

Kaiserslautern, den 23.07.2004

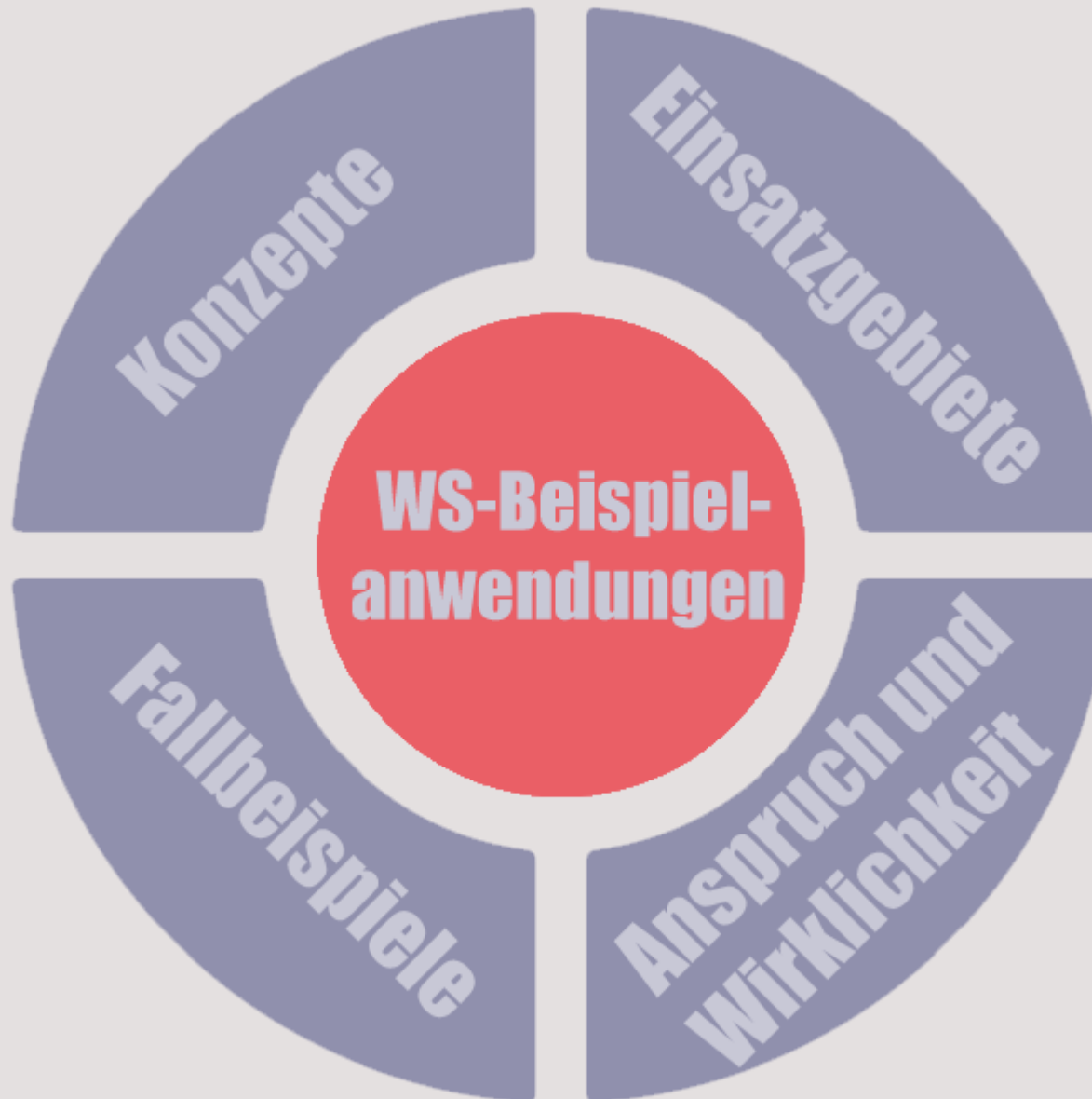
Sebastian Weber

**Vorstellung von
Web-Services-Beispielsystemen**

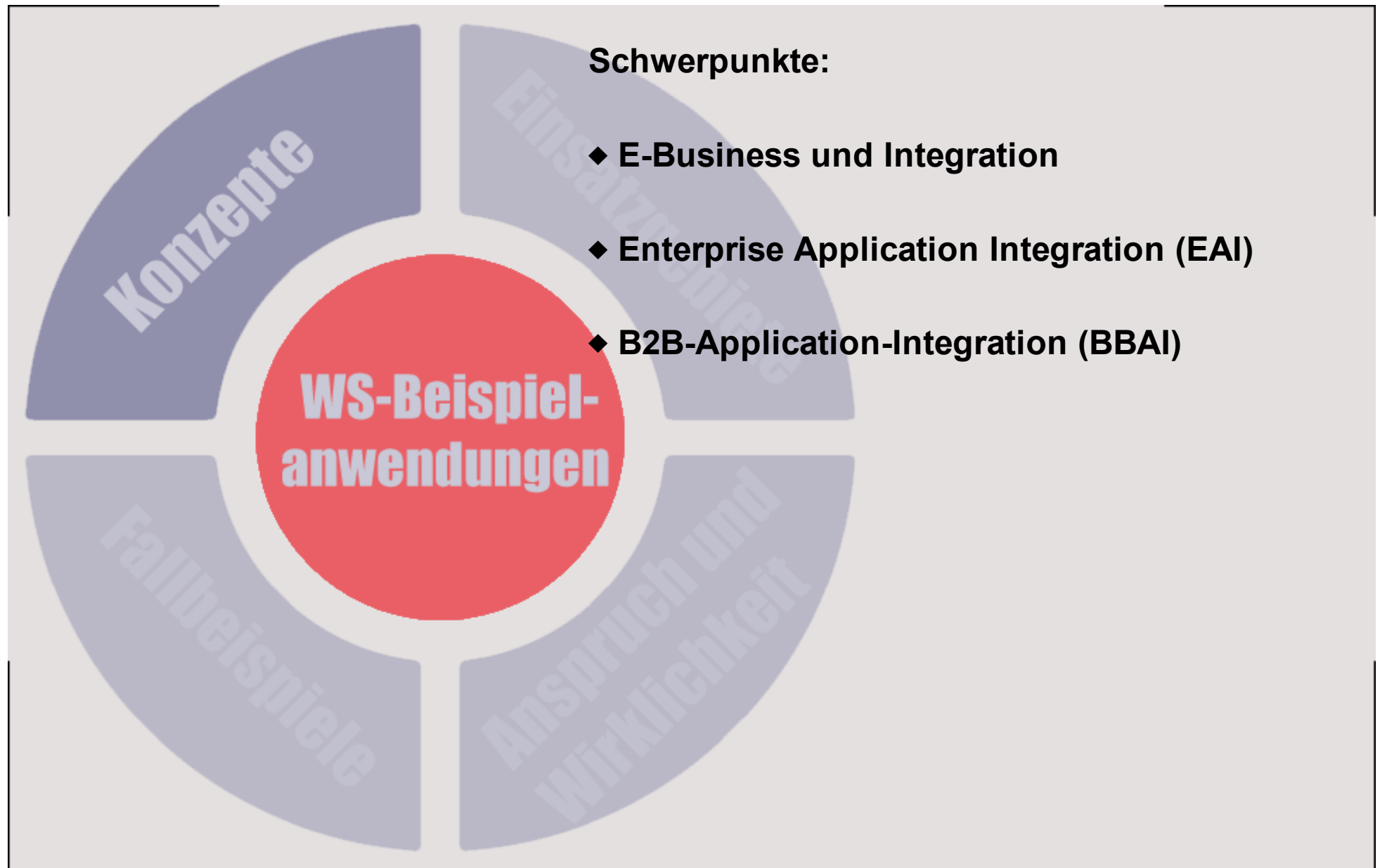
Lehrgebiet Datenverwaltungssysteme

**Fachbereich Informatik
TU Kaiserslautern**

Gliederung in Kapitel



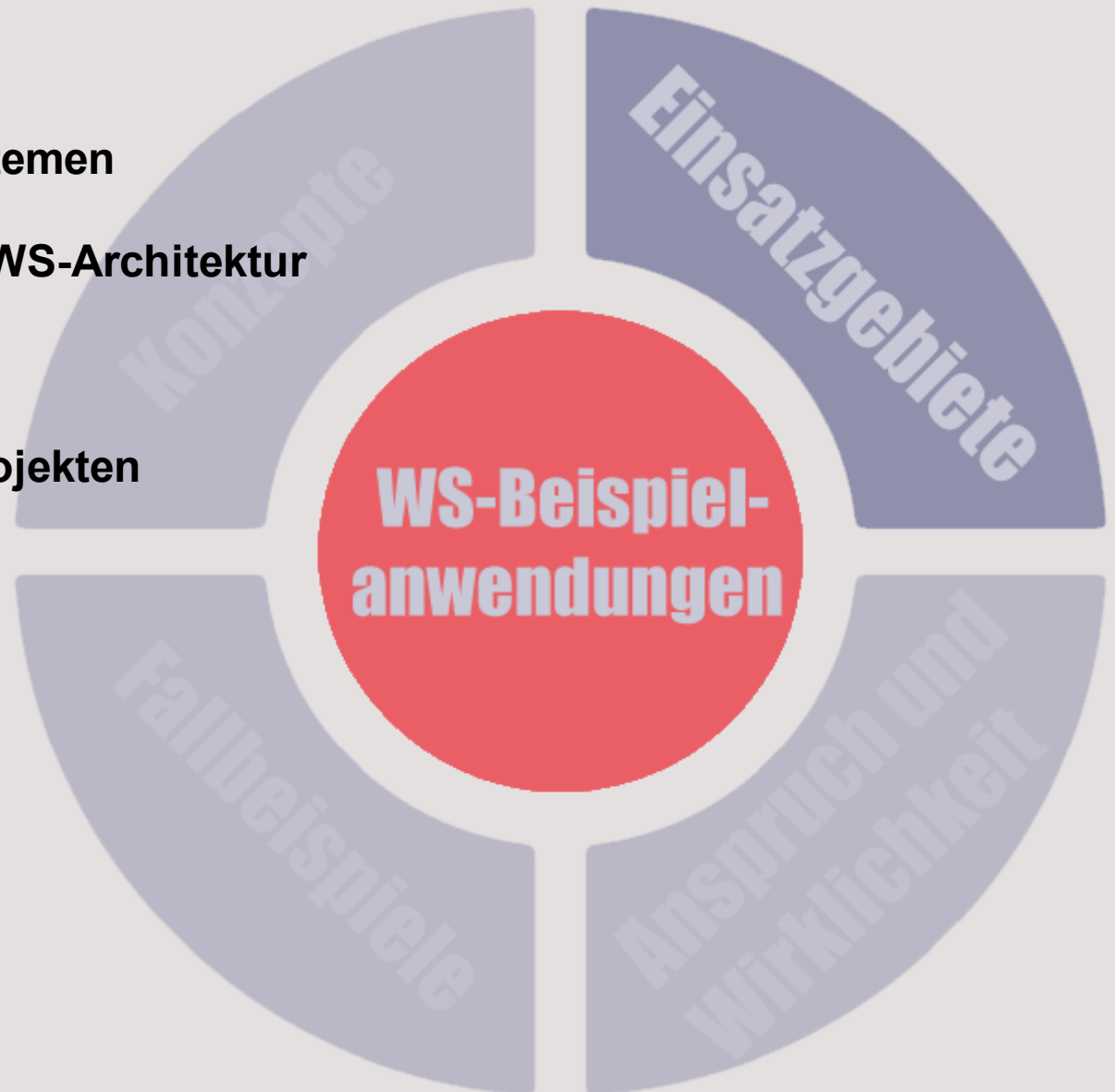
Kapitel 1 – Begriffe und Konzepte



Kapitel 2 – Häufige Einsatzgebiete von WS

Schwerpunkte:

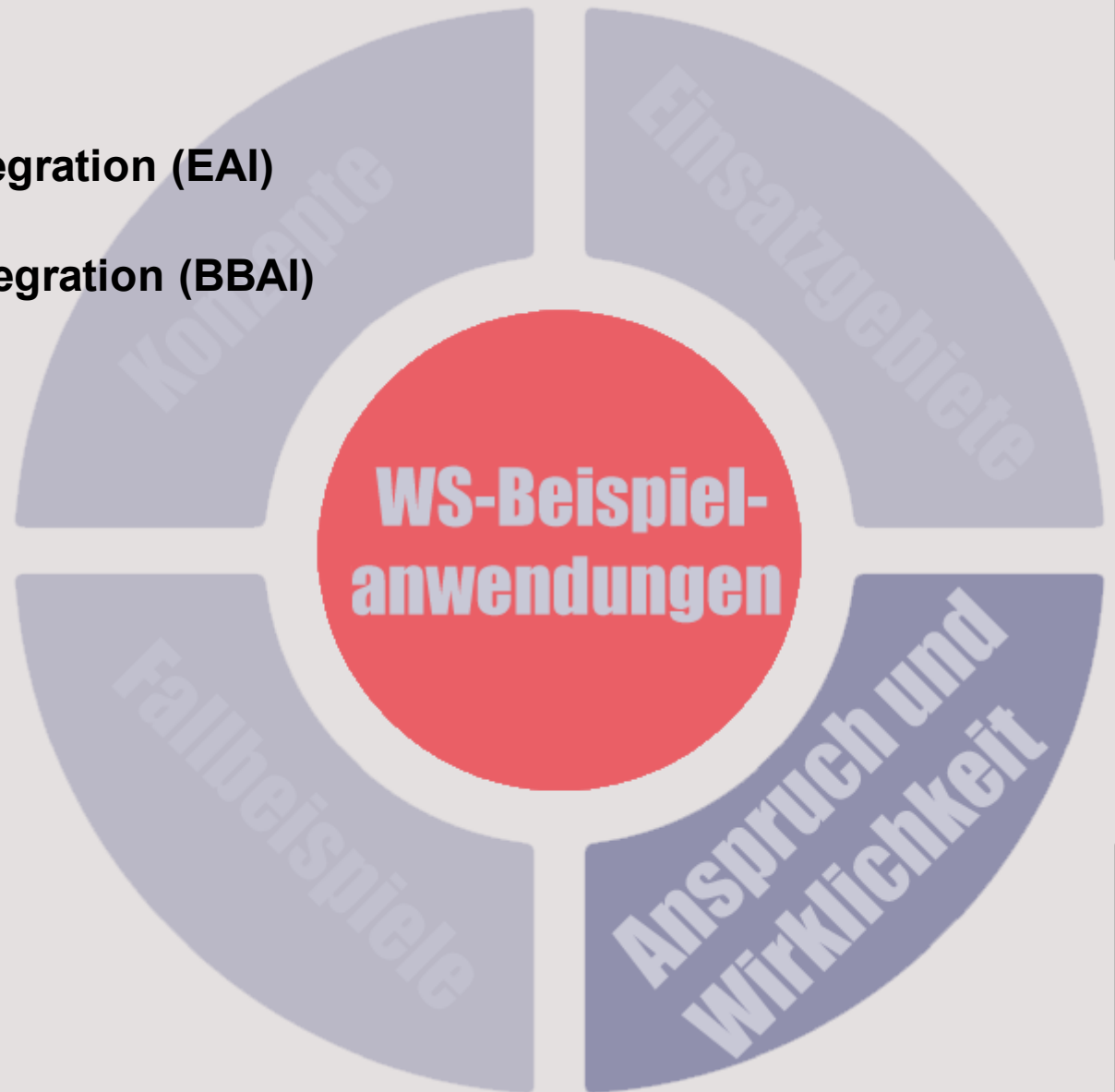
- ◆ Migration von Legacy-Systemen
- ◆ Prinzipieller Aufbau einer WS-Architektur
- ◆ Phasen einer Integration
- ◆ Vorteile bei Integrationsprojekten



Kapitel 3 – Anspruch und Wirklichkeit

Schwerpunkte:

- ◆ Verbreitung in interner Integration (EAI)
- ◆ Verbreitung in externer Integration (BBAI)
- ◆ Problem der Semantik



Kapitel 4 – Konkrete WS-Anwendungen



E-Business-Integration

◆ E-Business:

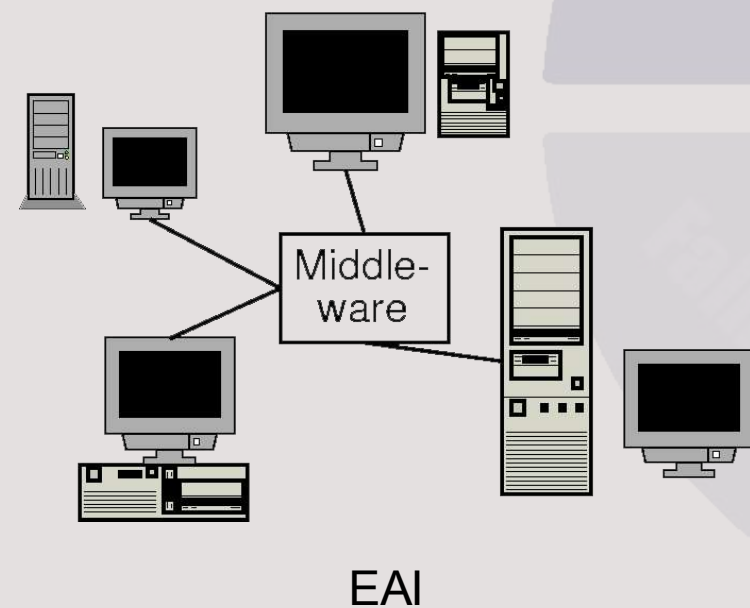
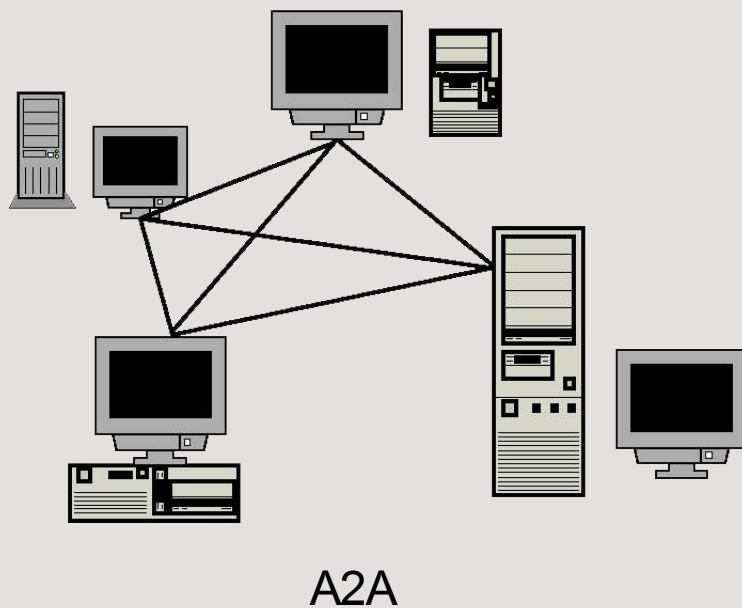
- **Geschäftstätigkeiten eines Unternehmens, welches seine Kernprozesse elektronisch abbildet und durchführt**

◆ Integration:

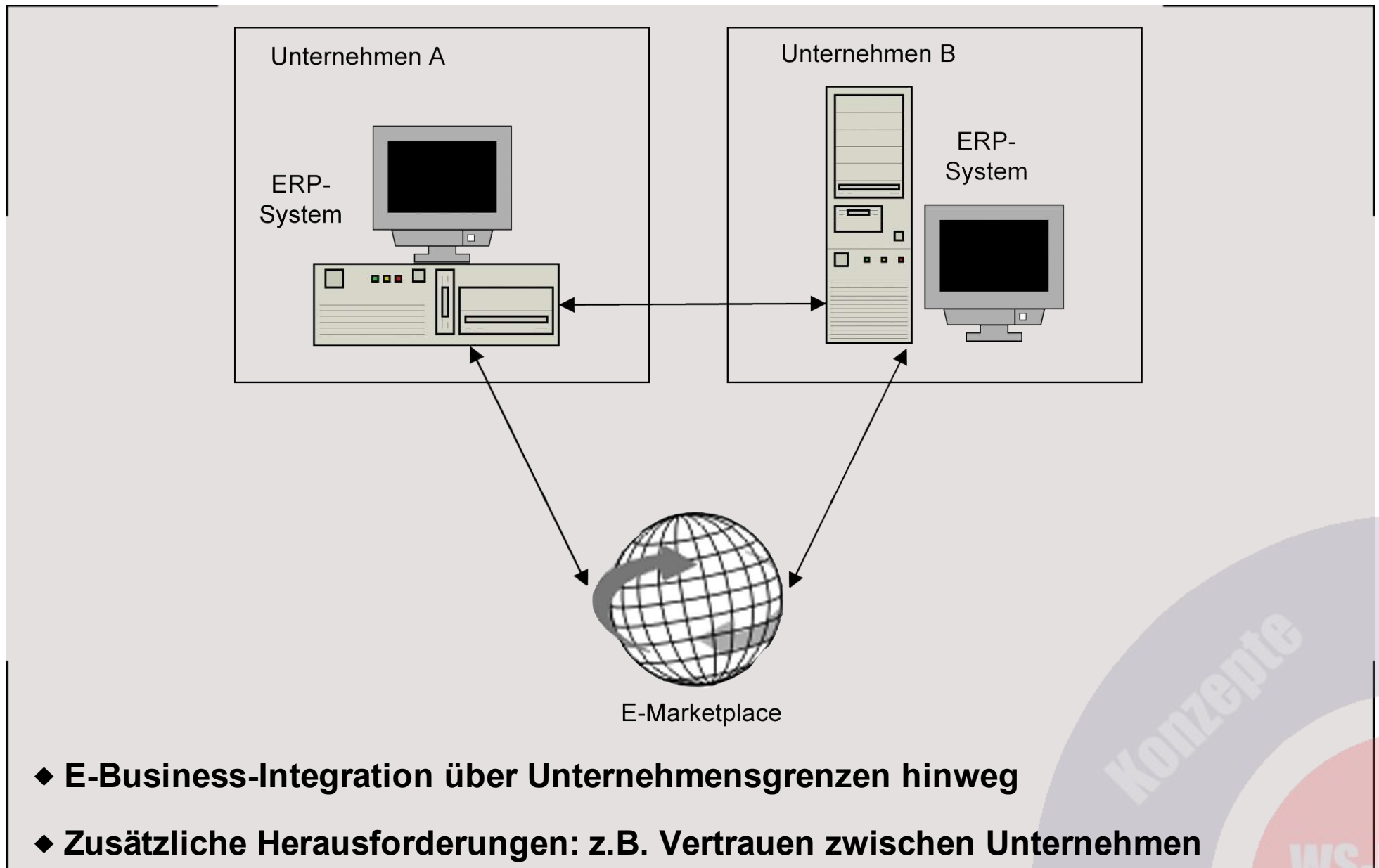
- **Verbindung von bisher allein stehenden (inkompatiblen) E-Business-Anwendungen**
- **Ziel: Austausch geschäftsbezogener Informationen und elektronische Abbildung von Geschäftsprozessen**

Enterprise Application Integration (EAI)

- ◆ **Application-to-Application-Integration (A2A):**
 - Viele Verbindungen zwischen den Systemen
 - Für jede Integration spezielle Integrationslogik erforderlich
- ◆ **Enterprise Application Integration (EAI):**
 - Systeme stehen nicht mehr direkt in Beziehung
 - Komplette Integrationslogik in Middleware-Komponente



B2B-Application-Integration (BBAI)

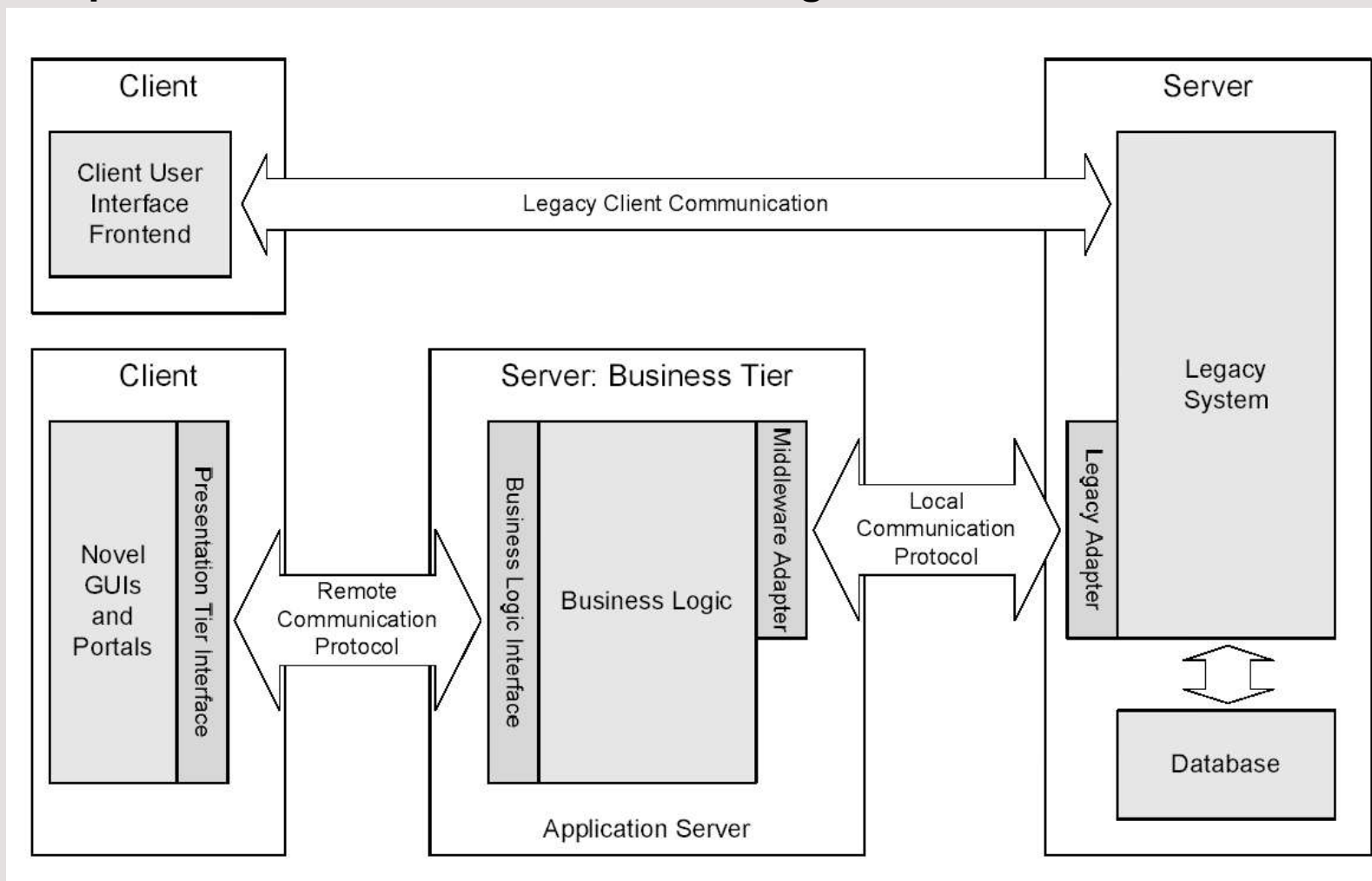


Migration von Legacy-Systemen

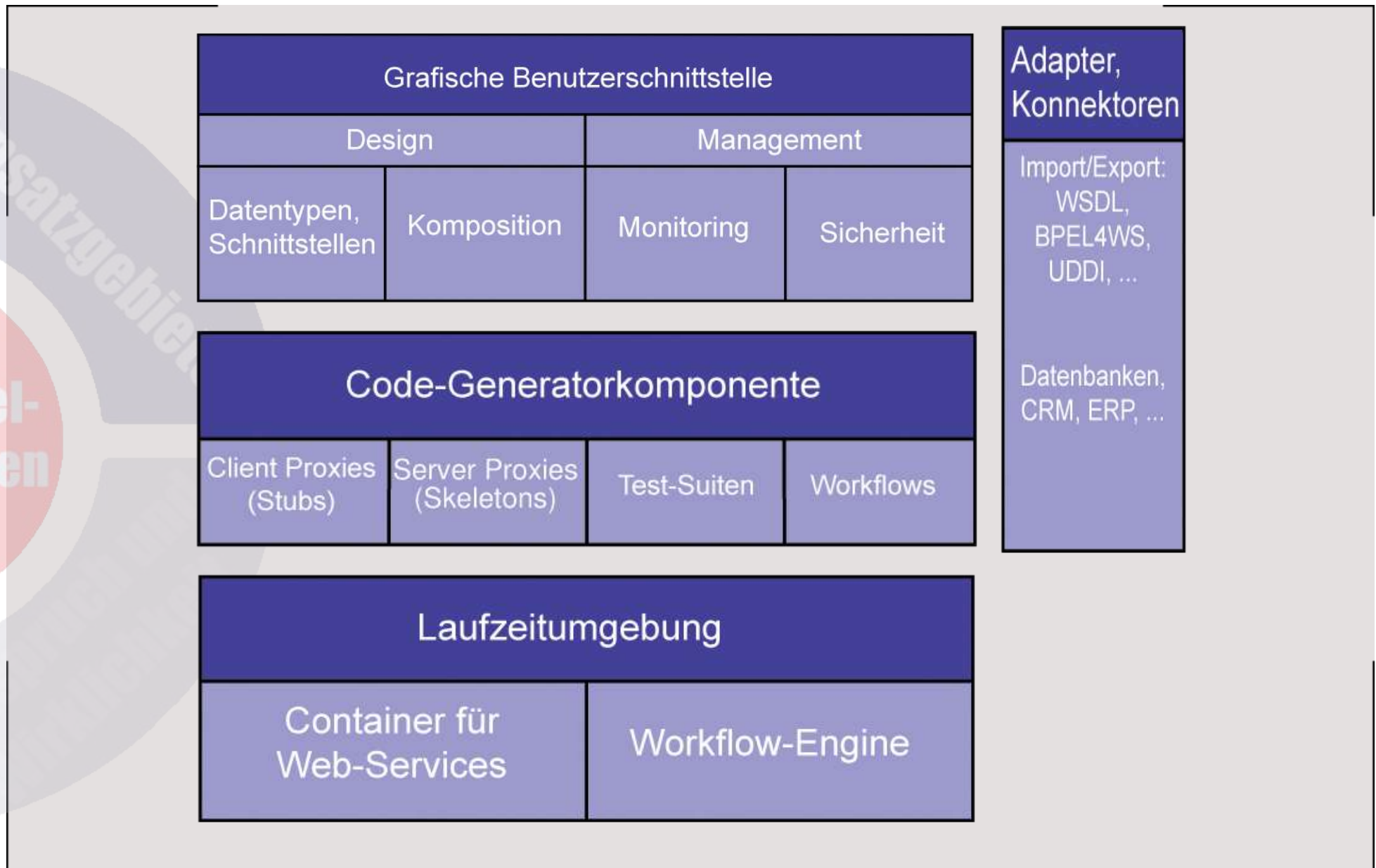
- ◆ **Legacy-Systeme sind häufig bestimmtem Unternehmensbereich zugeordnet**
 - Entsprechen nicht Ansprüchen des heutigen Software-Engineering-Stands
 - Monolithisch und komplex
 - Keine Trennung zwischen Präsentation, Geschäftslogik und Datenhaltung
- ◆ **Vereinfachte Integration durch Web Services**
 - Weiternutzung der Geschäftslogik des Altsystems
 - Investitionsschutz

Beispiel: Dublo-Architekturprinzip

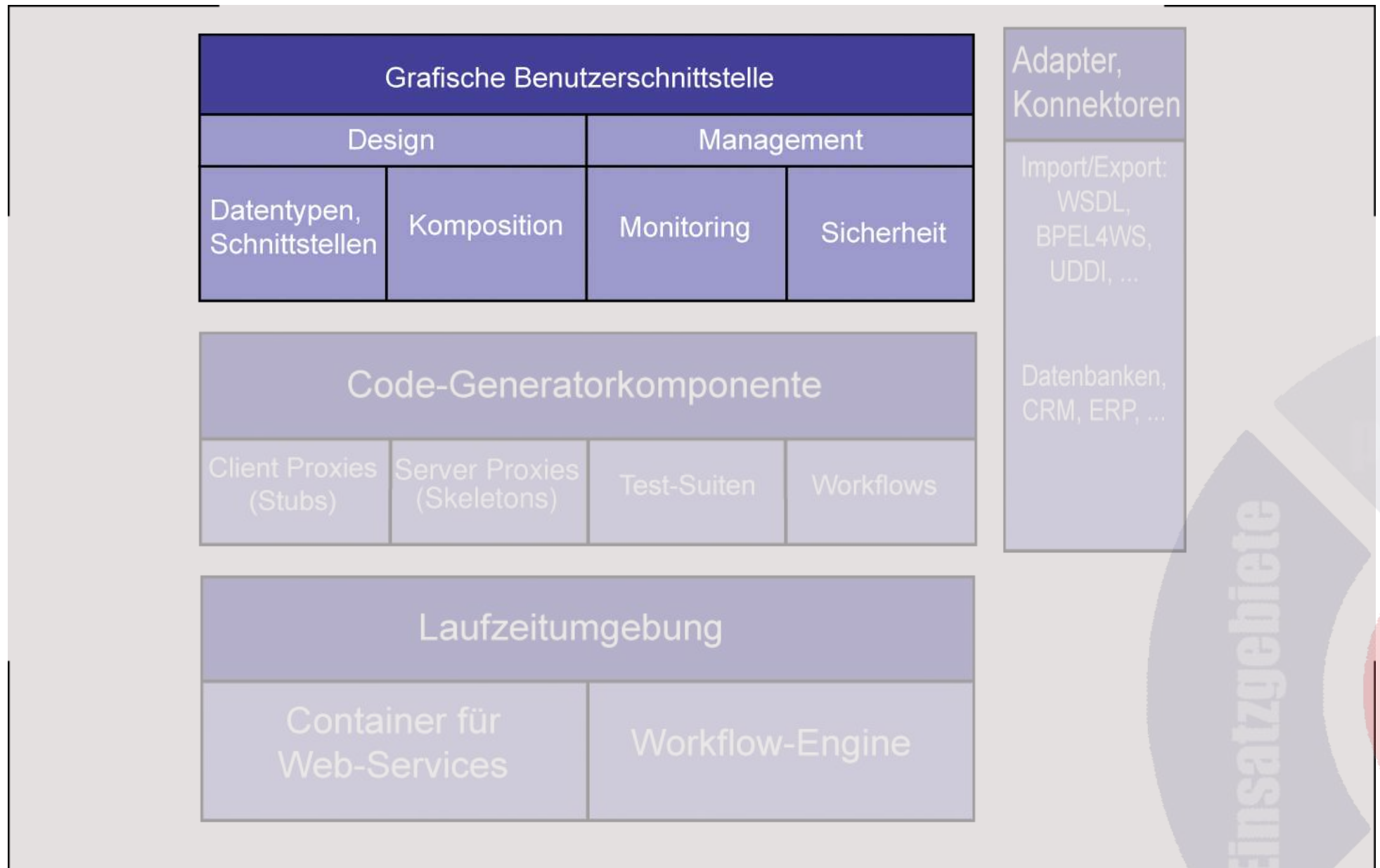
- ◆ Dublo = Dual business logic
- ◆ Über Adapter wird alte mit neuer Geschäftslogik verbunden



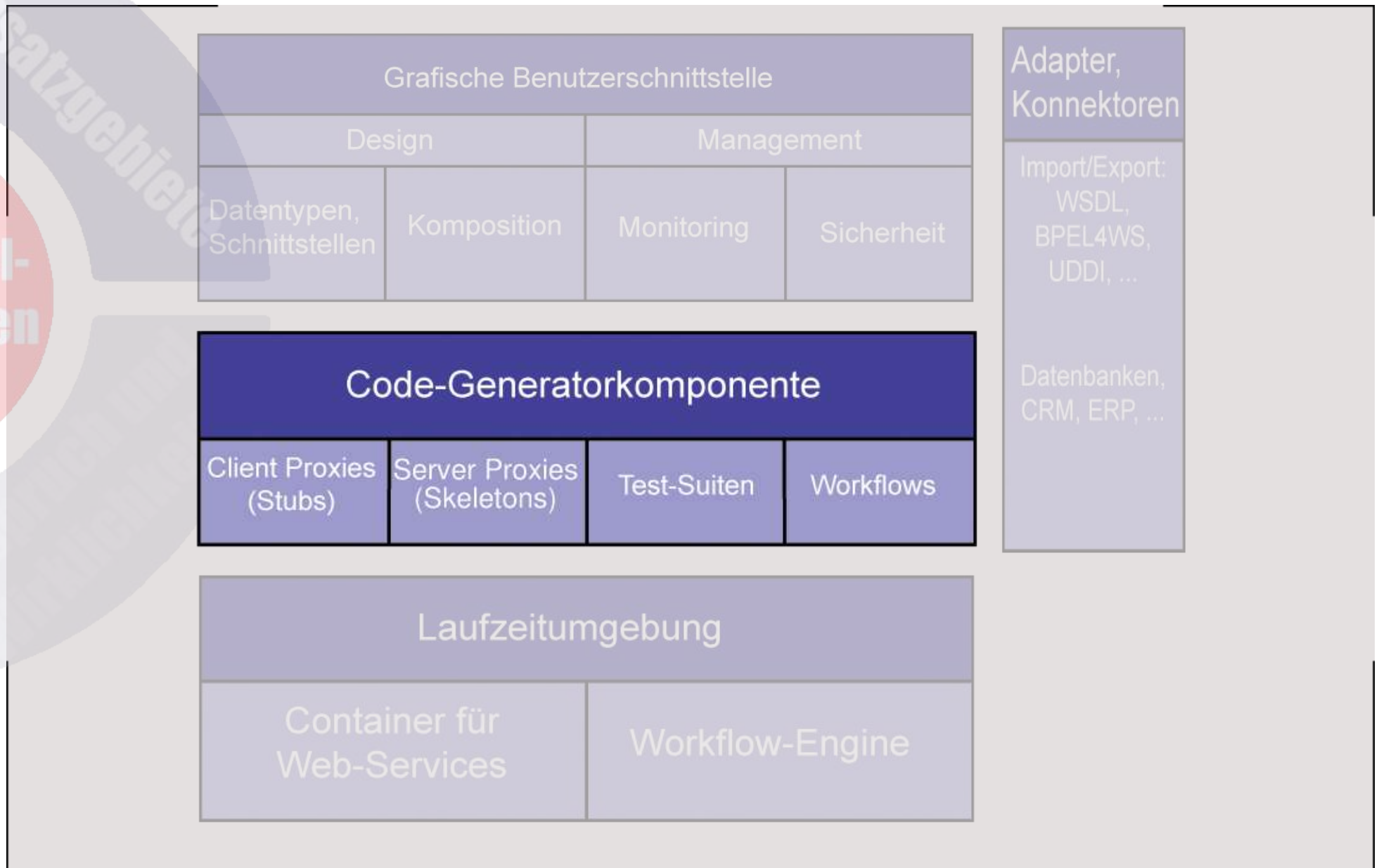
Architekturprinzip für WS-Plattformen



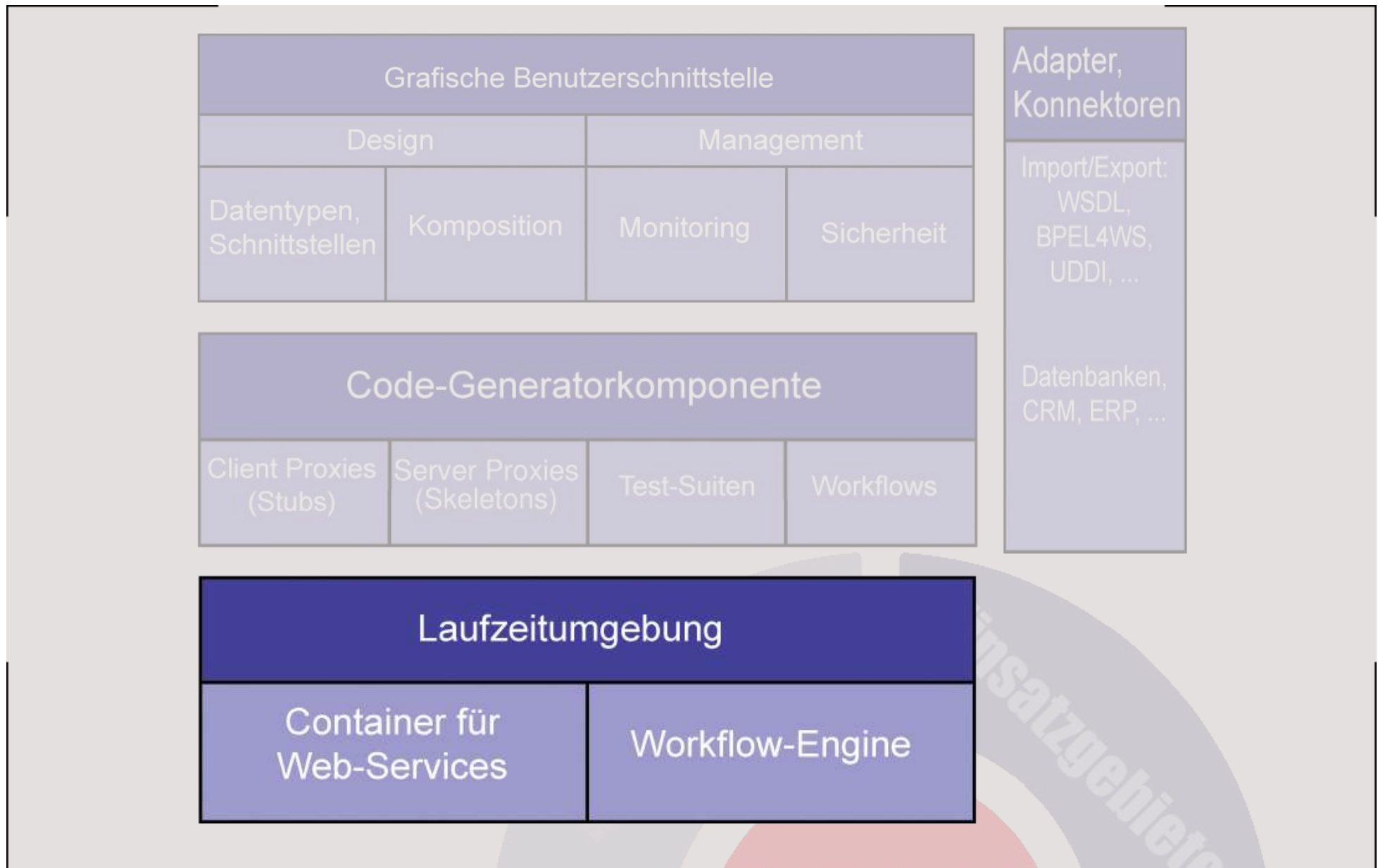
1. Komponente: GUI



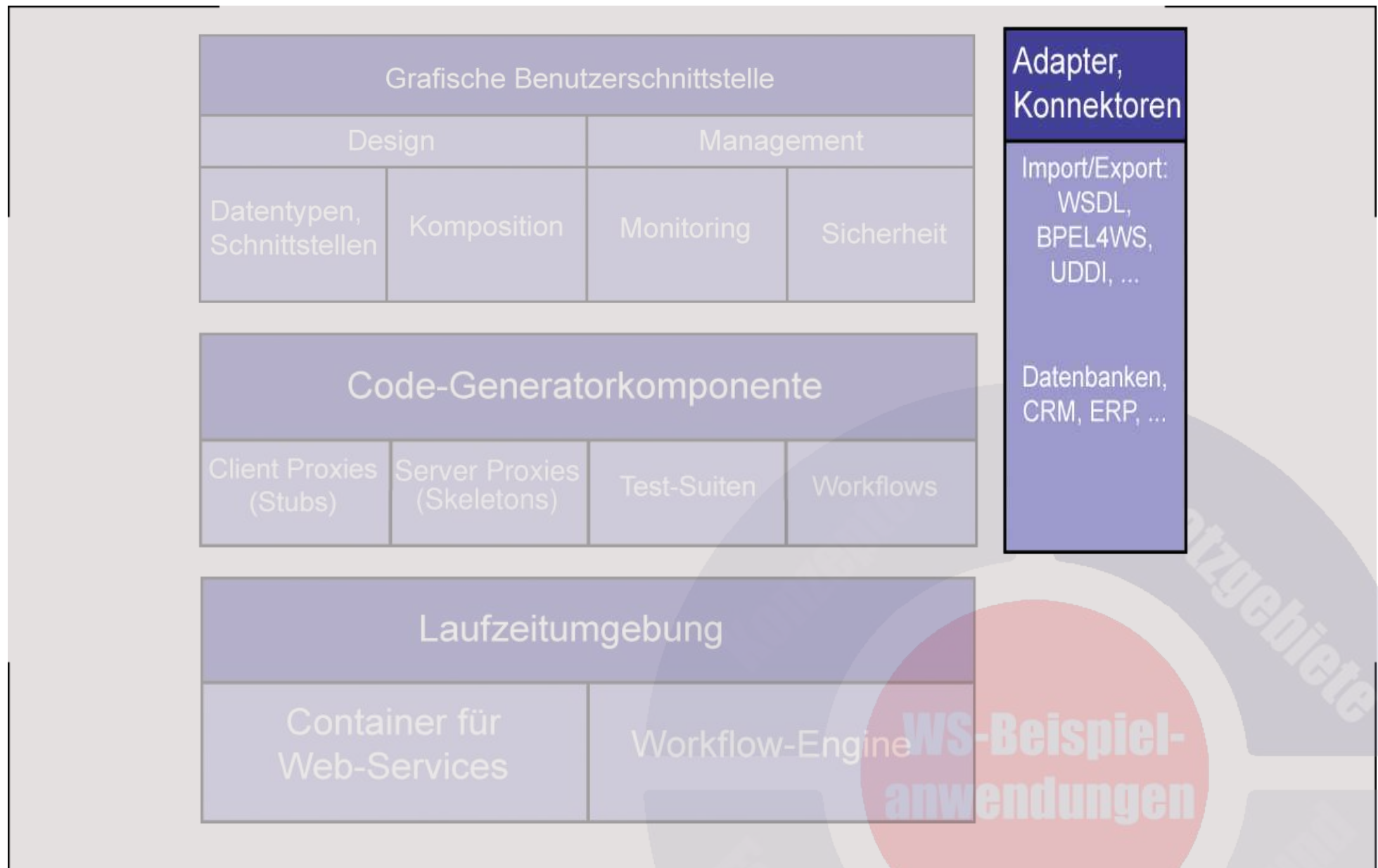
2. Komponente: Code-Generator



3. Komponente: Laufzeitumgebung



4. Komponente: Adapter und Konnektoren

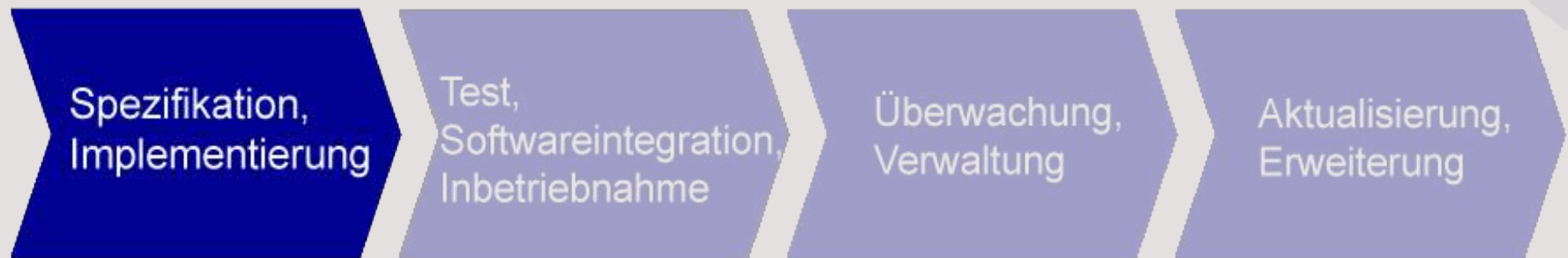


Phasen der Integration



Phase 1: Spezifikation und Integration

- ◆ Spezifikation von Web-Services-Schnittstellen und Datentypen
- ◆ Komposition von Workflows und Transaktionen
- ◆ Verarbeitung der Modelle:
 - Bottom-up-Verfahren
 - Top-down-Verfahren
- ◆ Manuelle Implementierung der Web-Services-Schnittstellenlogik erforderlich



Phase2: Integration, Inbetriebnahme

- ◆ Simulation der einzelnen Komponenten (Operationen, Transaktionen, ...)
- ◆ Überwachung von Geschäftsprozessinteraktionen durch Test-Suiten:
 - Künstliche SOAP-Requests, SOAP-Replies als Eingaben der Web-Services-Komponenten
 - Simulation der einzelnen Komponenten (Operationen, Transaktionen, ...)
- ◆ Inbetriebnahme nach der Test- und Integrationsphase



Einsatzgebiete

Phase3: Überwachung und Verwaltung



◆ Fortlaufende Überwachung notwendig

→ Sicherstellung von Konsistenz der Daten

→ Häufig Verwendung von Logging- und Recovery-Komponenten

◆ Bereitstellung grafischer Werkzeuge zur Visualisierung der Abläufe durch GUI

Phase4: Aktualisierung und Erweiterung

- ◆ Änderung und Erweiterungen
- ◆ Aufwand hängt stark von Art der Modifikation ab



Ultimative Vision von Web Services

- ◆ **Web Services als Basis für B2B-Integrationen**
 - **Clients suchen in UDDI-Repositories nach geeigneten Web Services**
 - **Automatisches Herausfinden wie Interaktion mit Web Service vonstatten geht**
 - **Automatisches Aufrufen des Web Services**

WS-Beispiel-
anwendungen

Web Services als Integrationstechnologie

- ◆ **In Realität häufiger Einsatz von Web Services für EAI**
 - Unternehmensrisiko kalkulierbar
 - Sammeln von Erfahrungen mit WS-Technologien für komplexeren Einsatz
- ◆ **Seltener Einsatz in BBAI**
 - Kosten-Risiko-Verhältnis von vielen Unternehmen als zu hoch erachtet
 - Sehr viel komplexer als EAI

Skepsis bezüglich Web Services

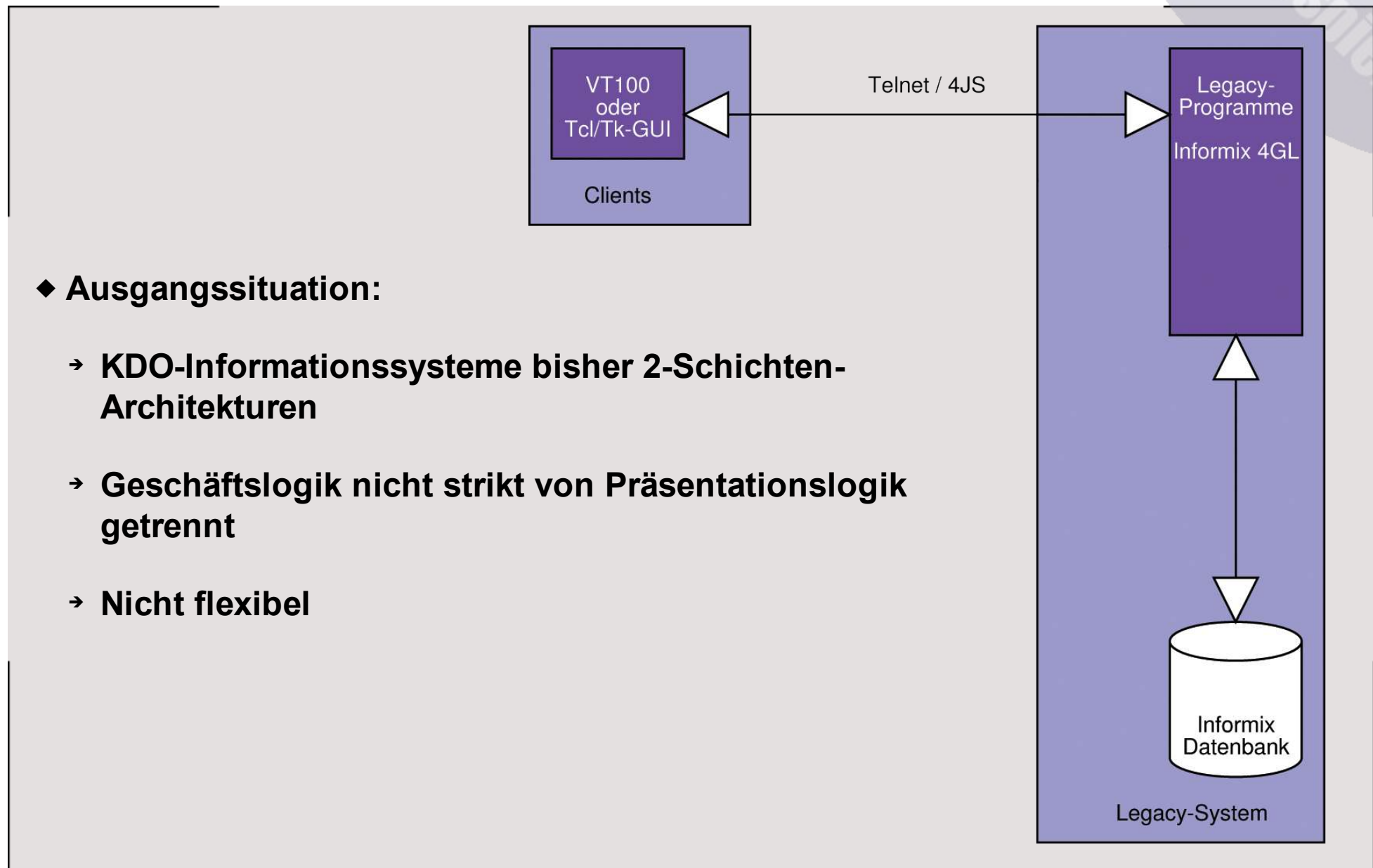
- ◆ Befragung von 108 deutschen Unternehmen zu Web Services, Sommer 2002

Hindernisse für eine Verwendung von Web-Services

Welche Hindernisse sehen Sie für eine Verwendung von Web-Services?

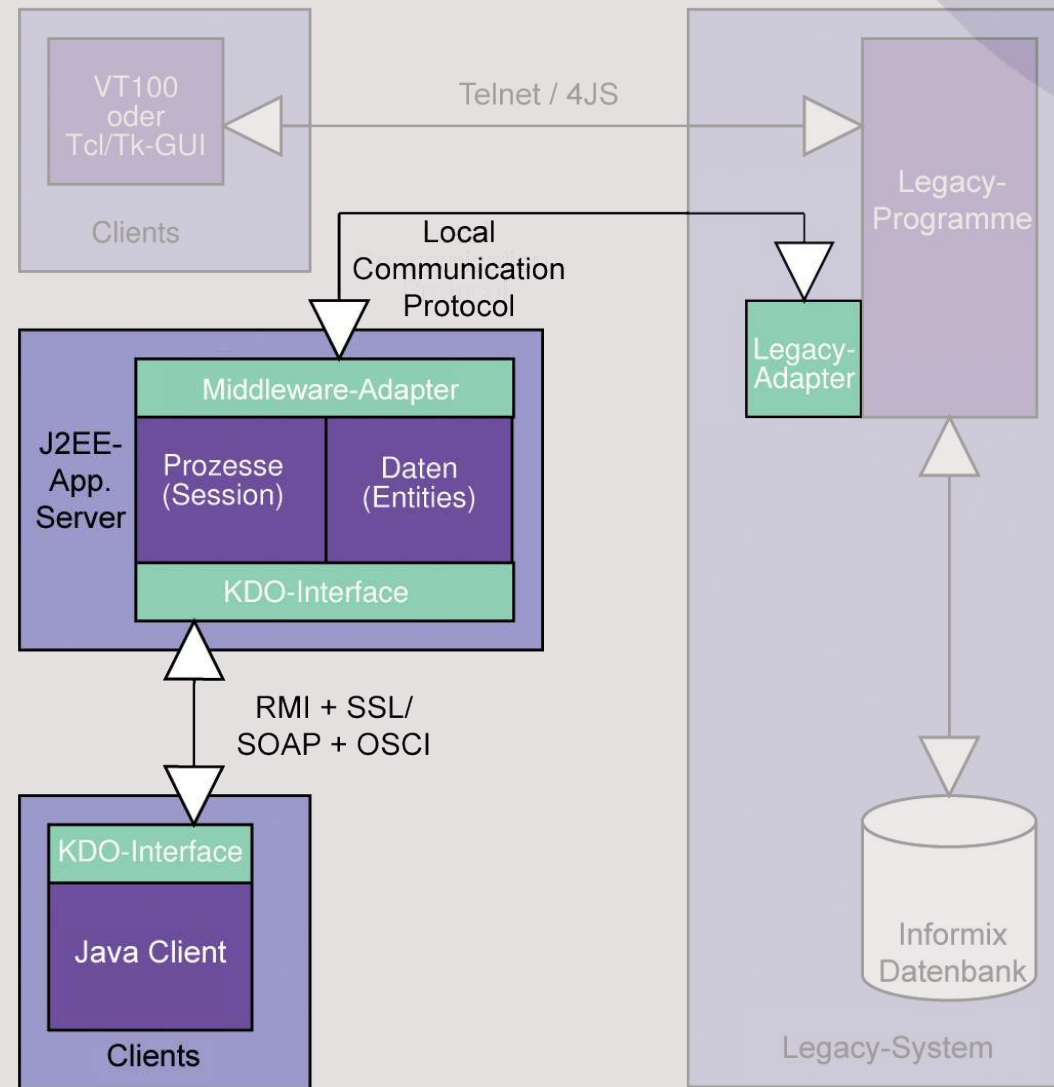


Fallstudie – Migration von Altsystemen

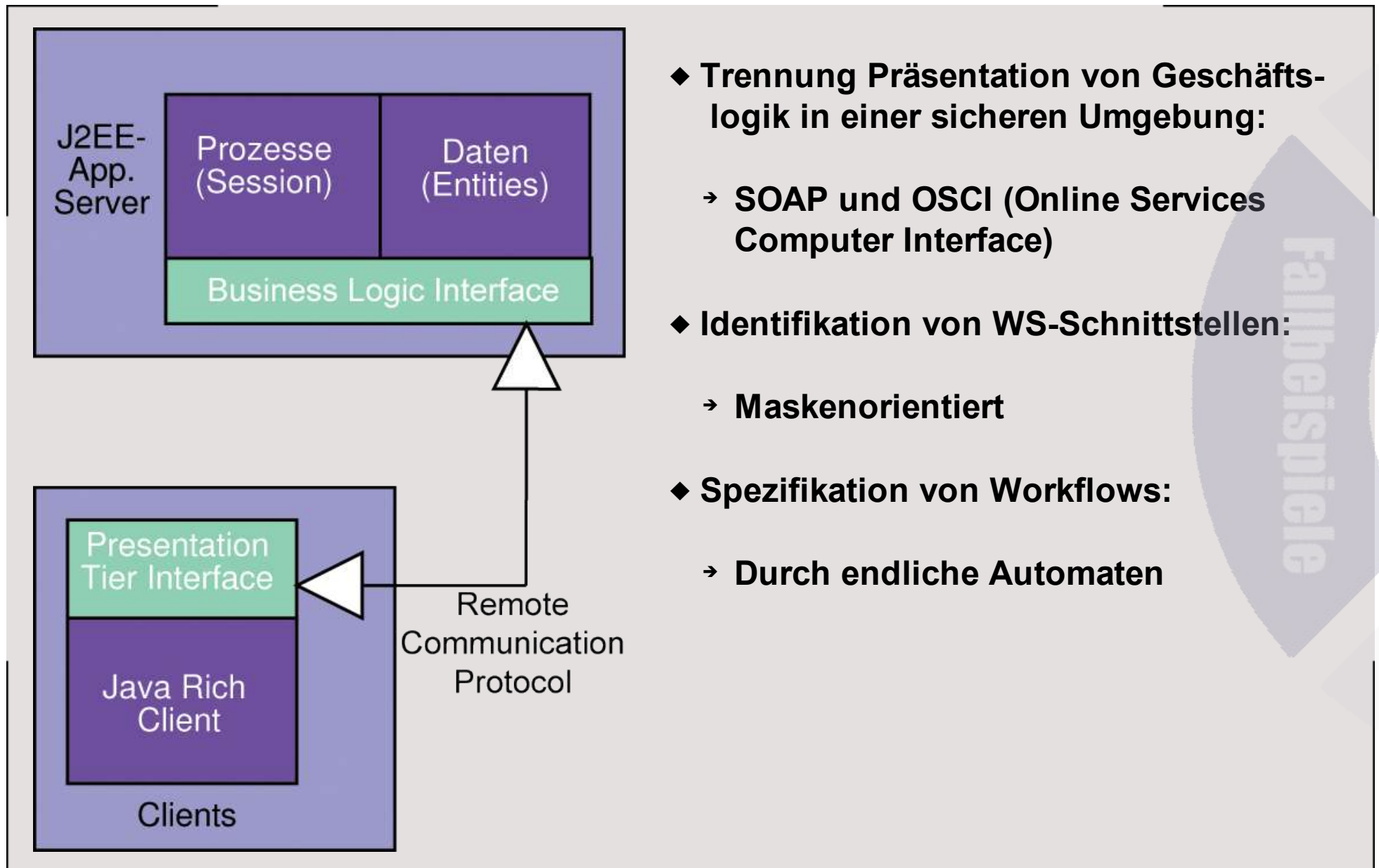


Integrationsphase

- ◆ **Moderne Mehr-Schichten-Architektur (J2EE)**
- ◆ **Trennung zwischen Präsentation, Geschäftslogik, Datenhaltung (EJB)**
- ◆ **Web Services als Integrationstechnologie**
- ◆ **Geschäftslogik der Legacy-Systeme wird weiter genutzt (Dublo-Architekturprinzip)**



Vollständige Migration des Altsystems



- ◆ **Trennung Präsentation von Geschäftslogik in einer sicheren Umgebung:**
 - **SOAP und OSCI (Online Services Computer Interface)**
- ◆ **Identifikation von WS-Schnittstellen:**
 - **Maskenorientiert**
- ◆ **Spezifikation von Workflows:**
 - **Durch endliche Automaten**

Fallstudie – WS als Integrationstechnologie

◆ Ausgangssituation:

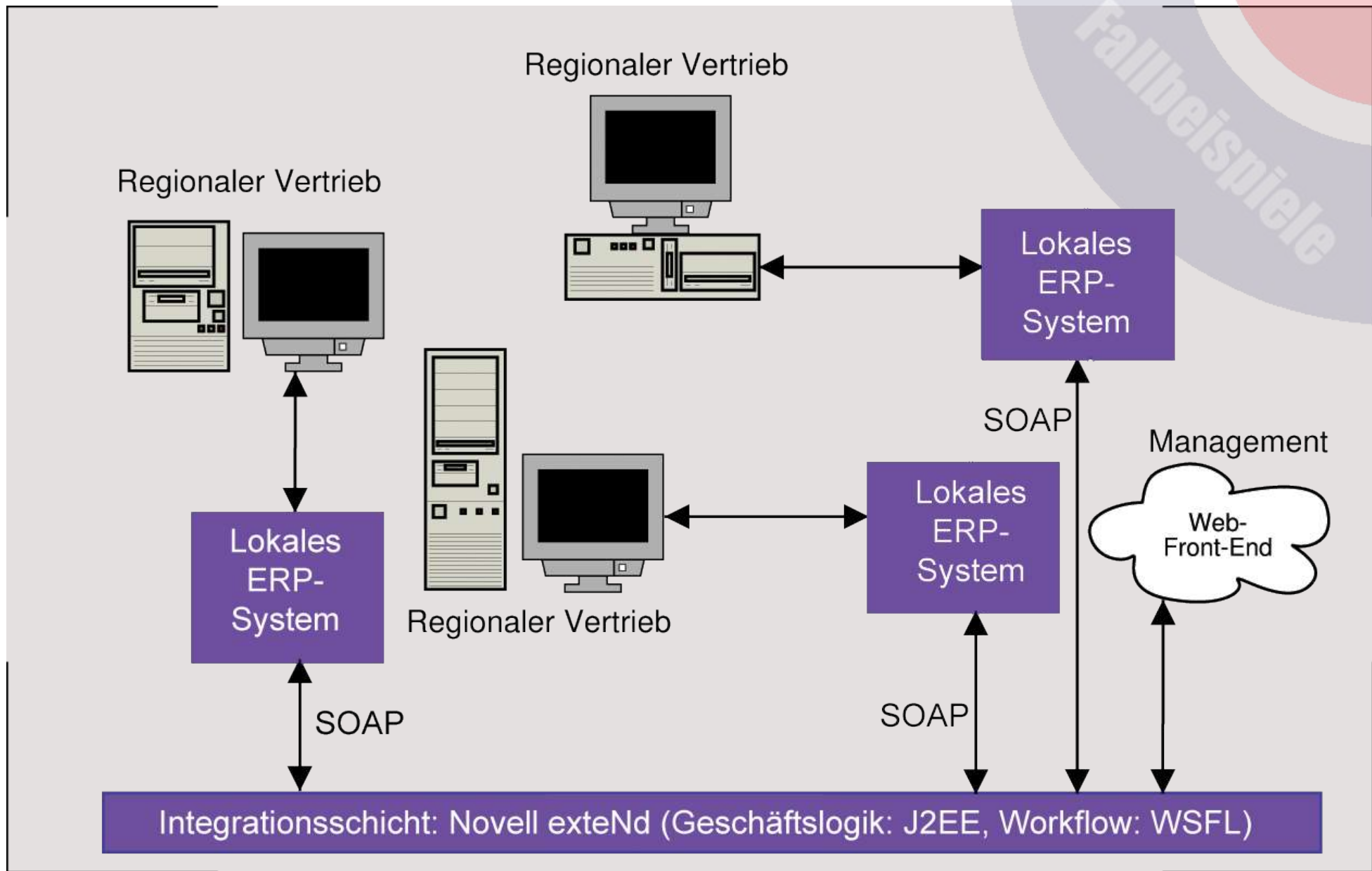
- Acer dezentraler Konzern mit vielen unabhängigen Vertriebsabteilungen
- 13 unterschiedliche ERP-Systeme weltweit

◆ Lösungsansatz:

- Integration der ERP-Systeme zu einer webbasierten Anwendung
- auf Web Services basierendes Bestellmanagementsystem

Technologische Umsetzung

Fallbeispiele



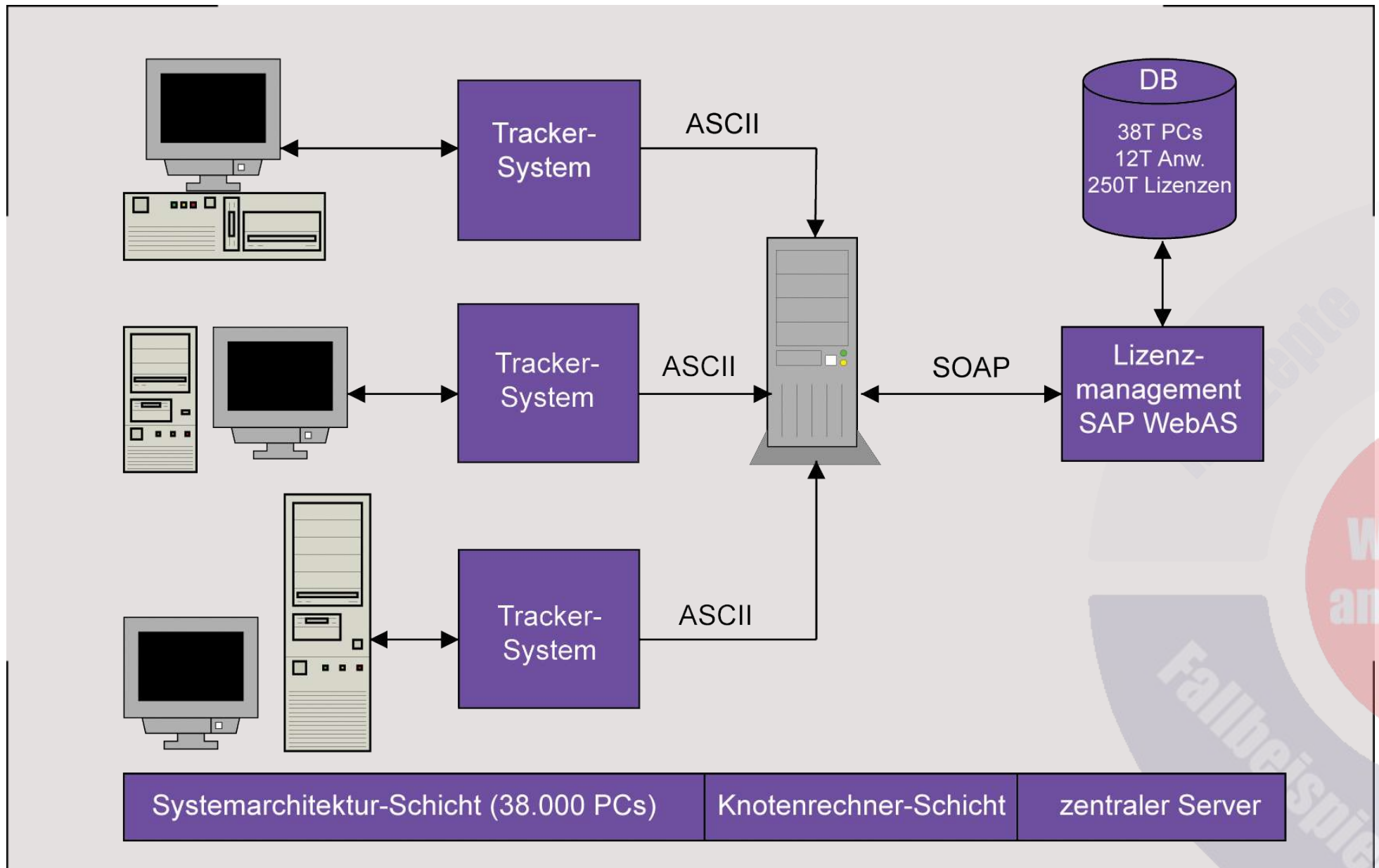
Technologische Umsetzung (2)

- ◆ **Basis: E-Business-Plattform ExteNd Application Server von Novell**
- ◆ **Implementierung der Geschäftslogik durch J2EE**
 - Erlaubt automatische Generierung der Web-Services-Schicht
- ◆ **Einsatz von WS: Kommunikation zw. ERP-Systemen / Teile der Geschäftslogik**
- ◆ **Workflows durch WSFL (Web Service Flow Language) realisiert**
- ◆ **3 verschiedene Service-Typen:**
 - SOAP
 - Proprietäre XML-Lösung
 - Java-API

Lizenzmanagement

- ◆ In Unternehmen: Häufig erhöhte Softwarekosten wegen Überlizenzierung
- ◆ Lösung: Lizenzmanagementsystem:
 - Basis: SAP Web Application Server (SAP WebAS), J2EE (EJB), Web Services
- ◆ 3-Schichten-Architektur:
 - Rechnerinfrastruktur (38.000 Rechner)
 - Für jede Abteilung einen Knotenrechner
 - Lizenzserver (Operationen durch Web Services realisiert)
- ◆ Umsetzung (WSDL-Spezifikationen von WebAS erzeugt):
 - Bottom-up: Generierung der Web Services direkt aus Session Beans
 - WSDL-Schnittstellen: Erzeugung von Proxies für Clients und Server

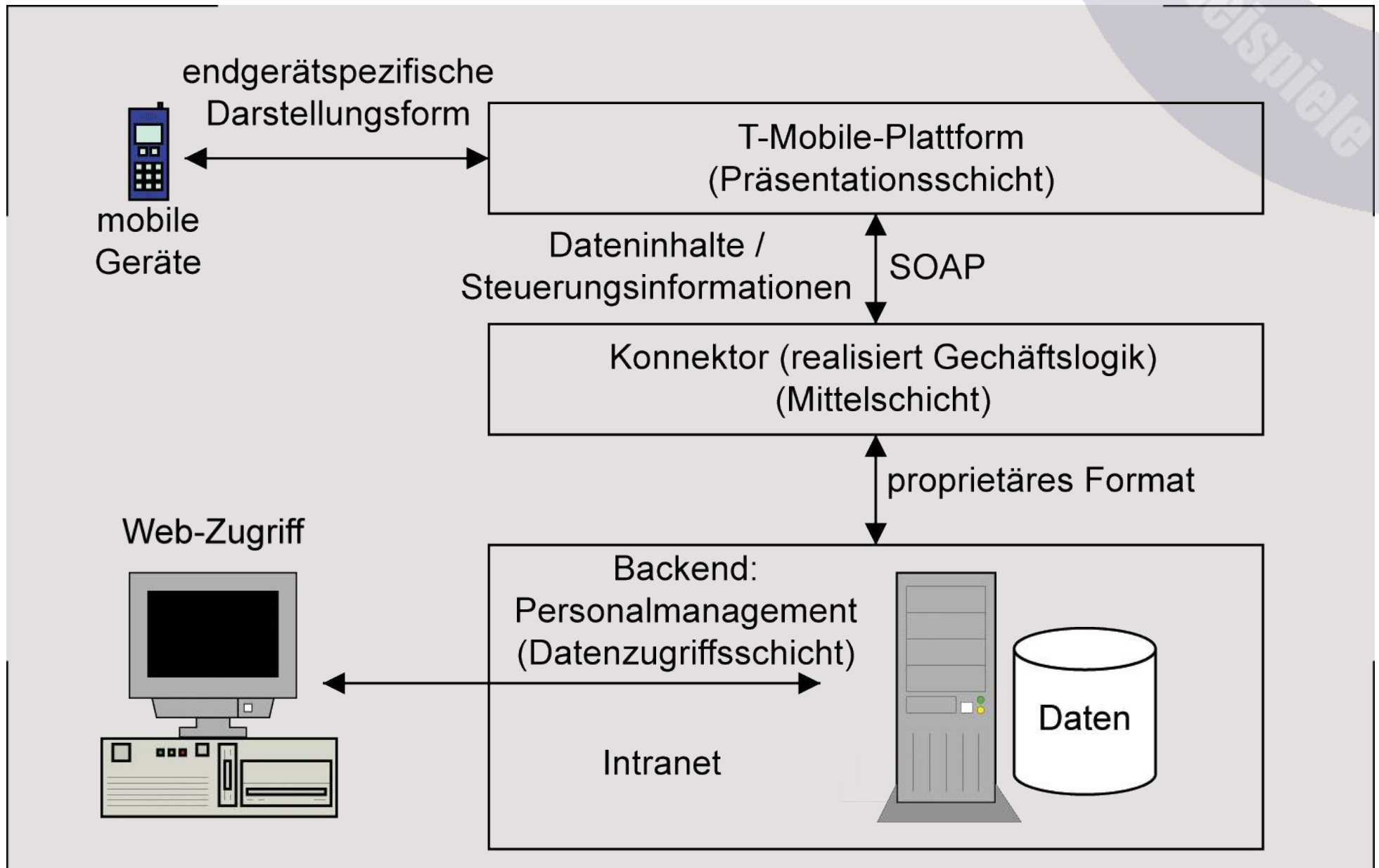
Infrastruktur Lizenzmanagementsystem



Leistungs- und Arbeitszeiterfassung

- ◆ **Ausgangssituation:** Mitarbeiter von IT-Unternehmen operieren beim Kunden
 - **Arbeitszeit-/Leistungserfassung** am Monatsende in Personalmanagementsys.
 - **Probleme:** Abrechnungsfehler, zu spät erkannte Fehlbuchungen, ...
- ◆ **Lösung:** Mobiles Leistungs- und Arbeitszeiterfassungssystem:
 - **Basis:** T-Mobile Service Integration Platform (SIP), basiert auf .NET
 - **Zugriff auf Backend** mit mobilen Geräten (Handy, Laptop, ...) möglich
- ◆ **3-Schichten-Architektur:**
 - **SIP (Präsentationsschicht)**
 - **Konnektor (Mittelschicht)**
 - **Backend: Personalmanagementsystem (Datenzugriffsschicht)**
- ◆ **Pragmatischer Ansatz:** WS auf Einsatz von SOAP beschränkt

Architektur des Systems



Fazit

- ◆ **Web-Services-Technologie erfüllt Anforderungen komplexer Integrationen:**
 - **Autorisierung, Authentisierung, Integrität, ...**
 - **Komplexe Workflows leicht modellierbar, änderbar, visualisierbar**
 - **Monitoring- und Simulationskomponenten**
 - **Interoperabilität, Flexibilität, Investitionsschutz**
- ◆ **Verbreitung von Web Services**
 - **Häufig unternehmensinterne Integration**
 - **Sobald verschiedenen Standards (Sicherheit, Komposition, Management) etabliert sind, auch externe Integrationen weiter verbreitet**