

**Vorlesung Wintersemester 2011/12**

# **Workflow-Management-Systeme**

## **Kapitel 12: Architektur und Implementierung von WfMS**

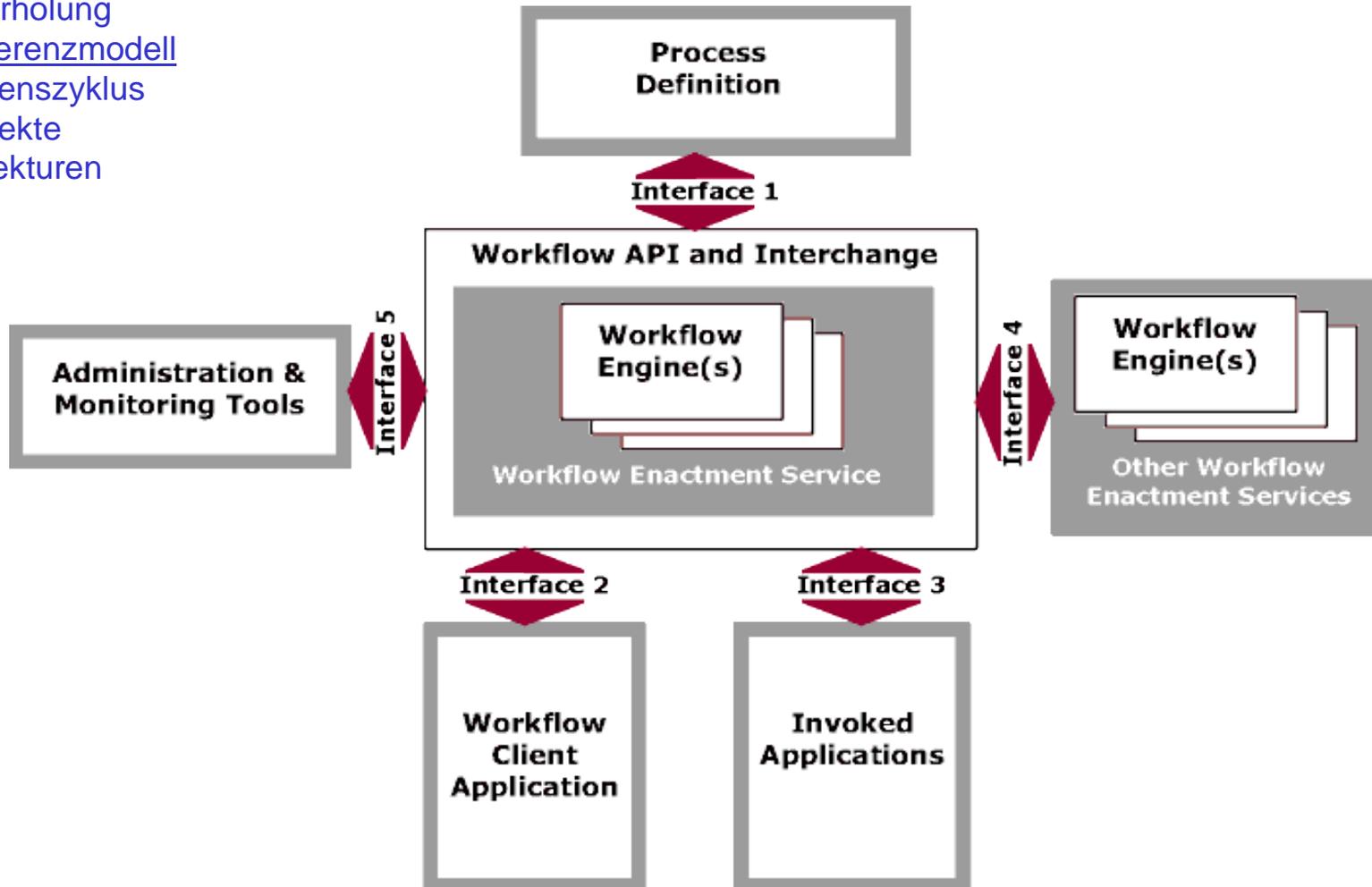
Lehrstuhl für Systeme der Informationsverwaltung, Prof. Böhm  
Institut für Programmstrukturen und Datenorganisation (IPD)

# Überblick Kapitel 12

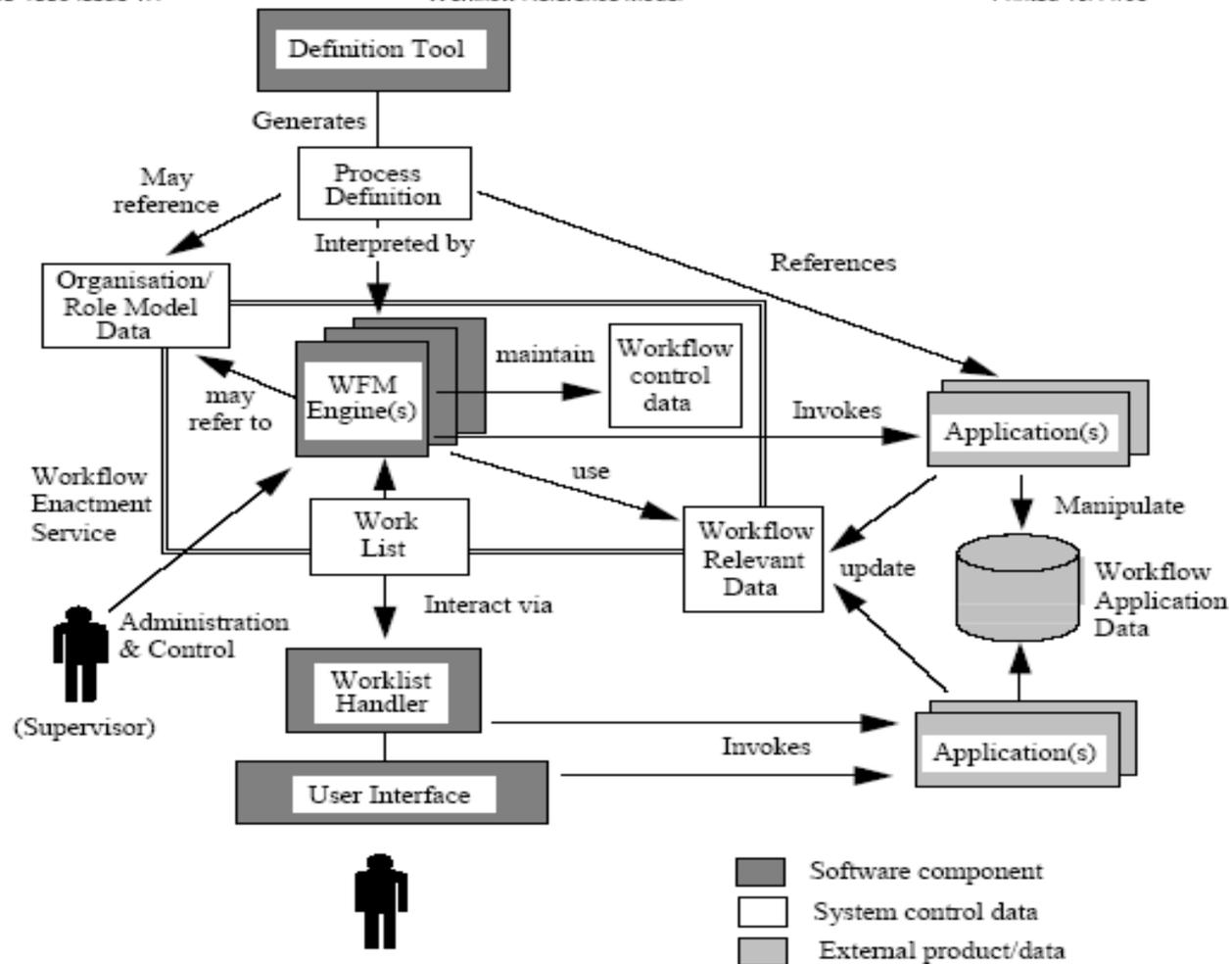
- ◆ Wiederholung Kapitel 2
  - Referenzmodell der WfMC
  - Lebenszyklus von Workflows
  - Aspekte und Komponenten
- ◆ Beispiele für Architekturen
  - MENTOR, Mentor lite
  - YAWL
  - ADEPT2

# Referenzmodell der WfMC (Wdh. Kap. 2)

Wiederholung  
Referenzmodell  
Lebenszyklus  
Aspekte  
Architekturen

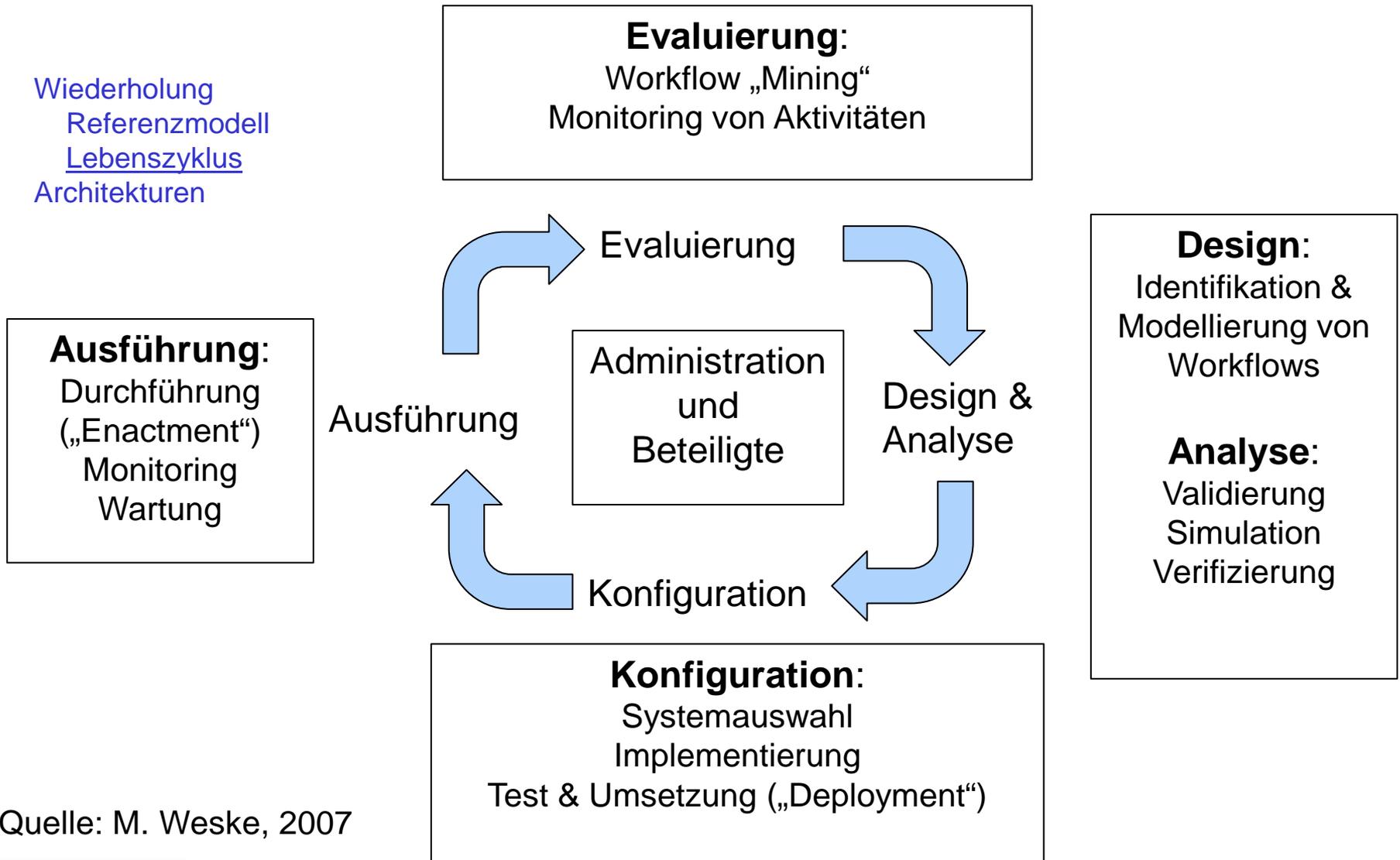


# Generische Architektur nach WfMC (Wdh. Kap. 2)



# Lebenszyklus von Workflows (Wdh. Kap. 2)

Wiederholung  
Referenzmodell  
Lebenszyklus  
Architekturen



Quelle: M. Weske, 2007

# Implementierungsmodell eines WfMS (Wdh. Kap. 2)

- ◆ Ziel: Beschreibung des implementierungs**unabhängigen** Teils der Architektur eines WfMSs.
- ◆ Beschreibung der wesentlichen Module oder Komponenten eines WfMSs.
- ◆ Die wesentlichen funktionalen Komponenten der Architektur eines WfMSs umfassen Module, welche die Aspekte eines WfMSs realisieren.
- ◆ Ergänzend ist ein Steuermodul (nicht unbedingt zentral!) erforderlich sowie
- ◆ Hilfsmodule, welche allgemeine Dienste zur Verfügung stellen.

# Wie kommt man zur Architektur eines Workflow Management Systems? (Wdh. Kap. 2)

- ◆ Beim Übergang vom Implementierungsmodell zur Implementierungsarchitektur sind folgende Entscheidungen zu treffen:
  - Werden Module durch Basissysteme (z.B. Betriebssystem, Middleware) unterstützt?
  - Wie werden die (persistenten) Daten einer Komponente verwaltet?
  - Wie wird die Kommunikation zwischen den Modulen realisiert?
- ◆ Architekturansätze unterscheiden sich stark hinsichtlich dieser drei Fragen, daher keine allgemeine Implementierungsarchitektur
- ◆ Die Konkretisierung der Implementierungskonzepte erfolgt in der eigentlichen Implementierungsphase

# Anforderungen an verteilte WfMSe (Wdh. Kap. 2)

- ◆ Funktionale Anforderungen, die vor allem durch die Workflow-Sprache und ihre Ausdrucksfähigkeit und durch Vorgaben bezüglich der Benutzerschnittstellen bestimmt sind.
- ◆ Nichtfunktionale Anforderungen, die vor allem durch das beabsichtigte Einsatzgebiet eines Workflow-Management-Systems beeinflusst werden.

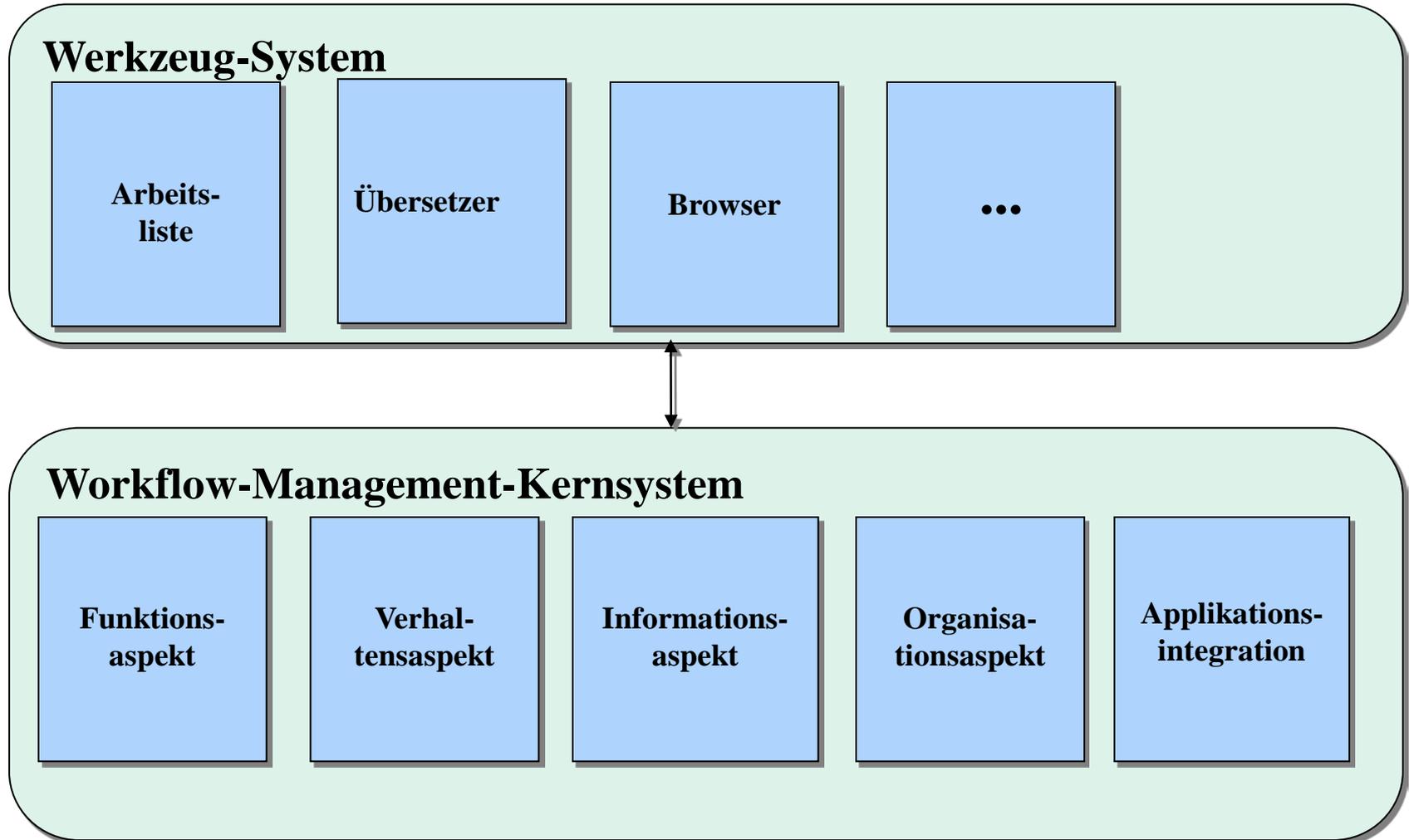
# Funktionale Anforderungen (Wdh. Kap. 2)

- ◆ Implementierung des Workflow-Meta-Schemas
  - Funktionsaspekt
  - Verhaltensaspekt
  - Informationsaspekt
  - Organisationsaspekt
- ◆ Applikationsintegration (Operationsaspekt): Nutzung der Eigenschaften externer Programme, z.B. hinsichtlich transaktionaler Ausführung
- ◆ Bereitstellung von Benutzerschnittstellen
  - Anwender
  - Entwickler
  - Administrator

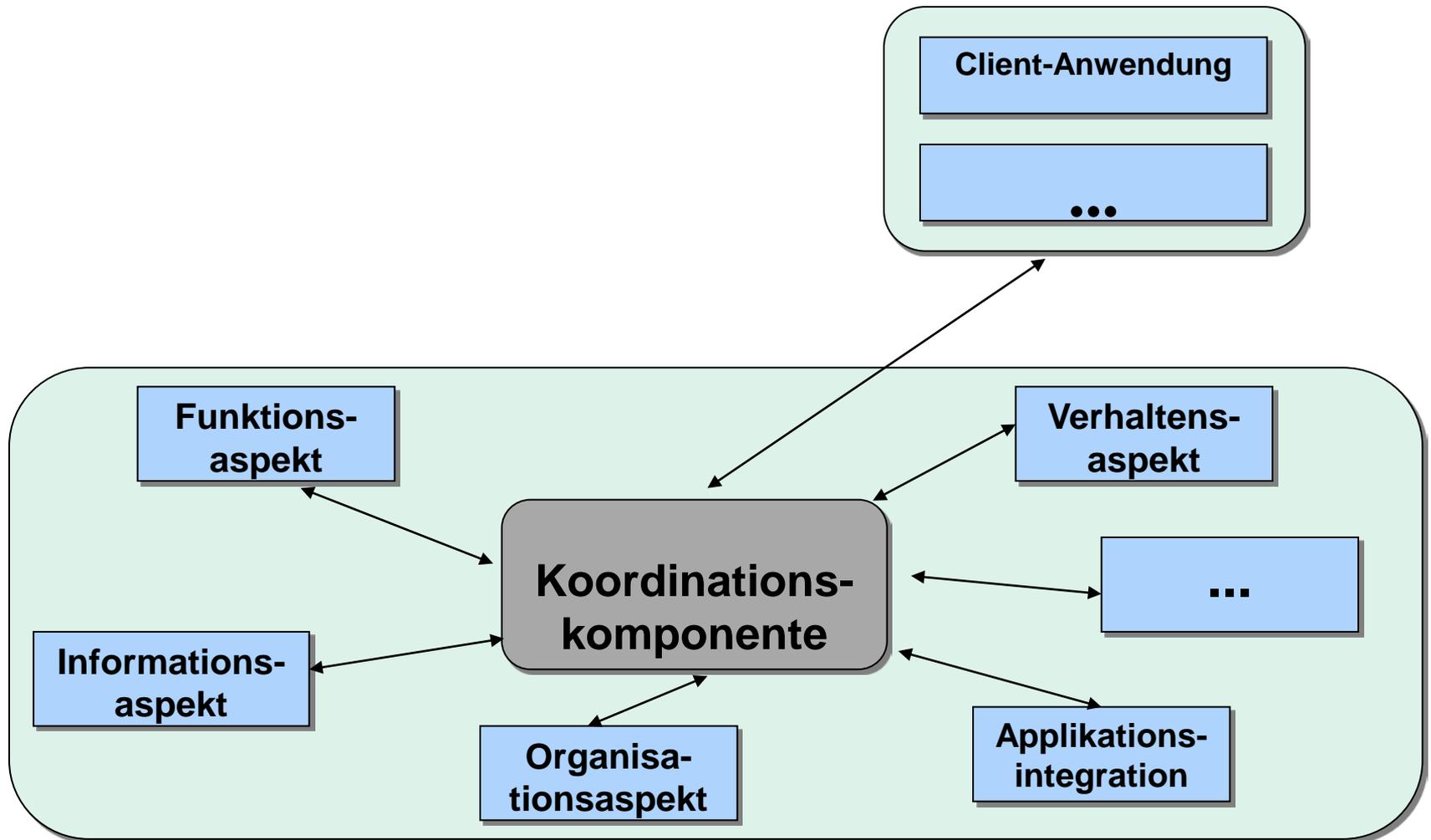
# Nichtfunktionale Anforderungen (Wdh. Kap. 2)

- ◆ Offenheit der System-Implementierung auf verschiedenen Ebenen
  - Funktionale Erweiterbarkeit und Konfigurierbarkeit
  - Unterstützung von verteilten heterogenen Einsatzumgebungen
- ◆ Zuverlässigkeit des Systems
  - Verfügbarkeitsgarantien für Workflow-Instanzen
  - Transparenz von Systemfehlern
- ◆ Analysierbarkeit von Workflow-Management-Systemen
- ◆ Skalierbarkeit
- ◆ Berücksichtigung organisatorischer Gegebenheiten

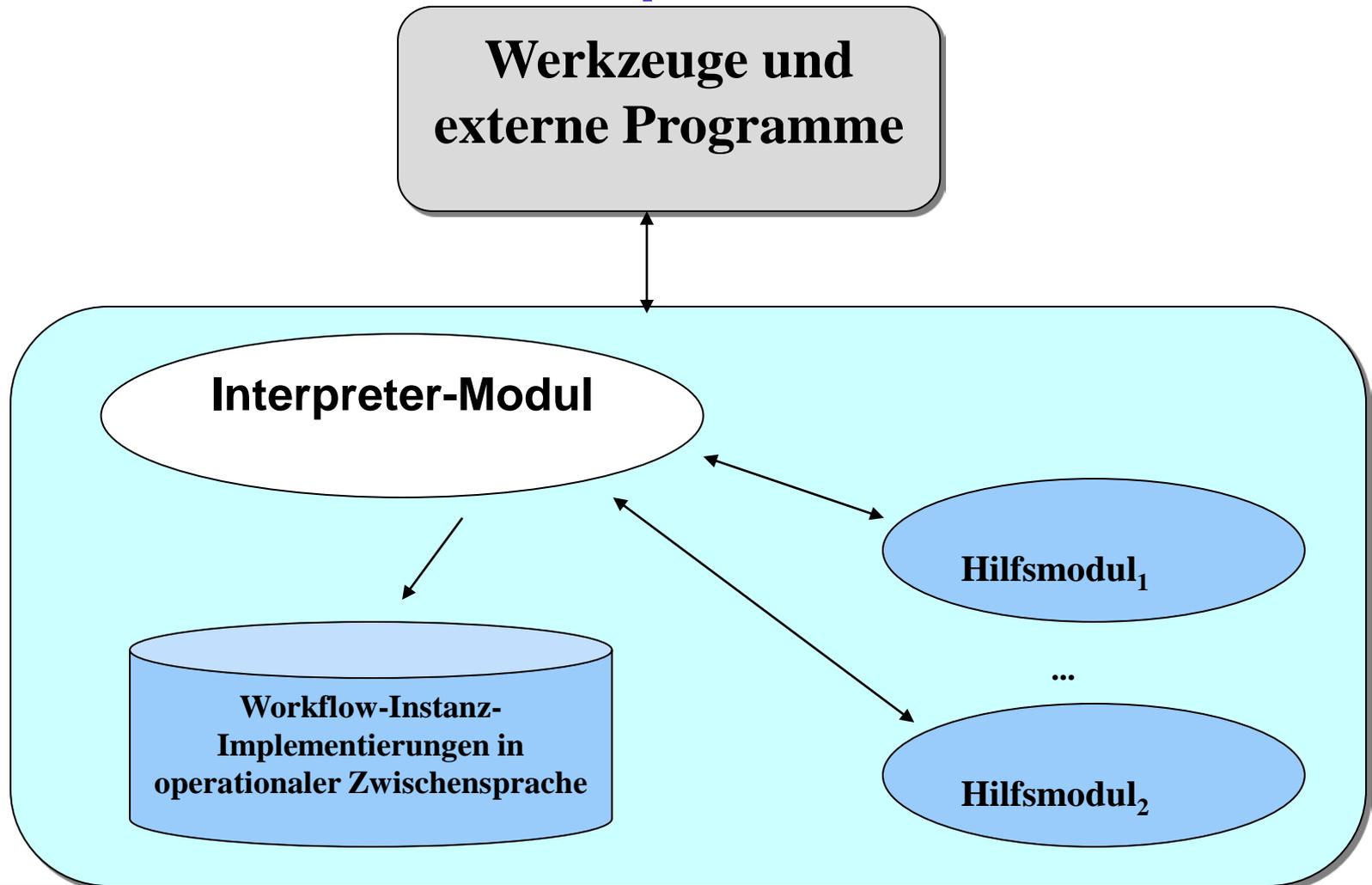
# Funktionale Komponenten eines WfMSs (Wdh. Kap. 2)



# Komponentenarchitektur eines WfMSs (Wdh. Kap. 2)



# Implementierungsarchitektur mit Workflow-Interpreter



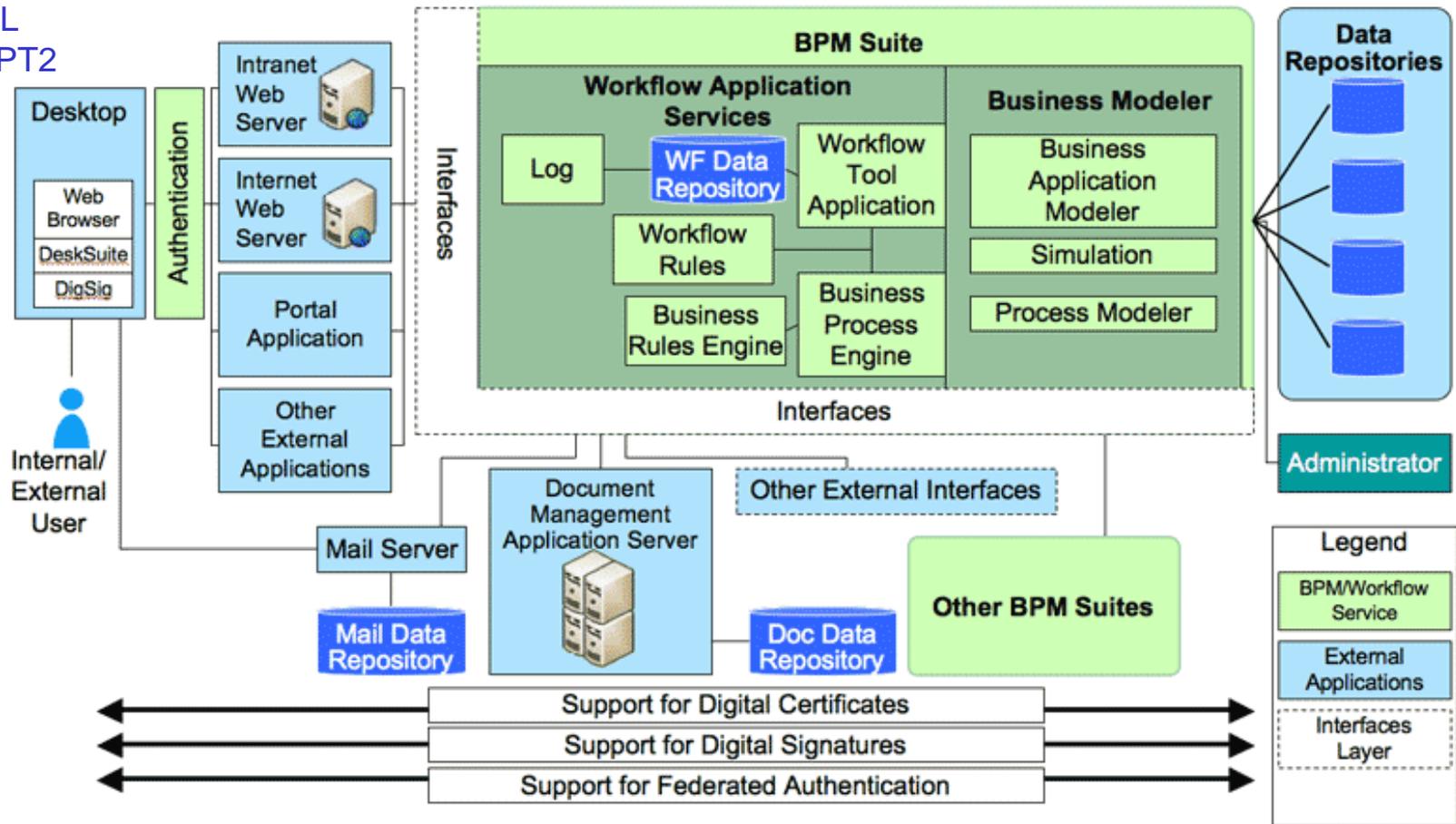
# Schichtenarchitektur eines WfMSs



# Generische Architektur eines WfMS

Wiederholung  
Architekturen

Mentor  
YAWL  
ADEPT2



Quelle: Wikipedia

# Middleware-Unterstützung für Implementierungsarchitekturen

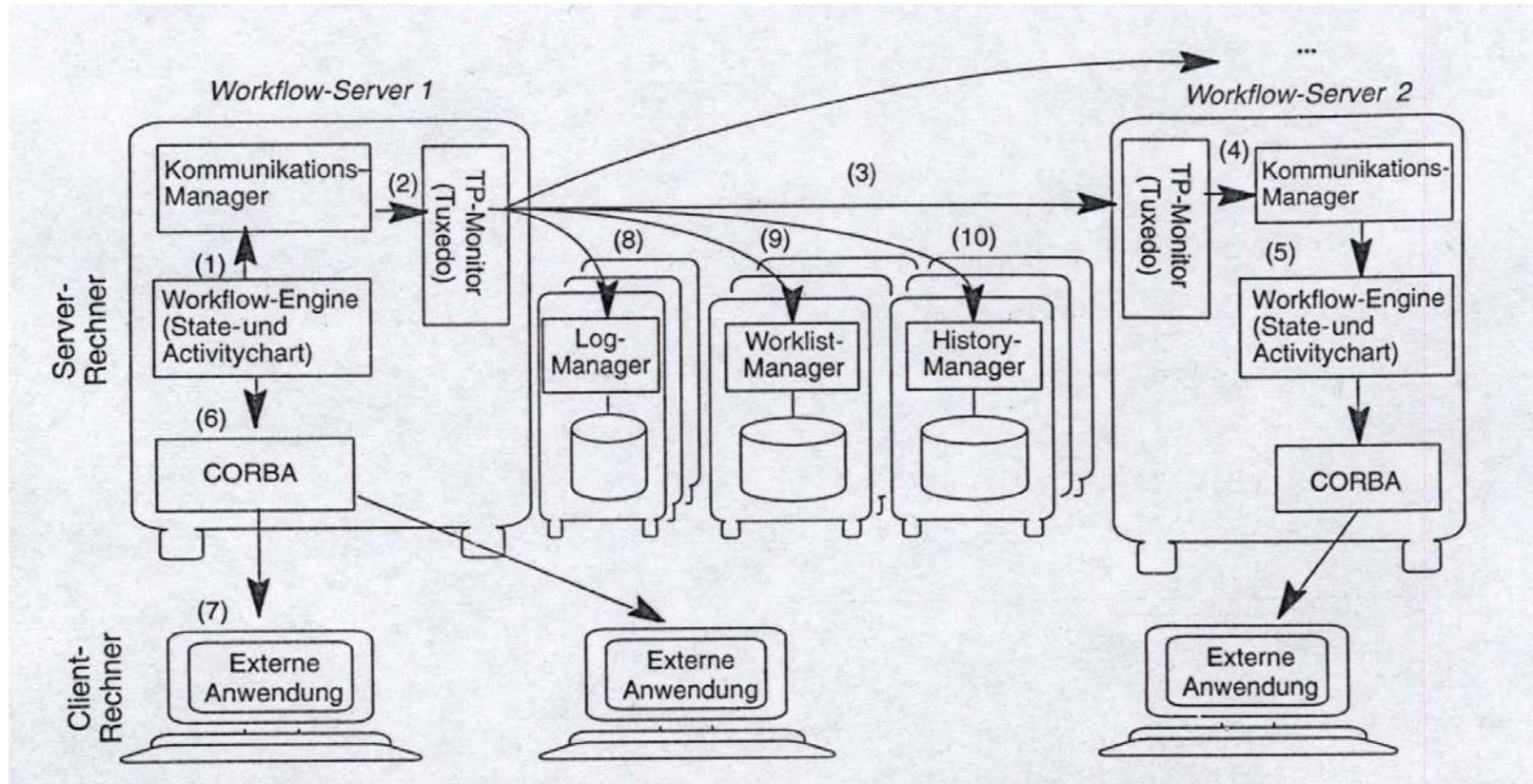
- ◆ Middleware als Verteilungsplattform
- ◆ Erfüllung nicht funktionaler Eigenschaften stehen im Vordergrund
- ◆ Middleware als Basis für die Kommunikation zwischen den Komponenten des verteilten Systems und zur Kommunikation mit Fremdapplikationen hat sich in der Praxis bewährt.
- ◆ Verteilung von Servern eines WfMS ist zwar Grundlage aber nicht Lösung für die nichtfunktionalen Anforderungen.
- ◆ Komponenten plattformunabhängig und in verschiedenen Programmiersprachen

# Mentor Workflow Management System

Wiederholung  
Architekturen  
Mentor  
YAWL  
ADEPT2

- ◆ Client/Server Architektur
  - Client: externe Applikationen
  - (mehrere) Server:
    - Steuerung des Prozesses; Überwachungs- und Administrations-Prozesse
- ◆ Standard-Middleware-Komponenten
  - ORB (Object Request Broker)
  - TP-Monitor (Transaction Processing Monitor)
    - Transaktionen, persistenter Speicher
- ◆ Orientierung am Referenzmodell der OMG
  - Darüber hinaus: jede Komponente kann als verteilte Komponente realisiert sein

# Komponenten der MENTOR Ausführungs-Engine



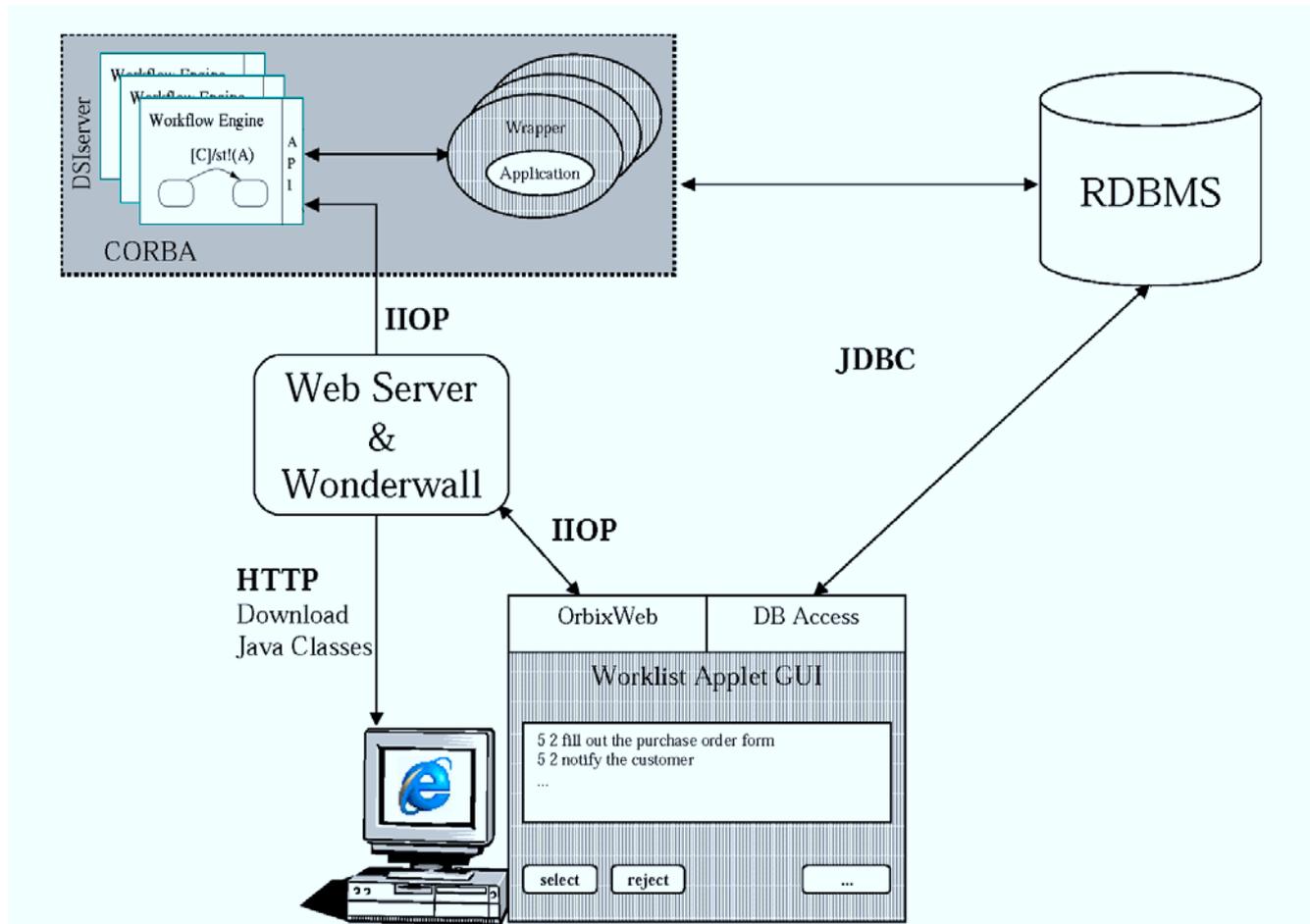
# Arbeitsschritte zur Kommunikation von Workflow-Engines in MENTOR (I)

1. Zustandsänderung, z.B. infolge Beendigung einer Aktivität, an lokalen Kommunikationsmanager melden.
2. Kommunikationsmanager kennt Workflow-Server-Rechner, TP-Monitor stellt Primitive zum Starten der Kommunikation zur Verfügung
3. Übertragung der Nachricht an den empfangenden Kommunikations-Manager
4. Ankommen der Nachricht auf Workflow-Server und weiterleiten an lokalen Kommunikationsmanager
5. Übergabe an Workflow-Engine, die die gemeldete Änderung lokal gültig macht.

# Arbeitsschritte zur Kommunikation von Workflow-Engines in MENTOR (II)

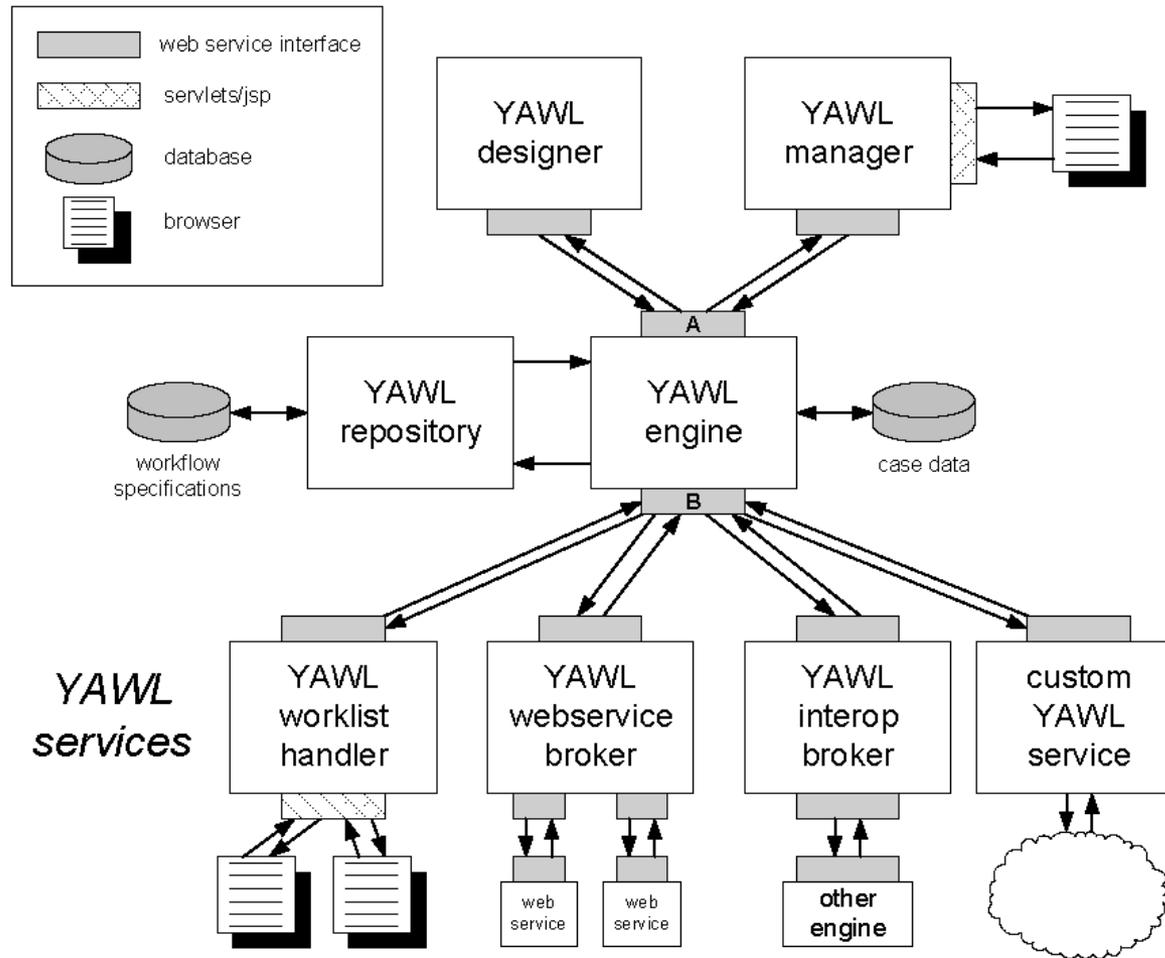
6. Aufruf einer CORBA-Komponente zur Durchführung einer Aktivität
7. Abbildung der Aufruf-Parameter der gestarteten Aktivität auf die Aufruf-Schnittstelle der Applikation
8. Protokollieren der Zustandsänderungen jeder Workflow-Engine durch Log-Manager
9. Worklist-Manager ermittelt geeignetes Ausführungsorgan (Ressource) unter all den Rolleninhabern, die die Aktivität ausführen dürfen
10. History-Manager protokolliert für Anfragen an die Workflow-Historie

# Mentor lite



# YAWL System (Wdh. Kap. 5)

Wiederholung  
Architekturen  
Mentor  
YAWL  
ADEPT2



# Komponenten in YAWL (I)

- ◆ Workflow-Spezifikationen (in der Sprache YAWL) werden mit dem *YAWL Designer* entwickelt
- ◆ *YAWL Engine* instantiiert und führt Workflows aus, die „deployed“ sind
- ◆ *YAWL Repository* dient zur Speicherung von ausführbaren Workflows
- ◆ *YAWL Manager* ermöglicht das manuelle Verwalten von Workflow-Instanzen oder Workflow Spezifikationen sowie die Verwaltung von Daten über beendete Instanzen

# Komponenten in YAWL (II)

## ◆ Services in YAWL

### ○ *YAWL Worklist Handler*

- Zuordnung von Work Items zu Personen
- Besonderheit: im Ggs. zu traditionellen WfMS Entkopplung von der Engine

### ○ *YAWL Web Services Broker*

- Mediator zwischen Engine und Web Services
- Grundgedanke, dass Web Services zu generisch sind, als dass sie direkt eingebunden werden können

### ○ *YAWL Interoperability Broker*

- Verknüpfung mit anderen Engines

### ○ *Custom YAWL Services*

- Erweiterbarkeit von Services (z.B. Telefondienste, Druckerdienste, etc.)

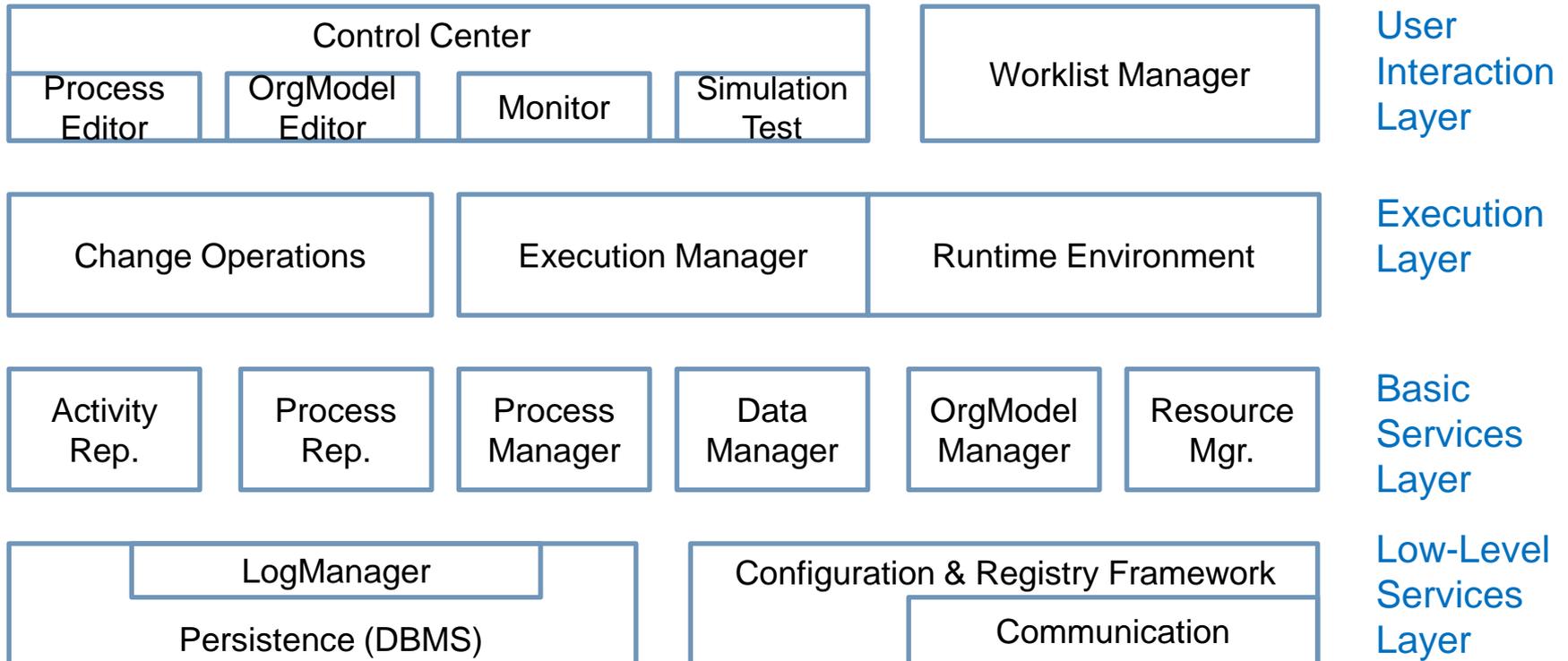
# Anmerkungen zu YAWL

- ◆ Schnittstellen des WfMC Referenzmodells sind abgebildet
- ◆ Umsetzung der Sprache YAWL (Kap.5) mit den Besonderheiten
  - OR-Join
  - Multiple Instanzen
  - Entfernen von Token
- ◆ Implementierung der Engine
  - Java
  - XML-Standards (XPath, XQuery, XML Schema)
    - Umsetzung des Datenaspektes (siehe Kap. 5)

# ADEPT

- ◆ **Application Development based on Encapsulated pre-modelled Process Templates (ADEPT)**
- ◆ **Historie**
  - Seit Ende der 90er: konzeptuelle Arbeiten und Entwicklung von Forschungsprototypen (ADEPT, ADEPT2)
  - Universität Ulm
  - Inzwischen auch kommerzielles Produkt (AristaFlow)
- ◆ **Anforderungen**
  - Flexibilität
    - Ad-hoc Anpassungen
    - Schema-Änderungen und Migration
  - Robustheit
    - Korrektheit von (adaptiven) Prozessen
  - Einfache Benutzbarkeit

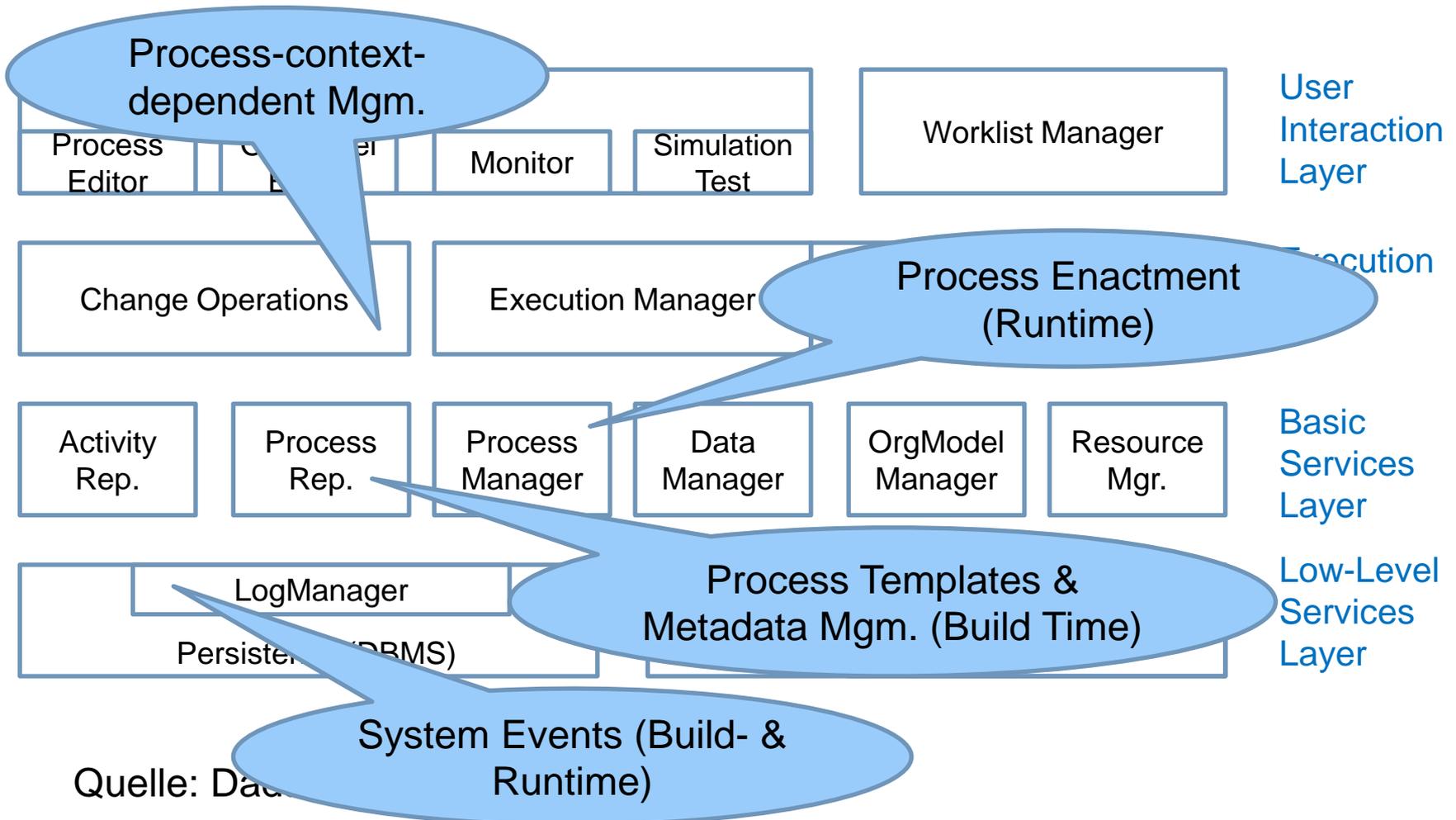
# Architektur von ADEPT2



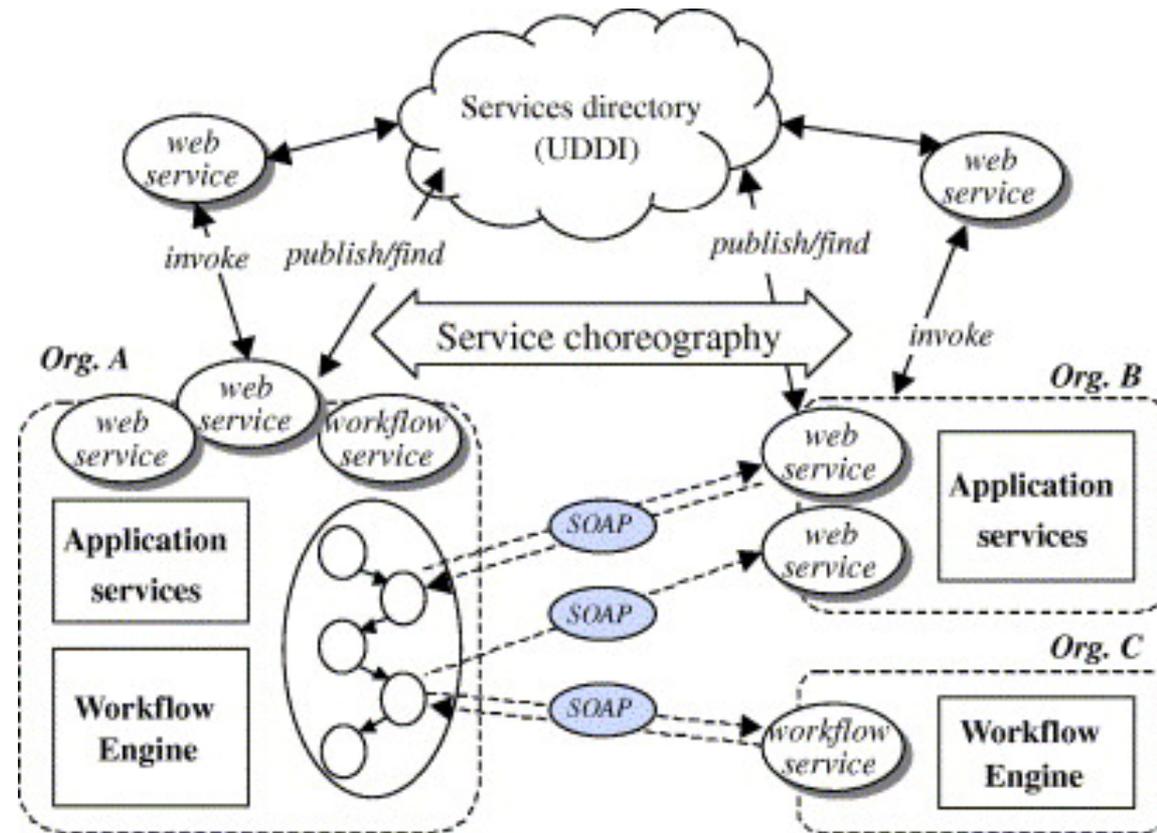
Quelle: Dadam 09

# Architektur von ADEPT2

## - ausgewählte Komponenten -

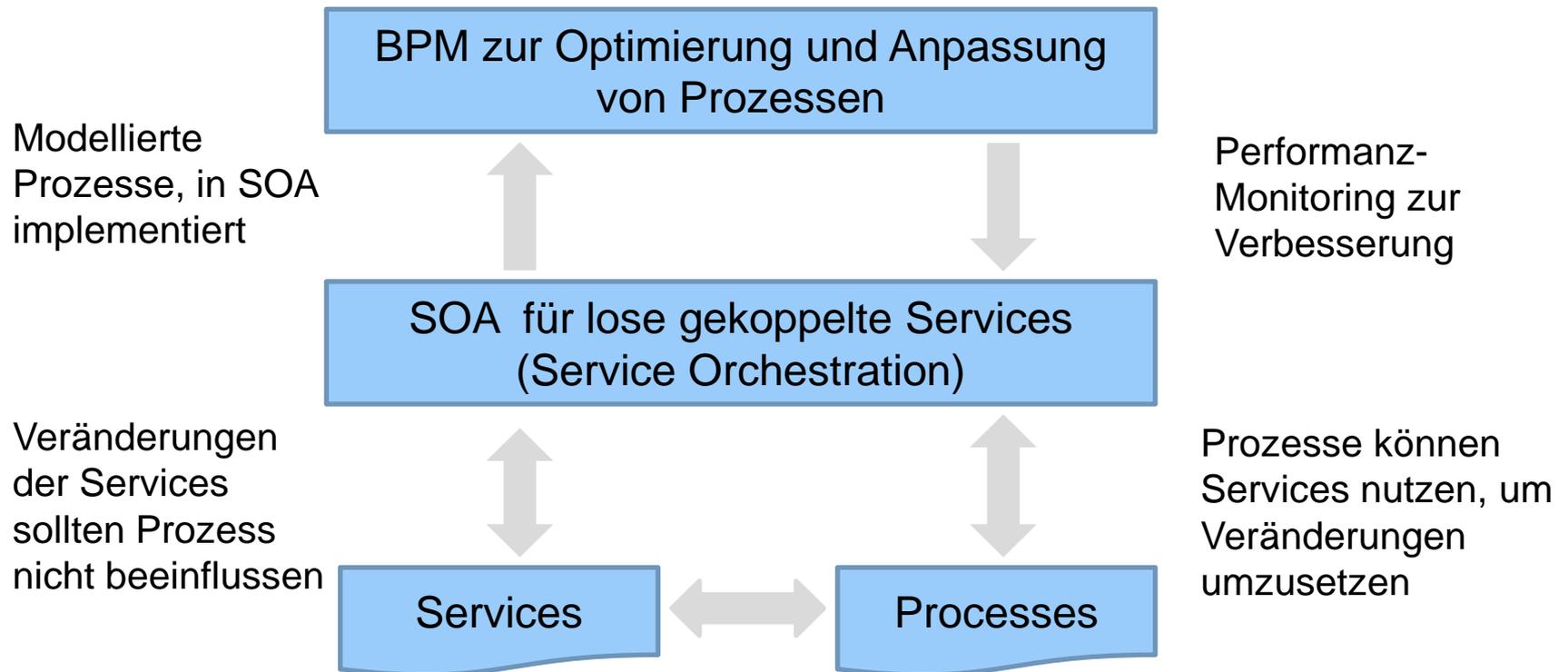


# WfMS und SOA



Quelle: Jung06

# BPM und SOA



Quelle: Bajwa09

# Ähnlichkeiten BPM und SOA

- ◆ Wiederverwendung
- ◆ Dynamische Veränderungen
- ◆ Iterative Entwicklung
- ◆ Lose Kopplung
- ◆ Verteilte Umgebungen

# Exemplarische Fragen zu Kapitel 12

- ◆ Beschreiben Sie die Architektur von MENTOR.
- ◆ Beschreiben Sie die Komponenten von YAWL.
- ◆ Beschreiben Sie die Besonderheiten von ADEPT2.
- ◆ Wozu werden DBMS in WfMS benötigt?

# Ergänzende Literatur zu Kapitel 12

- ◆ Zu Architekturen (generell)
  - S. Jablonski, M. Böhm, W. Schulze (Hrsg.): Workflow-Management - Entwicklung von Anwendungen und Systemen. dpunkt-Verlag, Heidelberg, 1997 (Kap. 14)
- ◆ MENTOR
  - S. Jablonski, M. Böhm, W. Schulze (Hrsg.): Workflow-Management - Entwicklung von Anwendungen und Systemen. dpunkt-Verlag, Heidelberg, 1997 (Kap. 15.1)
- ◆ Mentor lite
  - German Shegalov, Michael Gillmann, Gerhard Weikum: XML-enabled Workflow Management for E-Services across Heterogeneous Platforms. In: The International Journal on Very Large Data Bases, Volume 10, Issue 1, August 2001
- ◆ ADEPT
  - Peter Dadam, Manfred Reichert, Stefanie Rinderle-Ma, Kevin Goeser, Ulrich Kreher, Martin Jurisch: Von ADEPT zur AristaFlow BPM Suite - Eine Vision wird Realität: "Correctness by Construction" und flexible, robuste Ausführung von Unternehmensprozessen. EMISA Forum, 29(1): 9-28, Koellen-Verlag, 2009
- ◆ YAWL System
  - Design and implementation of the YAWL-System. Proc. of CAISE, 2004
- ◆ SOA und BPM
  - Imran Sarwar Baywa, Rafaqut Kazmi, Shahzad Mumtaz, M. Abbas Choudhary, M. Shahid Naweed: SOA and BPM Partnership: A paradigm for Dynamic and Flexible Process and I.T. Management. Int. Journal of Human and Social Sciences 4:7, 2009.
  - Jae-Yoon Jung, Hoontae Kim, Suk-Ho Kang: Standards-based approaches to B2B workflow integration, Computers and Industrial Engineering, Volume 51, Issue 2, October 2006, Pages 321-334.