integrata cegos

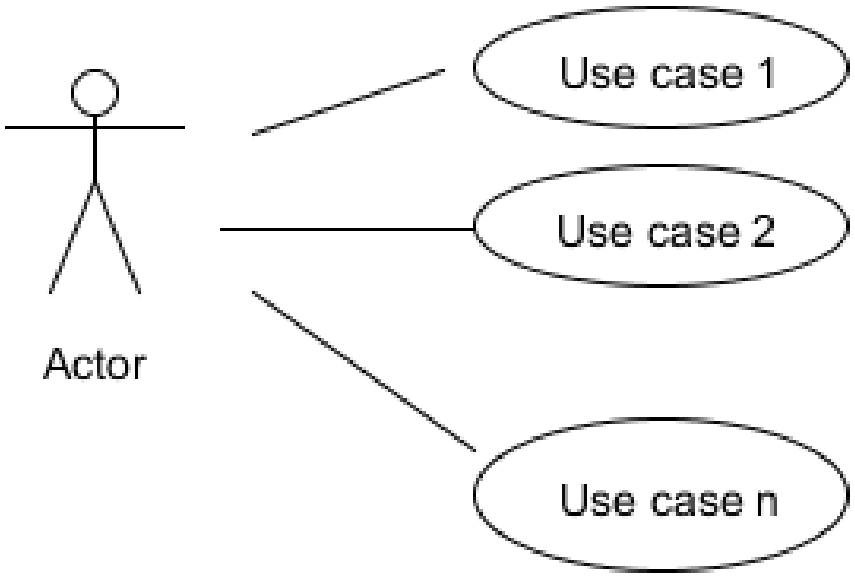
Einführung in die Java / Jakarta Enterprise Edition

Kapitel 4 – Programmierung von JEE Anwendungen

4.1

FOKUS UND TYPISCHE BEISPIELE

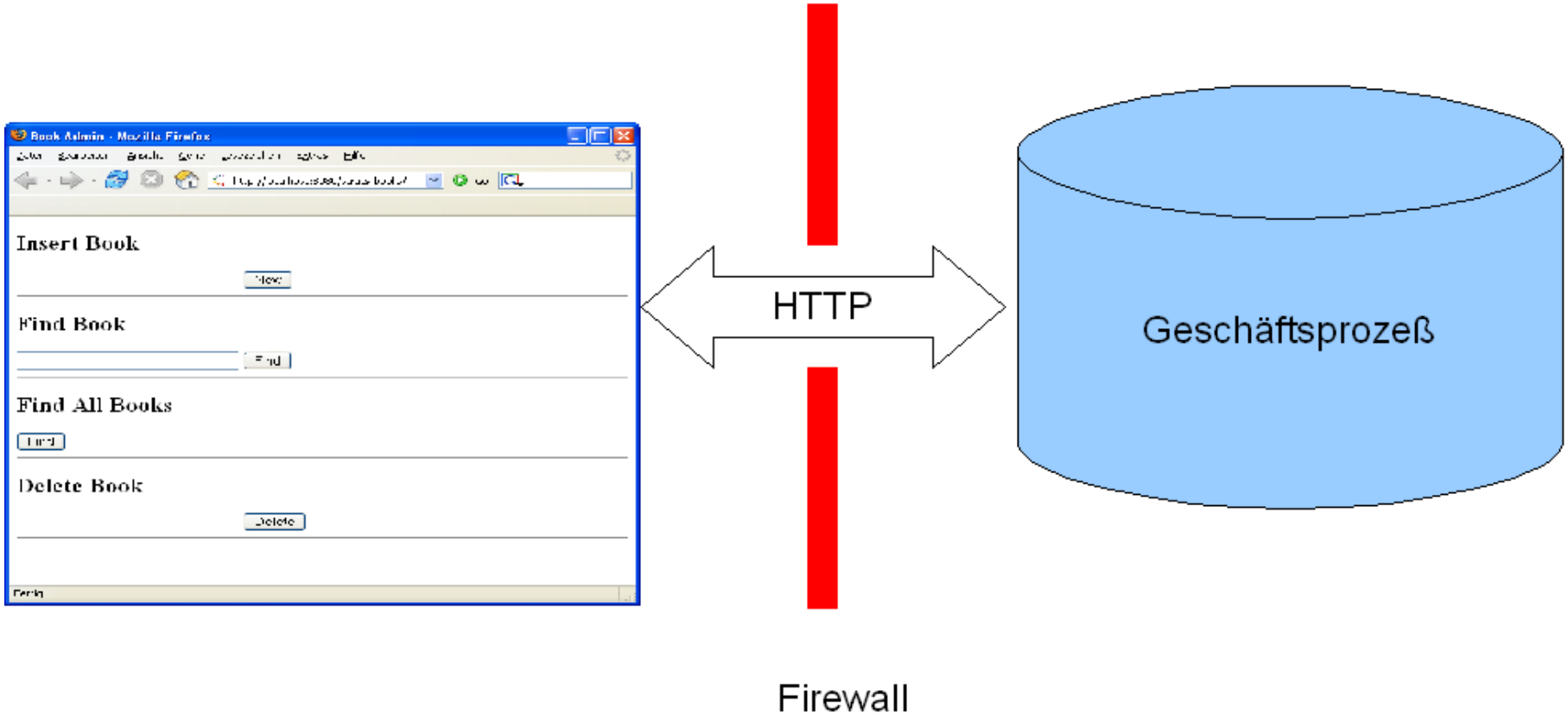
- Die JEE ermöglicht keine völlig neue Art von Anwendungen
  - Weder technisch noch fachlich
- Auch mit JEE bleibt die klassische Aufgabenstellung:
  - „Eine Fachvorgabe muss in eine funktionierende Anwendung umgesetzt werden!“



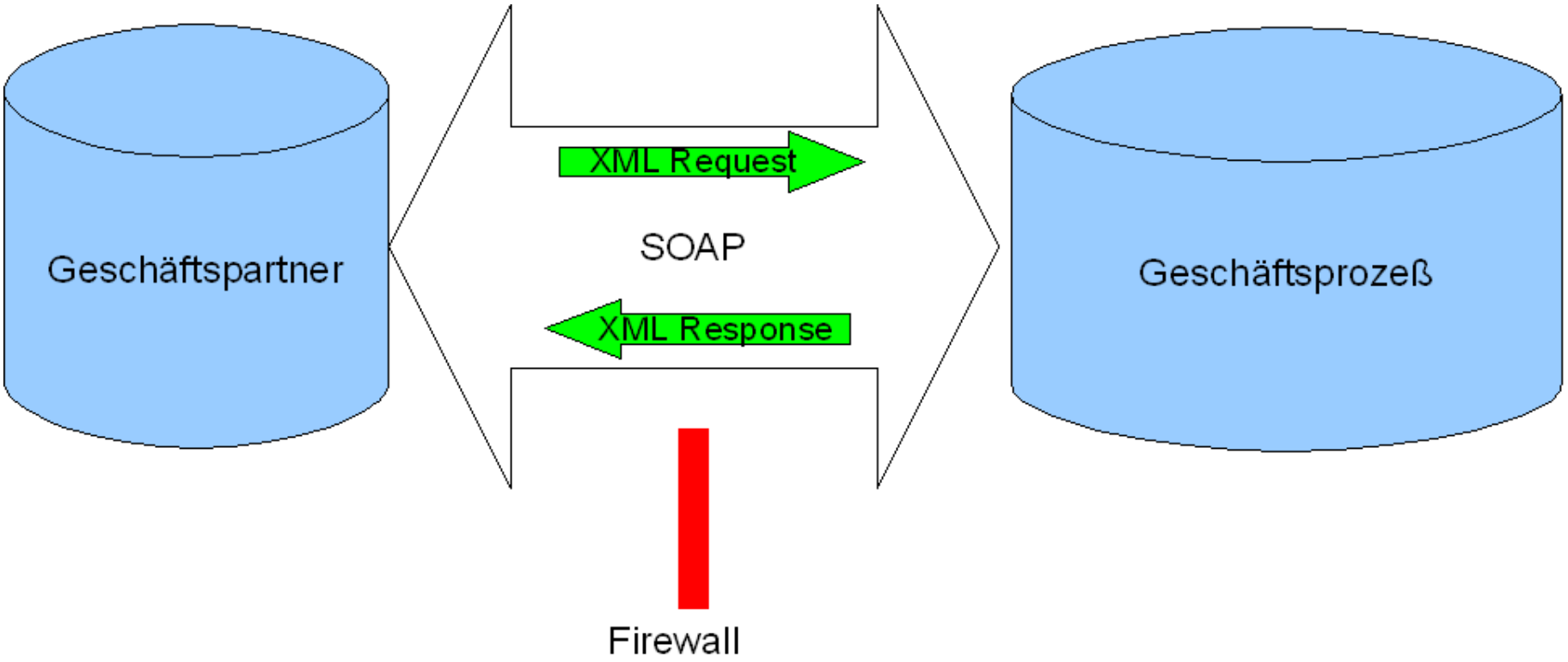
Anwendungs-Beispiele im JEE-Umfeld

- Die JEE ist auf verschiedene typische Anwendungen hin ausgerichtet, z. B.:
  - Web Applikation mit Browser-basiertem Front End
  - Unternehmens-übergreifende Prozesse
  - Web und Rich Clients
  - Komplexe Transaktionssteuerung
  - …
- Es können aber auch komplexe Spezial-Aufgaben umgesetzt werden!
  - Workflow-Engines
  - Bus-Systeme
  - Applikationsserver als Content Management System
  - …
- Auf Basis der JEE hat sich eine breite Produkt-Palette entwickelt
  - jBPM Workflow Engine
  - IBM Process Manager
  - Liferay Portal Server
  - …

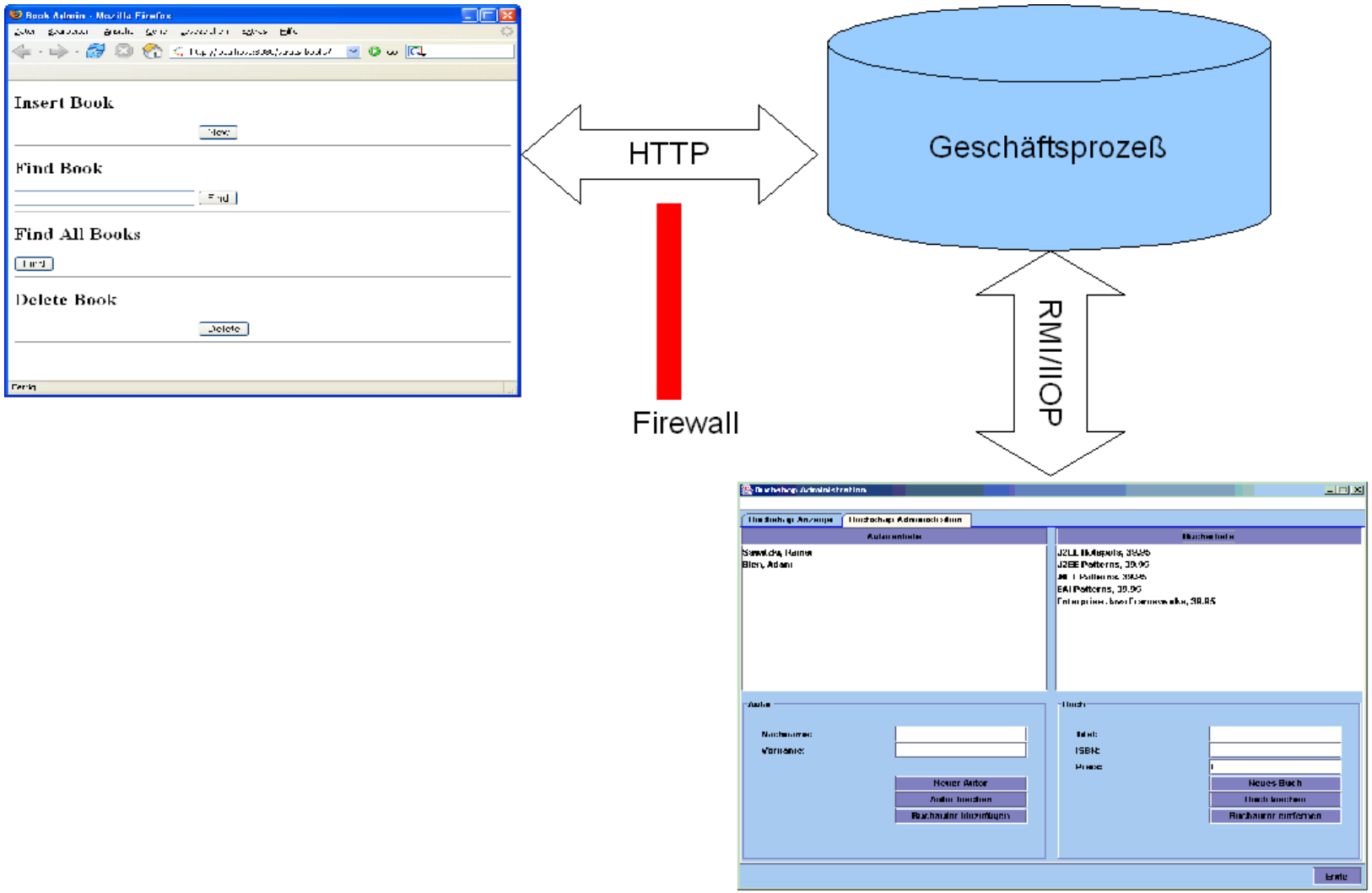
Beispiel 1: Business to Consumer



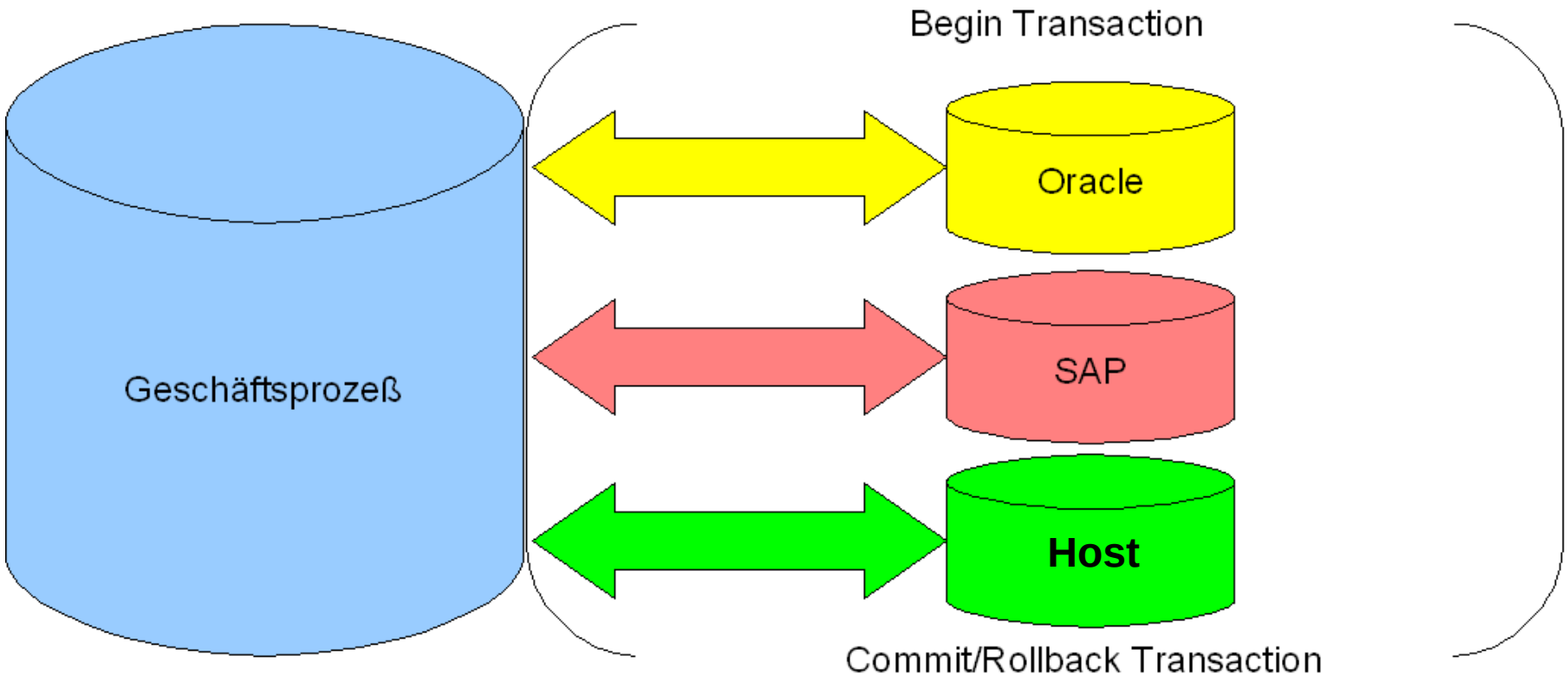
Beispiel 2: Business to Business



Beispiel 3: Interne und externe Anwendungen



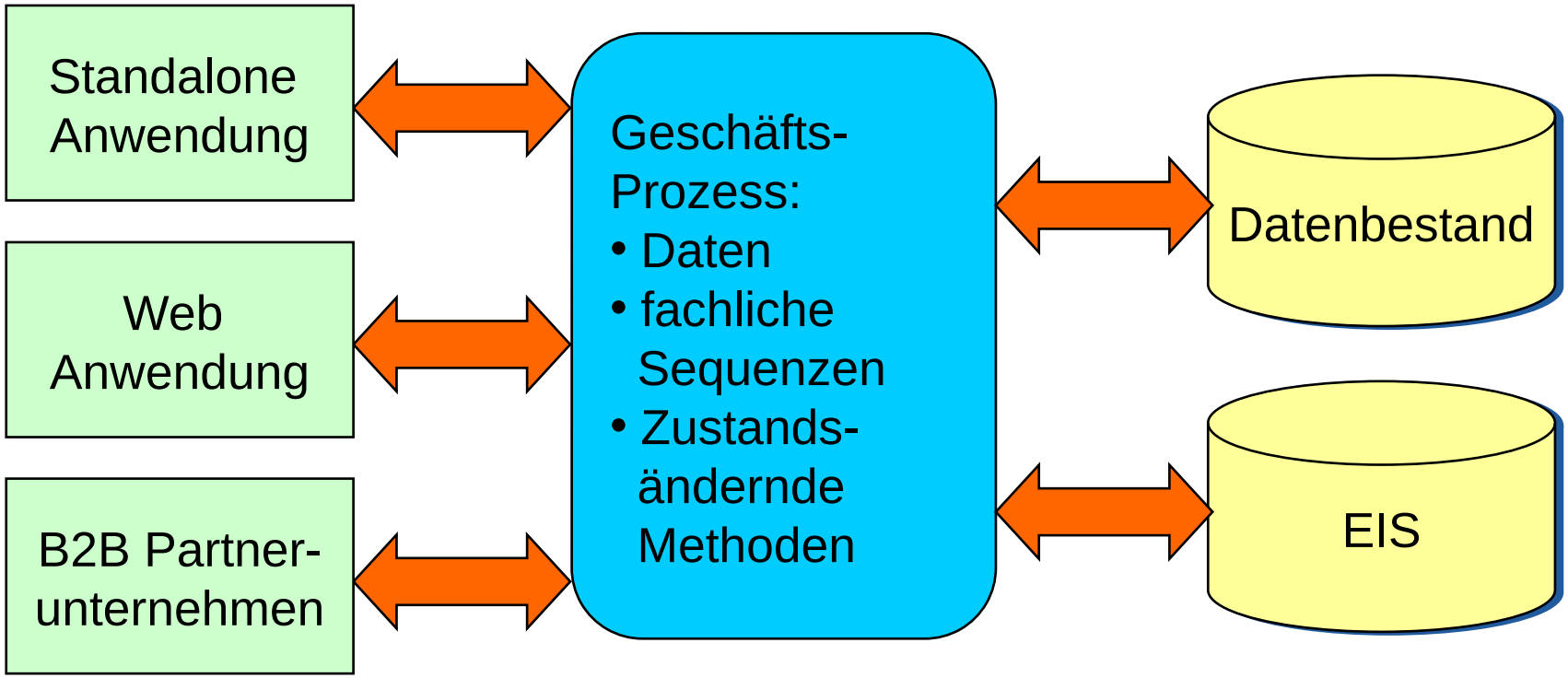
Beispiel 4: XA-Transaktionen



4.2

DIE VISION DER IDEALEN UMSETZUNG

Ideale Umsetzung



Eine ideale Umsetzung…

- enthält möglichst wenige Code-Zeilen
- ist modular und flexibel aufgebaut
- ist Fehler-tolerant und ausfallsicher
- ist einfach zu testen
- kann von verschiedensten Anwendungen aufgerufen werden
- kann beliebig verteilt werden und ist 100%ig skalierbar
- ist selbst Transaktionsfähig und orchestriert Transaktionen konsistent über beliebige Backend-Systeme hinweg

JEE: Eine Plattform für ideale Anwendungen?

- Der Applikationsserver nimmt dem Entwickler viele Aufgaben ab
  - Netzwerk
  - Skalierung/Cluster-Betrieb
  - Verwaltung von Ressourcen
- Deklarative Programmierung
  - Transaktionssteuerung
  - Security
  - Seiten-Navigation

Die JEE mag nicht perfekt sein, ist aber definitiv ein großer Schritt in die richtige Richtung!

4.3

KRISE UND WIEDERAUFERSTEHUNG: DIE JEE IM WANDEL DER ZEIT

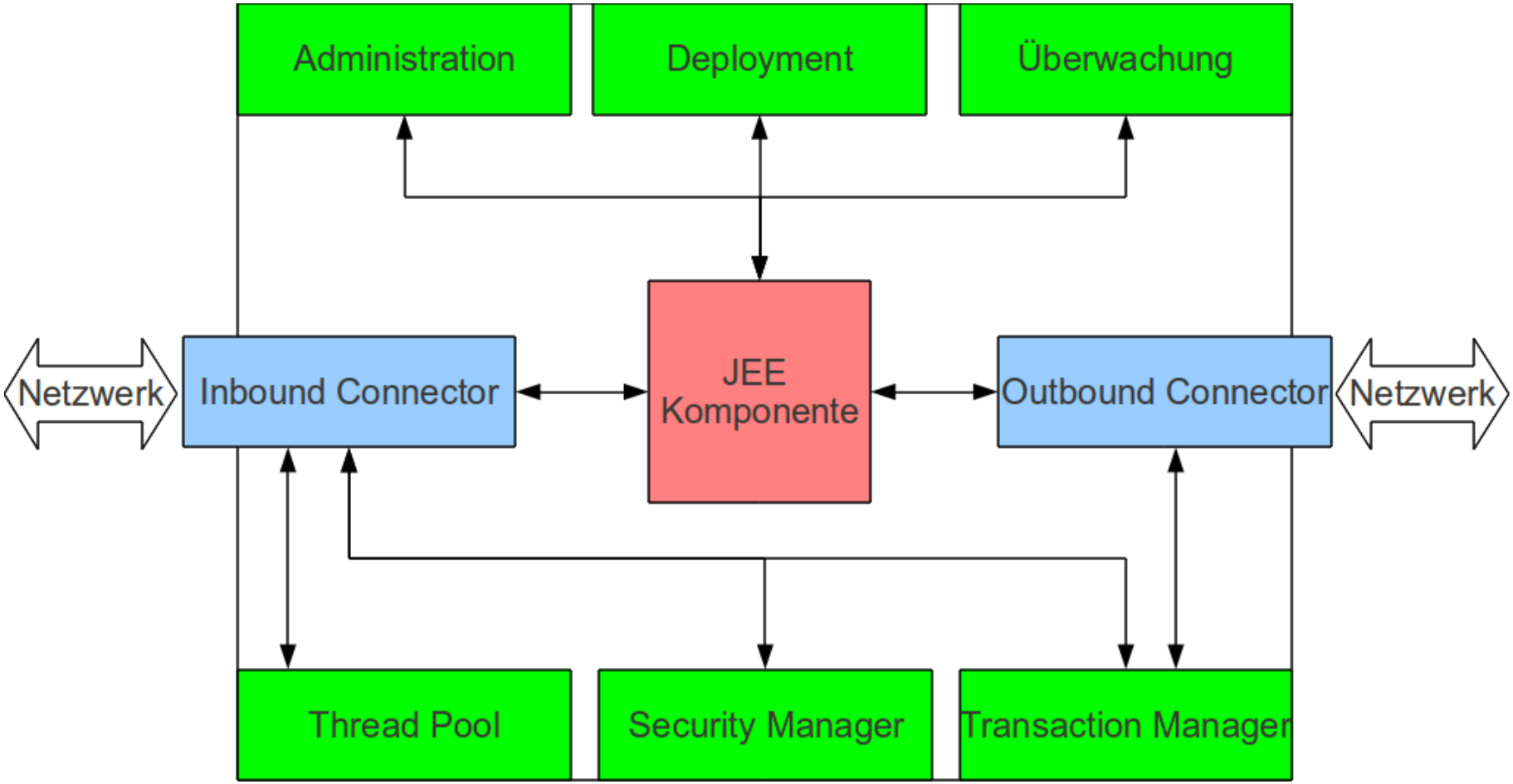
Diskussion

- Welche typischen Anwendungen kennen Sie in Ihrer Umgebungen?
- Was sind die großen Vorteile von Jakarta EE Anwendungen?
- Auf welche Probleme sind Sie bereits gestoßen (Benutzung, Betrieb, Entwicklung)?

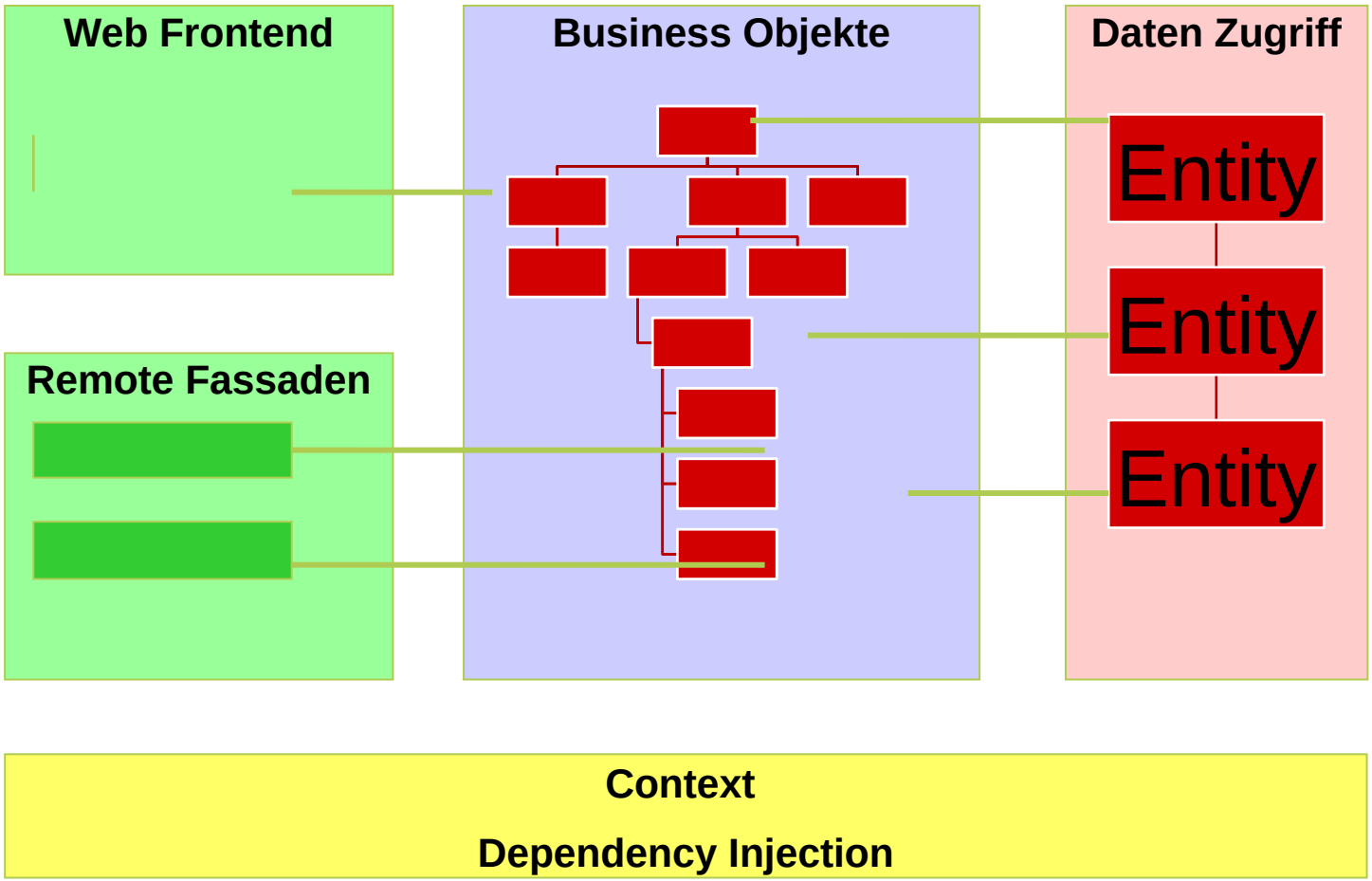
Die Krise der JEE

- Die Zahl der Projekt-Anfragen/Seminar-Teilnehmer, die den vollen Umfang der JEE benötigen, stagnierte bis 2009 auf etwa 25% des Maximalwerts von 2002
  - Unter Berücksichtigung allgemeiner Marktschwankungen
- Die JEE in der Version 1.4 wurde schon zum Zeitpunkt ihrer Veröffentlichung im November 2003 massiv kritisiert
  - „Umständlich, unzeitgemäß, unbrauchbar, …“
  - Effiziente Anwendungsentwicklung war mit reinen JEE-Mitteln de facto unmöglich!
    - JEE Projekte benutzten große Mengen an proprietären Werkzeugen zur Erleichterung des Entwicklungsprozesses

JEE ist ursprünglich eine Laufzeitumgebung



Was die Anwender brauchen…



JEE Alternativen

- Seit 2003 wurden eine Vielzahl von Frameworks entwickelt, deren Fokus alleine auf der Vereinfachung des JEE-Programmiermodells liegt
  - Web Frameworks wie Apache Struts, Code-Generatoren wie XDoclet
- Seit 2004 steht mit dem Spring Framework ein alternatives Programmiermodell zur Verfügung
  - In der Entwicklergemeinde als deutlich einfacher empfunden
- JBoss Seam übernahm auf Grund der Stagnation und langen Release-Zyklen der JEE die Weiterentwicklung
  - Allerdings stets darauf Bedacht, auf Standards zu achten!
  - JBoss Seam ist mittlerweile größtenteils in die JEE aufgegangen und wird nicht mehr aktiv weiterentwickelt
- Seit 2009 steht CDI (Context & Dependency Injection) zur Verfügung
  - Eigentlich eine neue Version der Java Enterprise Edition
    - Allerdings kompatibel mit den alten Versionen

Exkurs: Spring

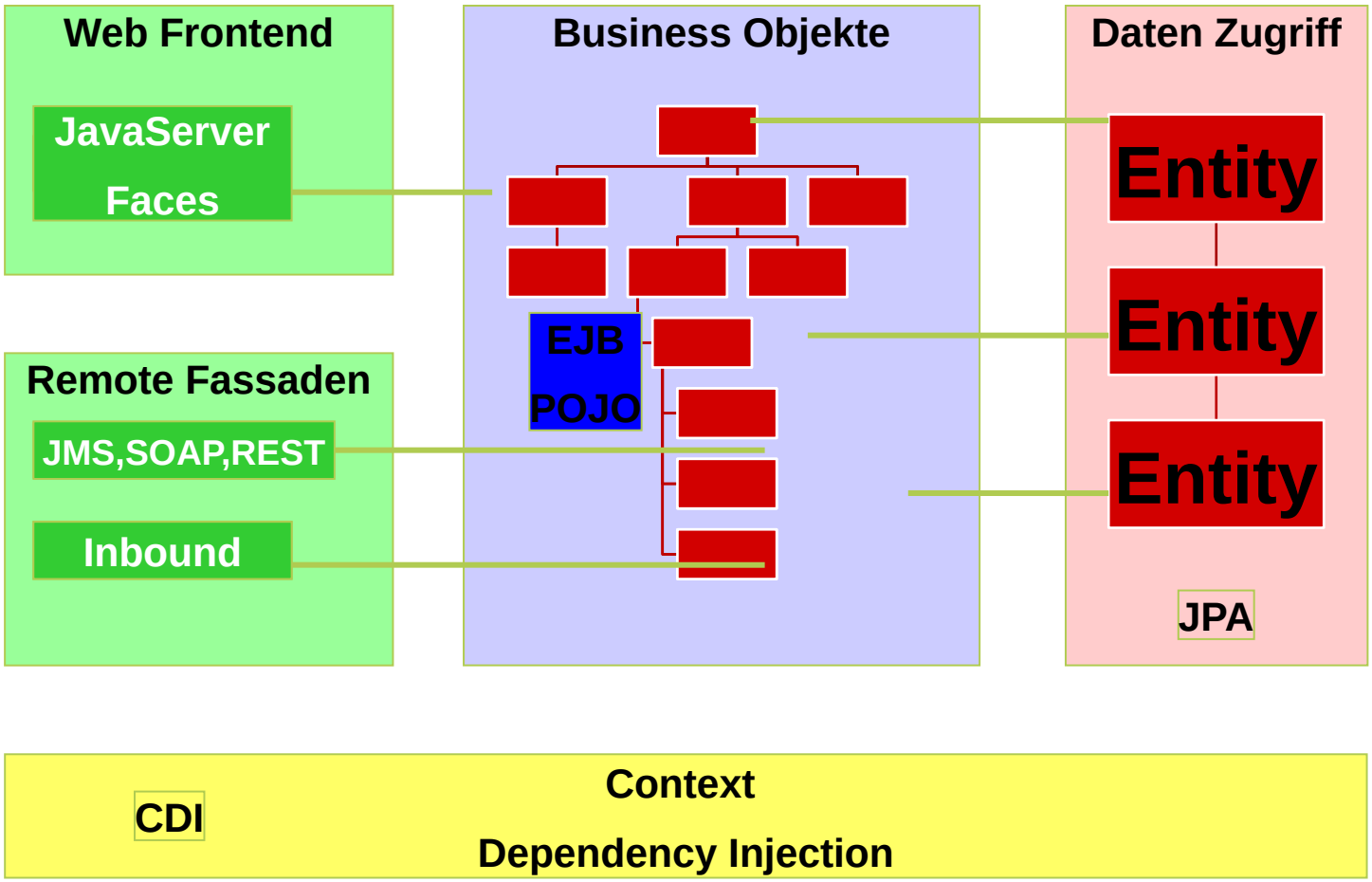
- Das Spring-Framework der springsource.org ist ein Open Source Framework, das aus sehr vielen Modulen besteht
- Die Core-Komponente ist ein hervorragendes Dependency-Injection- und AOP-Framework
  - Und deshalb prinzipiell zur Ergänzung der JEE wunderbar geeignet!
- Spring positioniert sich selbst jedoch recht aggressiv als Alternative zur JEE
  - Ein eigenes Web Framework
  - Mit dem Spring tc-Server (einem erweiterten Server) und dem dm-Server (OSGi-konformes Deployment) werden eigene Laufzeitumgebungen definiert.
- Spring ist Implementierungsgetrieben, es gibt keine Spezifikation!
  - Ein erster (?) Versuch der Springsource zur Kommerzialisierung durch die Einführung einer geschlossenen „Supported Version“ ist am massiven Protest der Community gescheitert
  - Spring enthält wie das Negativ-Beispiel Struts 1.x bereits eine ganze Menge von „historisch gewachsenen“ Bibliotheken, die nicht mehr benutzt werden sollten

JEE 6

- Spätestens mit der im Jahre 2009 veröffentlichten JEE 6 ist es jedoch gelungen, einen Großteil der proprietären Frameworks wieder einzufangen
  - Datenbankzugriffe und O/R-Mapping mit dem Java Persistence API 2
  - Das Web Framework JavaServer Faces 2 mit den Facelets Templates
  - Mit den Interceptors können Querschnittsfunktionen realisiert werden
  - Dependency Injection mit der Context and Dependency Injection-Bibliothek

- Damit wird die ursprüngliche Spezifikation der Laufzeitumgebung um ein komfortables Programmiermodell ergänzt!
  - Dies wird auch von den Entwicklern honoriert: Es ist eine deutlicher Trend zurück zur JEE zu beobachten!

Realisierung mit JEE-Bibliotheken



JEE im Wandel: Eine Referenz-Applikation

- 2003
  - Java Enterprise Edition, Version 1.4
  - Direkte Datenzugriffe mit „Bean Managed Persistence“ und Apache Torque
  - Web Frontend mit Servlets und JavaServer Pages
- 2004
  - Erster Umstieg auf Hibernate
  - Code-Generierung mit XDoclet
  - Gescheiterter Umstieg auf JavaServer Faces, statt dessen Apache Struts
- 2005
  - Design-Änderung auf Dependency Injection, erste Integration von Spring
  - Generische SessionBeans als Fassaden
  - Web Services mit Apache Axis
- 2006
  - Migration auf JEE 5
  - Umstellung auf MyFaces
  - Kompletter Umstieg auf Hibernate

JEE im Wandel: Eine Referenz-Applikation

- 2007
  - Aufsplittung in eine Variante mit EJBs und eine Variante mit Spring
  - Teilweiser Rückbau von Hibernate auf Java Persistence API
- 2008
  - Integration von AJAX-Funktionalität mit Ajax4JSF bzw. RichFaces
  - Umstellung auf Apache CXF
- 2009
  - Weitere Aufsplittung in eine JBoss Seam-Variante
  - Teilweise Umstellung auf JAX-WS

JEE im Wandel: Eine Referenz-Applikation

- 2010
  - Umbau auf JEE 6
    - Kompletter Verzicht auf Hibernate API
    - Teilweiser Rückbau von RichFaces auf JSF 2.0
    - RESTful Aufrufe mit JAX-RS
  - Die Anwendung ist zum ersten mal praktisch unabhängig von externen Bibliotheken!
- 2013
  - Konsequente Benutzung von CDI auch zur Transaktionssteuerung
    - Fast alle EJBs sind verschwunden
- Jakarta EE 11 soll auch ein Gegenstück zu Spring Data beinhalten

Zusammenfassung Zukunftsfähigkeit Jakarta EE

- Auch wenn JEE aufgeholt hat, Spring Boot bleibt eine sehr gute Alternative
- Aber auch Spring Boot nutzt Teile der JEE Spezifikation (JPA)
- Alte komplette Applikationsserver sind oft in der Microservice-Welt schwerfällig und langsam für die Entwicklung
- Container wie Quarkus nutzen einen Großteil der JEE Spezifikation und sind schnell und gut in Docker-Container benutzbar
- ...steht und fällt mit den Unternehmen, die dahinter stehen → es lohnt, sich mit deren Geschäftsmodellen auseinander zu setzen