3.– 6. September 2012
in Nürnberg

Herbstcampus

Wissenstransfer
par excellence

Lucky Seven

Java Enterprise Edition 7

Wolfgang Weigend

ORACLE Deutschland B.V. & Co. KG

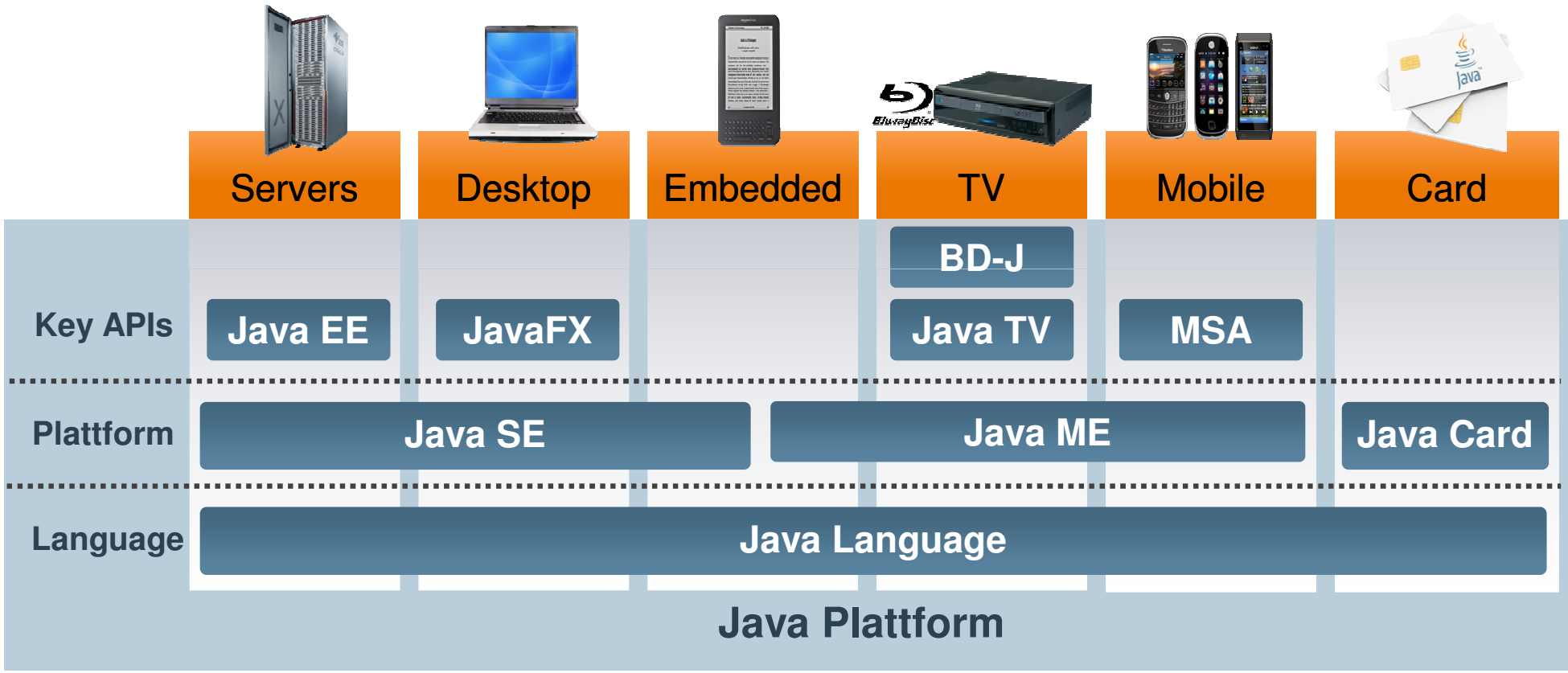
Lucky Seven

Java Enterprise Edition 7

Java EE on the way to PaaS

Wolfgang Weigend
Sen. Leitender Systemberater
Java Technologie und Architektur

Die Java Plattform



ORACLE

Agenda

- Java EE 6
- Java EE 7/8
- Demo
- Zusammenfassung
- Q & A

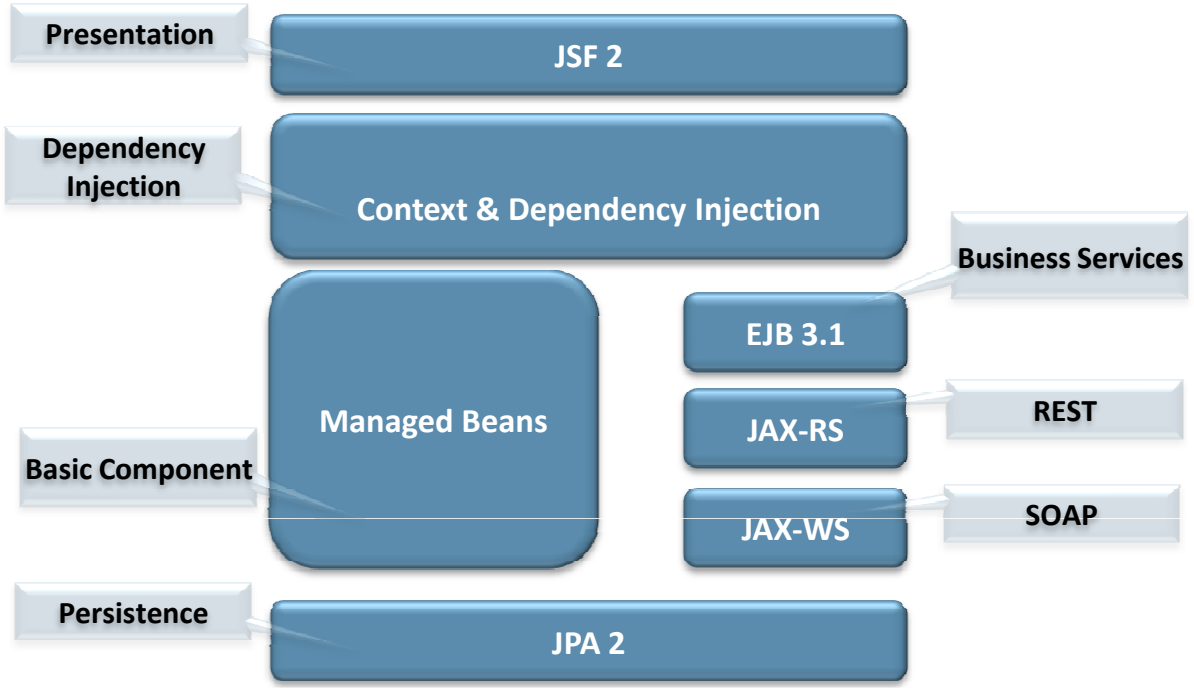
Java EE 6 die aktuelle Ausgabe der Java EE Plattform seit Dez. 2009

Wesentliche Vorteile vom Java EE 6 Standard:

- Weniger Abhängigkeiten
- Konvention vor Konfiguration: Reduzierung der Notwendigkeit von XML-Konfigurationen
- Schlankere standardisierte APIs
- Plain Old Java Objects (POJO): ein einfaches und leichter testbares Programmiermodell
- Annotationsgestütztes Programmiermodell: dekorieren und injizieren
- Weniger oder überhaupt keine Deployment-Deskriptoren
- Freie Wahl der Entwicklungsumgebung
- Herkömmliche API für erfahrene Anwender
- Migration von Java EE 5 nach Java EE 6

Java EE 6 Technologie

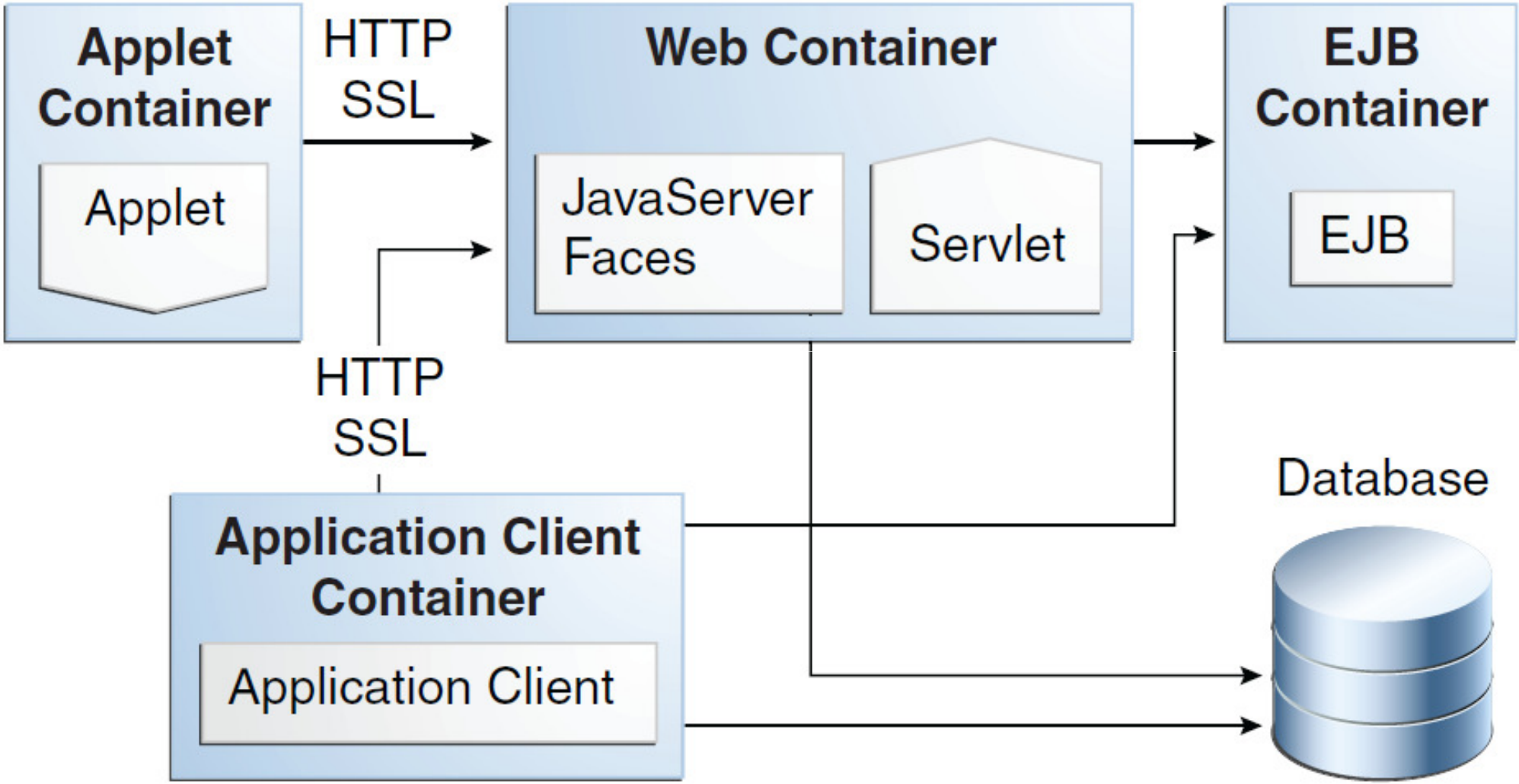
Flexibilität, Erweiterbarkeit und Entwicklereffizienz



- **Neue Funktionalität**
  - Web Profile
  - Erweiterbarkeit/Plugin-fähig
  - Context und Dependency Injection, Managed Beans
  - REST, Validation
- **Erweiterte APIs**
  - EJB 3.1, JPA 2.0, JSF 2.0, Servlet 3.0
- **Verbesserte Nutzbarkeit und vereinfachte Konfiguration**
  - Konventionen über Konfigurationen (weniger XML)
  - Annotationen-basiertes Programmiermodell (Decorate und Inject)
  - POJO Modell - Managed Beans zum Test von Komponenten
  - Bessere Testmöglichkeiten und Zusammenspiel mit Betriebssystem-Werkzeugen
    - Ant / Maven / Hudson
  - Keine Deployment Deskriptoren (optional)

ORACLE

Java EE 6 API's



ORACLE

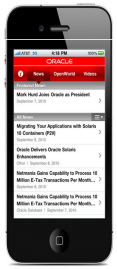
Benutzerumgebung und Client-Interaktion

- **HTML 5 Clients**
  - Verwendung von bi-direktionaler Kommunikation zur Nutzung Event-gesteuerter Anwendungen
  - Umfassendes, standardisiertes Client-Side-Programmiermodell:
    - APIs, schnelles JavaScript und standardisierte offline-Fähigkeiten
- **Cloud / PaaS**
  - Vorhandene Building Blocks: WebSockets (Comet), JSON (JAX RS), RESTful WS
  - Fehlende Benachrichtigung bei Datenänderungen, asynchrone Infrastruktur, …
- **Programmier Modell**
  - Mehr als die Summe aller Einzelteile
  - Web-native, bi-directionales Binding zur Cloud
  - Vereinigung von Java ME, Java SE und Java EE

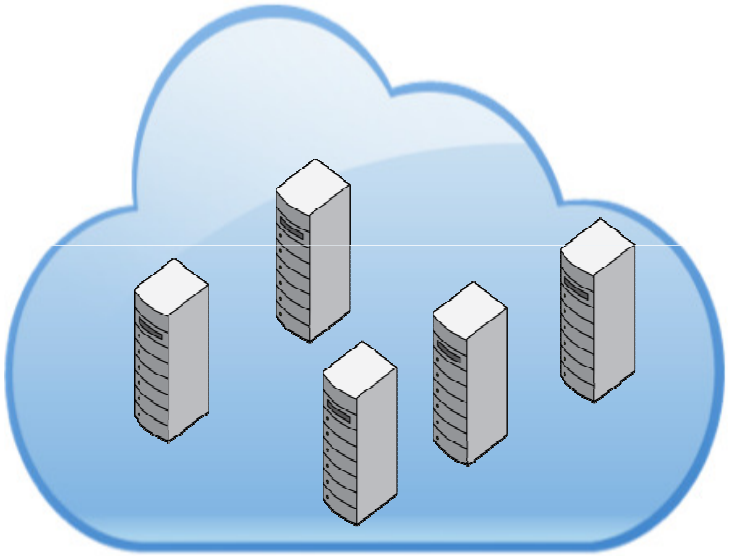
Ganzheitliche Betrachtung zur Unterstützung von dynamische Rich-Clients

HTML 5 Browser

HTML & Java hybride Anwendungen



JSON over WebSocket



Java Anwendungen

Java EE und PaaS

ORACLE

Wie sich die Entwicklung durch die Wolke schrittweise verändert

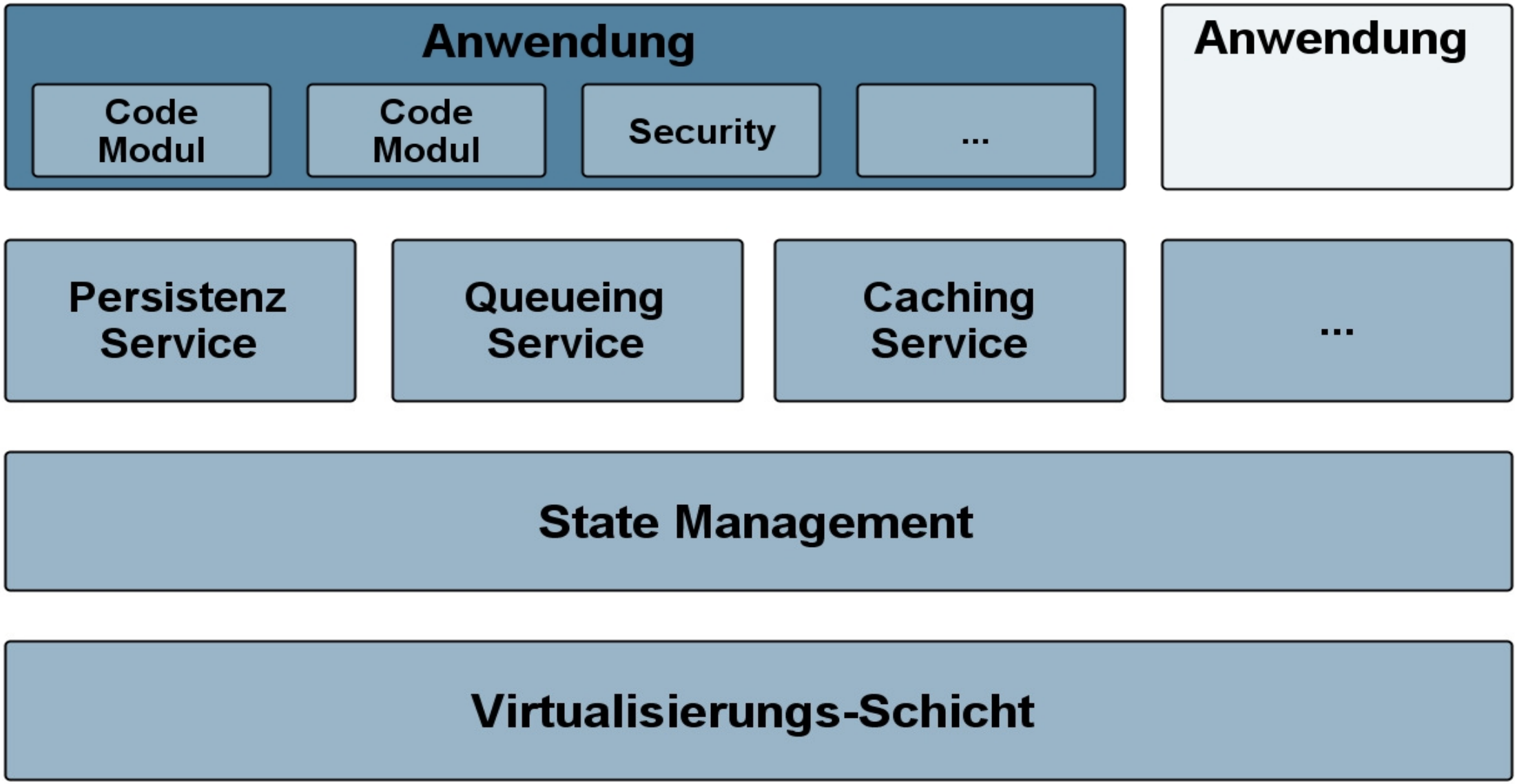
- **Entwickler für Unternehmensanwendungen wollen Cloud-Lösungen von der eigenen IT-Abteilung**
  - IaaS als neuer “Self-Service Data Center”
  - Unmittelbar, On-Demand Provisioning
  - Hosted, sichere Cloud Services

- **Virtualisierung ist ein wertvoller technischer “Building Block”, aber keine Plattform**
- **Entwickler schauen nach einem PaaS Standard für die nächste Generation Cloud-basierter Anwendungen**
  - Die Java EE Plattform bietet Fähigkeiten für PaaS an
- **Java EE hat bereits vergleichbare Herausforderungen für die IT gelöst**

Java EE 7 und PaaS-Ausrichtung

- **Definition neuer Rollen für die Plattform**
  - Anpassung und Adaption vom PaaS-Modell
- **Einführung von Metadaten für:**
  - Service-Provisionierung und Konfiguration
  - Service-Qualität (QoS), Elastizität
  - Gemeinsam genutzte Applikationen und Ressourcen
  - Konfigurationsänderung und Anpassung (Customization)
- **Hinzufügen neuer API‘s für PaaS-Umgebungen**
  - JAX-RS Client API
  - Caching API
  - State Management
  - JSON, ..
- **Erweiterung existierenden API’s für Mandantenfähigkeit als Dienst**

Java EE 7 Architektur mit PaaS-Ausrichtung



Java EE 7 Services

- PaaS Anwendungen konsumieren Services
  - Persistenz, Queueing, Caching, …
- Service Metadaten ermöglichen einfache Nutzung beim Deployment in PaaS

```
@DataSourceDefinition(
    name="java:app/jdbc/myDB",
    className="oracle.jdbc.pool.OracleDataSource",
    isolationLevel=TRANSACTION_REPEATABLE_READ,
    initialPoolSize=5
)
```


Java EE 7 Services

- PaaS Anwendungen konsumieren Services
  - Persistenz, Queueing, Caching, …
- Service Metadaten ermöglichen einfache Nutzung beim Deployment in PaaS

```
@JMSConnectionFactory (
    name="java:app/myJMSConnectionFactory",
    resourceType="javax.jms.QueueConnectionFactory")

@JMSDestination(
    name="java:app/myQueue",
    resourceType="javax.jms.Queue")
```


Java EE 7 Services

- PaaS Anwendungen konsumieren Services
  - Persistenz, Queueing, Caching, …
- Service Metadaten ermöglichen einfache Nutzung beim Deployment in PaaS

```
@ConnectorService (
    name="java:app/myCustomConnector",
    type="com.extraServices.customConnector.class",
    properties = {…}
)
```


Java EE 7 Services

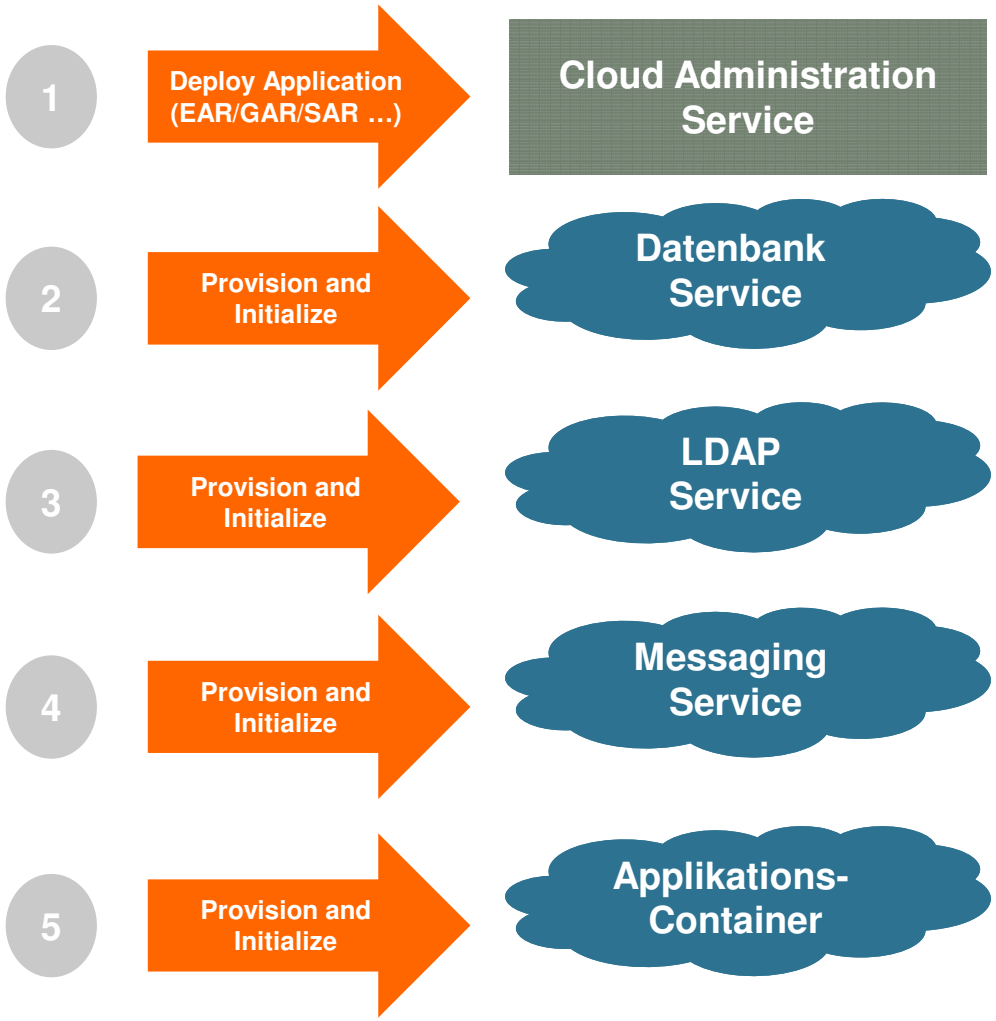
- Erzeugte Entitäten provisionieren, verwalten und überwachen innerhalb der Ablaufumgebung mit Zielrichtung PaaS
- Signifikante Software-Funktionalität die zur Ausführung einer Anwendung notwendig ist
- Typen
  - **Provisioniert:** Installiert, konfiguriert, von der Plattform verwaltet, Anwendungsbezogen oder gemeinsam verwendet
  - **Extern:** Enterprise-existent, Plattform kennt die Konfiguration
  - **Shared:** Mehrfache Verwendung in verschiedenen Umgebungen pro Benutzer, pro System
- Beispiele:
  - Java EE Applikations-Service (mit Transaktions-Konzept JTS/JTA, 2PC)
  - Datenbank
  - Load Balancer

Java EE 7 Services

- Service-Definition
  - Verwendet Metadaten zur Service-Provisionierung und Konfiguration
    - Spezifikation von funktionalen und nicht-funktionalen Service-Charakteristiken
    - Template-Abgleich
    - Explizite Template Spezifikation
- Service-Referenz
  - Repräsentiert die Abhängigkeit der Applikations-Komponente zum Service
  - Explizit: Benutzer-spezifisch durch Deployment Deskriptoren
  - Implizit und Auffinden (Discovery): Information ist im Archiv enthalten

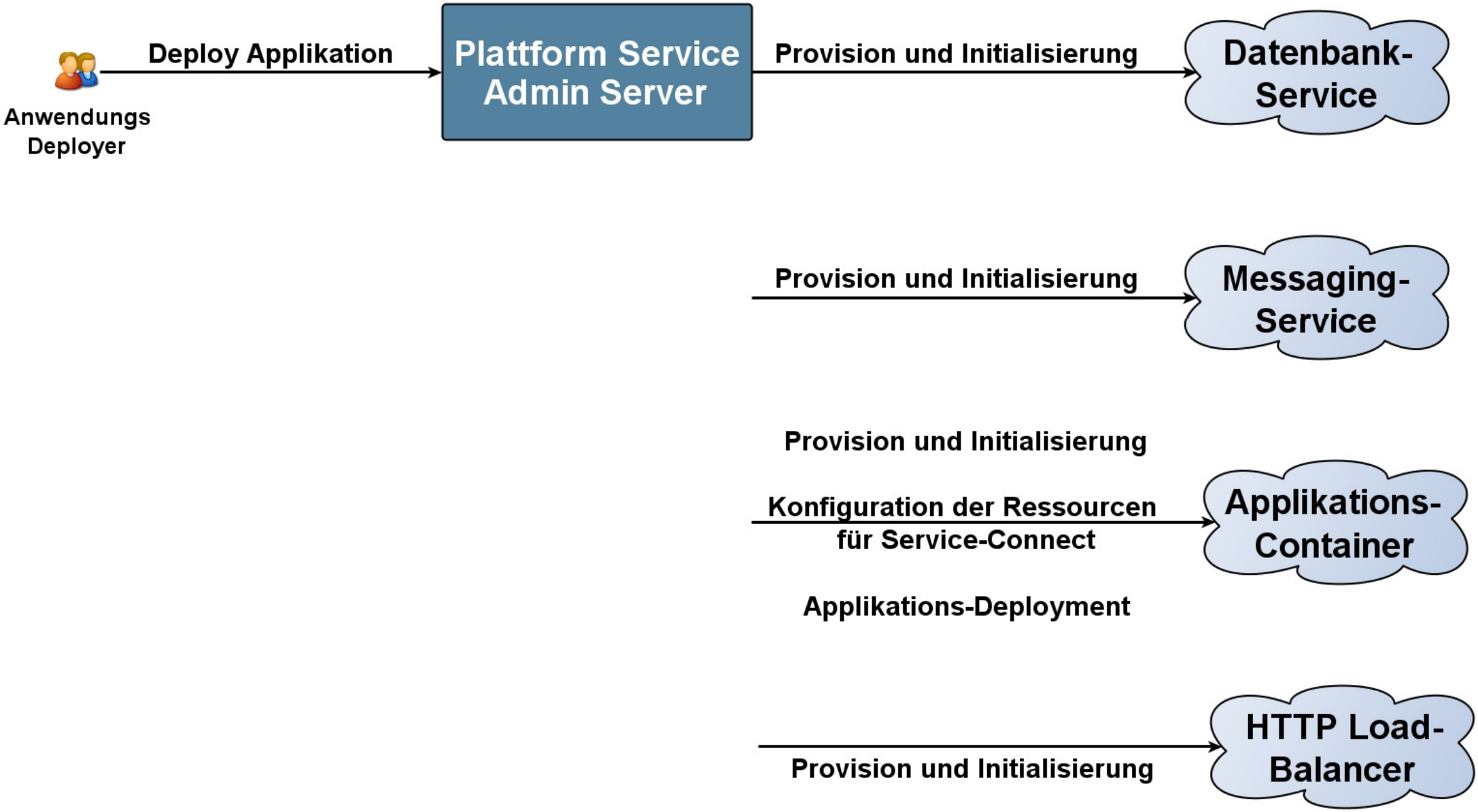
Zielsetzung: Java EE 7 – Provisioning

- **Java EE Vorhandenes Modell**
  - Konfiguration Java EE Ressourcen – JDBC, JMS, etc.
  - Deploy Application Archive (.EAR)
- **Java EE 7 Modell**
  - Auto-Provision-Services von Applikationsabhängigkeiten e.g. Datenbank, LDAP
- **Erweiterbare Deployment-Modelle für Frameworks**
  - Spring, Seam, etc.
- ***Ziel: „Eine Anwendung soll eigenständig in der PaaS-Umgebung einer Cloud-Infrastruktur lauffähig sein.“***

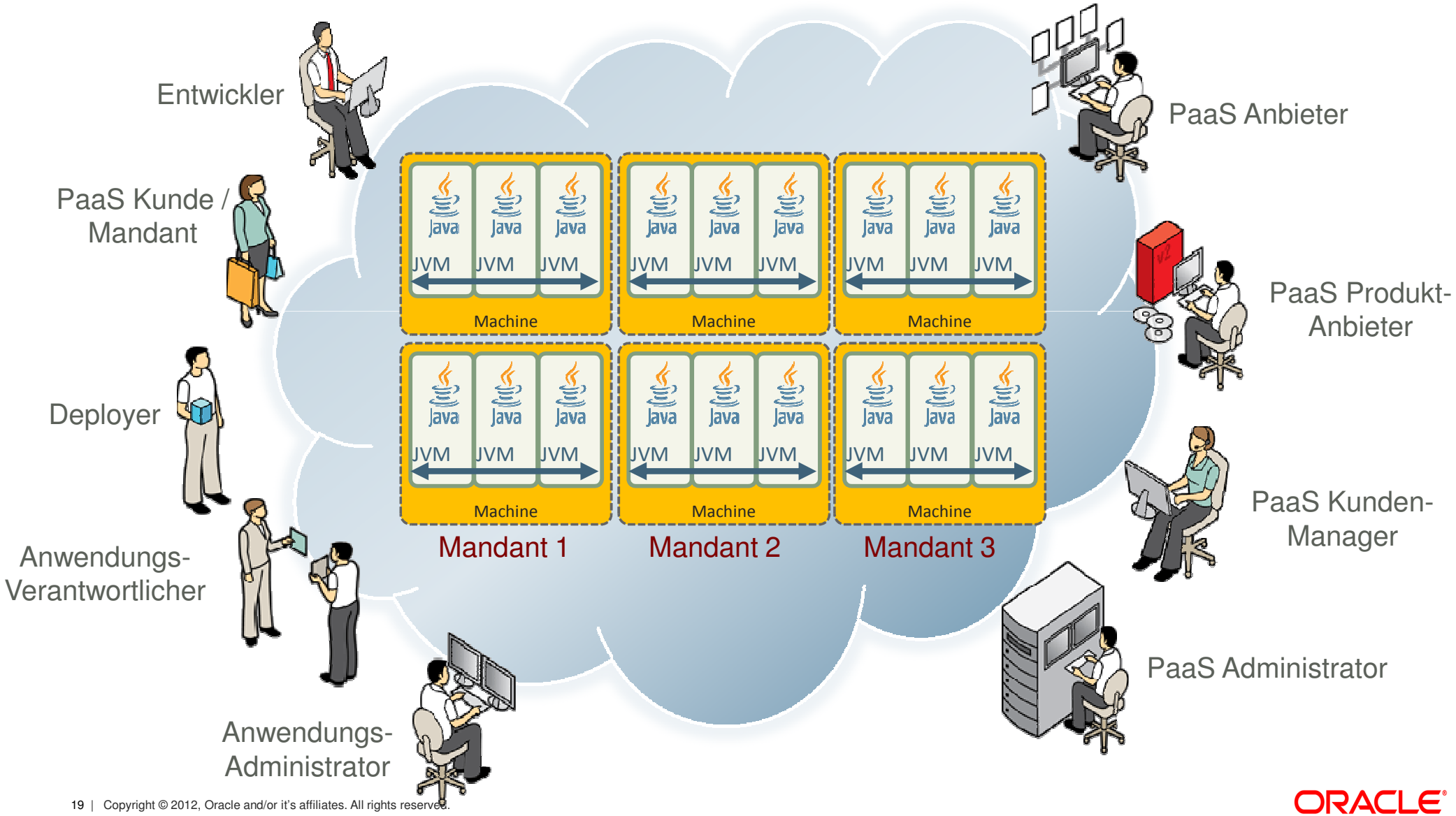


ORACLE

Java EE – Applikations-Deployment

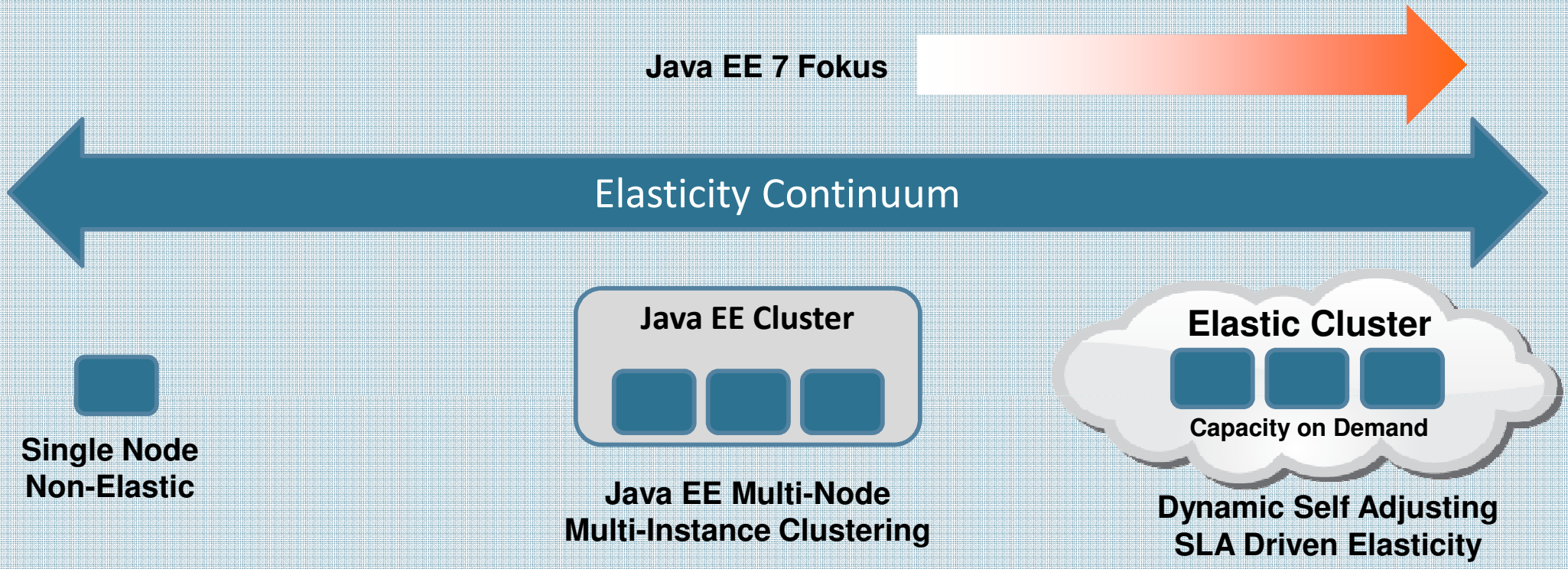


Java EE 7/8 - Rollenmodell



ORACLE

Java EE – Elastizität

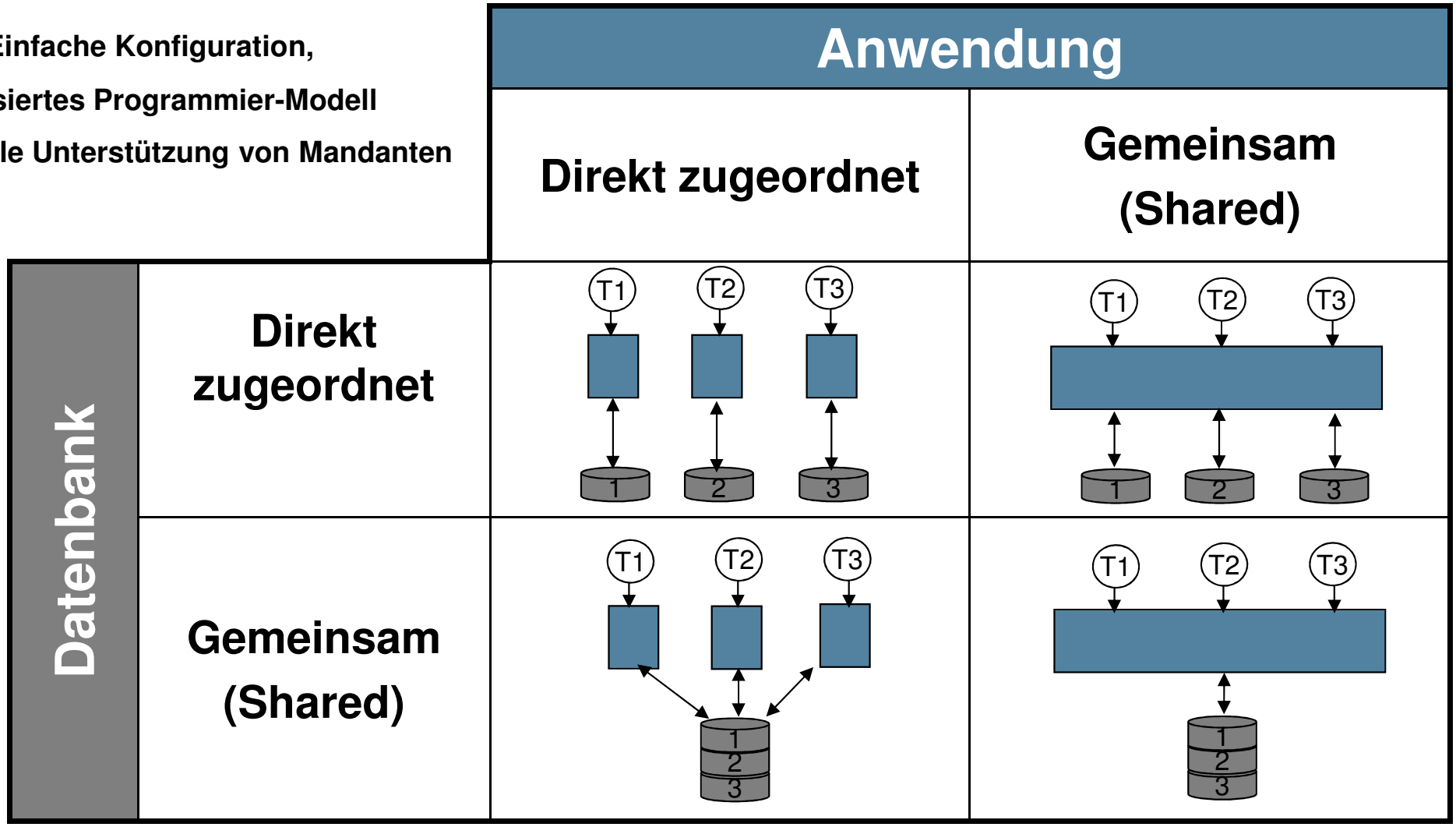


- **Elastisches Cluster – "Capacity On Demand"**
- **Eigenständiges Service Level Management**
  - Antwortzeit, CPU's, Speicherbedarf, Plattenverbrauch, ..
- **Deployment von "Single Machine" bis IaaS**

ORACLE

Java EE Mandantenfähigkeit und Persistenz

- Ziel: Einfache Konfiguration, CDI-basiertes Programmier-Modell
- Flexible Unterstützung von Mandanten
- JPA

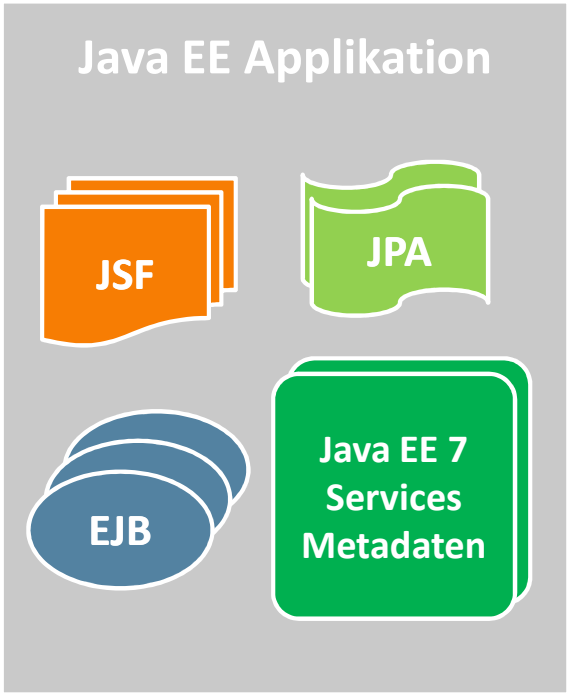


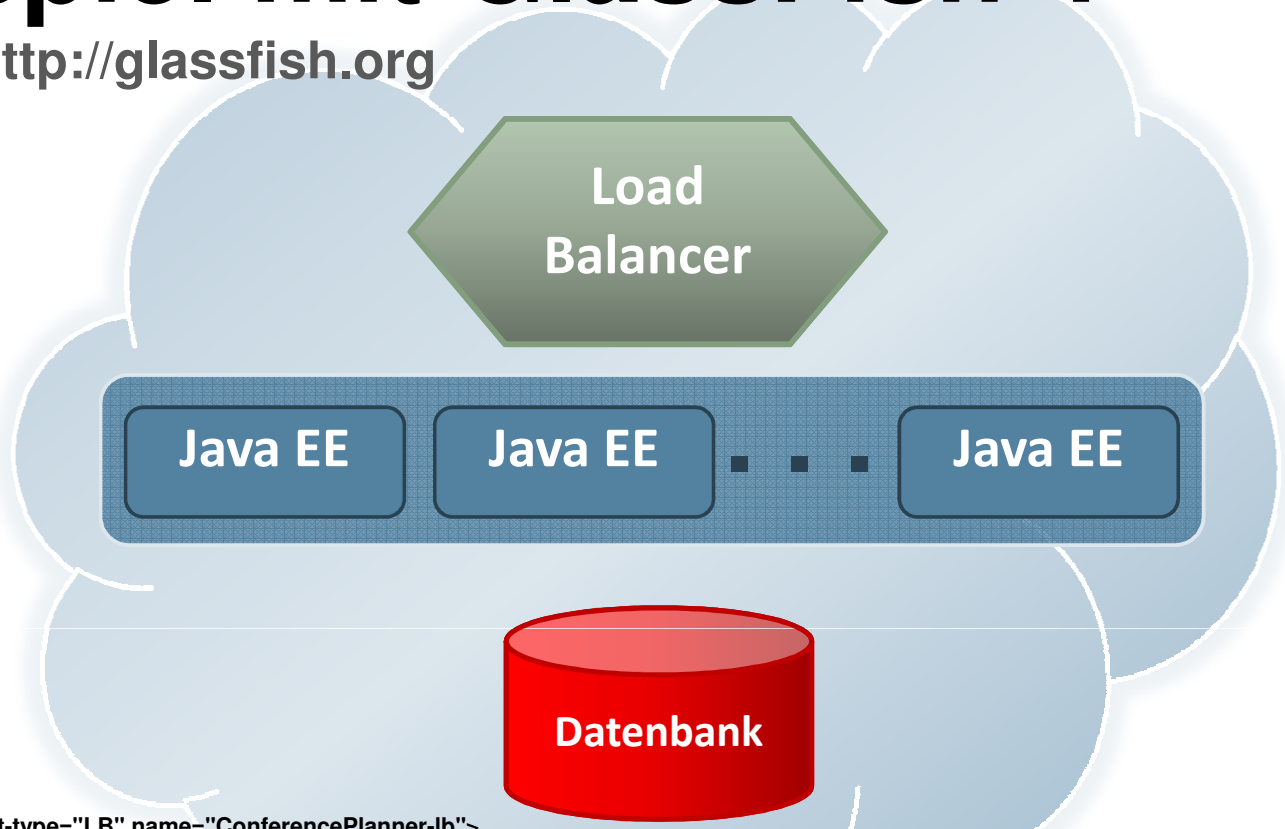
Einzelanwendungs-Deployment mit Unterstützung für unterschiedliche Mandanten-Architekturen (Multi-Tenancy)

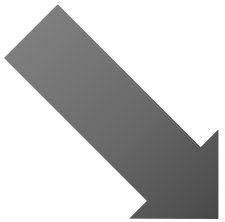
ORACLE

Java EE Beispiel mit GlassFish 4

http://glassfish.org



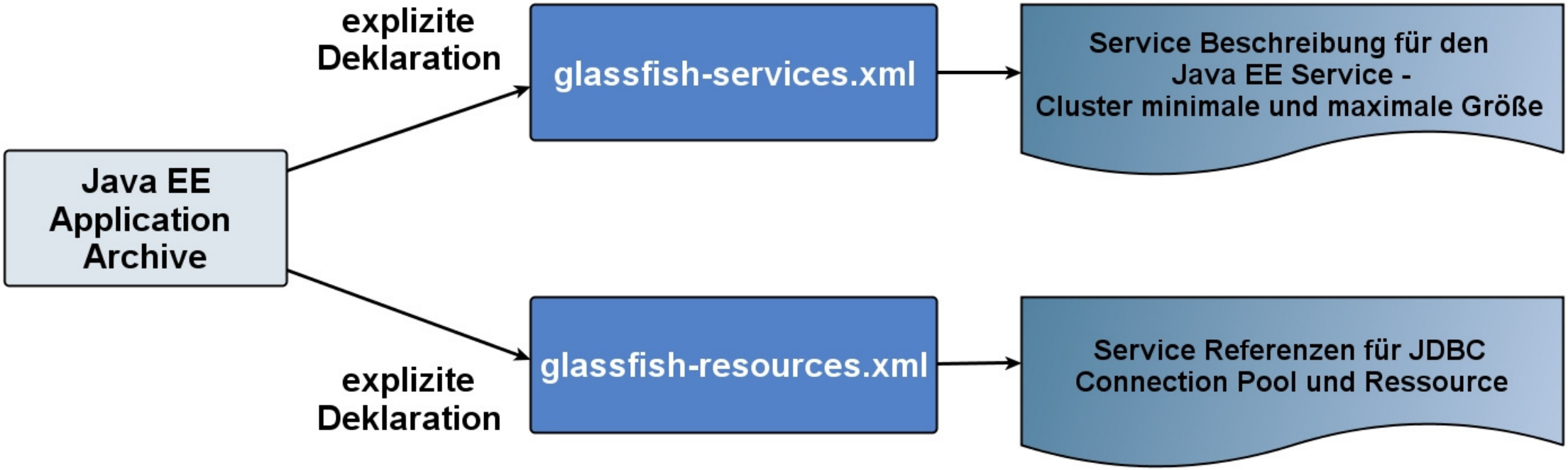




```
<glassfish-services>
<service-description init-type="LB" name="ConferencePlanner-lb">
                    <template id="LBNative"/>
                    <configurations>
                                        <configuration name="https-port" value="50443"/>
                    <configuration name="ssl-enabled" value="false"/>                                                    <configuration
name="http-port" value="50080"/>
                    </configurations></service-description>
<service-description init-type="JavaEE" name="ConferencePlanner">
                     <characteristics>
                                        <characteristic name="service-type" value="JavaEE"/>
                    </characteristics>
                    <configurations>
                                        <configuration name="max.clustersize" value="4"/>
                                        <configuration name="min.clustersize" value="2"/>
                    </configurations>
</service-description>
. . .
</glassfish-services>
```


ORACLE

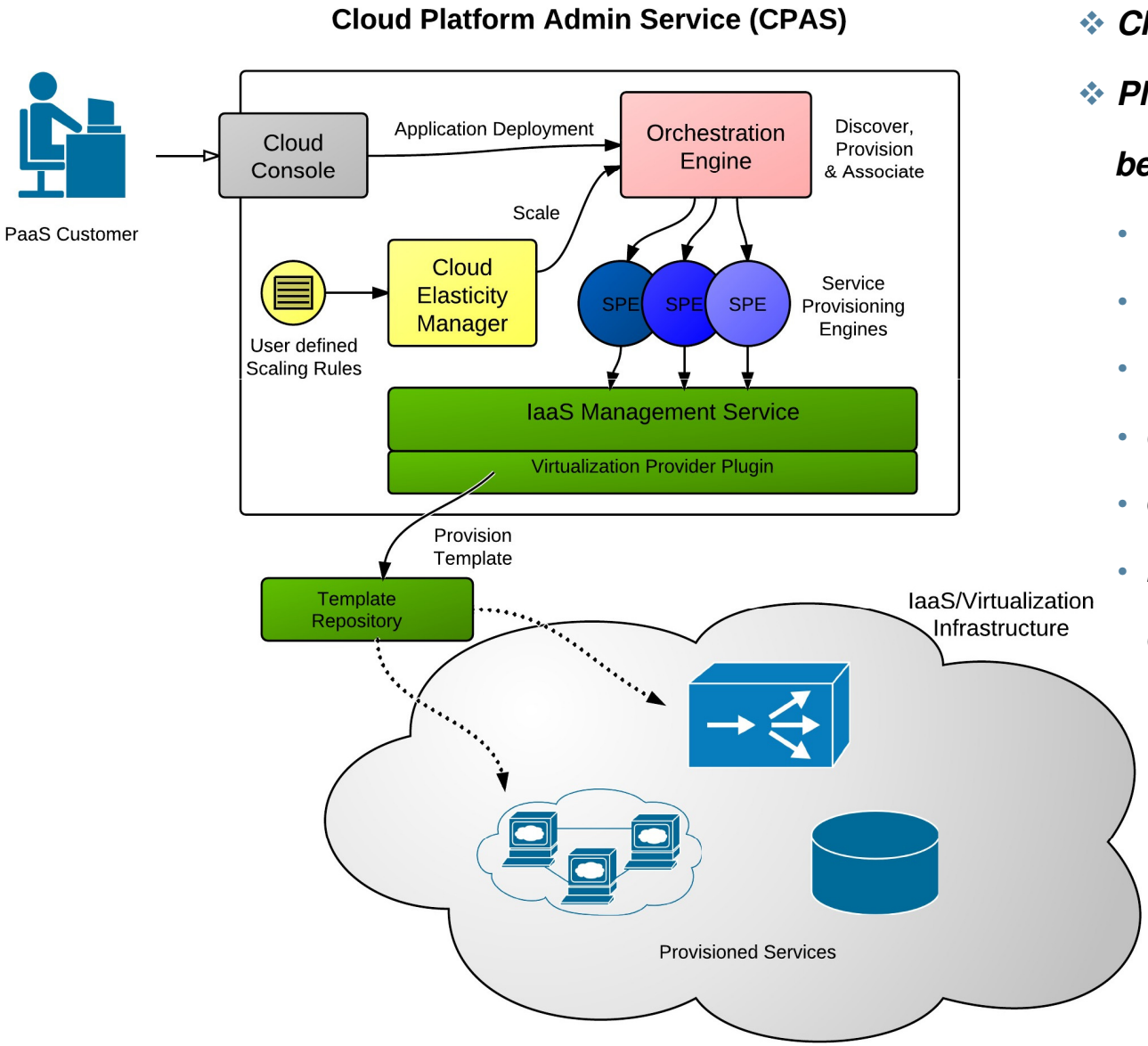
Java EE Services mit Spezifikation von Abhängigkeiten



ORACLE

GlassFish Plattform-Services

Runtime-Architektur

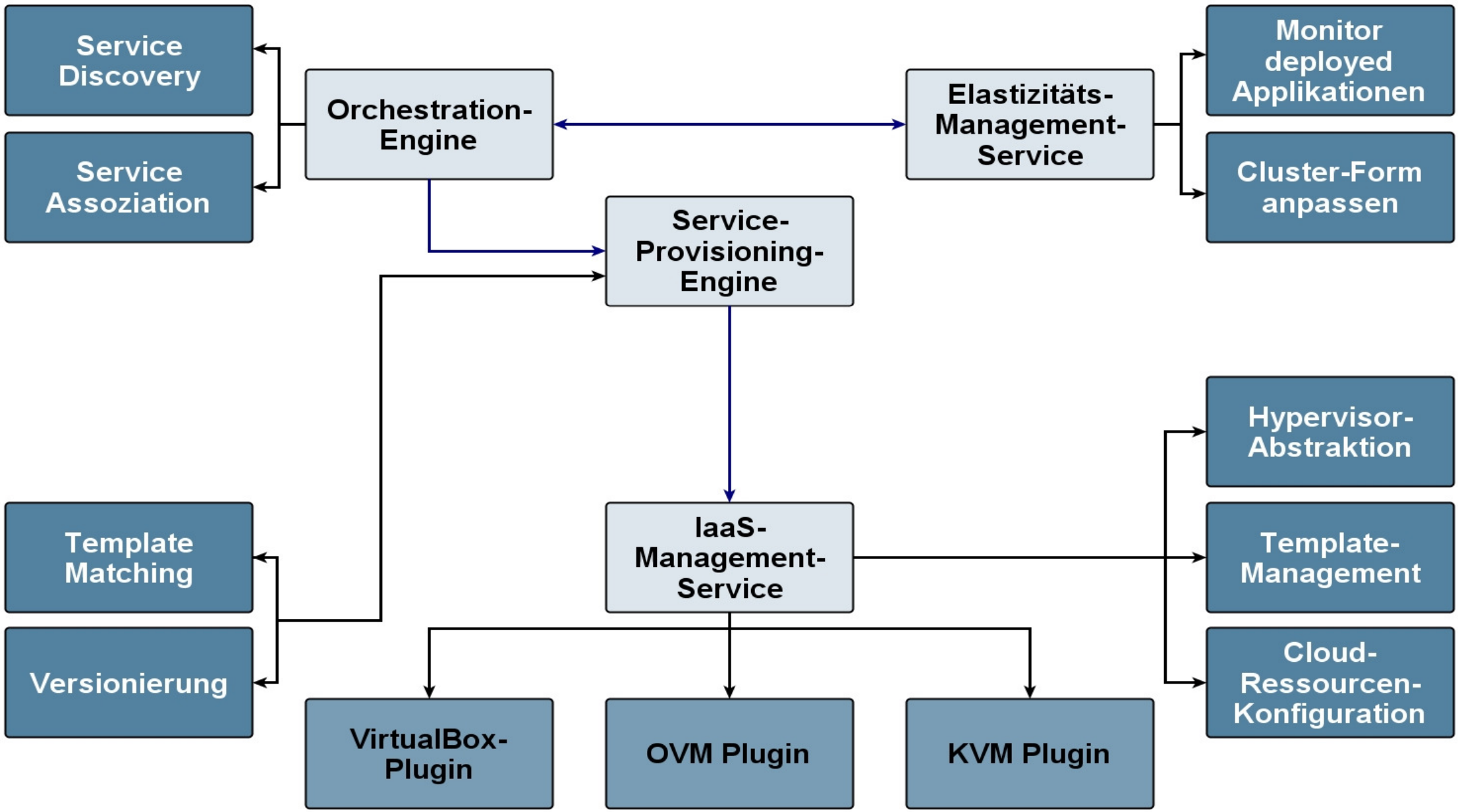


- ***Cloud Application Management for Platforms (CAMP)***
- ***Plattform-as-a-Service Management API Spezifikation bei OASIS eingereicht***
  - *Künftigen Industriestandard mitgestalten*
  - *Multi-Cloud-fähige Management Werkzeuge entwickeln*
  - *REST-basierter Ansatz für Anwendungs-Management*
  - *Common Entwicklungs-Set & API für PaaS und Non-PaaS*
  - *On-Premise-Anwendung in Privat- und Public-Cloud betreiben*
  - *Re-Deployment von Anwendungen über mehrere Cloud-Plattformen unterschiedlicher Hersteller*

ORACLE

PaaS Enablement mit GlassFish

http://glassfish.org



PaaS Auswirkungen für Deployment

Service Management

- **Automatic Service Provisioning and Management**
  - Service Orchestration
    - Automatic Service Dependency Discovery
    - Service Provisioning and Association
  - Handle operational infrastructure concerns automatically
    - Network Configuration, HA, Clustering, Load Balancing ..
  - Application and Service deployment versioning

ORACLE

PaaS Auswirkungen für Deployment

Virtualisierte Ablaufumgebung

- **Scalable virtualized on-demand environment**
  - Support multiple cloud deployment models
    - Public, Private, Hybrid
  - PaaS Provider decoupled from IaaS infrastructure
  - Multi-tenancy

ORACLE

PaaS Auswirkungen für Deployment

Skalierbarkeit und Betrieb

- **Automatic Scaling of Services**
  - Scale to application's needs
  - User-defined alerts and actions
- **Control over application hosting environment**
  - Flexibility in choice of application services, frameworks
  - Rich service configuration
  - Shared services
  - Extensible runtime to allow new Services

ORACLE

Java EE 7 und Java EE 8 – Themenschwerpunkte

PaaS *	Modularität **	HTML5
• Provisioning • Elastizität & eigenständige Skalierbarkeit • Mandantenfähigkeit	• Building on Jigsaw • Fokus auf OSGi interop • Supporting Profiles & Modulare Applikationen	• Emerging Web Standards erfordern ein Programmier-modell • JSON, Web Sockets, off-line, APIs & DOM

* Aligning with delivery schedules of PaaS in Java EE 8
** Aligning with delivery schedules of Jigsaw in Java SE 9

Java EE 7/8 – Geplante Inhalte

Thema	Beschreibung/Inhalt
PaaS * Enablement	• Service Definitions and Provisioning to enable Java as Platform as a Service • Enable Multi-Tenancy in APIs
Web Profile	• Provide popular additions to the Web Profile including JAX RS 2.0 Support
JMS 2.0	• Simplify the programing model for building messaging based applications • Dependency Injection support
CDI	• Tighter Integration with JSF • Expand scope of container managed transactions • Expanded service metadata and improved configuration
Caching	• Provide APIs for accessing caching systems
Concurrency Utilities	• Support for Java concurrency APIs within the container
Pruning	• Allow vendors to optionaly support older APIs • EJB CMP/BMP, JAX-RPC
Open Source and Transparency	• Open development under project GlassFish on java.net • Java EE 7 JSRs run in open with publicly viewable Expert Group mail archive

* Aligning with delivery schedules of PaaS in Java EE 8

Java EE 7 JSR's

javaee-spec.java.net

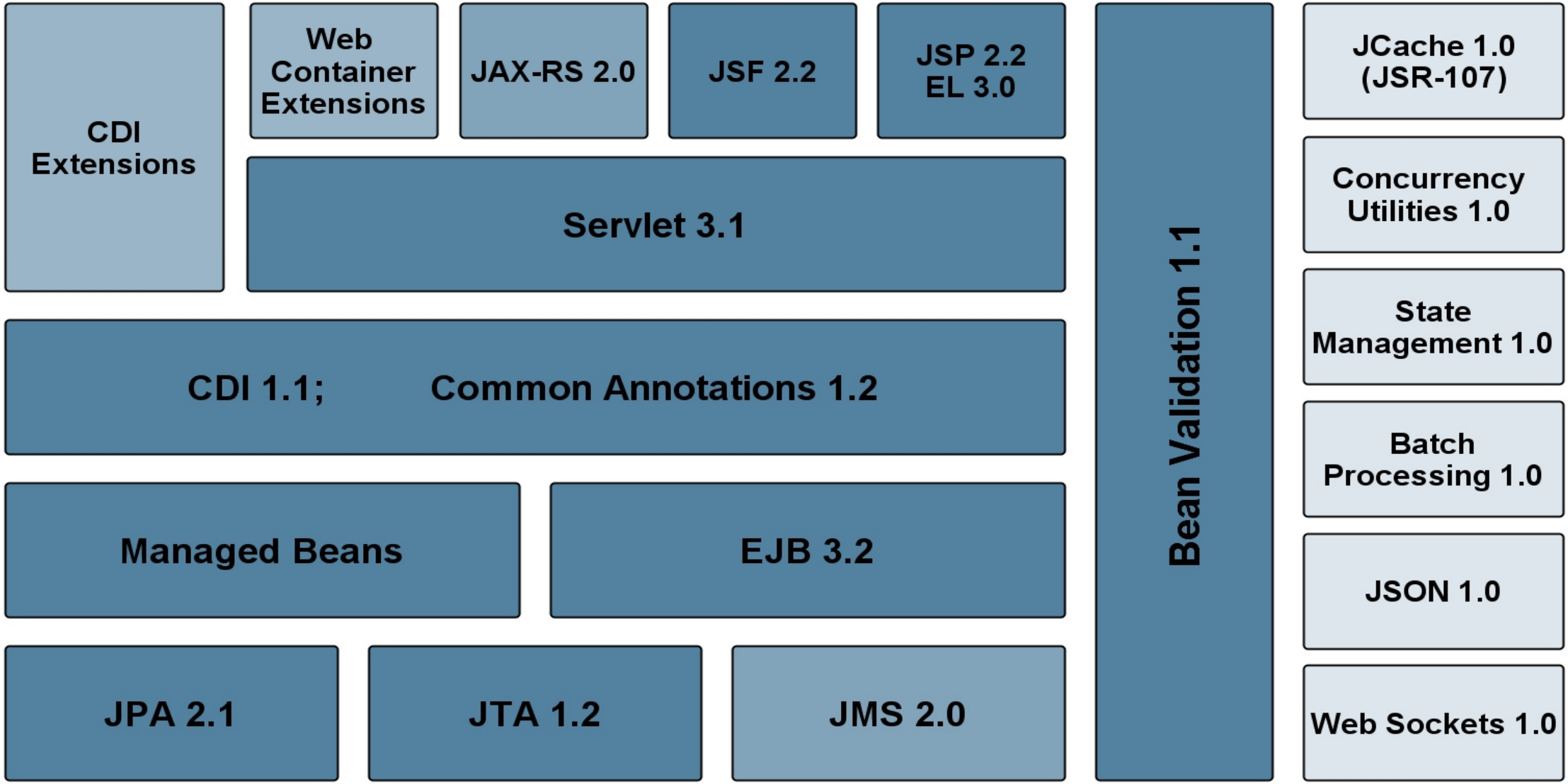
- **Java EE Plattform 7 / Web Profile 7**
- Java Persistence API JPA 2.1 (JSR-338)
- Java API for RESTful Web Services JAX-RS 2.0
- JavaServer Faces JSF 2.2 (JSR-344)
- Servlet 3.1 (JSR-340)
- Enterprise JavaBeans EJB 3.2 (JSR-345)
- JavaServer Pages 2.3 MR
- Expression Language 3.0 (JSR-341)
- Java Messaging Service JMS 2.0 (JSR-343)
- Java API for XML-based Web Services JAX-WS 2.3 MR
- Dependency Injection for the Java Platform (JSR-330)
- Contexts and Dependency Injection for Java EE CDI 1.1
- Bean Validation 1.1 (JSR-349)
- Common Annotations 1.2 MR (JSR-250)
- Java Connector Architecture (JCA 1.6 mit JSR-322)

- Java Web Sockets API 1.0 (JSR-356)
- JSON API 1.0 (JSR-353)

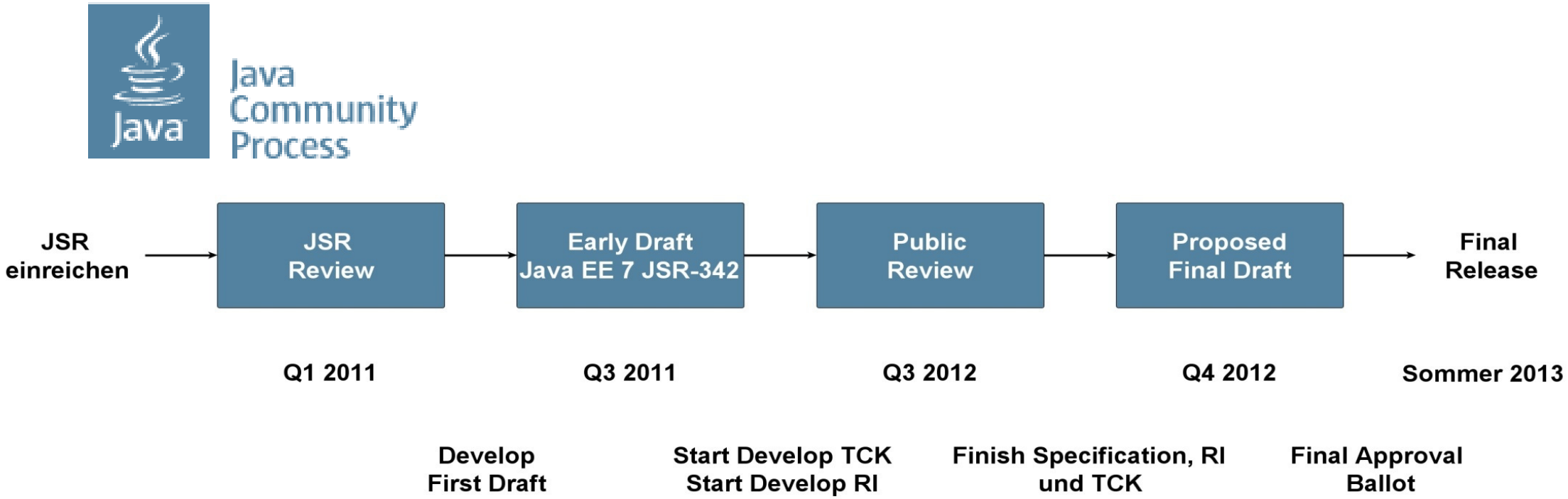
- JCache 1.0 (JSR-107)
- Concurrency Utilities 1.0 (JSR-236)
- State Management 1.0 (JSR-350)
- Batch Processing 1.0 (JSR-352)
- JTA 1.2 MR
- JASPIC 1.2 MR

Java EE 7 JSR's

Architektursicht



Java EE 7 – Zeitplan (1)



- Q3 2011 Early Draft Java EE 7 JSR-342
- Q3 2012 Public Review
- Q4 2012 Proposed Final Draft
- Sommer 2013 Final Release

ORACLE

Java EE 7 – Zeitplan (2)

- ***Providing solid support for standardized PaaS-based programming and multi-tenancy would delay the release of Java EE 7 until the spring of 2014***

- Proposed to the Java EE 7 Expert Group that we adjust our course of action - namely, stick to our current target release dates, and **defer the remaining aspects of our agenda for PaaS enablement and multi-tenancy support to Java EE 8**

- **We continue to believe that Java EE is well-suited for use in the cloud**, although such use might not be quite ready for full standardization. Even today, without Java EE 7, Java EE vendors such as **Oracle, Red Hat, IBM, and CloudBees have begun to offer the ability to run Java EE applications in the cloud**

Java EE 7 – Zeitplan (3)

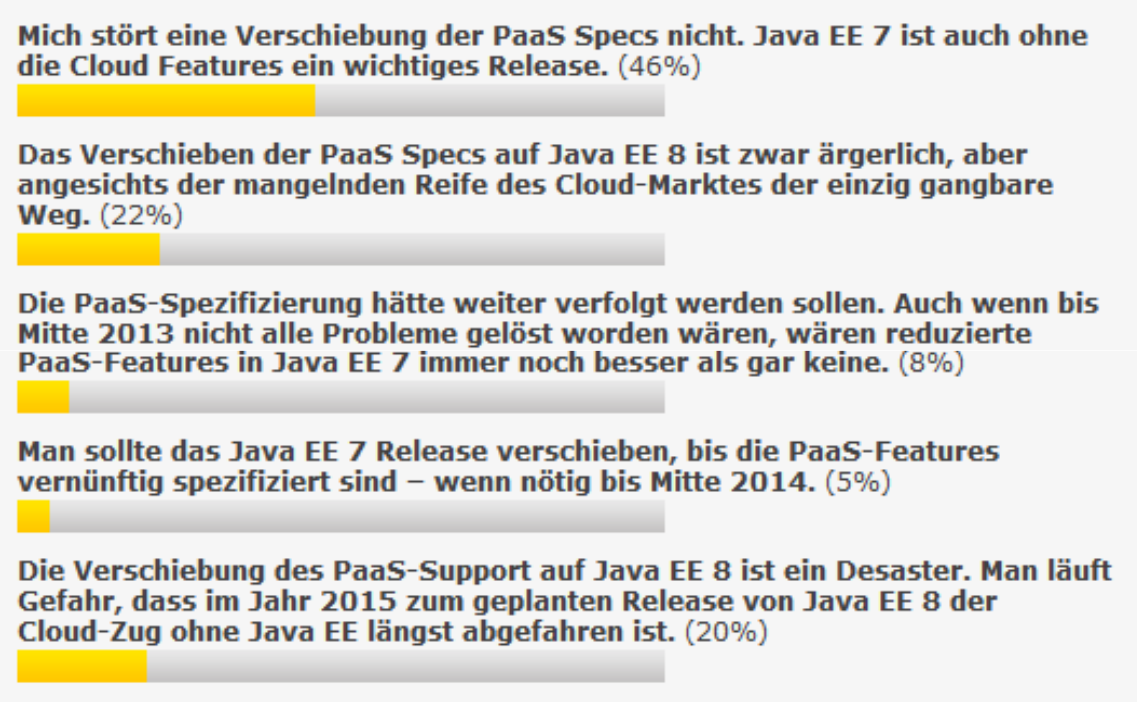
Deferring the remaining cloud-oriented aspects of our agenda has several important advantages:

- It allows Java EE Platform vendors to gain more experience with their implementations in this area and thus helps us avoid risks entailed by trying to standardize prematurely in an emerging area
- It means that the community won't need to wait longer for those features that are ready at the cost of those features that need more time
- Because **we have already laid some of the infrastructure for cloud support in Java EE 7, including resource definition metadata, improved security configuration, JPA schema generation, etc.,** it will allow us to expedite a Java EE 8 release. **We therefore plan to target the Java EE 8 Platform release for the spring of 2015**

Java EE 7 – Zeitplan (4)

- This shift in the scope of Java EE 7 allows us to better retain our focus on enhancements in simplification and usability and to deliver on schedule those features that have been most requested by developers.
  - These include the support for HTML 5 in the form of Web Sockets and JSON-P
  - the simplified JMS 2.0 API's
  - improved Managed Bean alignment, including transactional interceptors
  - the JAX-RS 2.0 client API
  - support for method-level validation
  - a much more comprehensive expression language
  - and more
- We feel strongly that this is the right thing to do, and we hope that you will support us in this proposed direction

Umfrage: Java EE 7 ohne Cloud – was halten Sie davon?



Quelle: it republik JAXenter vom 4.9.2012

ORACLE

Zusammenfassung

- Die Java Plattform nutzt Innovationen im Ökosystem und wird sich weiterentwickeln
  - Dafür sind signifikante Entwicklungs-Ressourcen notwendig
- Unternehmen profitieren von existierende Investitionen in Java EE
- Java Plattform liefert Mehrwert
- Java EE 7 Ressourcen Definition auf Basis Metadaten
- GlassFish ist die RI für Java EE mit Service-Konzept
- Java EE 8 wird PaaS beinhalten

The preceding is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

ORACLE

Vielen Dank für Ihre Aufmerksamkeit!

Wolfgang.Weigend@oracle.com

ORACLE

Vielen Dank!

Wolfgang Weigend
ORACLE Deutschland B.V. & Co. KG