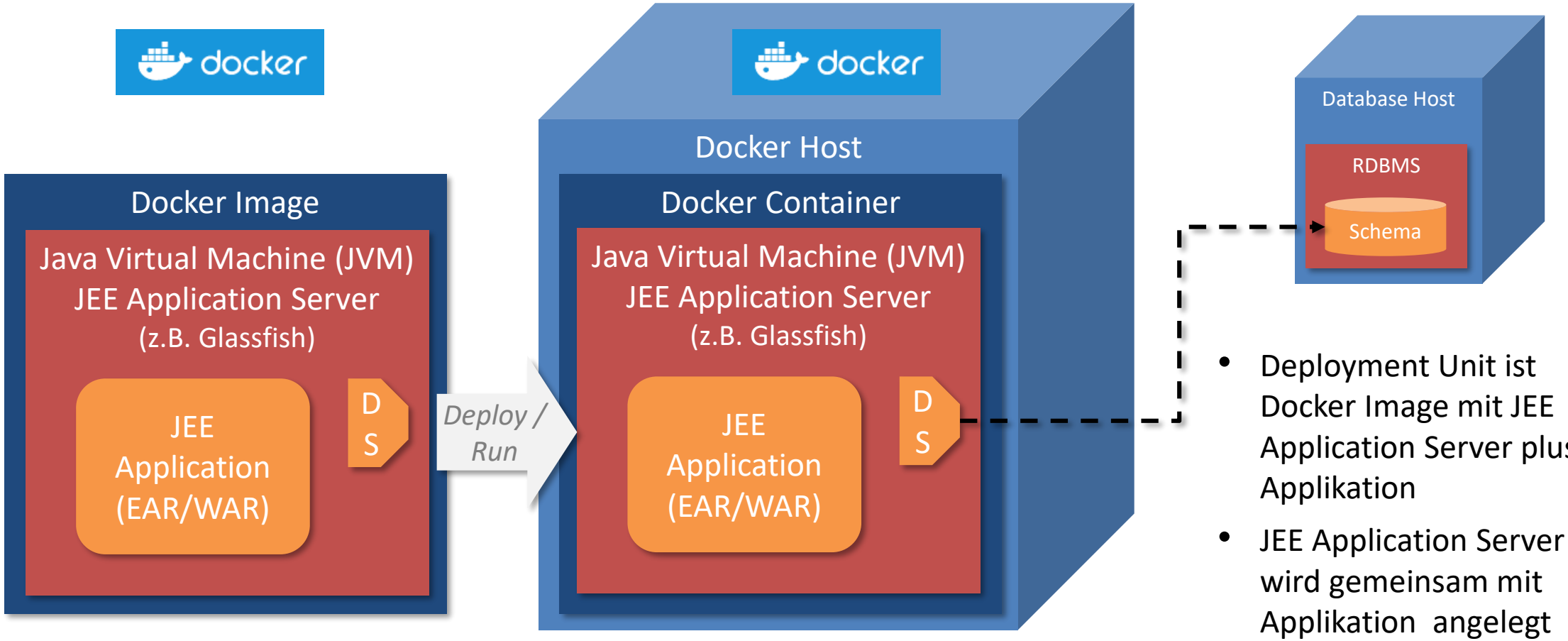
BETRIEB VON JEE-APPLIKATIONEN

Betrieb einer klassischen JEE App



- Deployment Unit ist JEE Applikation
- JEE Application Server muss getrennt angelegt, konfiguriert und betrieben werden

Betrieb einer containerisierten JEE App



- Deployment Unit ist Docker Image mit JEE Application Server plus Applikation
- JEE Application Server wird gemeinsam mit Applikation angelegt und konfiguriert

Fragen?

ANHANG

Quellen

- Eric Jendrock et. al.: **The Java EE 7 Tutorial**
  http://docs.oracle.com/javaee/7/tutorial/
  Oracle September 2014
- Marcus Schießer, Martin Schmollinger: **Workshop Java EE 7**
  dpunkt.verlag 2015, 2. Auflage; ISBN 978-3-86490-195-9
- Adam Bien: **Real World Java EE Patterns: Rethinking Best Practices**
  press.adam-bien.com September 2012; ISBN 978-0-300-14931-6
- Eric Evans: **Domain Driven Design:**
  Tackling Complexity in the Heart of Software
  Addison Wesley 2004; ISBN 0-321-12521-5
- Rod Johnson: **J2EE Design and Development**
  Wrox Press 2002; ISBN 1-86100-784-1
- Martin Fowler: **Patterns of Enterprise Application Architecture**
  Addison Wesley 2003; ISBN 0-321-12742-0

Kontakt

Michael Theis
Lehrbeauftragter Hochschule München

email michael.theis@hm.edu
mobile + 49 170 5403805
web http://www.tschutschu.de/Lehrauftrag.html