LEHRSTUHL 14
SOFTWARE ENGINEERING

Willkommen zur Vorlesung

Methodische Grundlagen des Software-Engineering

im Sommersemester 2011

Prof. Dr. Jan Jürjens

TU Dortmund, Fakultät Informatik, Lehrstuhl XIV

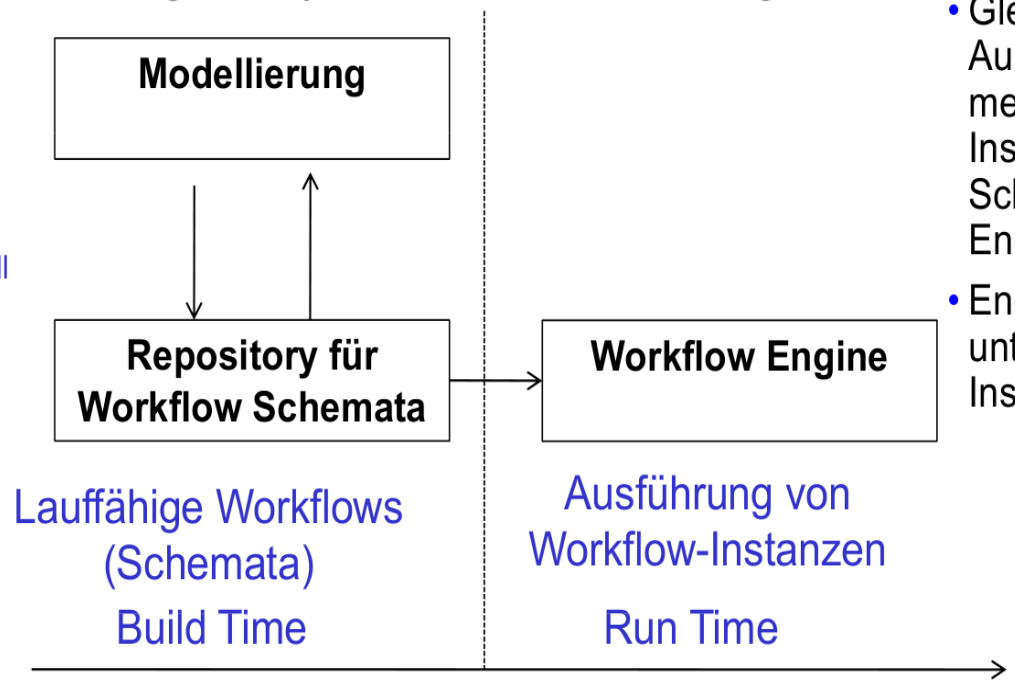
LEHRSTUHL 14
SOFTWARE ENGINEERING

08. Workflow-Management-Systeme

[inkl. Beiträge von Jutta Mülle und Dr. Silvia von Stackelberg, (LS Prof. Böhm, Karlsruher Institut für Technologie) und Professor Dr. Frank Leyman (Universität Stuttgart)]

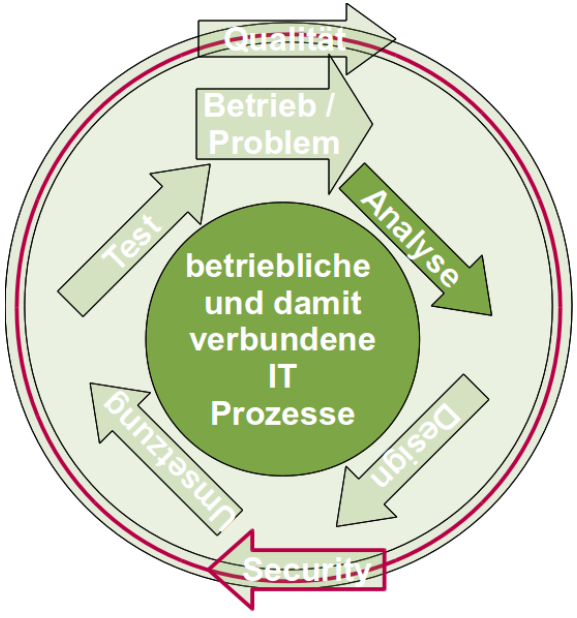
Einordnung
Business Process Execution

Lebenszyklus
Überblick
Aspekte
Integration
Referenzmodell



Beziehungen:

- Gleichzeitige Ausführung mehrerer Instanzen eines Schemas in Engine
- Engine führt unterschiedliche Instanzen aus



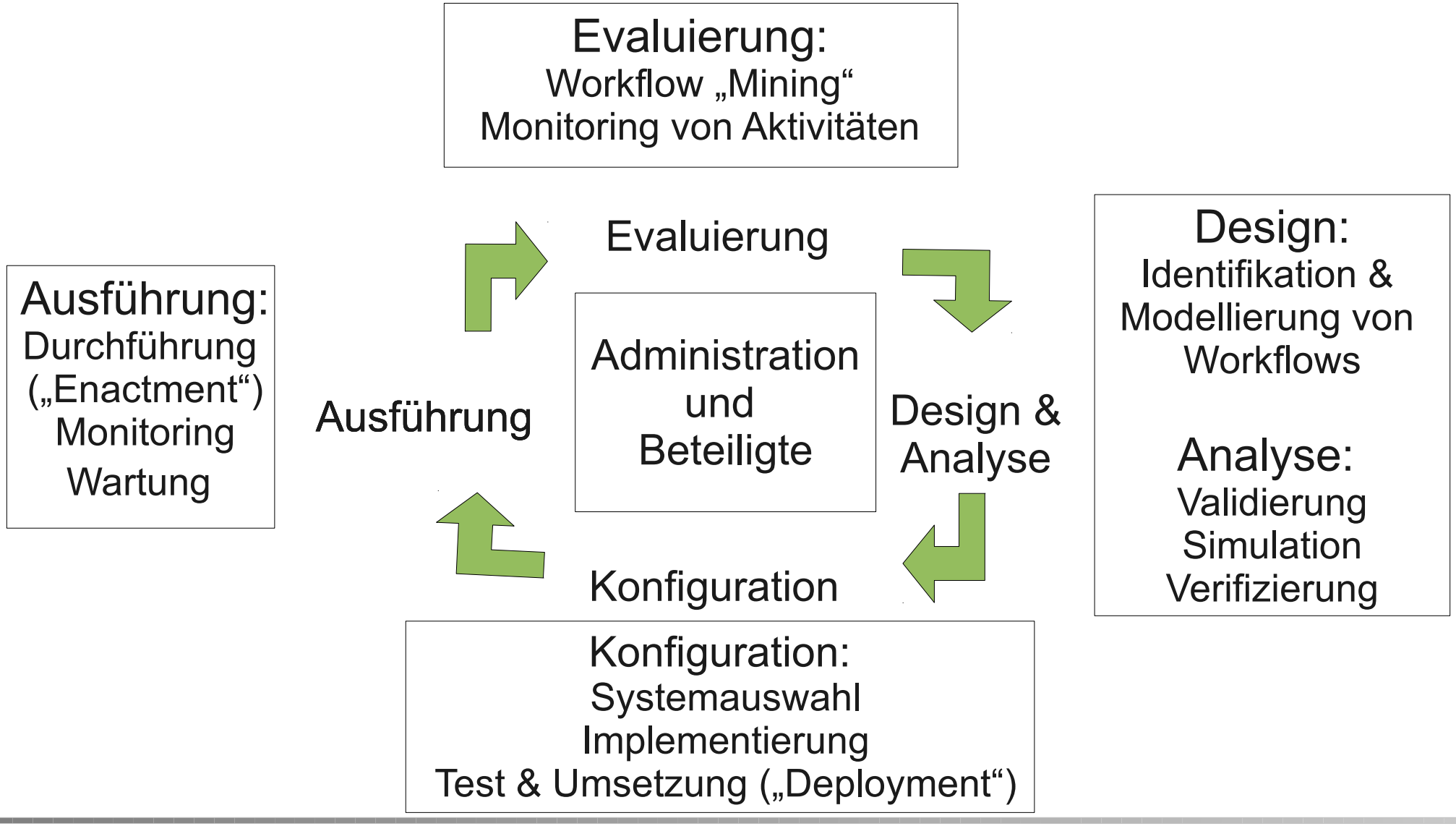
Einordnung
Grundlagen Business Prozesse

- Business Prozesse
  - Anwendungsbeispiel Finanz- und Versicherungsdomäne
  - Grundlagen Geschäfts-Prozesse
  - Elektronische Prozessketten
  - Einführung in die BPMN
  - Business-Process-Mining
  - Workflow-Management-Systeme
- Qualitätsmanagement
- Testen
- Sicherheit
- Sicheres Software Design

Überblick Workflow-Management

- Lebenszyklus von Workflows
- Überblick Modellierung und Ausführung
- Workflow-Aspekte
  - Begriffe
  - Architekturansätze
- Integration von Workflow-Management-Systemen (WfMS) in Anwendungssysteme
- Referenzmodell der Workflow Management Coalition (WfMC)
  - Überblick
  - Komponenten und Schnittstellen
  - Anmerkungen

Lebenszyklus von Workflows (1)



Quelle: M. Weske 2007

Lebenszyklus von Workflows (2)

- Design und Analyse
  - Design:
    - Identifikation, Abgrenzung, Spezifikation von Zielen
    - Modellierung der Workflow Schemata typischerweise in graph. Notation (Kommunikationsbasis mit Anwendern)
  - Analyse:
    - Validierung (Feedback der Anwender)
    - Simulation (Prozessverhalten, z.B. Dauer, Lastverteilung)
    - Verifizierung (z.B. Erkennung von Deadlocks)

Lebenszyklus von Workflows (3)

- Konfiguration
  - Systemauswahl (z.B. WfMS), Spezifikation von technischen Informationen, Integration existierender Systeme, Spezifikation von Benutzerinteraktionen
  - Implementierung
  - Test und Umsetzung („Deployment“)

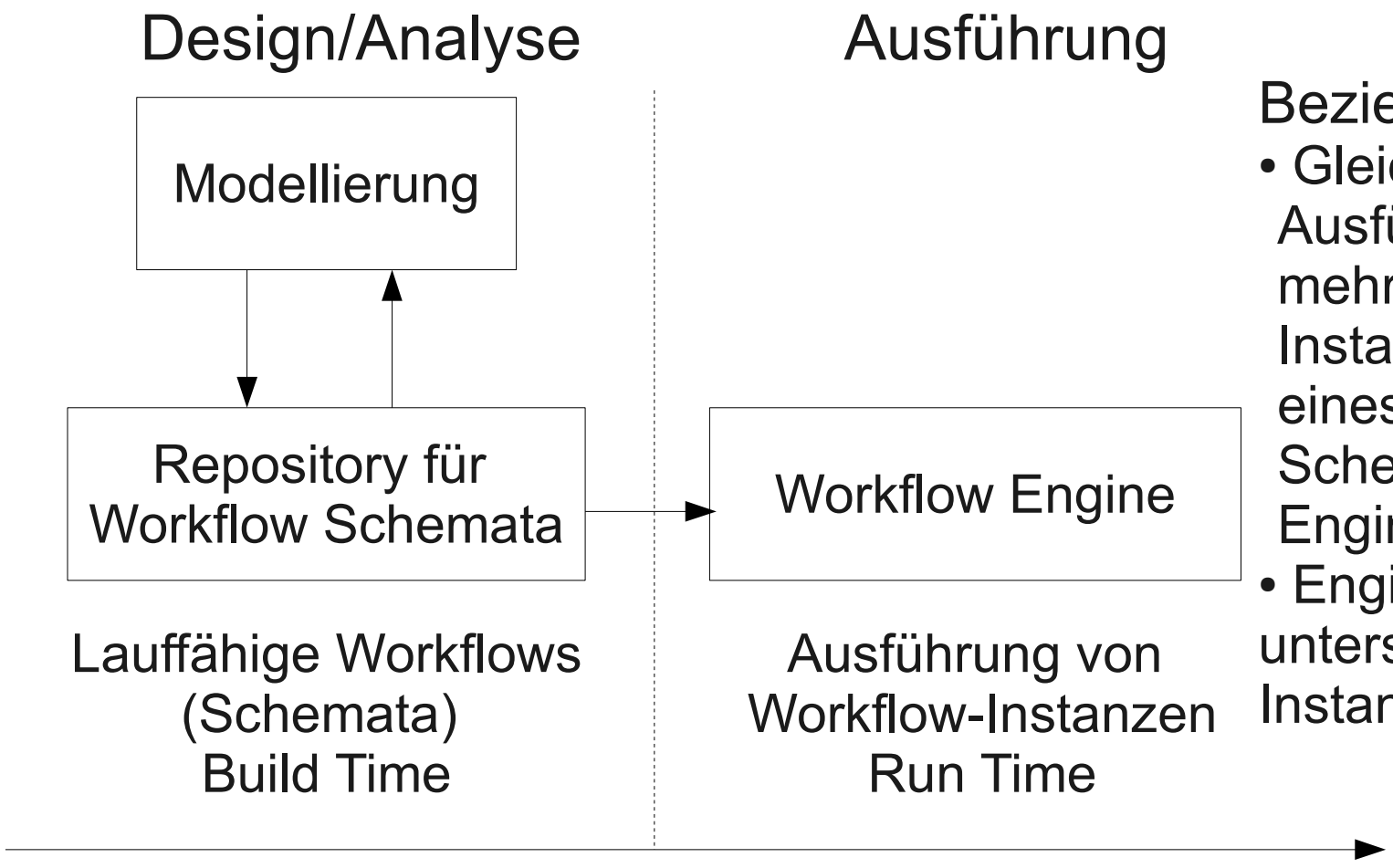
Lebenszyklus von Workflows (4)

- Ausführung
  - Durchführung („Enactment): korrekte Ausführung („Orchestration“) von Workflowinstanzen gemäß der spezifizierten Bedingungen (zeitl. Ablauf, Rechte für Endbenutzer etc.).
  - Monitoring (z.B. Statusinformationen von aktiven Instanzen, Events in Log-Files)
  - Betrieb und Wartung

Vergleich Luftfahrt und Workflow-Management

Luftfahrt-Management	Workflow-Management
Gesamtes Flugnetz einer Luftfahrt-gesellschaft (alle Flugverbindungen)	Gesamtheit aller Workflow-Modelle (Schemata) eines Unternehmens
Einzelstrecke (z.B. Karlsruhe-Mailand)	Workflow-Modell (Schema) eines Prozesses (z B Reisekostenabrechnung)
Gesamtheit aller Ressourcen (Flugzeuge, Piloten, Flugbegleiter, Bodenpersonal)	Ressourcen-Modell (Personen, externe Applikationen)
Zugeordnetes Flugzeug (z.B. HH1007 Karlsruhe-Mailand, 27.10.2010)	Workflow-Instanz (z.B. Reisekosten-abrechnung Nr. 08/007 vom 18.10.2010)
Flugüberwachung der fliegenden Jets in der Flugleitzentrale	Monitoring der aktiven Workflow-Instanzen in Echtzeit
Analyse von Kennzahlen über y Flugverspätungen, Störungen, Passagierzahlen u.a.	Analyse von ausgeführten Workflow-Instanzen hinsichtlich Durchlaufzeit, Liegezeit, Kosten u.a.

Überblick Modellierung – Ausführung (1)

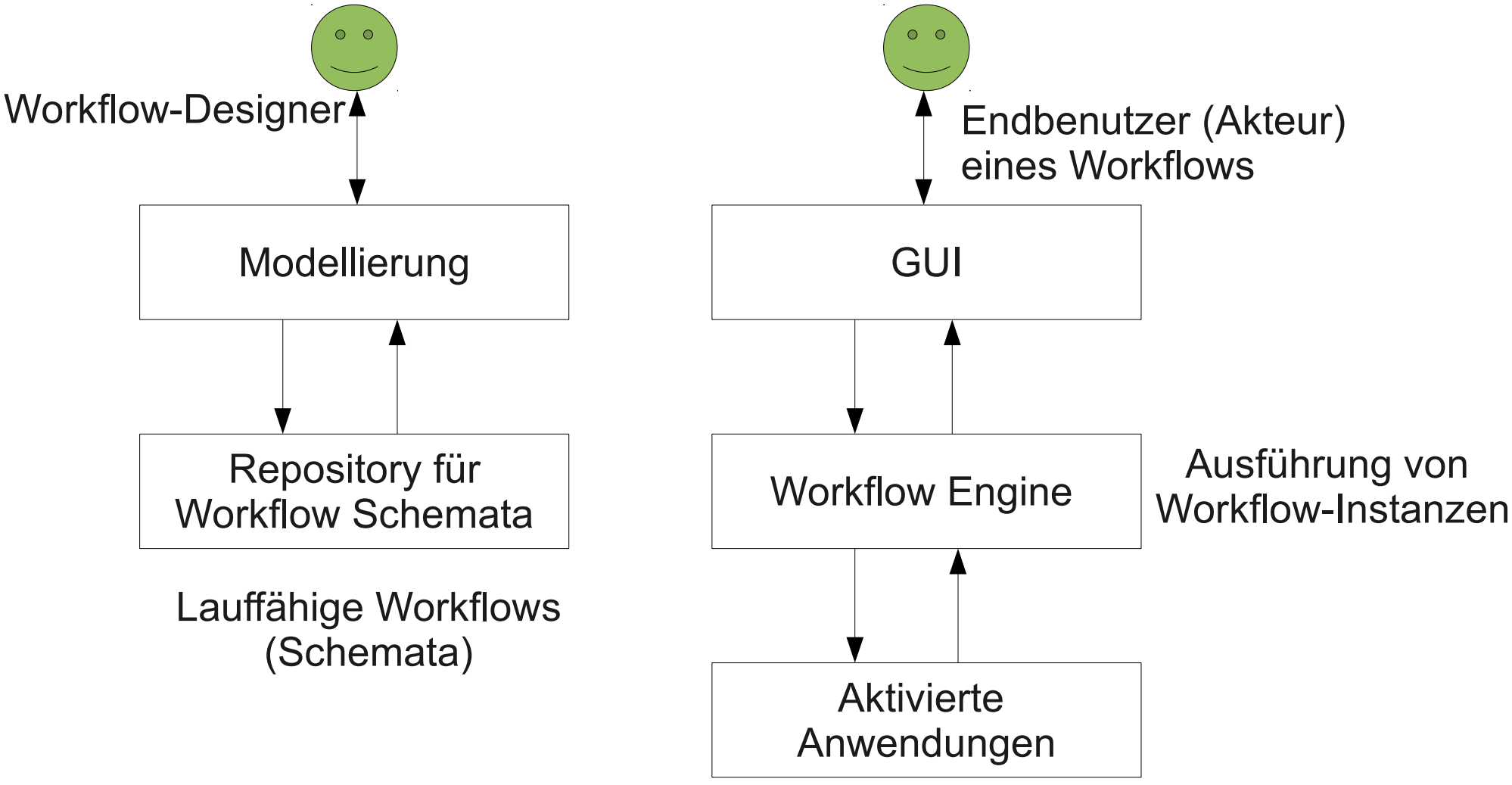


Beziehungen:

- Gleichzeitige Ausführung mehrerer Instanzen eines Schemas in Engine
- Engine führt unterschiedliche Instanzen aus

Quelle: M. Weske, 2007

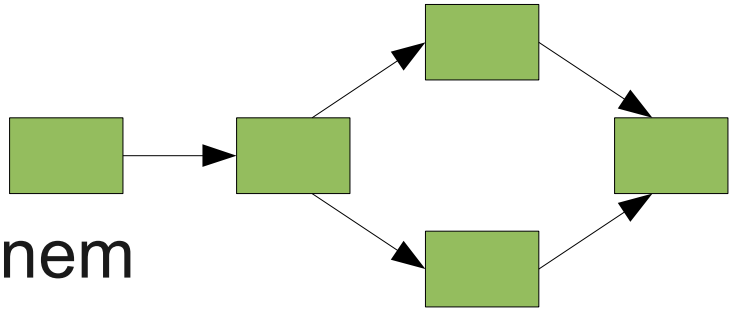
Überblick Modellierung – Ausführung (2)



Quelle: M. Weske

Erstellung eines lauffähigen Workflows

Grobes Vorgehen:



1. Definition des Workflowmodells (Schema) und anderer Daten mit einem Modellierungstool

2. Spezifikation von Ressourcen (z.B. Personen)

3. Implementierung oder Integration der von den einzelnen Arbeitsschritten ausgeführten Programme

Ausführung einer Workflow-Instanz

1. Initialisieren und Starten einer neuen Workflow-Instanz (gesteuert durch ein Ereignis, z.B. Eintreffen einer Dienstreisegenehmigung oder eines Auftrages)

2. Solange Arbeitsschritte auszuführen sind und kein „Terminate“-Arbeitsschritt erreicht wird:
   - Ausführung eines Arbeitsschrittes (automatisiert, semi-automatisiert)
   - Das weitere Vorgehen nach Beendigung des Arbeitsschrittes erfolgt nach den Vorgaben des Workflow-Schemas (Bedingungen, Verzweigungen, …)

3. Terminierung der Workflow-Instanz

Workflow-Aspekte (1)

- **Funktionaler Aspekt:** beschreibt die funktionalen Einheiten, d.h. die Struktur des Workflows mit elementaren und zusammengesetzten Arbeitsschritten (Unter-Workflows).
- **Verhaltensbezogener Aspekt:** konstituiert den Kontrollfluss zwischen den elementaren und zusammengesetzten Arbeitsschritten eines Workflows unter Berücksichtigung von kausalen und temporalen Zusammenhängen.
- **Datenbezogener Aspekt:** betrifft den Datenfluss eines Workflows.

Workflow-Aspekte (2)

- **Organisations-Aspekt:** beschreibt die organisations-bezogenen Inhalte mit der Definition von Organisations-strukturen und deren Population, sowie die Festlegung, wer die verschiedenen Operationen eines Workflows ausführen kann/darf.
- **Operationaler Aspekt:** regelt die Einbindung von Applikationen in einen elementaren Arbeitsschritt. Applikationen sind Programme oder Programmsysteme, die für die Ausführung eines Workflows benötigt werden.

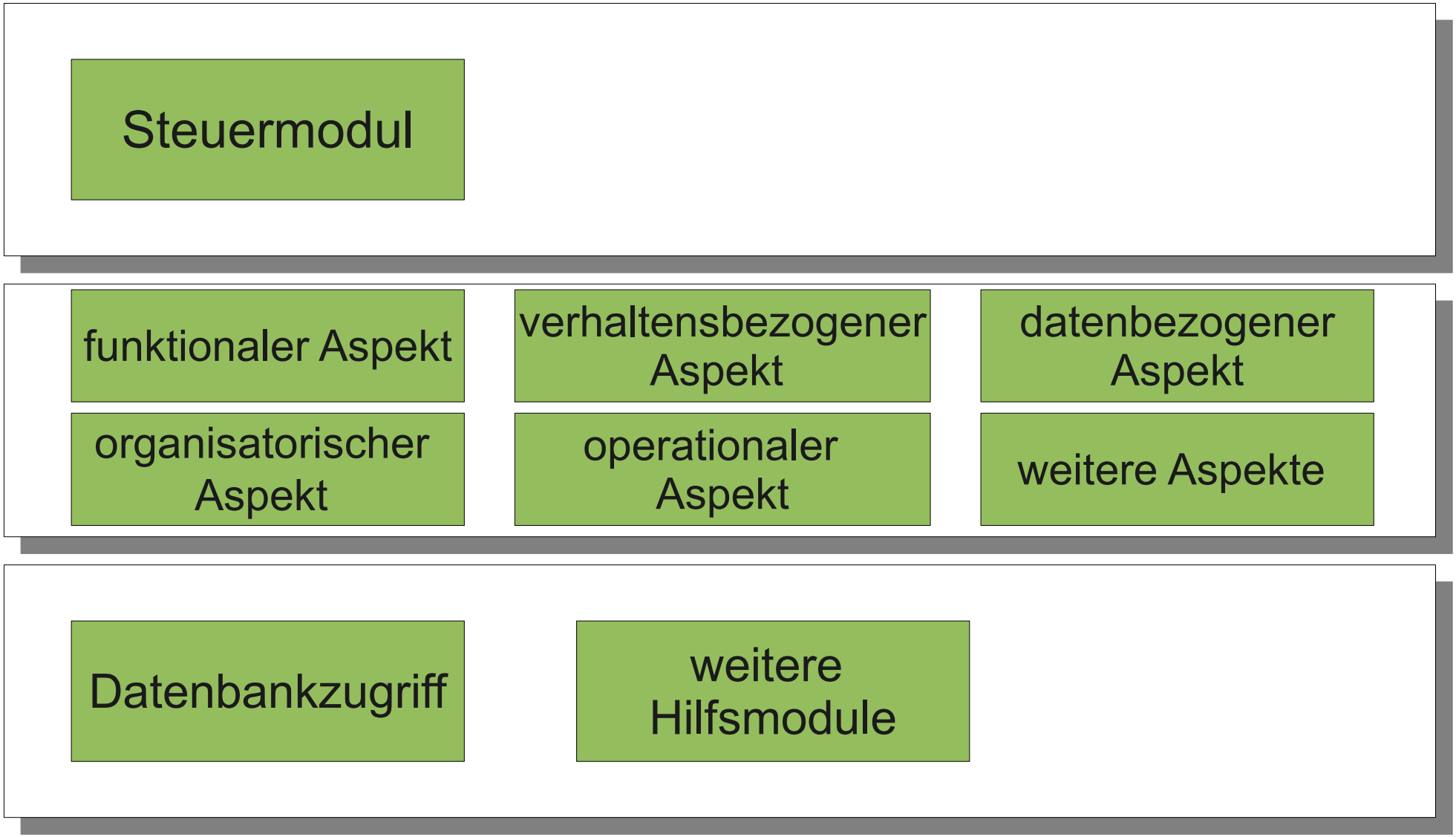
Die genannten Aspekte bilden das Fundament eines WfMSs.

Erweiterungen sind möglich und in vielen Anwendungsfällen notwendig, z.B. ist in vielen Umgebungen ein Sicherheitsaspekt erforderlich, um Zugriffskontrolle zu regeln.

Implementierungsmodell eines WfMS (1)

- Ziel: Beschreibung des implementierungs-unabhängigen Teils der Architektur eines WfMSs
- Spezifikation 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.

Implementierungsmodell eines WfMS (2)



Wie kommt man zur Architektur eines Workflow Management Systems?

- 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 WfMS

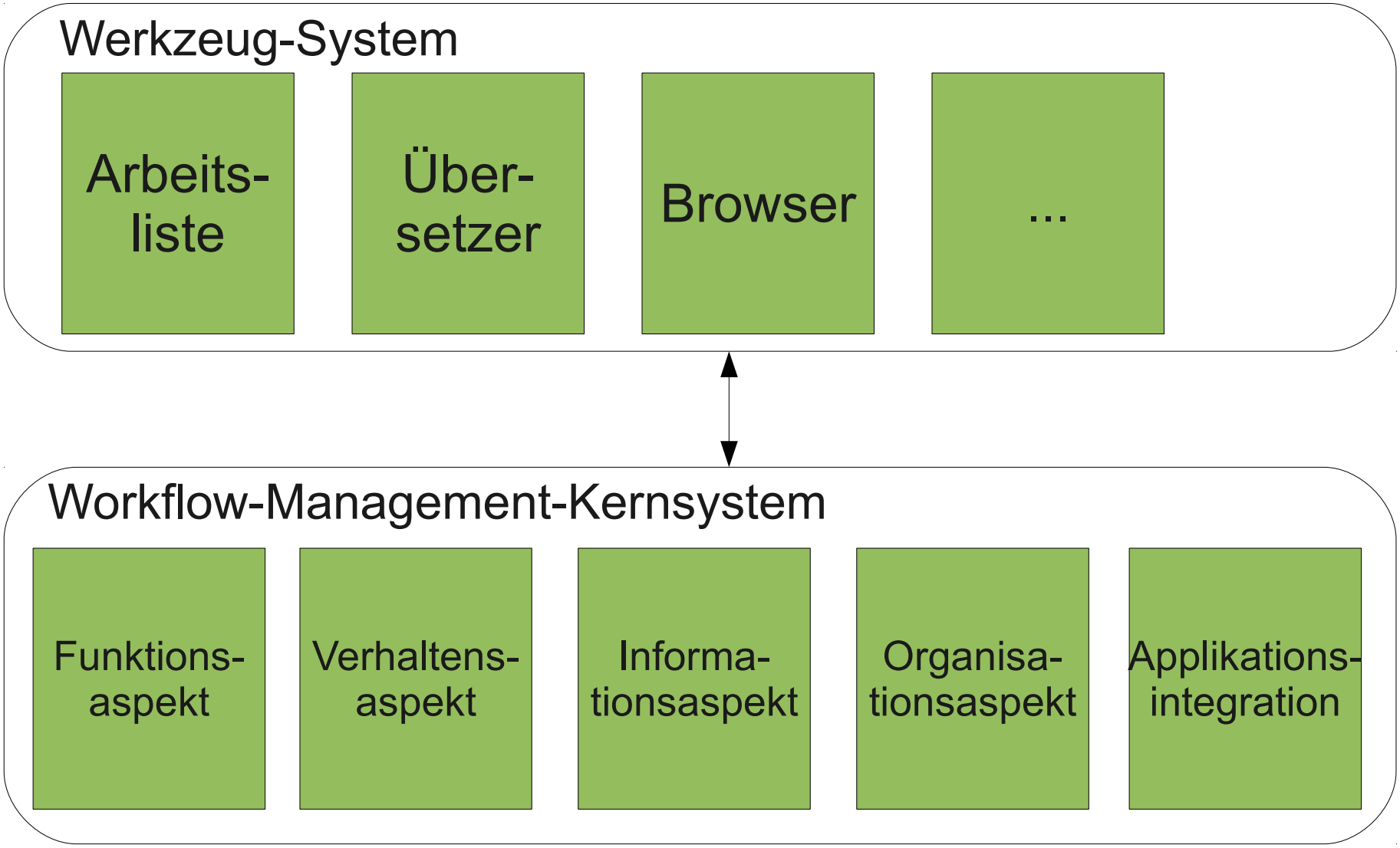
- **Funktionale** Anforderungen
  - Werden vor allem durch die Workflow-Sprache und ihre Ausdrucksfähigkeit und durch Vorgaben bezüglich der Benutzerschnittstellen bestimmt.

- **Nicht-Funktionale** Anforderungen
  - Werden vor allem durch das beabsichtigte Einsatzgebiet eines Workflow-Management-Systems beeinflusst.

Funktionale Anforderungen

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

Funktionale Komponenten eines WfMSs



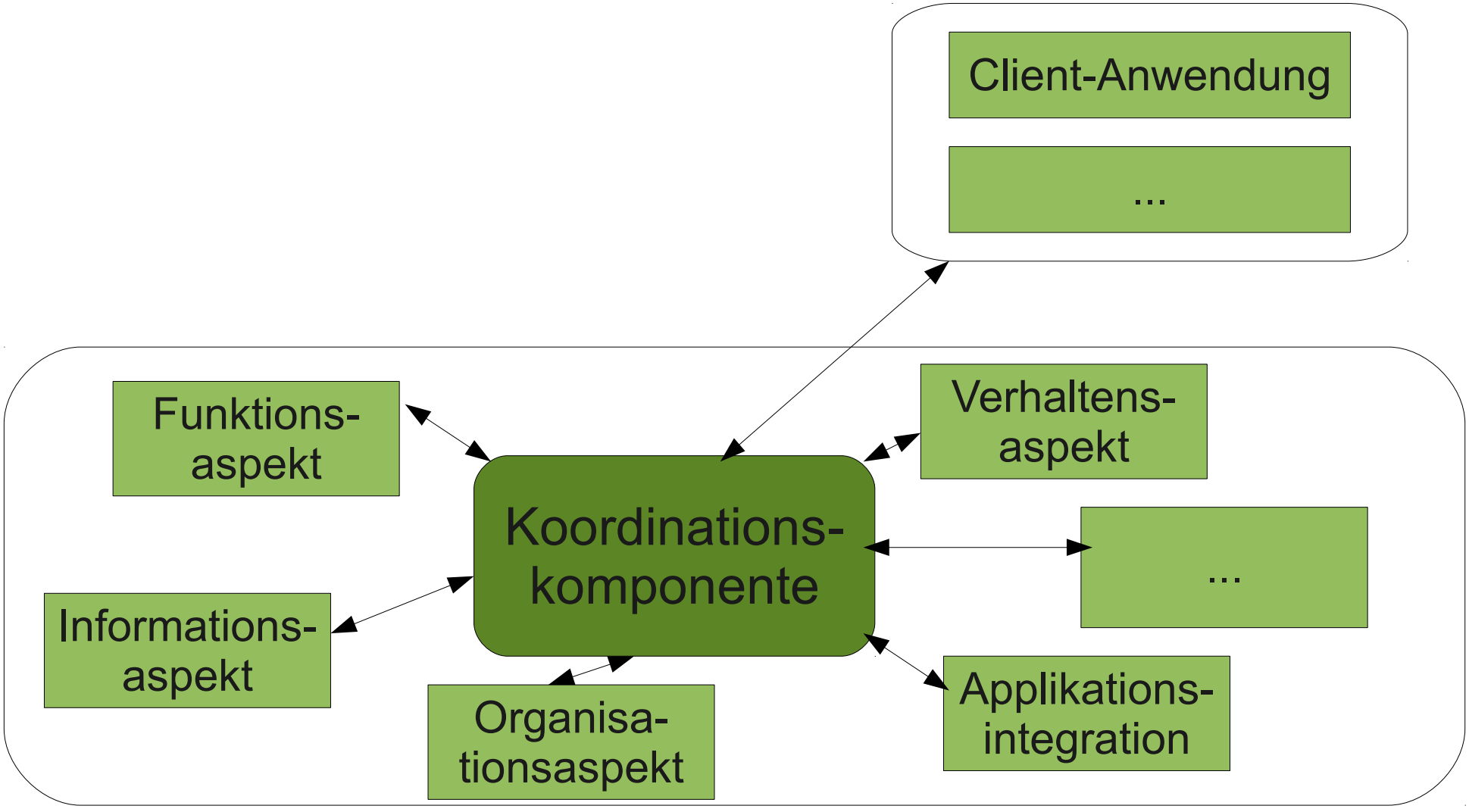
Frage

- Sicherheit als Workflow-Aspekt wird eine Orthogonalität zu allen klassischen Workflow-Aspekten zugeschrieben. Was könnte damit gemeint sein? Welche Konsequenzen hat das?

Nicht-funktionale Anforderungen

- Offenheit der Systemimplementierung 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

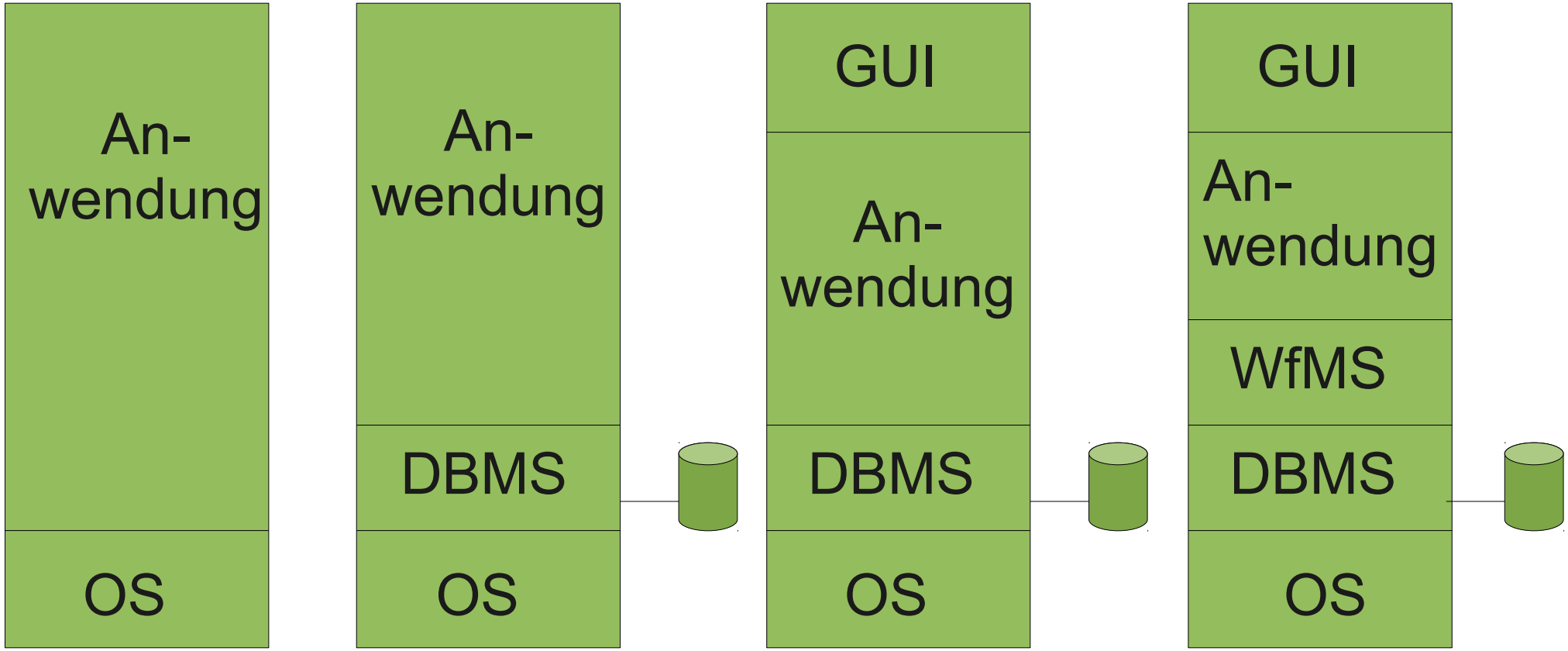
Komponentenarchitektur eines WfMSs



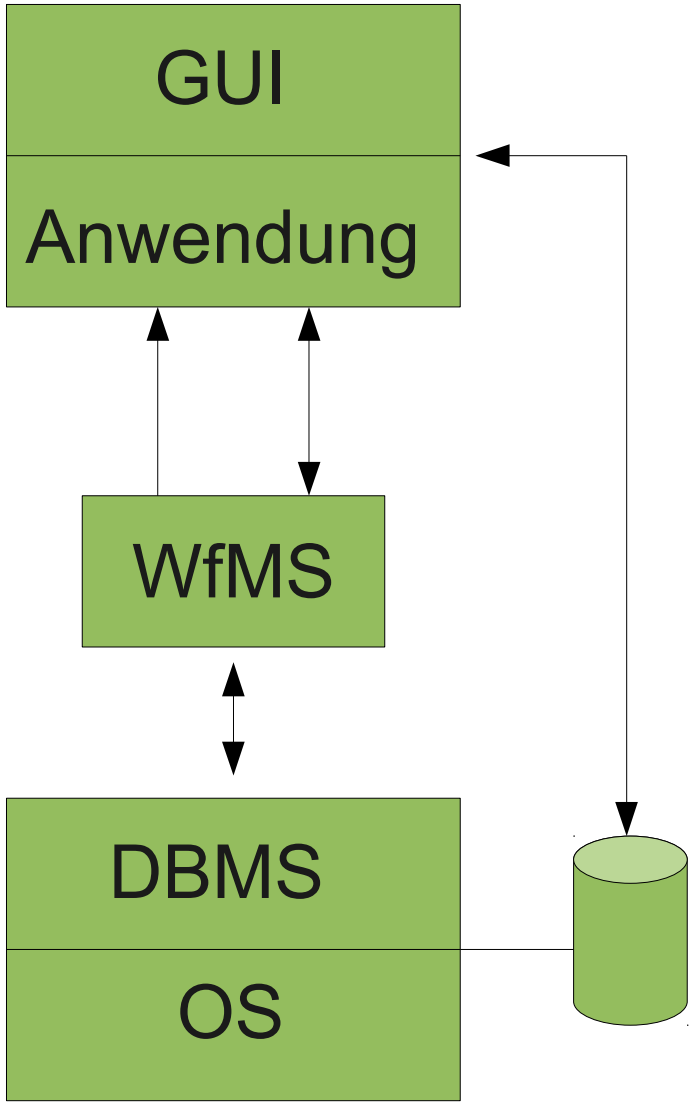
Schichtenarchitektur eines WfMSs

Einbindung von WfMS in Anwendungssysteme (1)

Evolution von Anwendungssystemen

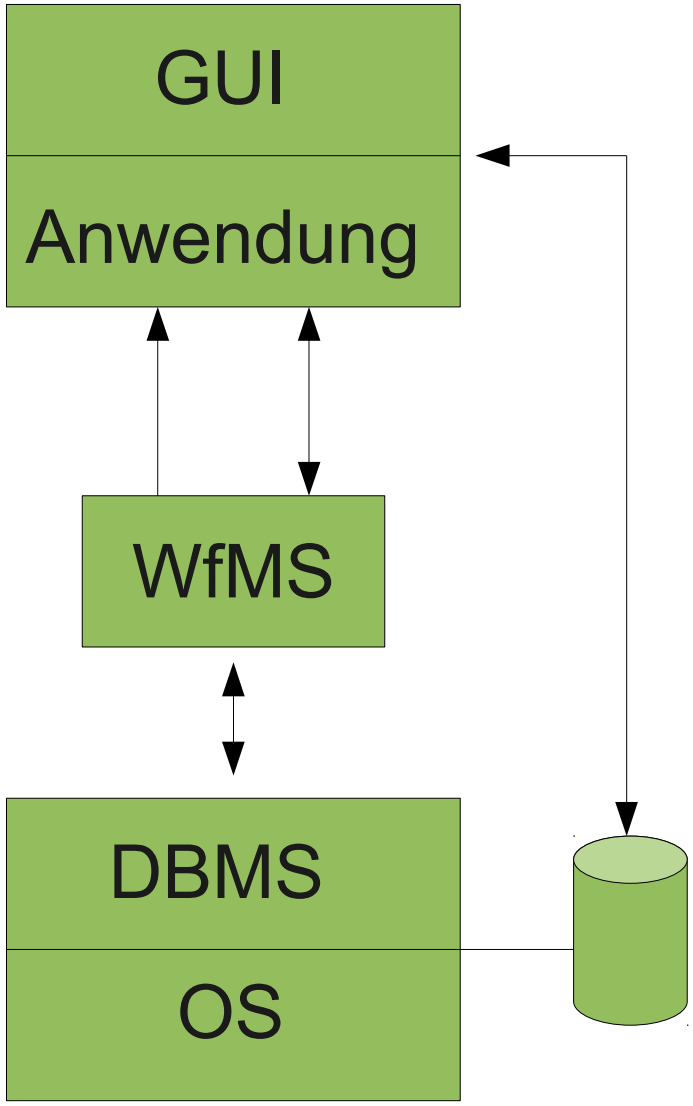


Einbindung von WfMS in Anwendungssysteme (2)



- Explizite Trennung von Workflow-Management und Anwendung
- Workflow-Management
  - Zuständig für „Logistik“ des Workflows
    - Zeitliche Steuerung
    - Aufruf von Anwendungen
    - Berücksichtigung von „Constraints“
    - Zuordnung von Endbenutzern (Anwender, Akteure)
- Anwendung
  - Ausführen von Arbeitsvorgängen
    - automatisiert – semi-automatisiert

Einbindung von WfMS in Anwendungssysteme (3)



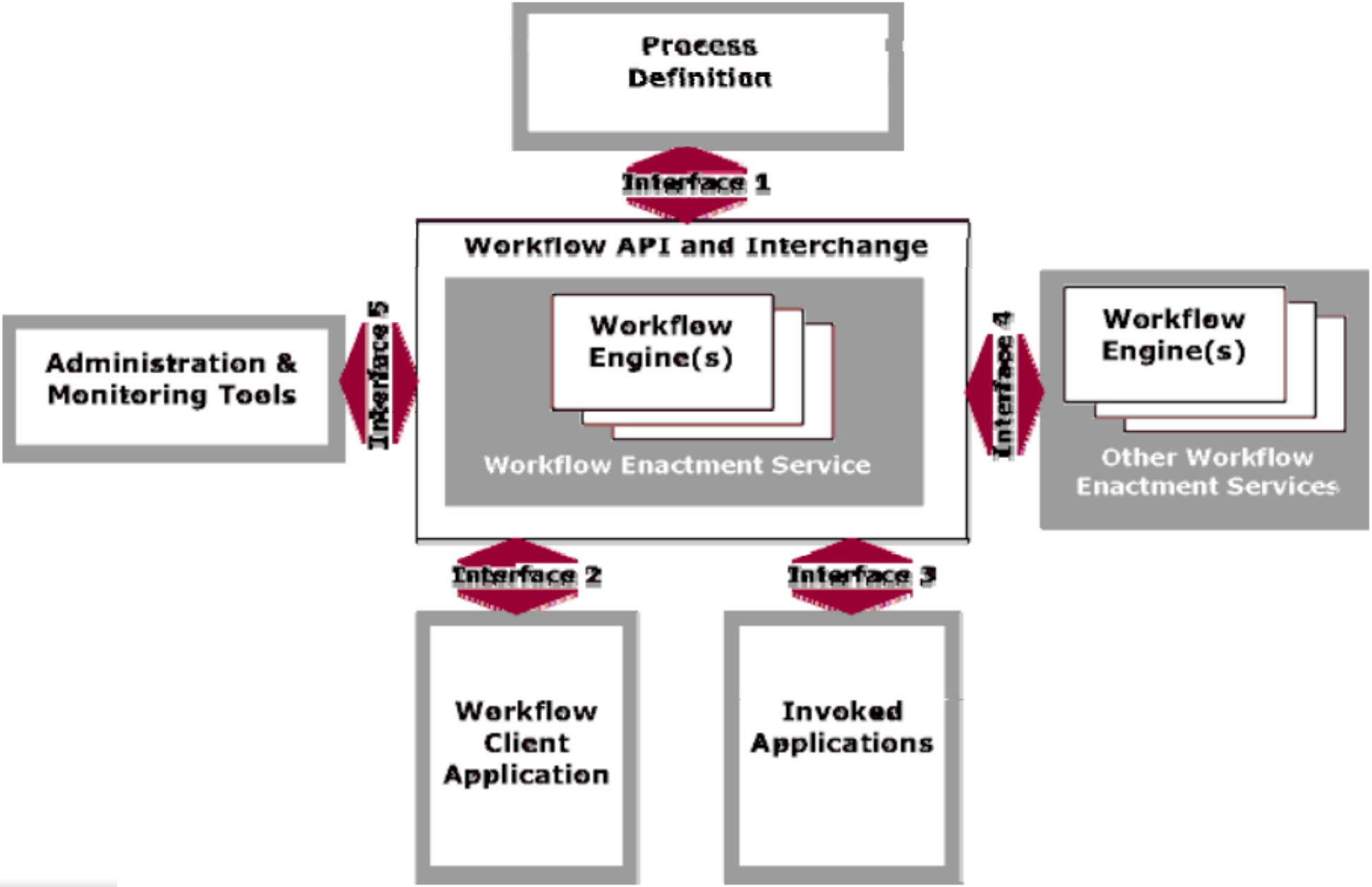
Vorteile Trennung Workflow-Management und Anwendungslogik

- Komplexitätsreduktion
- Wiederverwendung generischer Prozessfunktionalitäten
- Einfache Prozessanalyse
- Einfache, modulare Anwendungsentwicklung

Referenzmodell der WfMC - Überblick (1)

- WfMC (Workflow Management Coalition)
  - Non-Profit-Organisation zur Standardisierung
  - davon ca. 50 Prozent WfMS Anbieter (Forschung ca. 25 Prozent)
- Referenzmodell von 1995
- Ziele:
  - Festlegung von standardisierten Komponenten und Schnittstellen zur Abdeckung der Basisfunktionalität eines WfMS

Referenzmodell der WfMC - Überblick (2)



Referenzmodell der WfMC – Komponenten (1)

- **„Workflow Enactment Service“** (Ausführung):
  - Workflow Engine(s) als „Herz des Systems“
    - Aus Performanzgründen ggf. mehrere Engines, nicht relevant für Endbenutzer (Anwender)
  - Ziel der Steuerung des Prozessablaufes:

    „*right activities at right order to right people*“

Referenzmodell der WfMC – Komponenten (1)

- Aufgaben des Workflow Enactment Service:
  - Initialisieren (→ „Process Definition Tools“) und Beenden von Workflowinstanzen
  - Planung des Ablaufs („Routing“) von Instanzen
  - Zuordnung von Tasks zu Ressourcen (gemäß Ressourcenklassifikation → „Process Definition Tools“)
  - Starten von Anwendungen zur Ausführung von Aktivitäten (→„Invoked Applications“)

Referenzmodell der WfMC – Komponenten (2)

- **„Process Definition Tools“**
  - Prozessdefinitionen im engeren Sinn:
    - Erstellung und Spezifikation von Prozessen: zeitliche (z.B. Sequenz, Parallelität) und konditionale (z.B. Verzweigungen) Abfolge von Komponenten (z.B. Aktivitäten), Constraints, etc.
    - Sprachen: z.B Petri-Netze, BPMN (Business Process Modeling Notation)
    - Spezifikation von Aktivitäten (z.B. Rollen zur Ausführung, aufzurufende Anwendungen sowie Übergabeparameter)
    - Überprüfung auf syntaktische Korrektheit und Konsistenz
  - Ressourcenklassifikation
    - z.B. Rollen, organisatorische Einheiten

Referenzmodell der WfMC – Komponenten (2)

- Workflow-Analyse
  - Semantische Korrektheit, Simulation zur Analyse des Prozessverhaltens

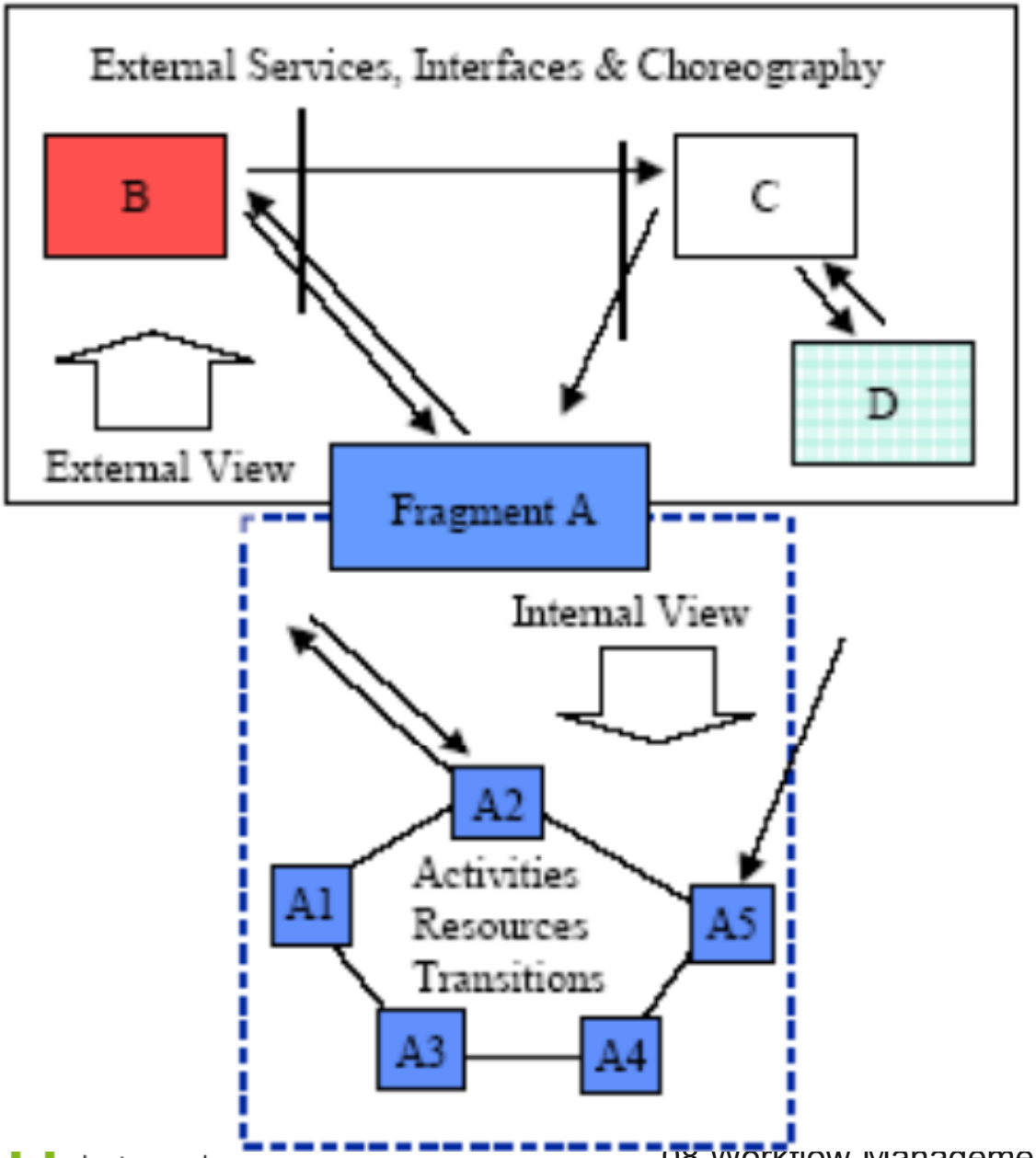
Referenzmodell der WfMC – Komponenten (3)

- **„Workflow Client Application“**
  - Präsentation der Aufgaben („work items“) an Endbenutzer (Anwender) in Form von sog. Arbeitsliste („worklists“)
  - Informationen zu „work items“
  - Initiieren und Beenden von Aufgaben
  - Übergabe von Zustandsparametern (→ Workflow Engine)
- **„Invoked Applications“**
  - Anwendungen im eigentlichen Sinne (Start durch → Workflow Engine) sowie Anwendungs-Parameter (Austausch mit →Workflow Engine);
  - Konfigurationsdateien
  - Datenzugriff
  - Automatisierte sowie interaktive Anwendungen

Referenzmodell der WfMC – Komponenten (4)

- **„Other Workflow Enactment Service(s)“**
  - Workflow Interoperabilität: Verknüpfung von autonomen Workflow- Systemen
  - Sequentielle und hierarchische Ausführung von Teilprozessen (Sub-Workflows)
  - Beispiele: bereichs- bzw. unternehmensübergreifende Kopplung
- **„Administration & Monitoring Tools“**
  - Operationales Management
    - Verwaltung von Endbenutzern (Stammdaten Verfügbarkeiten)
    - Zustandsinformationen einer Prozessinstanz
  - Beobachtungs- und Report-Tools
    - Verarbeitung der vergangenheitsbezogenen „Rohdaten“ (→Workflow Engine) zur Berechnung von Performanz-Indikatoren

Externe und Interne Prozess-Sichten (WfMC)

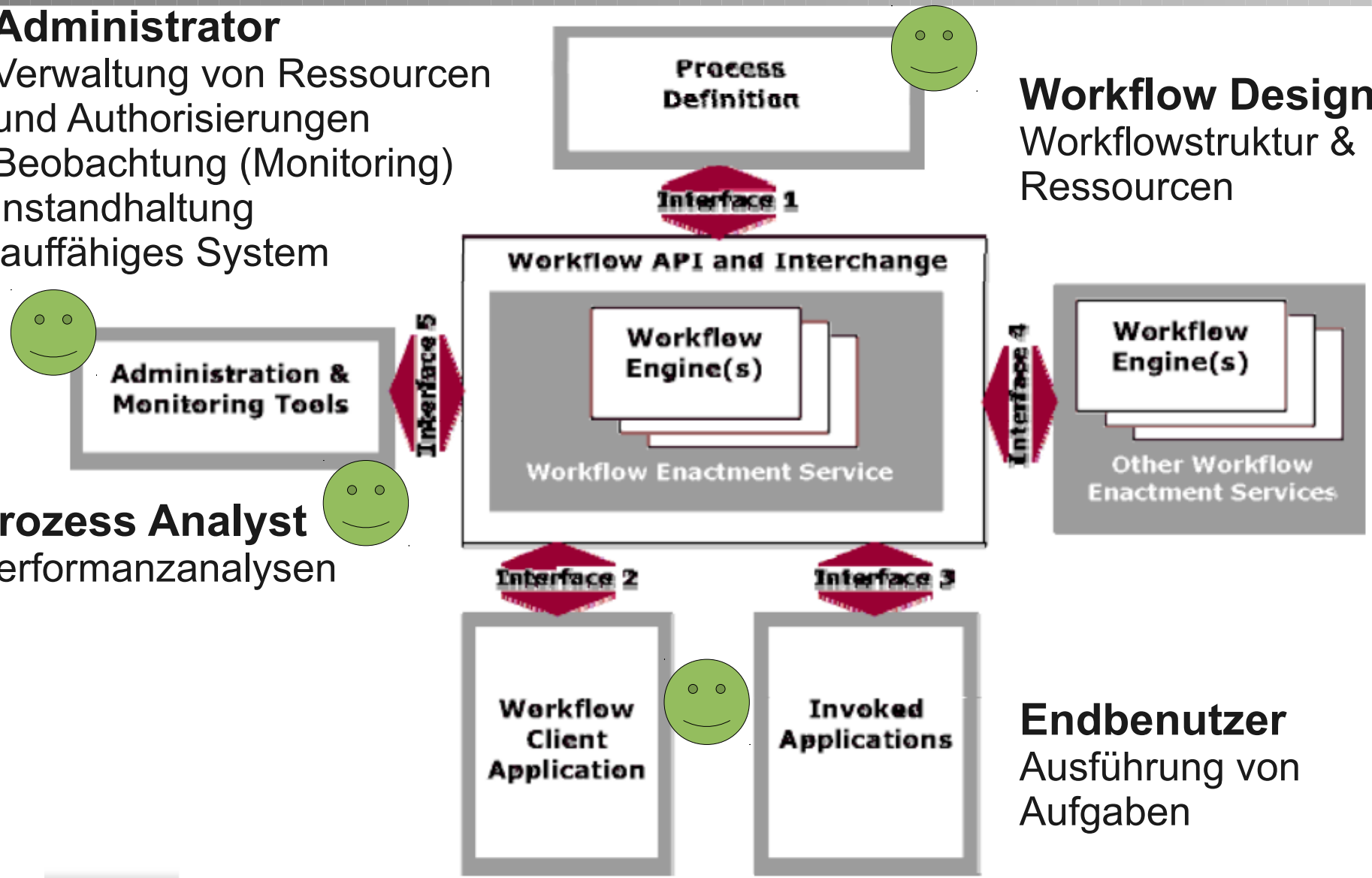


- **Externe Schicht:**
  - Nur Schnittstellen, ansonsten Black Box
  - Choreographie
- **Interne Schicht:**
  - Spezifikation von Aktivitäten, Ressourcen, Schnittstellen

Einbindung von Rollen am Beispiel des Referenzmodells der WfMC

Administrator
Verwaltung von Ressourcen und Authorisierungen
Beobachtung (Monitoring)
Instandhaltung
lauffähiges System

Workflow Designer
Workflowstruktur & Ressourcen



Prozess Analyst
Performanzanalysen

Endbenutzer
Ausführung von Aufgaben

Referenzmodell der WfMC – Schnittstellen (1)

- Generisches Ziel Interoperabilität
  - Standards zum Datenaustausch
  - Verknüpfung von Ausführungsmaschinen unterschiedlicher Hersteller
  - Unabhängigkeit der Anwendungen von konkreten WfMS
- Schnittstelle 1: Workflow Process Definition Read/Write-Interface
  - Verbindung von Workflow-Modellierungswerkzeugen mit der zentralen Komponente - Austausch von Workflow-Schemata
- Schnittstelle 2: Notifikationsdienst auf Clientseite (Workflow Client Application Programming Interface) – Problem: ein Eingangskorb für mehrere Systeme
- Schnittstelle 3: Programmierschnittstelle für den Applikationsaufruf

Referenzmodell der WfMC – Schnittstellen (1)

- Schnittstelle 4: Austausch von workflow-relevanten Daten zwischen verschiedenen, autonomen Workflow-Engines. Ziel ist die Übergabe eines Subworkflows zur Ausführung an ein anderes WfMS.
- Schnittstelle 5: Administrations- und Monitoring-Werkzeuge anderer Hersteller zur Information über den aktuellen Ausführungszustand von Workflows und zur Administration laufender Workflows Administration laufender Workflows.

Schnittstellen sind nicht völlig unabhängig voneinander spezifizierbar.

Vorgehensweise: minimale Menge an Funktionen bzw. Funktionalität standardisieren

Design-Primitive des Referenzmodells

- Abstraktion
  - Beschreibung der wesentlichen Komponenten und Schnittstellen, unabhängig von technischer Umsetzung
  - Spezifikation der Interfaces (abstrakte Terme und konkrete „Bindings“)
  - Konkrete Bindings konnten im Lauf der Zeit den Entwicklungen angepasst werden (API in C → Middleware → Web Services)
- Lebenszyklus eines Prozesses
  - Phasen des Lebenszykluses tragen zum Gesamtbild bei
- Integrierte Sicht von Daten, Prozess und Organisation

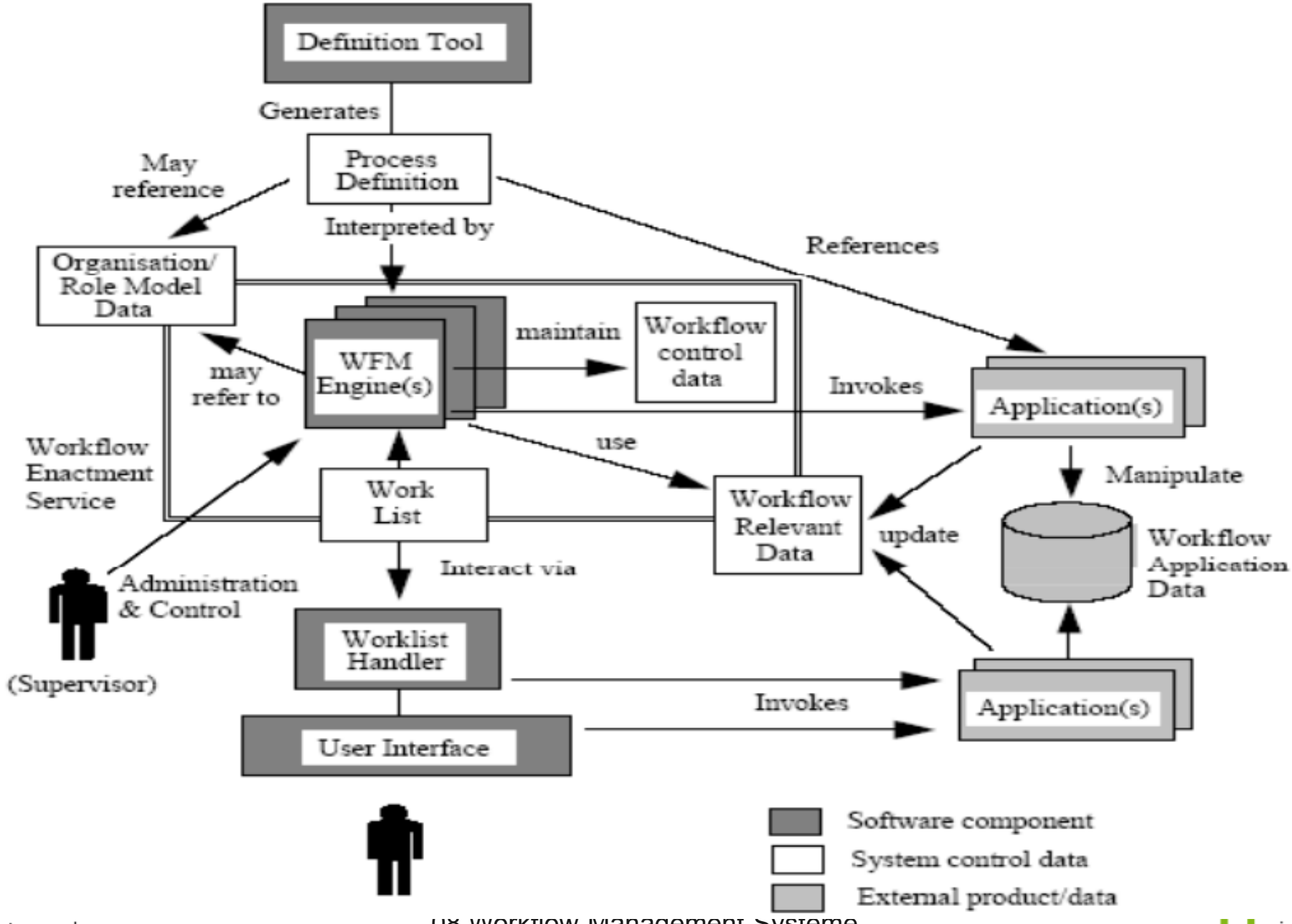
Frage

- Die Verknüpfung von Workflow Enactment Services findet man in der Praxis selten. Woran könnte das liegen?

Einige Schwachstellen des Referenzmodells

- Datensicht etwas vernachlässigt
- Spezifikation der Schnittstellen „nach außen“
  - Schnittstelle 3: Synchronisation von Anwendungen mit Arbeitslisten und Engine (ACID-Eigenschaften von Transaktionen)
  - Schnittstelle 4: lose Kopplung (z.B. unternehmens-übergreifend) nicht adäquat unterstützt, keine Datenstruktur-Spezifikation
- Zu geringe Flexibilität
  - Adaptive Workflows → Anpassungen zur Laufzeit
  - Ad-hoc-Entscheidungen (Problem lokale Autonomie)
- Berücksichtigung neuer Workflow-Paradigmen
  - Z.B. Daten-zentrierte Workflows

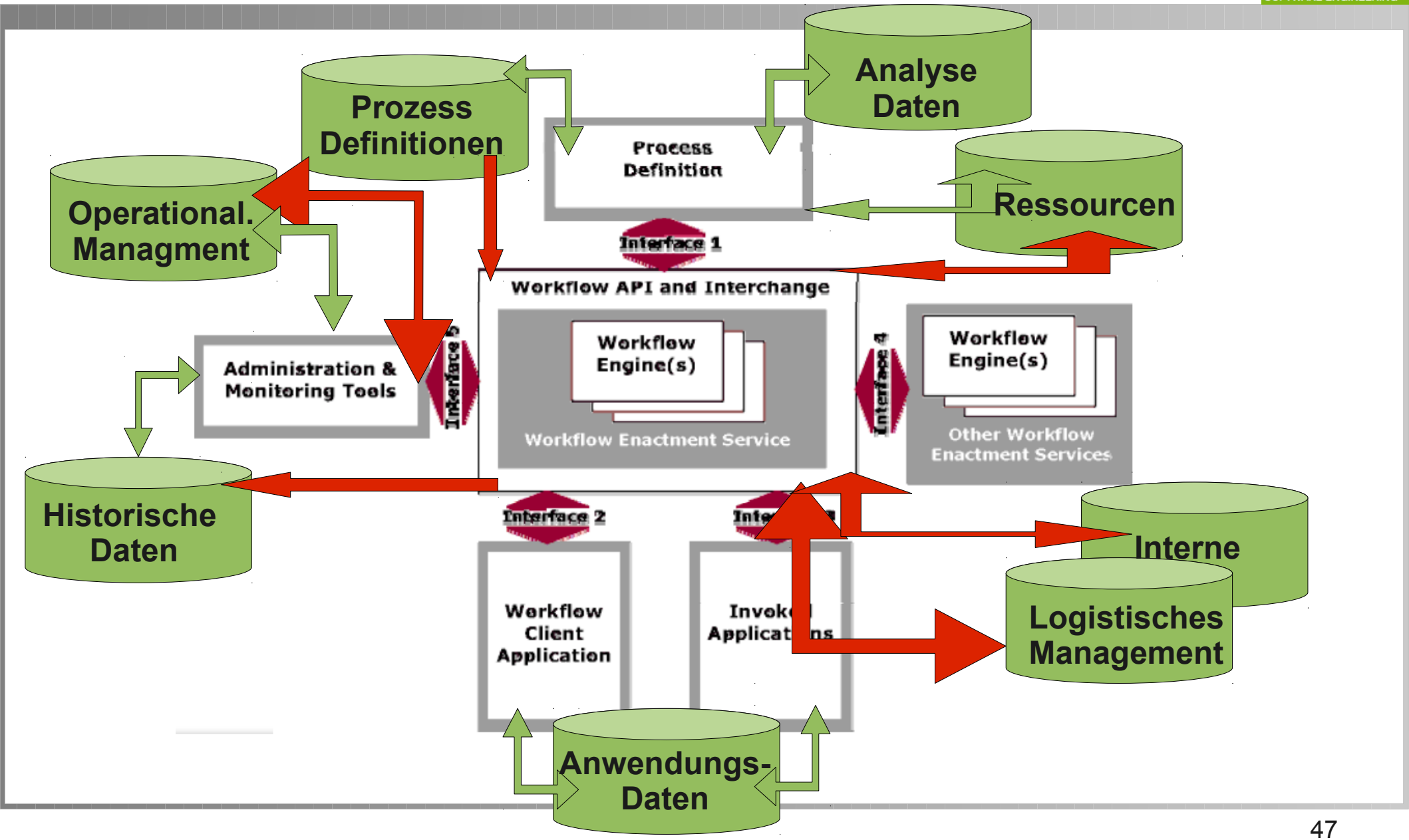
Generisches WF-Management System (WfMC)



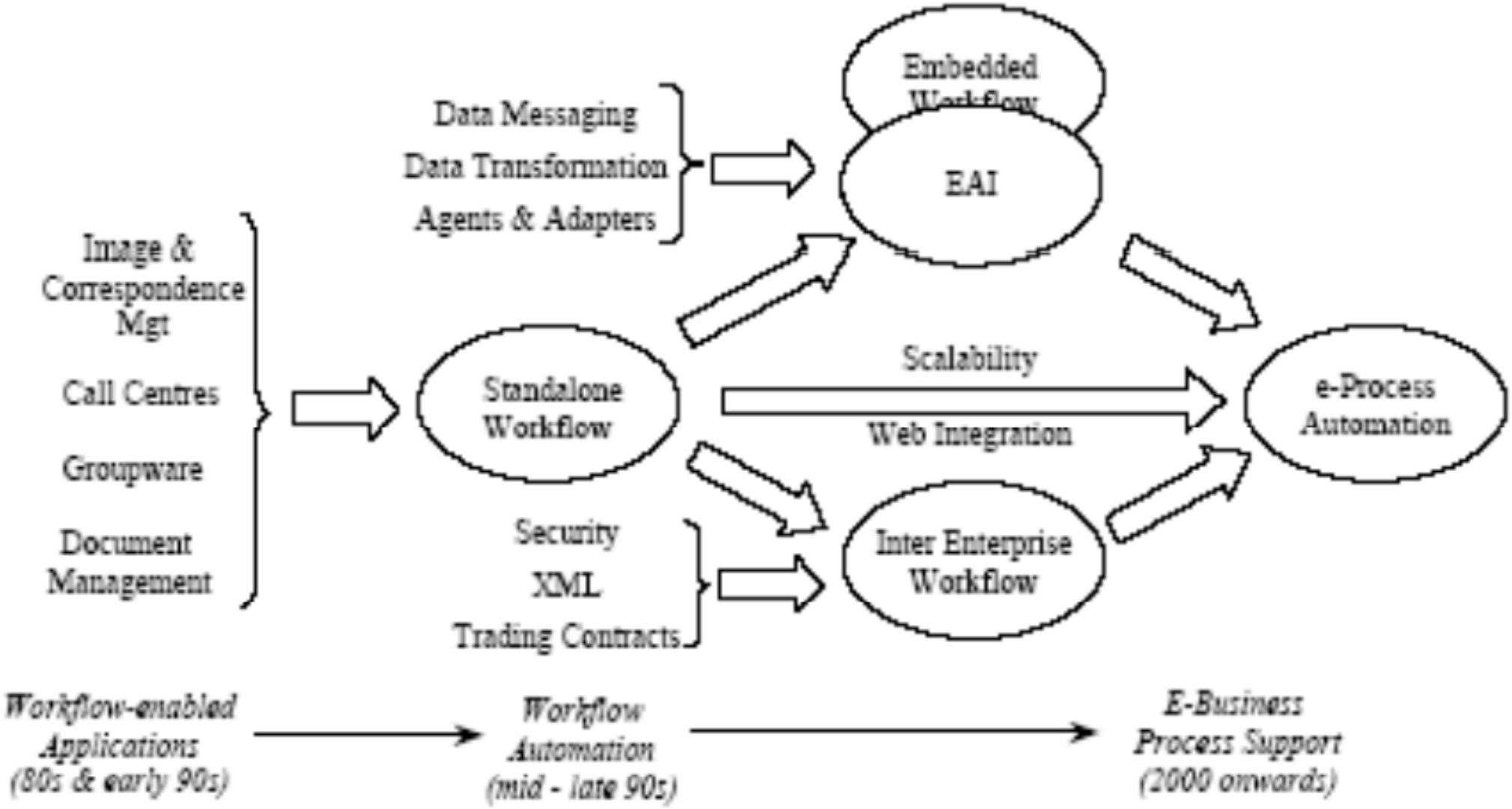
Klassifikation von Daten in einem WfMS (v.d. Aalst)

- Prozess-Definitionen (Schemata), inkl. Definitionen der Aktivitäten
- Ressourcen (Rollen, Organisationen)
- Analysedaten
- Operationales Management (technische Parameter)
- Historische Daten: Nachverfolgung einer Prozessinstanz
- Anwendungsdaten
  - Zugriff nur über Applikation, nicht über Engine
  - Masterdaten (generisch, z.B. Stammdaten) und Prozess-Instanzdaten (spezifisch)
- Interne Daten: technische Daten (z.B. Netzwerkadressen)
- Logistisches Management (z.B. Prozesszustände inkl. Attribute, Ressourcenzustände)

Daten am Beispiel des WfMC Referenzmodells



Weiterentwicklung von WfMS



WfMC-Implementierungen
Beispiel WfMOpen

- Open-Source
  - http://wfmopen.sourceforge.net/
- J2EE-basiert
  - EJB Container, JBoss
- Beherrscht nativ XPDL
  - WfMC-Standard zur Prozessdefinition
  - Wird aktuell nicht mehr separat weiterentwickelt, sondern an BPMN 2.0 ausgerichtet.
- Stark modulare Architektur, die die WfMC Referenzarchitektur umsetzt.

WfMC-Implementierungen
Beispiel WfMOpen

Process Definition

Interface 1 XPDL (WfMC)

Workflow-Engine

EJB-Container

WfProcess

Logging

WfActivity

Services

WfResource

Resource Assignment Service

Resource Management Service

Data-base

JDBC/ SQL

Administration & Monitoring

Servlet-Container

Cocoon

Management-Client

Interface 5

RMI / SOAP

Interface 4

WF-XML

HTML

Interface 2 RMI / SOAP

Interface 3 RMI / SOAP

Workflow Client Application

Servlet-Container

Cocoon

WebForm

HTML

XML, Corba Tuxedo etc.

Operating Systems

Invoked Applications

Mail-Tool

Script-Tool

⋮

ür
k

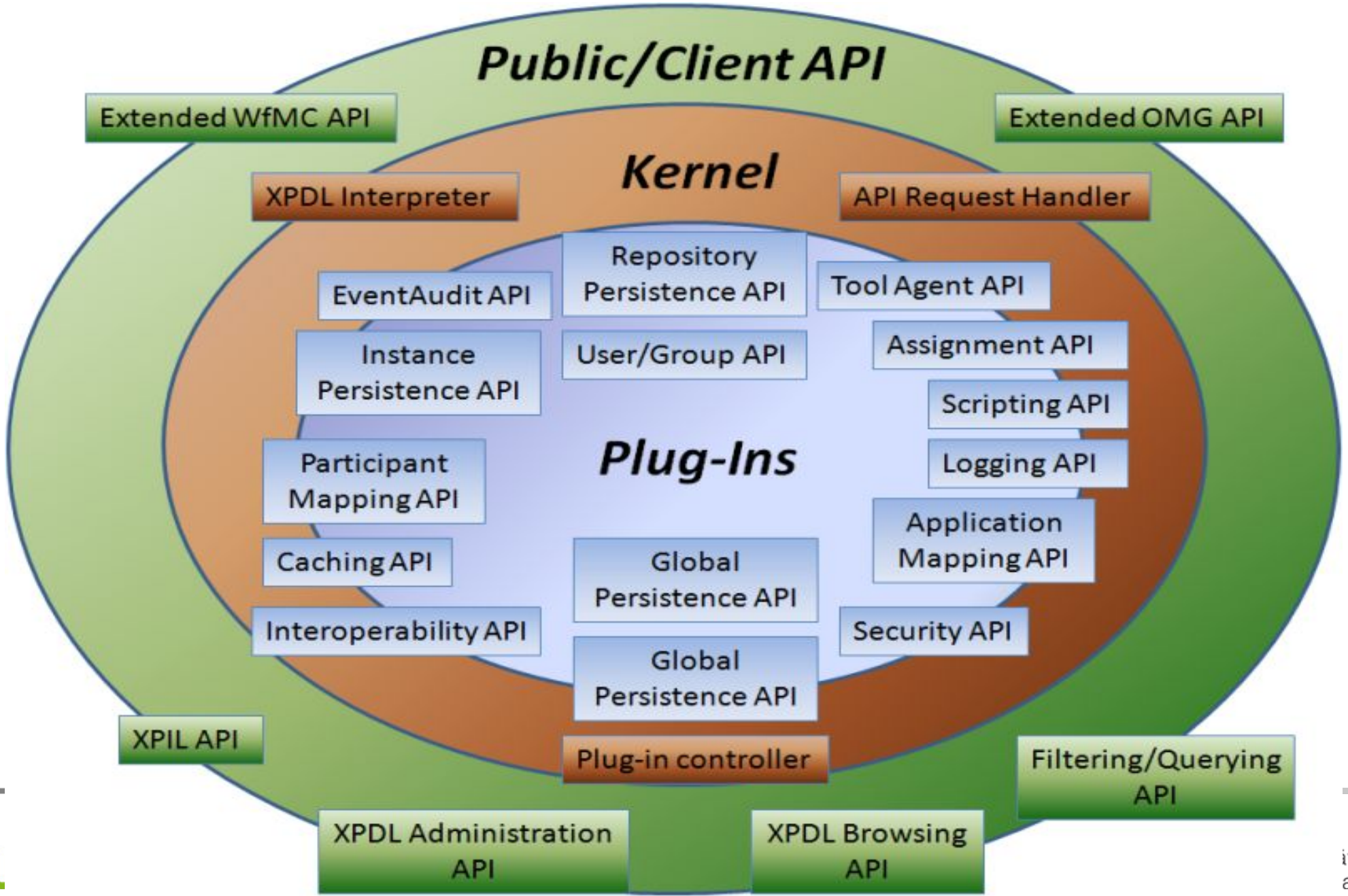
WfMC-Implementierungen
Beispiel WfMOpen

- Wird unter anderem eingesetzt von
  - STAIRS
    - Structured Audio Information Retrieval System
    - Forschungsprojekt SAP und TU Darmstadt
    - Augmented Working Environments
  - IniT
    - Im Bereich öffentliche Verkehrssysteme
  - T Mobile
    - Unterbau für T-Traffic (digitale Karten für Navis)
  - Und einige andere mehr

WfMC-Implementierungen
Beispiel Enhydra Shark

- Open-Source
  - http://sourceforge.net/projects/sharkwf/
  - Kommerziell: Together Workflow Server
- Java-basiert
  - Als Library implementiert (POJO)
  - Verschiedene Wrapper für unterschiedliche Szenarien
    - EJB, CORBA, RMI, WebService
- Adaptiert unter anderem WfMC
  - Aufbau WfMC konform und implementiert Interfaces
  - Bietet Schnittstellen konform zu anderen Standards

WfMC-Implementierungen
Beispiel Enhydra Shark



WfMC-Implementierungen
Beispiel Enhydra Shark

- Partner
  - Abacus
  - Adobe
  - BOC
  - IBM
  - Microsoft

WfMC-Implementierungen

- Es existieren noch weitere Open Source Implementierungen
  - Open Business Engine (http://obe.sourceforge.net/)
  - PL/Flow (http://plflow.sourceforge.net/)
  - …
- Viele kommerzielle Anbieter haben den Conformance-Test der WfMC durchlaufen
  - Testen allerdings nur die Existenz der Schnittstellen, nicht die eigentliche Architektur
- Veröffentlichte Liste dazu etwas veraltet
  - Es ist anzunehmen, dass die Anbieter schon wesentlich weiter sind.

WfMC-Implementierungen
Conformance Test List (Auszüge)

Vendor	Product/Version	Release Date	WfMC Interface Supported	Implemented	Web Address for Vendor Conformance Statement
Action Technologies	ActionWorks Metro 5.0	27/4/98	IF2/3 IF4	Prototypes Demonstrated	Action Technologies
Argo 21	Nautica Workflow 0.9	9/2005	IF1 IF2/3 IF4 IF5	Implemented XPDL 1.0 with GUI based editor Implemented Yes(Original) Yes	Nautica Project
BancTec/Plexus Software	FloWare V 5.0	9/98	IF1 IF2/3 IF4	Yes Prototype Demonstrated Planned	Plexus
Blockade Systems	ManageID™4.2	2/2003	IF1 IF4	Yes (also available in 4.1) Yes (used internally in 4.1)	Blockade Systems
BOC GmbH	ADONIS 3.81	2005	IF1 IF5	Implemented Prototype	BOC GmbH
CARNOT AG	CARNOT Process Engine 3.1	March 2005	XPDL, Wf-XML 2.0	Yes	CARNOT AG

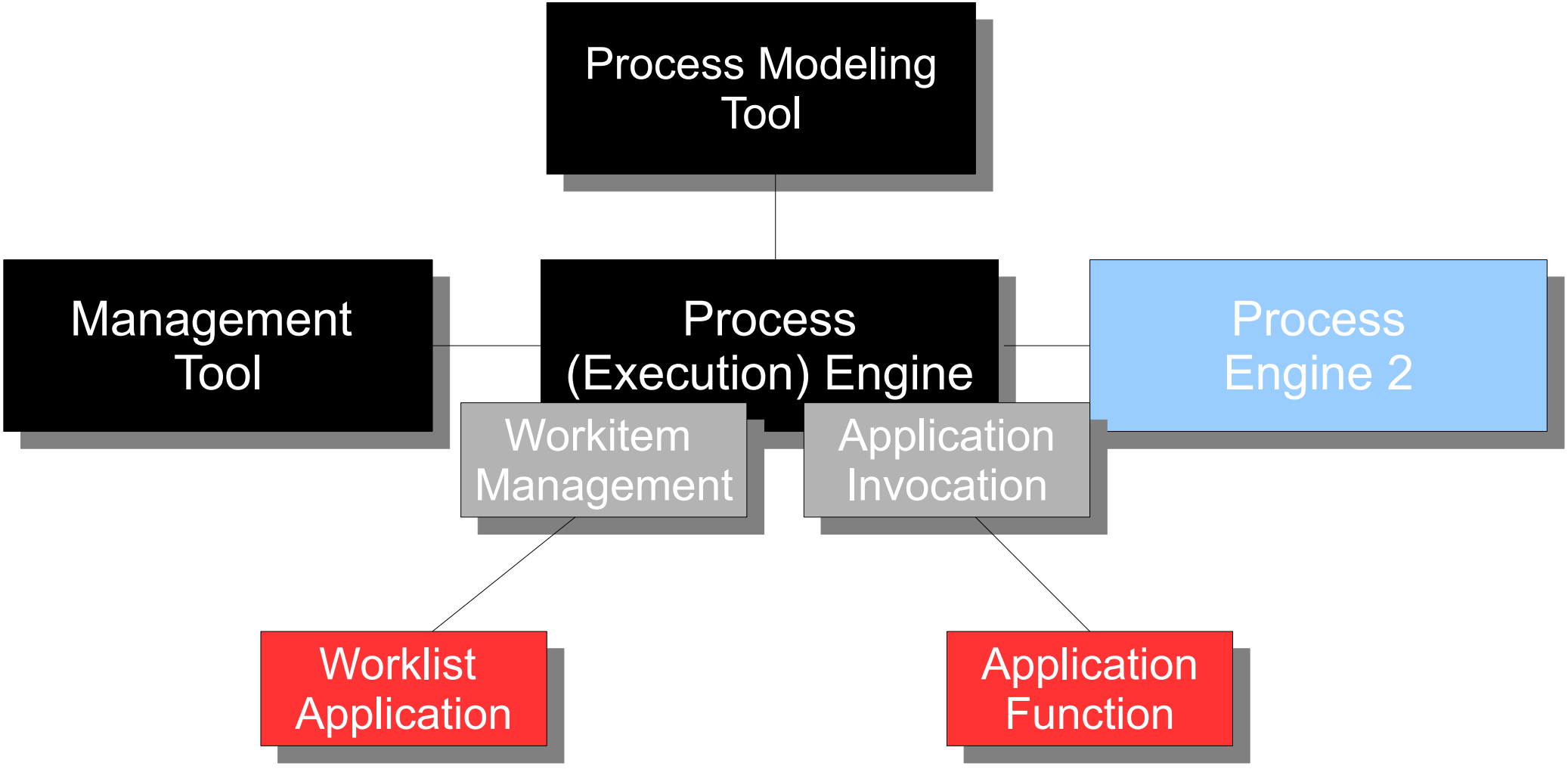
WfMC-Implementierungen
Conformance Test List (Auszüge)

Filenet (IBM)	Visual WorkFlo 3.0 Integrated WorkFlo 1.22	9/98	IF4	Yes	FileNET
Fujitsu	Interstage BPM (i-Flow)	2003	IF1,IF2/3,IF4	Yes	Fujitsu
Hitachi Ltd	Groupmax Workflow V5	5/99	IF4	Yes	Hitachi
	WorkCoordinator	11/98	IF2/3	Yes	
			IF4	Planned	
IBM	WebSphere MQ Workflow	2003	IF1	Implemented	IBM
	WebSphere Business Integration		IF2/4/5	Implemented	
Identitech, Inc.	FYI Workflow™	2003	WF-XML and XPDL	Planned	Identitech
IDS Scheer	ARIS Toolset 3.2 Modelling and BPR Tool	1997	IF1	Prototype	IDS Prof Scheer
			IF2/3,IF4,IF5	NO	

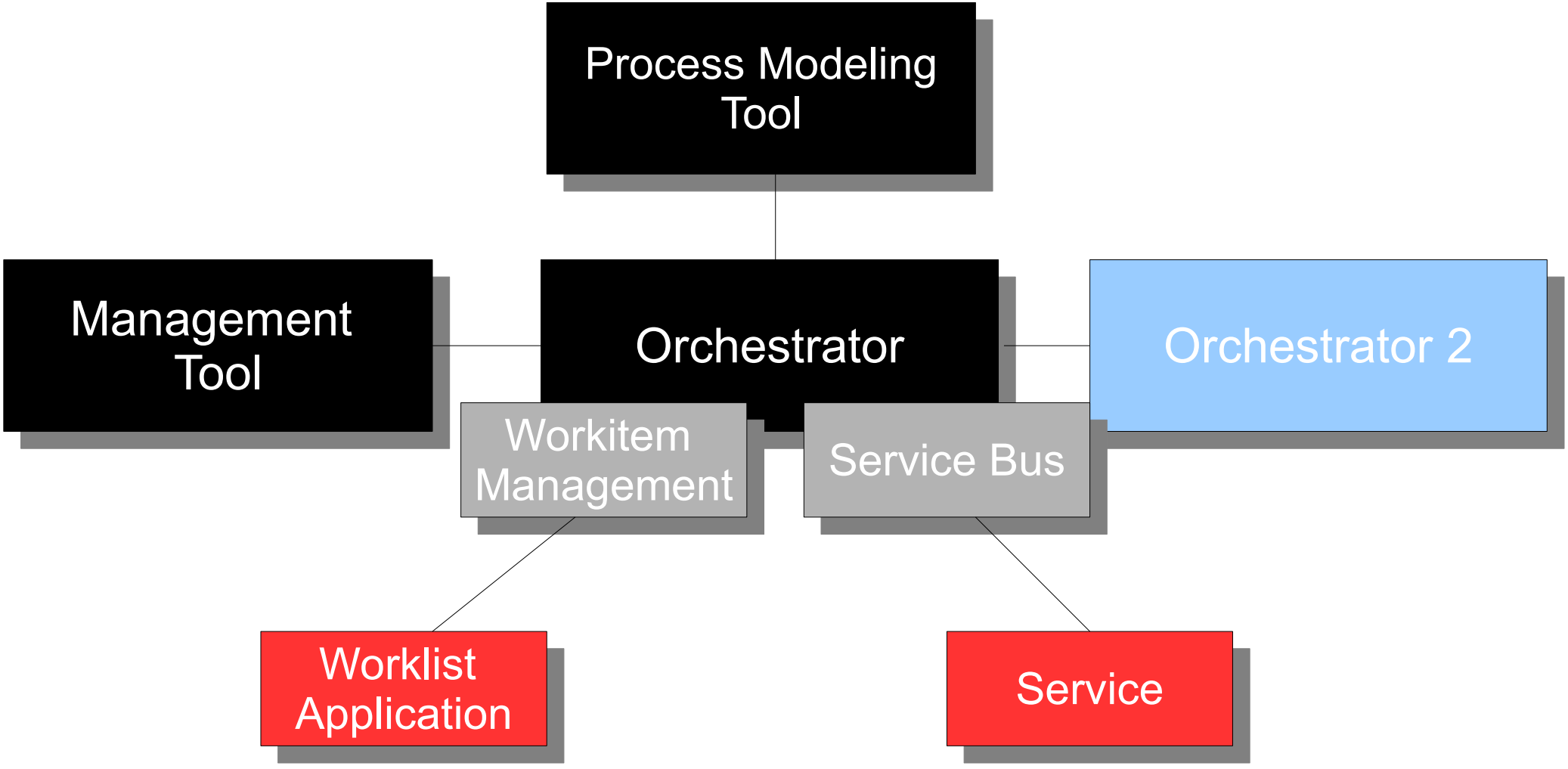
WfMC-Implementierungen
Conformance Test List (Auszüge)

SAP	SAP WebFlow/ SAP Business Workflow	1997 R/3. Rel. 3.1 2000 Technology Release 4.6C	IF2/3 Wf-XML	Implemented Implemented	SAP
Savvion	WebDeploy: WorkFlow 1.2	6/98	IF2	Yes	TDI
Together Teamlösungen and ObjectWeb	Enhydra Shark 1.1-2	1/2006	IF1 IF2/3 IF4 IF 5	Yes Yes Demonstrated adherence to draft versions of standards yes	Enhydra Shark http://shark.enhydra.org

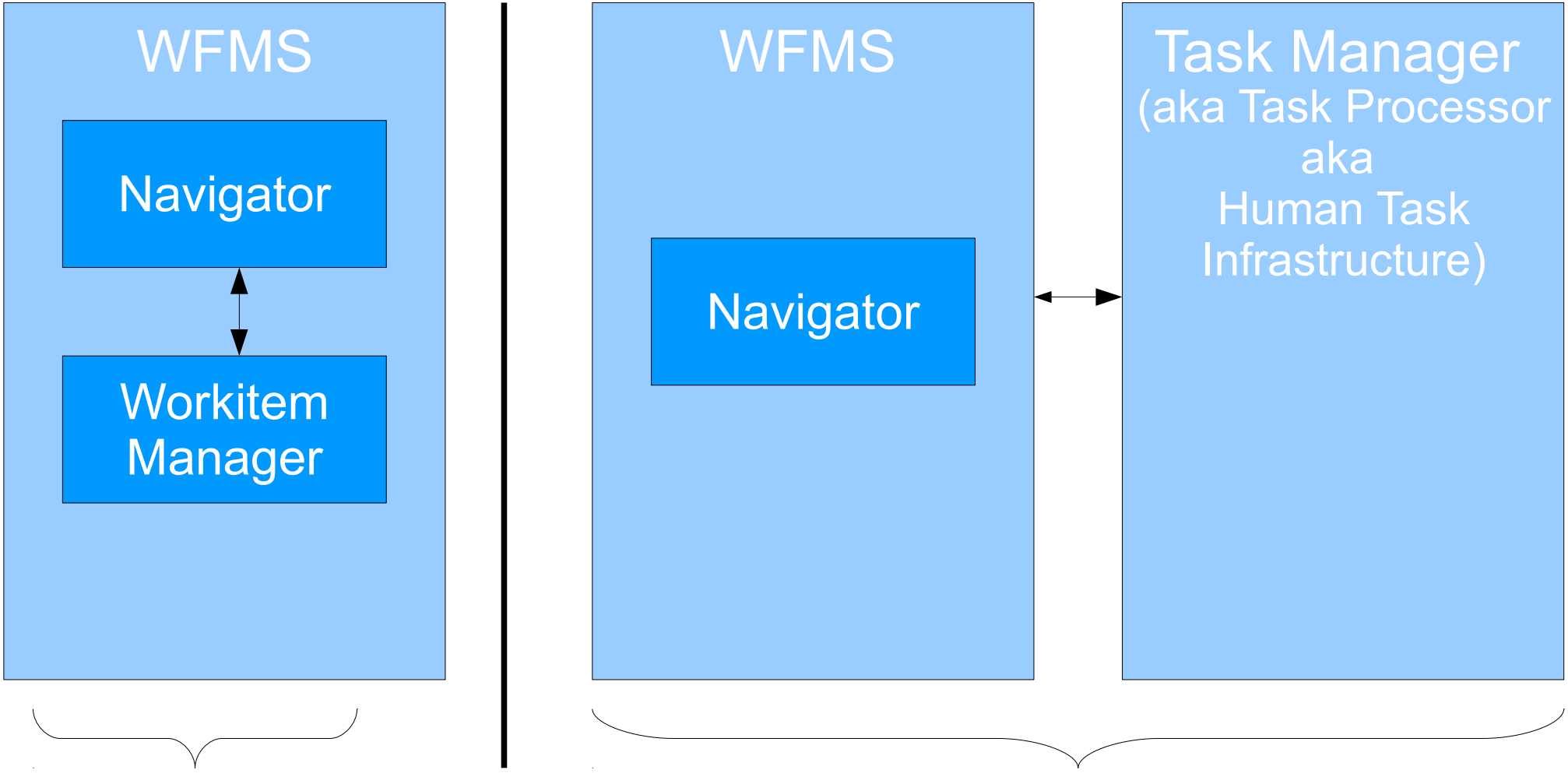
Workflow Management Systeme: Moderne Terminologie



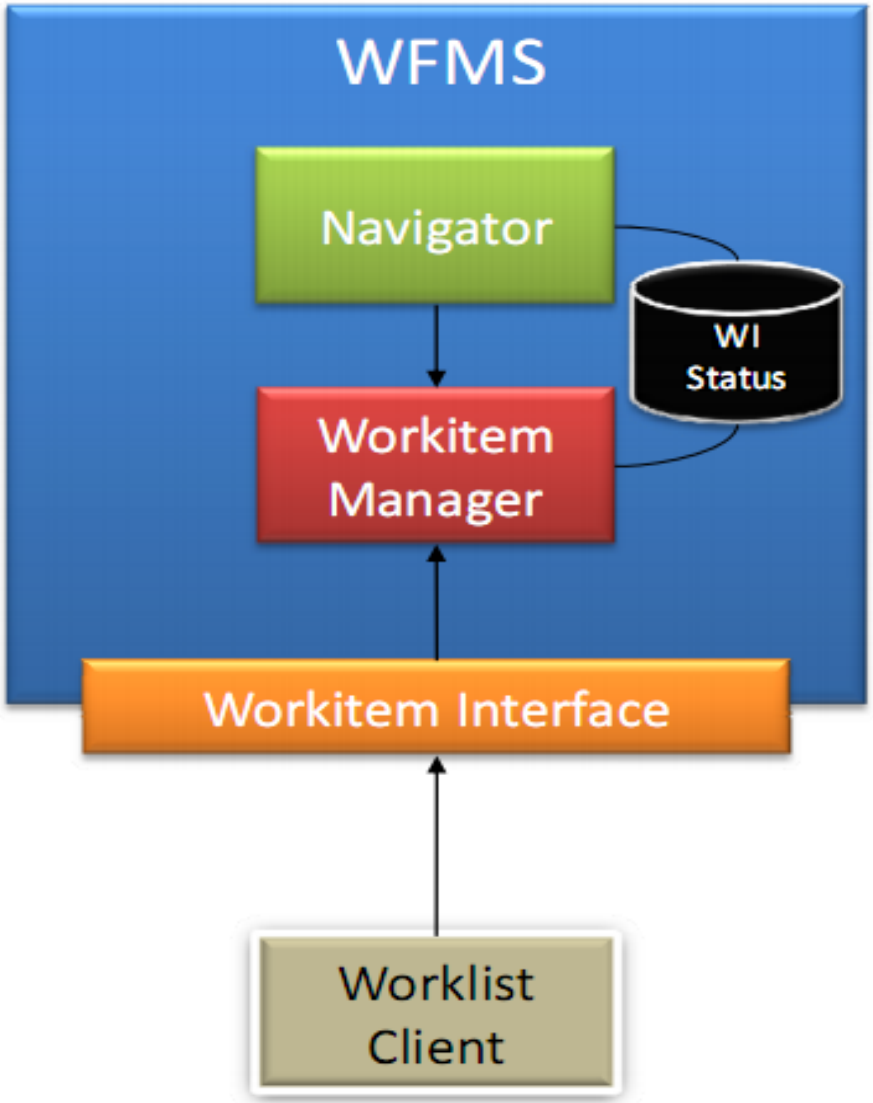
Workflow Management Systeme: Noch modernere Terminologie

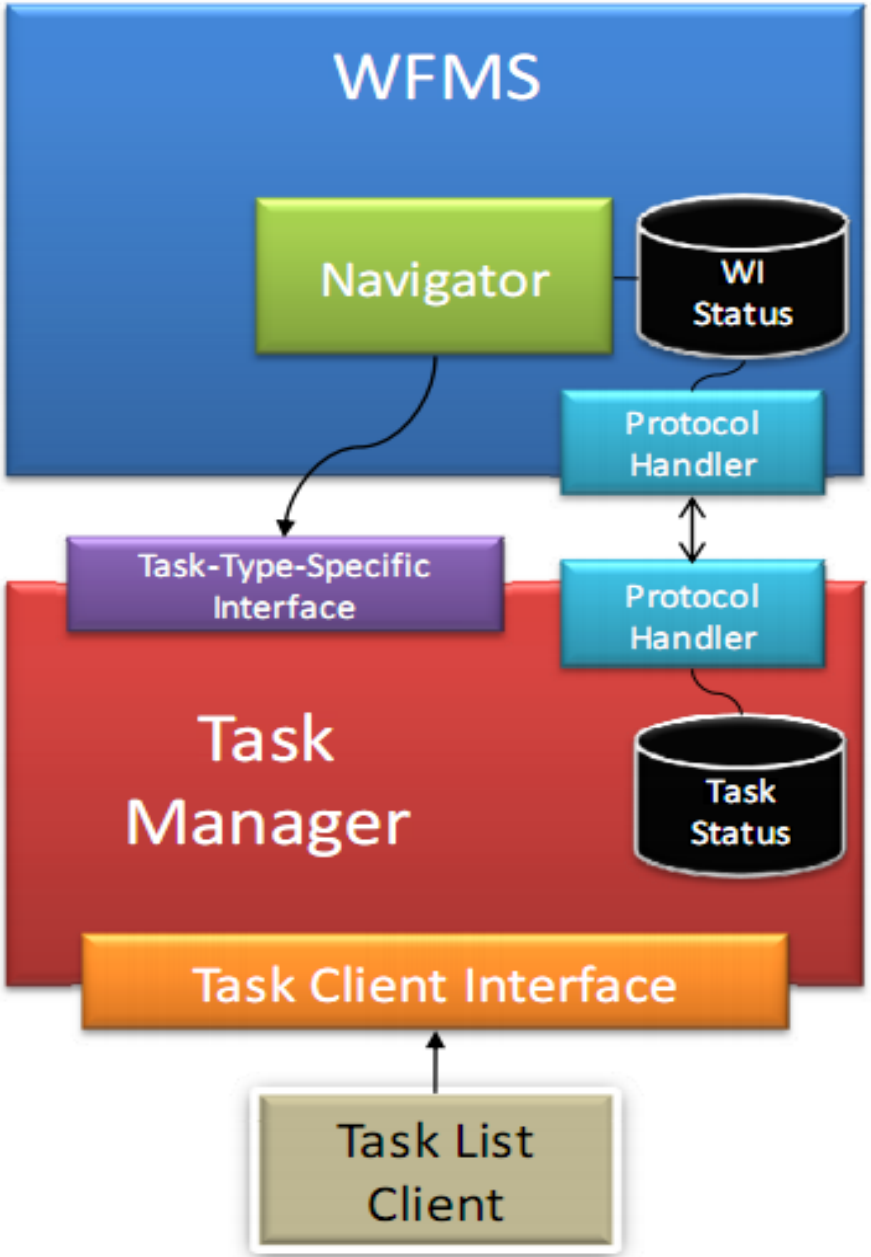


Architektonische Auswirkung von BPEL4People

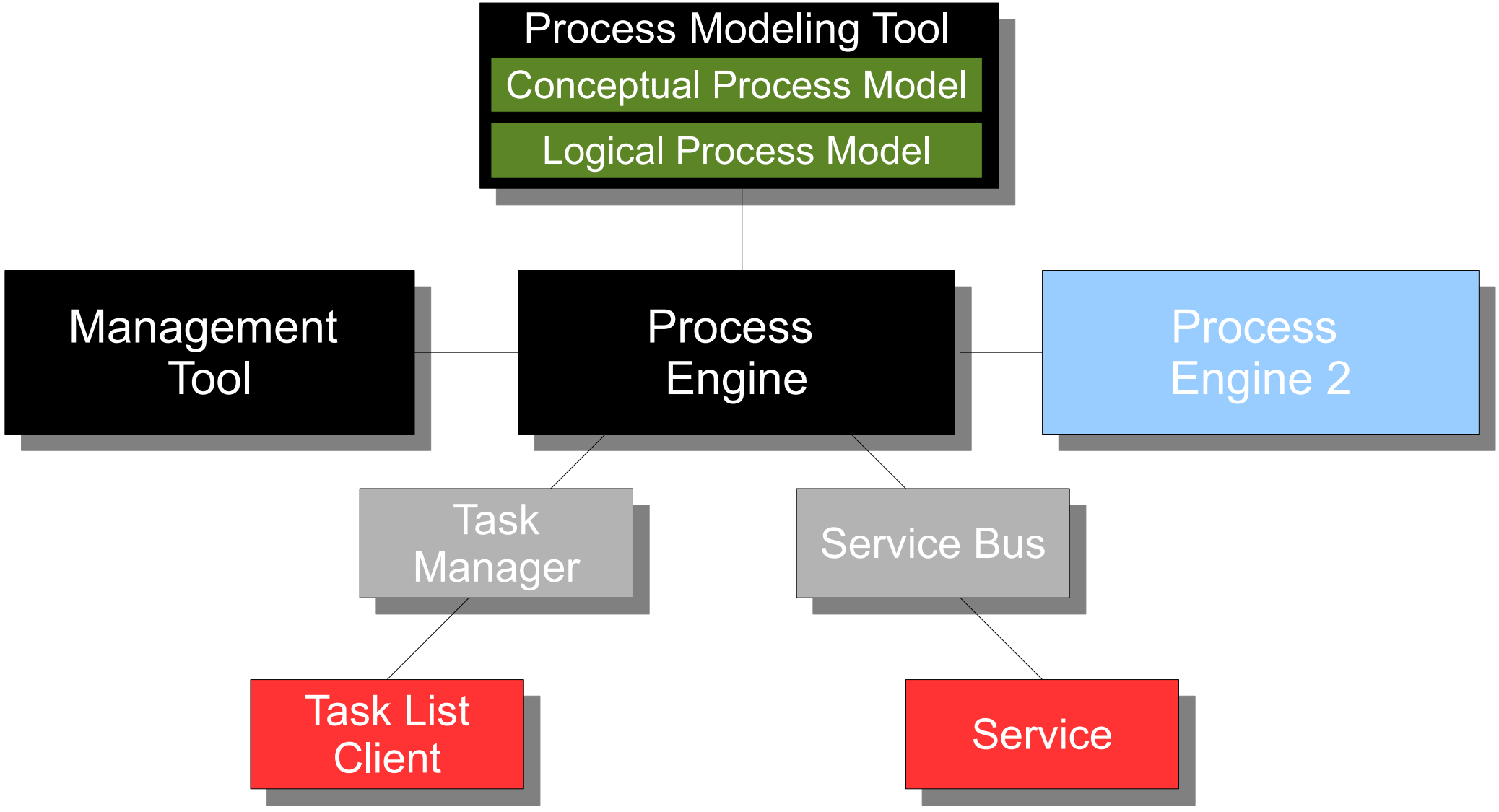


Architektonische Auswirkung:
Verfeinerte Ansicht



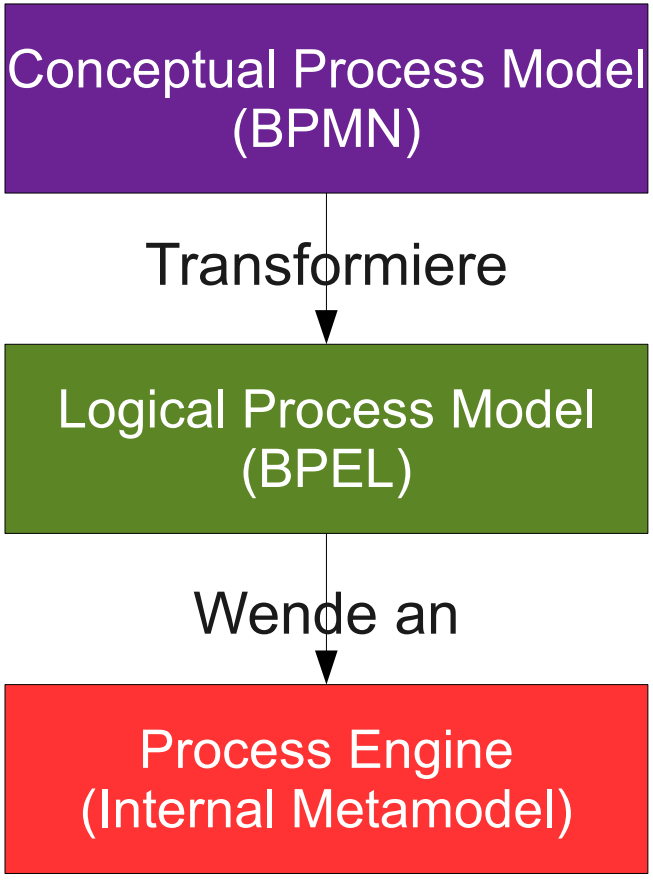


Resultierende SOA-Ansicht auf das WfMC Referenz Model



Schichten des Process Modeling

Process Management Stack



Vergleich: Database Management System

Process Management Stack

Conceptual Process Model (BPMN)

Transformiere

Logical Process Model (BPEL)

Wende an

Process Engine (Internal Metamodel)

Data Management Stack

Conceptual Data Model (Entity-Relationship)

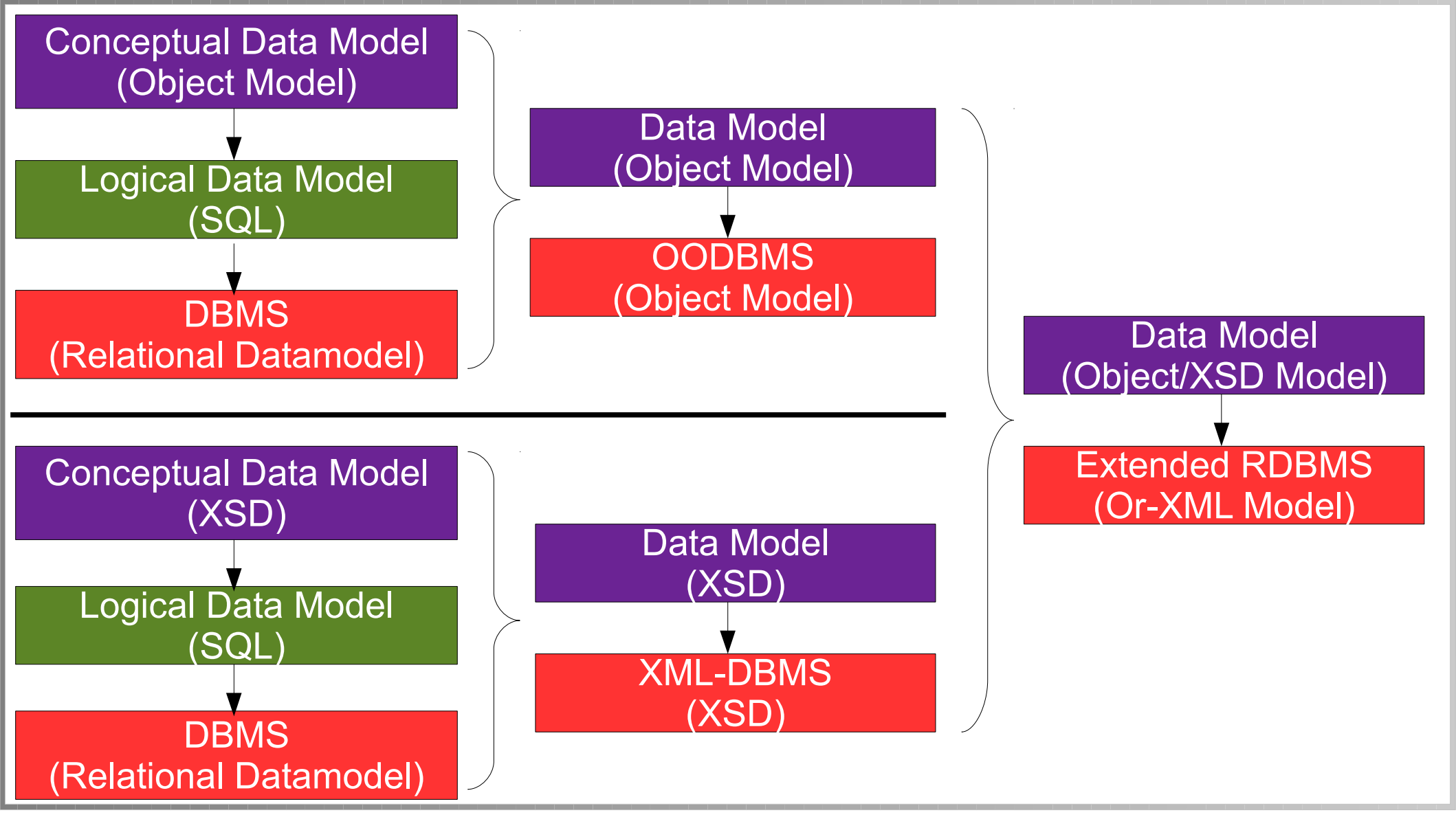
Transformiere

Logical Data Model (SQL)

Wende an

DBMS (Relational Datamodel)

DBMS: Was geschah in der Vergangenheit?



Die offene Frage

Wird sich die Geschichte in der „Prozess Welt“ wiederholen?

Der Begriff des „nativen Metamodells“

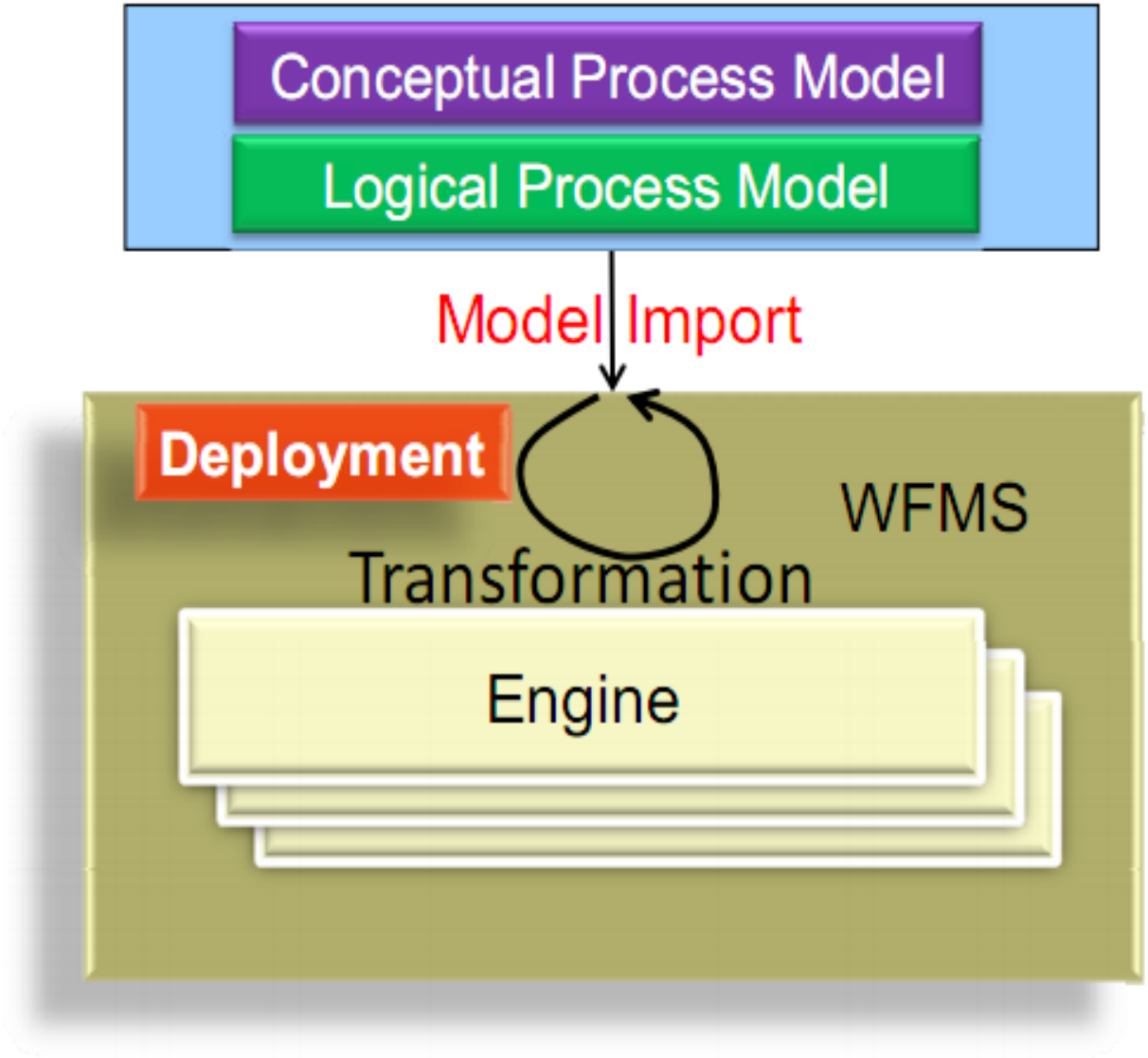
Die Absicht eines nativen Metamodels

- Was bedeutet es, eine BPEL (BPMN) Engine zu sein?
- Die Engine kann Prozessmodelle importieren, die in BPEL (BPMN) spezifiziert sind, *entsprechend der operationalen Semantik von BPEL (BPMN).*
- Oft werden während des Imports die BPEL- (BPMN-) Artefakte bereits in interne Artefakte der Engine abgebildet:
  - **Die meisten Engines implementieren ihr eigenes Process Metamodel (Native Metamodel).**
- „Native“ BPEL (BPMN) Engines sind seltene Ausnahmen!
  - *Native BPEL (BPMN) := BPEL (BPMN) ist das interne Metamodel*

Was bestimmt ein Natives Metamodel einer Engine?

- Die Hersteller der Engines müssen sich dafür entscheiden, wie sie die Effizienz, Skalierbarkeit etc. der Anwendung sichern.
- Modellierungssprachen müssen ab dem ersten Release der Prozess-Engine unterstützt werden, dies hat eine große Auswirkung auf die nativen Metamodelle!
- Gute Engine-Ersteller kümmern sich um die Allgemeingültigkeit und Erweiterbarkeit des nativen Metamodels, wobei eine möglichst einfache Unterstützung von Erweiterungen existierender (und sogar von neuen) Modelierungssprachen angestrebt wird.
- **Normalerweise kennen die Benutzer der BPEL Engine nicht das native Metamodel und wissen nicht, wie unterschiedlich dieses zu BPEL als Metamodel selber ist.**

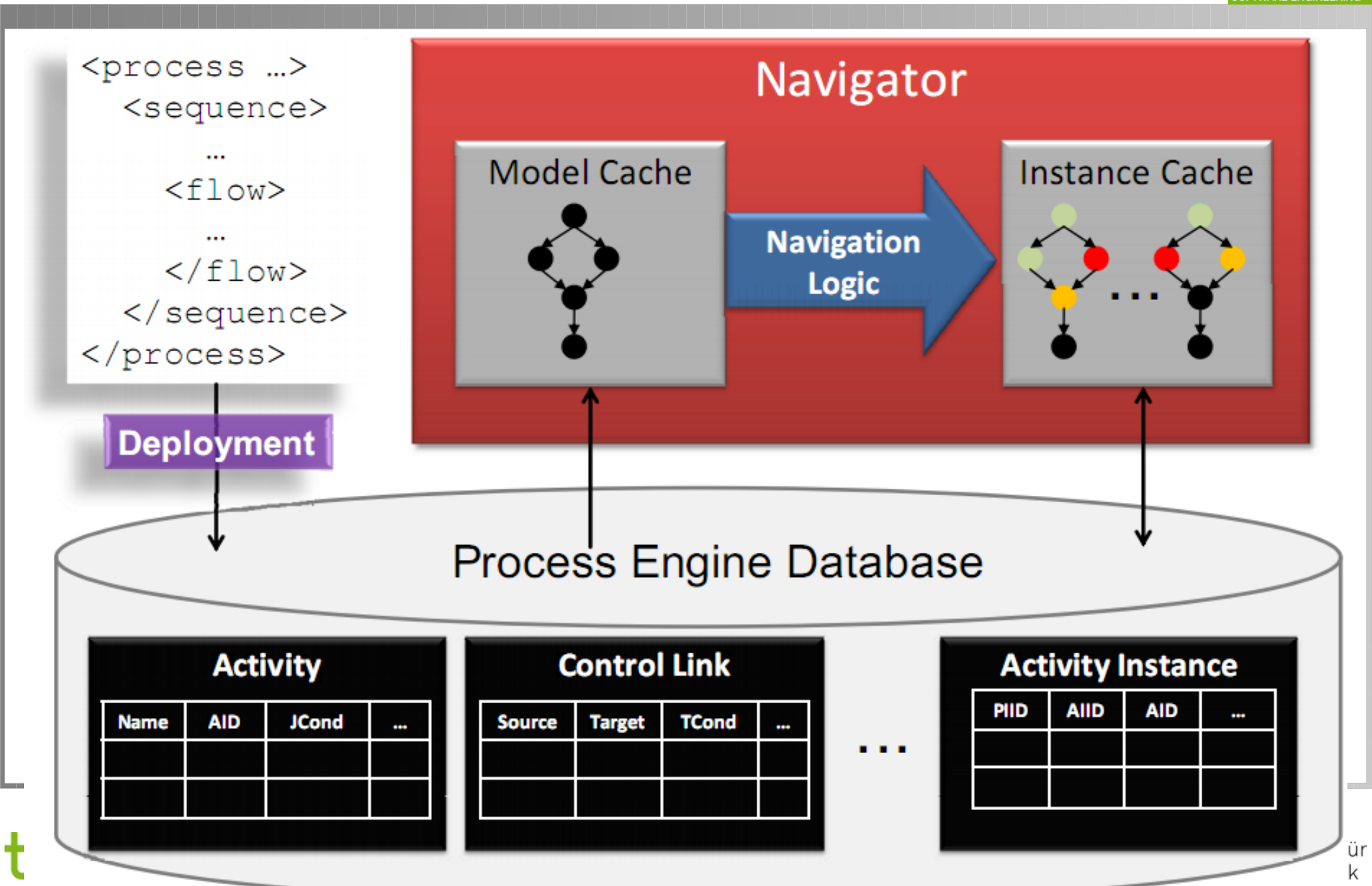
Die Rolle der Anwendung



Anwendung: Deployment

- **Deployment** = Das Prozessmodell produktiv schalten.
  - z.B. bereit für die Ausführung machen
- Die Anwendung übersetzt normalerweise **das korrespondierende Modell** in unterschiedliche Formate.
  - Das importierte Modell kann in einem Metamodell spezifiziert sein, welches sich von dem, was das WFMS direkt unterstützt, unterscheidet.

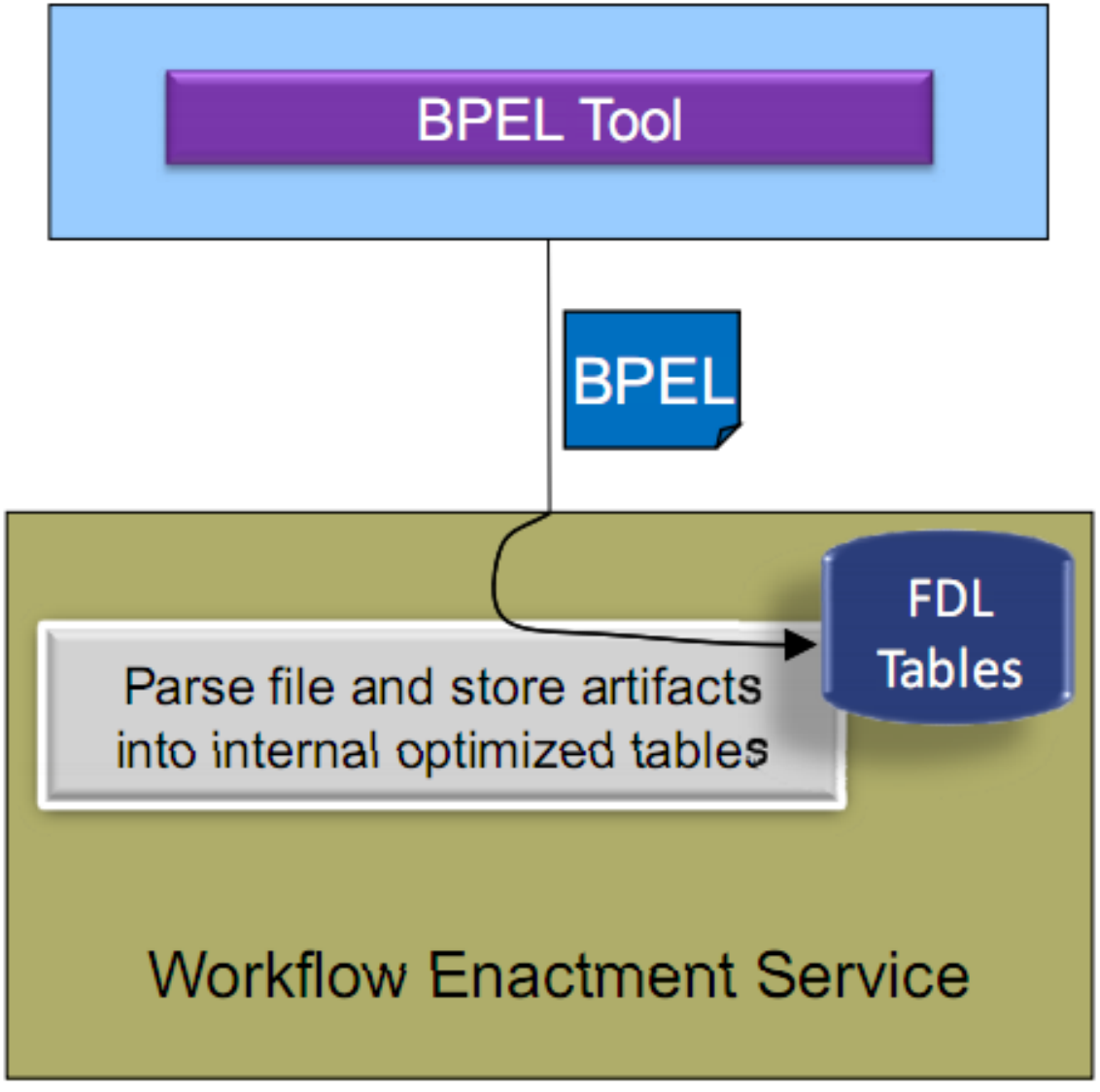
Anwendung: Überblick



Native Unterstützung eines Metamodels

- Jede Workflow Engine unterstützt **nativ** ein bestimmtes Metamodell.
- Üblicherweise wird das native Metamodell...
  - direkt unterstützt in der Datenbank des WFMS.
    - Das Datenbankschema enthält sofort Instanzen des Metamodellkonstrukts
    - Das Datenbankschema ist zur Unterstützung des Navigators angepasst.
  - direkt unterstützt in dem Zustandsmodell des WFMS.
    - Alle Metamodellkonstrukte haben eine Menge von Zuständen und Transitionen.
    - Das Zustandsmodell ist im Monitoringmodell und Protokoll reflektiert.
  - direkt im Navigator des WFMS implementiert.
    - Navigator versteht direkt jedes Metamodellkonstrukt, dessen Zustände, dessen gültige Transitionen und die Relation zwischen den Zuständen verschiedener Artefakte.
    - die Implementierung ist „optional“.

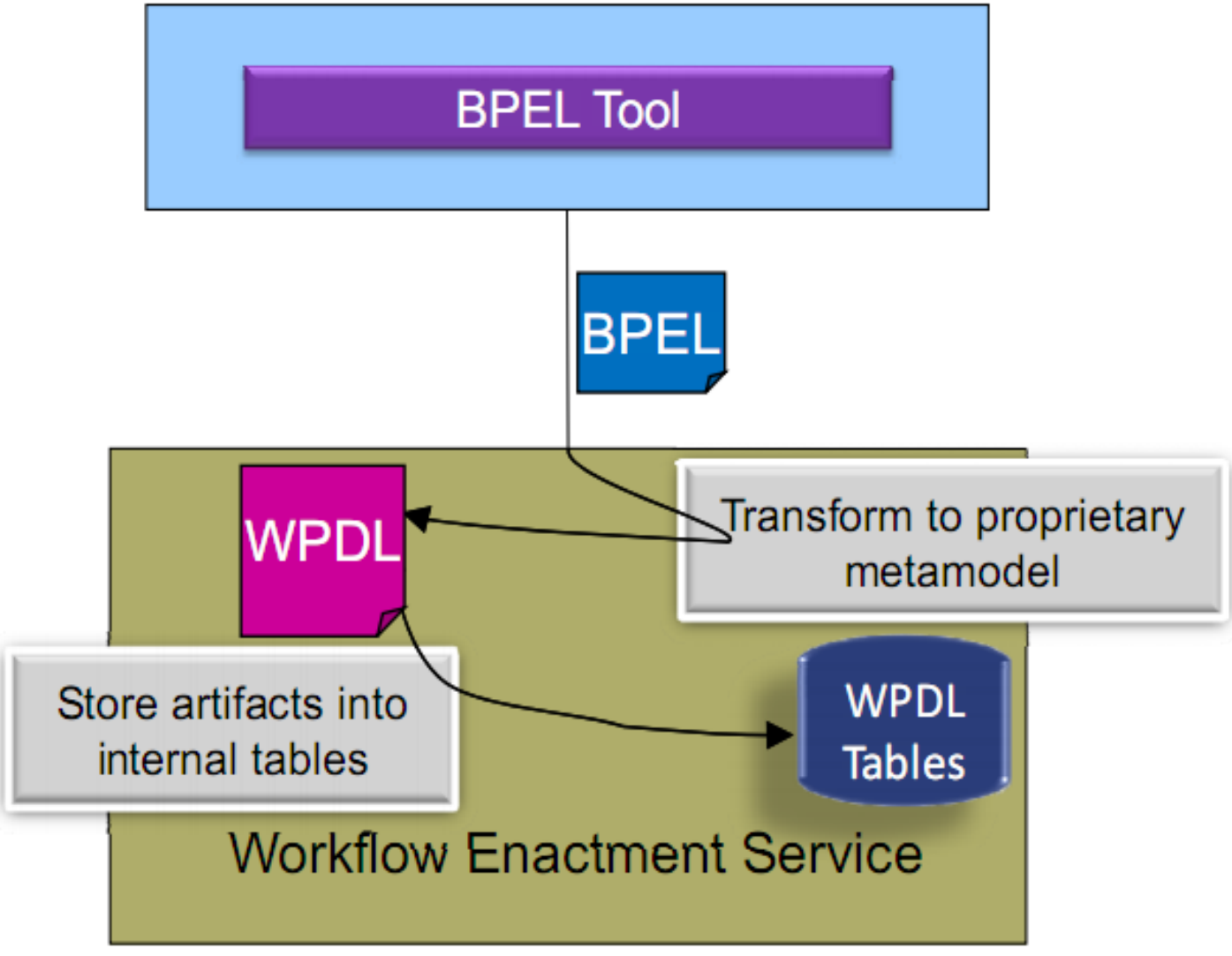
Transformation während Anwendung: Beispiel



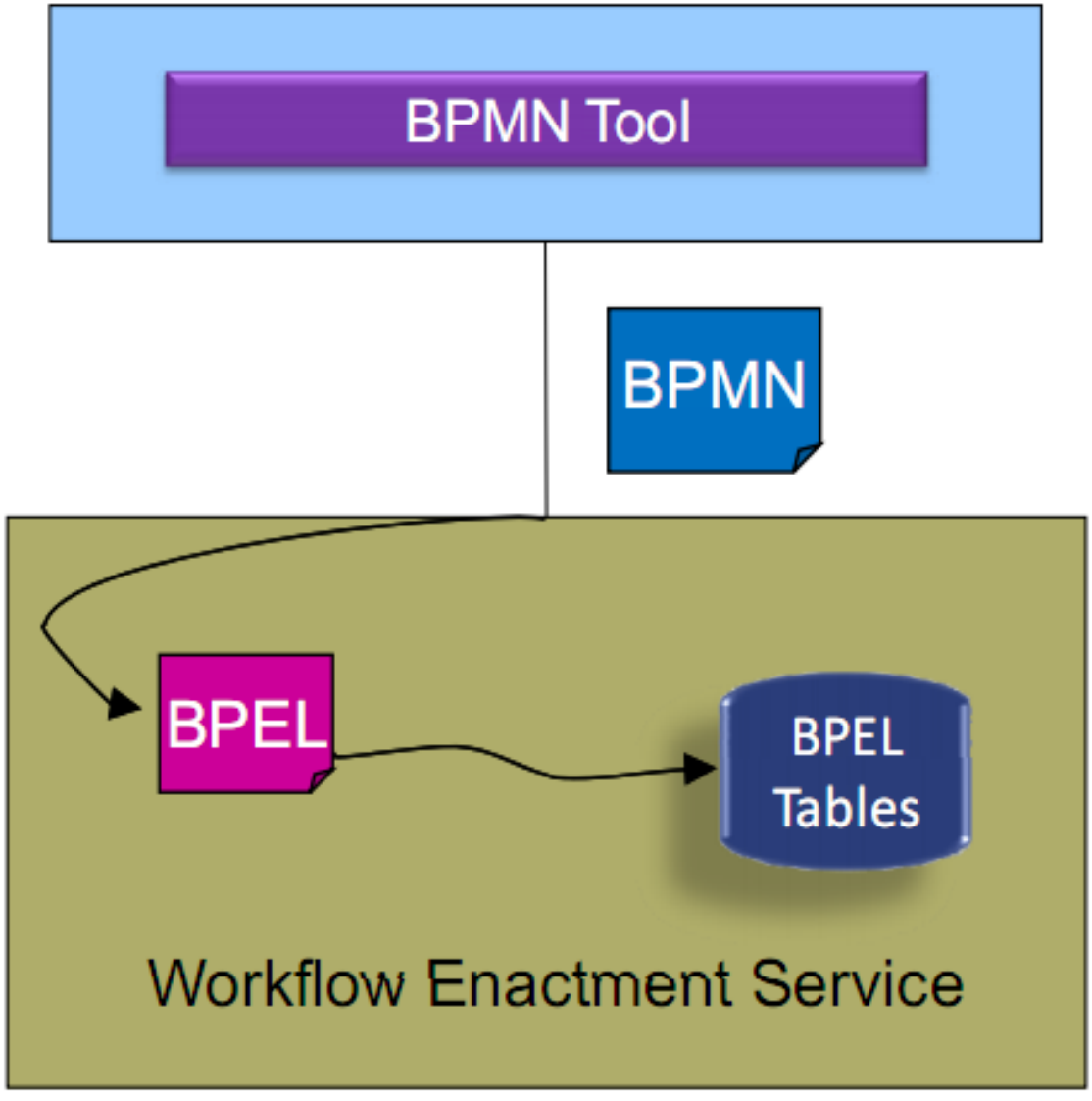
Anm.: FDL = Flow Definition Language. Vgl IBM Websphere.

(http://publib.boulder.ibm.com/infocenter/dmndhelp/v6rxmx/index.jsp?topic=/com.ibm.btools.help.modeler.doc/doc/concepts/overviews/productfamilyoverview.html).

Transformation während Anwendung: Beispiel



Transformation während Anwendung: Beispiel



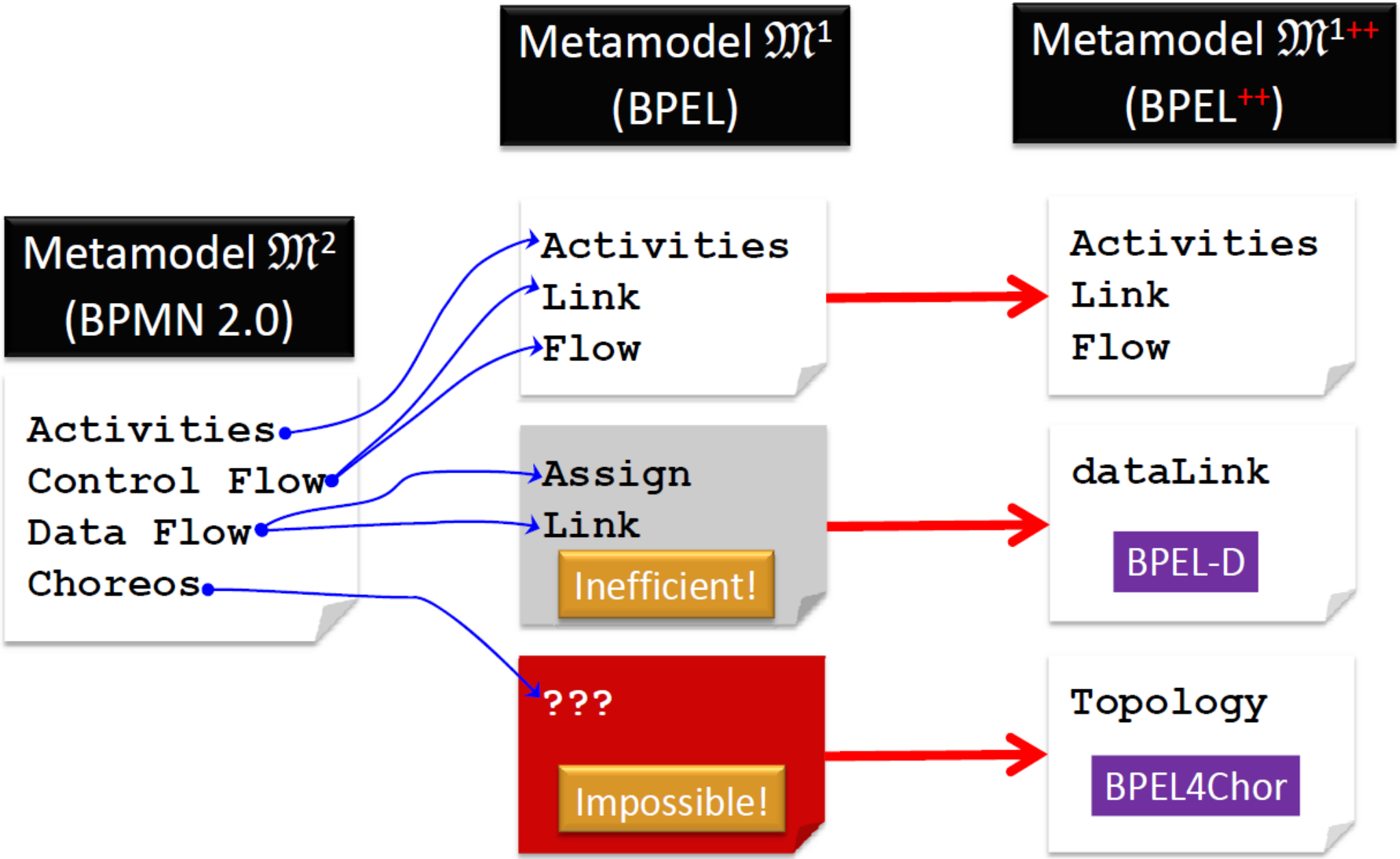
Modell-Transformationen

- Prozessmodelle, die in einem anderen Metamodell spezifiziert sind, als das der Engine, müssen in dieses native Metamodell, welches von einer eigenen WFMS unterstützt wird, umgewandelt werden.
- Beispiel: Auf diese Art kann ein WFMS, das WPDL nativ unterstützt, ein BPEL-Modell nach der dazugehörigen Transformation ausführen.
- Beispiel: Auf diese Art darf ein WFMS, das BPEL nativ unterstützt, ein BPMN-Modell nach der dazugehörigen Transformation ausführen.

Probleme der Modell-Transformationen

- Solche Transformationen sind nicht immer genau, d.h. das transformierte Modell hat ein anderes...
- … semantisches Verhalten als das Originalmodell
  - z.B. das Verhalten des Ausgangsmodell muss im Zielmodell emuliert werden – wenn möglich
    - z.B. BPEL Exception-Verhalten ist schwer zu emulieren
- … operatives Verhalten als das Originalmodell
  - z.B. der Navigator führt das transformierte Modell weniger effizient aus als ein WFMS, das das Metamodell des Originalmodells unterstützt.
    - z.B. die Unterstützung eines FDL-Datenflusses in BPEL ist schwerfällig.

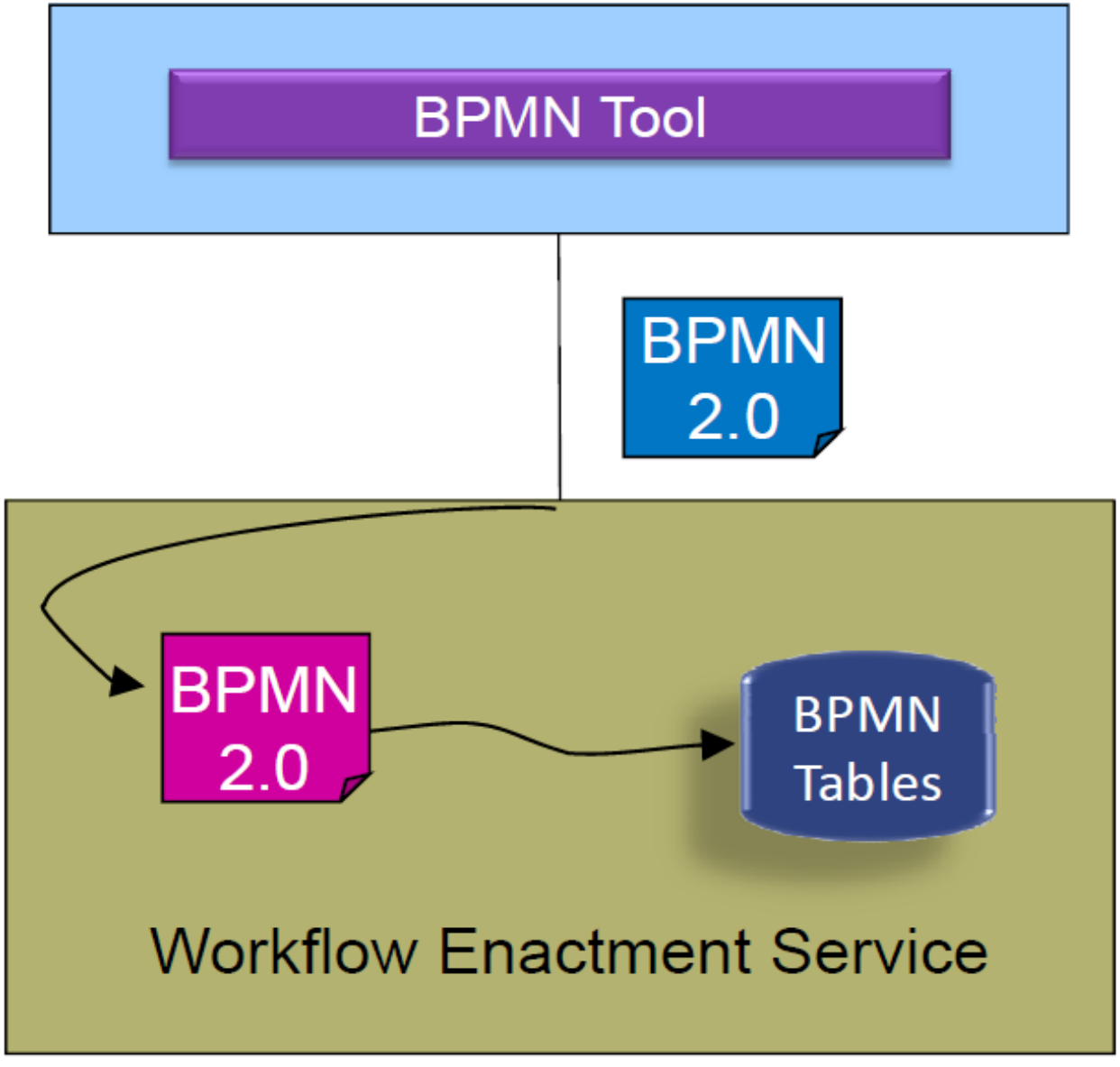
Unterstützung eines Metamodells in einem anderen



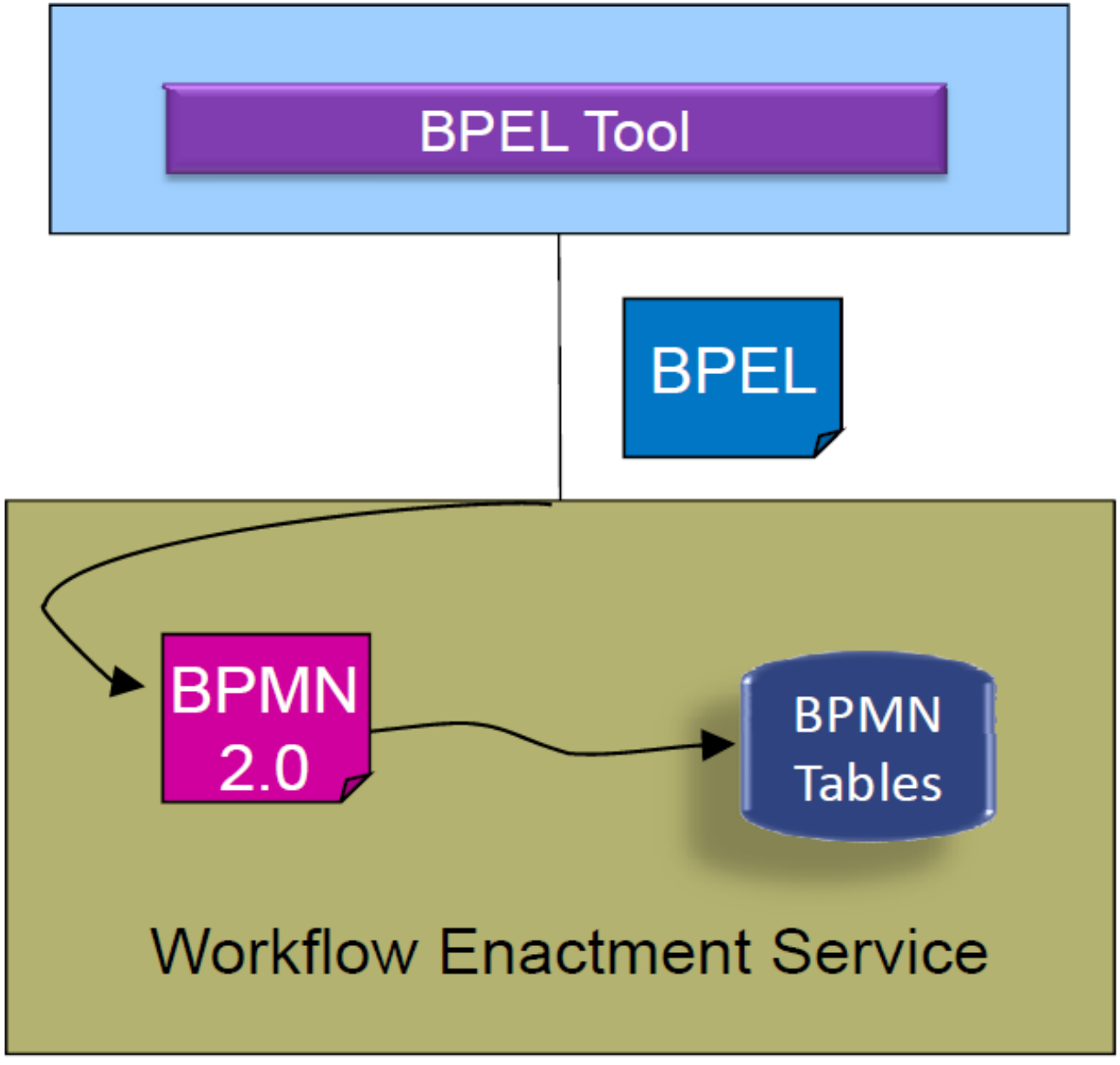
Das Erweitern eines gegebenen Metamodells

- Modelle, die in einem bestimmten Metamodell M^2 spezifiziert wurden, müssen in einer Engine mit einem anderen Metamodell M^1 perfekt unterstützt werden.
- Demnach müssen Konstrukte aus M^2, die schwer in M^1 zu emulieren sind, identifiziert werden, um entsprechende Konstrukte in M^1 hinzuzufügen.
- Das ist der Grund, warum BPEL erweiterbar entworfen wurde: Damit neue Konstrukte hinzugefügt werden können, um optimale Zuordnungen in verschiedenen Metamodellen zu gewährleisten.
- Eine erweiterte Variante einer gegebenen Engine (eine M^{1++} Engine) könnte so z.B. Prozessmodelle von anderen Metamodellen M^2, M^3, ..., M^n unterstützen.

Die Anfänge der BPMN Engines



Neues Anwendungsszenario: Transformation von BPEL nach BPMN



Konsequenzen für BPMN 2.0 Engines

- Ob ein BPMN 2.0 Prozessmodell durch eine BPEL Engine oder eine neue BPMN 2.0 Engine ausgeführt wird, ist nicht wichtig.
- Ein natives Metamodell einer BPMN 2.0 Engine sieht sehr wahrscheinlich anders aus, als BPMN 2.0 selber als Metamodell.
- Ein natives Metamodell einer BPMN 2.0 Engine könnte auch durch BPEL unterstützt werden.
- D.h. ein Anbieter einer BPEL Engine, welche die Fähigkeit besitzt, ein BPMN 2.0 Prozessmodell zu importieren, kann auch zum Anbieter einer BPMN 2.0 Engine werden, basierend auf der gleichen 'execution engine'.
- Ggf. kann ein vorläufiges BPEL Prozessmodell aus einem BPMN 2.0 Prozessmodell generiert werden, bevor das Importieren stattfindet.
  - Die Benutzer müssen sich der Transformationsprozesse nicht einmal bewusst sein.

Was ist wichtig für eine Prozess Engine

Sehr viel wichtiger als das native Metamodell einer Prozessengine ist ihre Stabilität, Effizient, Skalierbarkeit, etc ...

- Anbieter von aktuellen BPEL Engines haben typischerweise viel in die nicht-funktionalen Eigenschaften ihrer Engines investiert.
- So haben diese Anbieter auch die Möglichkeit, BPMN 2.0 Engines mit sehr ähnlichen nicht-funktionalen Eigenschaften anzubieten.

Frage

- „Oftmals wird von Entscheidern übersehen, dass die Wahl der Workflow-Engine und der darin enthaltenen Transformation zu einem irreversiblen Lock-in-Effekt führt“. Wie interpretierst du diese Aussage? Worin besteht die Gefahr, die durch die Transformation entsteht?

Eine DBMS Analogie

- In der Vergangenheit haben ambitionierte Anbieter dediziert objekt-orientierte Datenbanksysteme (OODBMS) implementiert.
  - Das Objektmodell wurde nativ von diesen DBMS unterstützt.
- Aber an Stabilität, Effizienz, Skalierbarkeit, … mangelte es solchen Systemen.
- Über die Zeit anerkannte RDBMS Anbieter unterstützten Schlüsselkonstrukte dieses Objektparadigmas während sie gleichzeitig Stabilität, Effizienz, Skalierbarkeit, … boten.
- OODBMS sind kein „Mainstream“ mehr.

Extrapolierung von Prozess Engines

- BPMN 2.0 ist signifikant komplexer als BPEL.
- Also werden Anbieter Teile von BPMN 2.0 auswählen, basierend auf den Anforderungen ihrer Kunden.

Es ist wahrscheinlich, dass kein Anbieter alle Aspekte der BPMN 2.0 in seinem Produkt vorweisen kann.

- BPEL Engines werden mit der Zeit erweitert, um Schlüsseleigenschaften von BPMN 2.0, die in BPEL und BPEL Engines fehlen, anzubieten.
- BPEL wird sich weiterentwickeln, um relevante BPMN 2.0 Eigenschaften (welche BPEL zur Zeit fehlen) aufzuweisen.
- Also sollte man warten, bis aktuelle Prozess Engines die fehlenden BPMN Eigenschaften aufweisen.

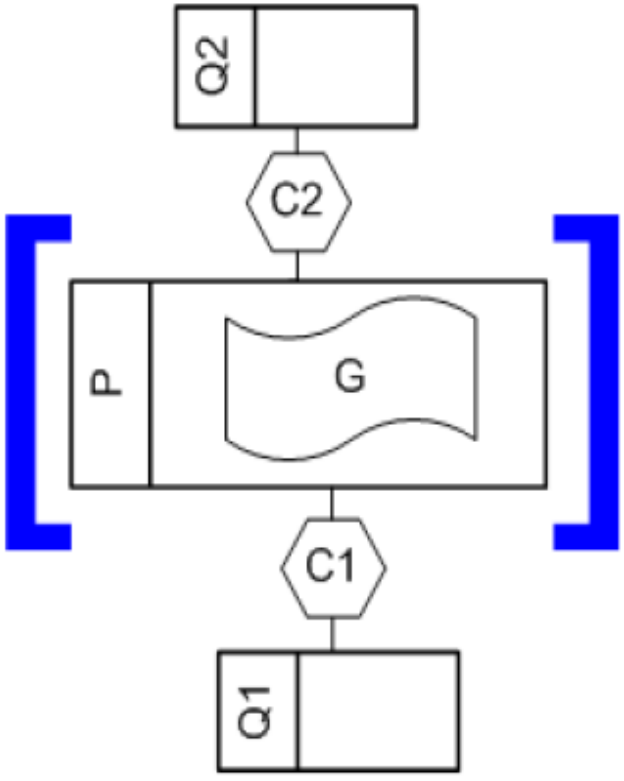
Wie die Ausführbarkeit ermöglicht wird

- Viele für die Ausführung relevante Informationen sind nur durch das BPMN 2.0 Schema verfügbar.
- D.h. die grafische Notation kann bei weitem nicht die Informationen, die zu einer Instantiierung und Ausführung eines BPMN Prozessmodells nötig sind, liefern.
- Die dazugehörigen syntaktischen Details werden in der BPMN 2.0 Spezifikation durch UML Klassendiagramme oder XML Schema Definitionen geliefert.
  - Aber die gegebene operationale Semantik stimmt <u>nicht</u> mit der Detailtiefe überein.

Grundprinzip

- Eine (rekursive) Funktion […] wird spezifiziert, die es erlaubt, eine Untermenge von BPMN auf BPEL abzubilden.
- Diese Abbildung wird spezifiziert durch folgende Beschreibungen:
  - [t] für alle elementaren BPMN Aufgaben t die auf BPEL abgebildet werden können
  - [e] für alle elementaren BPMN Events e die auf BPEL abgebildet werden können
  - [s] für alle BPMN Strukturen s, die eine direkte Abbildung auf BPEL haben
- Das definiert konstruktiv eine BPMN Untermenge, welche abgebildet werden kann, sowie die assoziierte Abbildung [...] selber.

Ein Prozess Abbilden



=

```
<process name="[P-name]"
   targetNamespace="[targetNamespace]"
   expressionLanguage="[expressionLanguage]"
   suppressJoinFailure="yes"
   xmlns="http://docs.oasis-open.org/wsbpel/2.0/process/executable">
 <partnerLinks>
   [ {P-Interfaces} UNION {Qi-Interfaces} ]
 </partnerLinks>
 <variables>
   [ {dataObjects} UNION {properties} ]
 </variables>
 <correlationSets>
   [ {Ci-CorrelationKeys} ]
 </correlationSets>
 [G]
</process>
```

Anm.: C1, C2 sind BPMN „Conversations“.
(Vgl. http://www.bpmn-introduction.com/BPMN20AllweyerExcerpt.pdf , Abs. 12.1).

Tasks einer Abbildung

[Service Task] =

```
<invoke name="[Task-name]"
  partnerLink="[Q, Task-operation-interface]"
  portType="[Task-operation-interface]"
  operation="[Task-operation]">
  <correlations>
    <correlation set="[Task-messageFlow-conversation-correlationKey]"
        initiate="[initialInConversation? 'join':'no']"/>
  </correlations>
</invoke>
```

[Receive Task] =

```
<receive name="[Task-name]"
  createInstance="[instantiate? 'yes':'no']"
  partnerLink="[Task-serviceRef]"
  portType="[Task-operation-interface]"
  operation="[Task-operation]">
</receive>
```

[Send Task] =

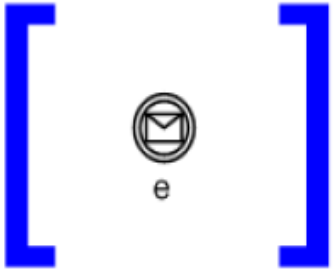
```
<invoke name="[Task-name]"
  partnerLink="[Task-serviceRef]"
  portType="[Task-operation-interface]"
  operation="[Task-operation]">
</invoke>
```

[None Task] =

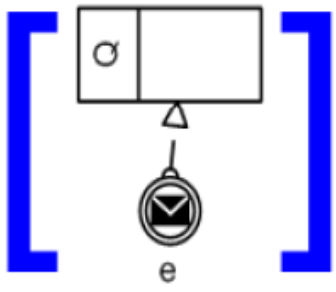
```
<empty name="[Task-name]">
</empty>
```

Events einer Abbildung (1/2)

```
<receive name="[e-name]"
  createInstance="yes"
  partnerLink="[e-operation-interface]"
  portType="[e-operation-interface]"
  operation="[e-operation]">
</receive>
```




```
<receive name="[e-name]"
  createInstance="no"
  partnerLink="[e-operation-interface]"
  portType="[e-operation-interface]"
  operation="[e-operation]">
</receive>
```




```
<invoke name="[e-name]"
  partnerLink="[Q, e-operation-interface]"
  portType="[e-operation-interface]"
  operation="[e-operation]">
</invoke>
```

```
<invoke name="[e-name]"
  partnerLink="[Q, e-operation-interface]"
  portType="[e-operation-interface]"
  operation="[e-operation]">
</invoke>
```

```
<throw faultName="[e-name]">
</throw>
```

```
<wait name="[e-name]" for="[e-TimeCycle]"/>
or
<wait name="[e-name]" until="[e-TimeDate]"/>
```

Events einer Abbildung (2/2)

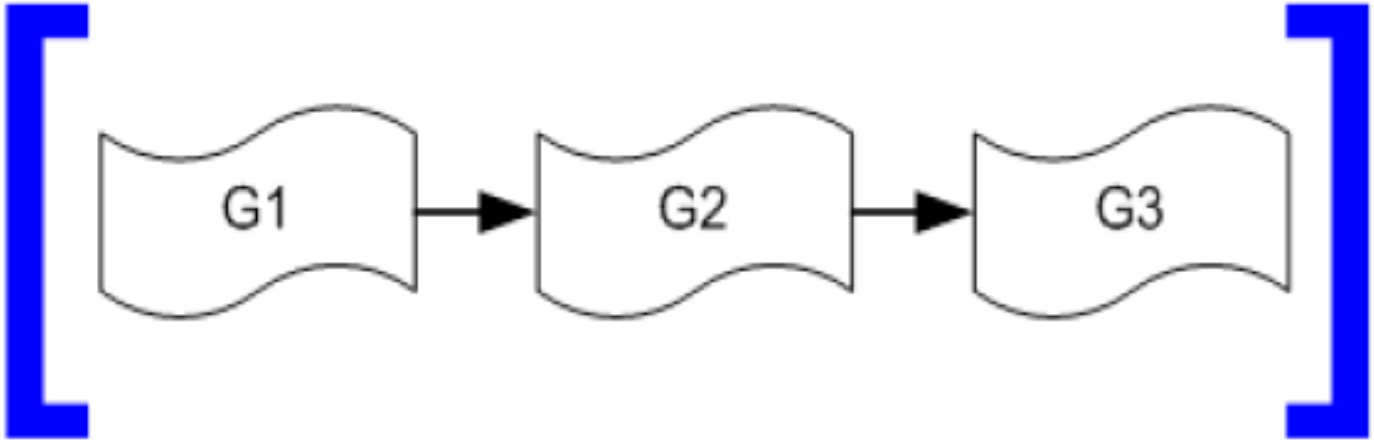
= <empty name="[e-name]">
</empty>

= <exit>
</exit>

= <compensate/>
or
<compensateScope target="[referencedActivity]"/>

= <compensate/>
or
<compensateScope target="[referencedActivity]"/>

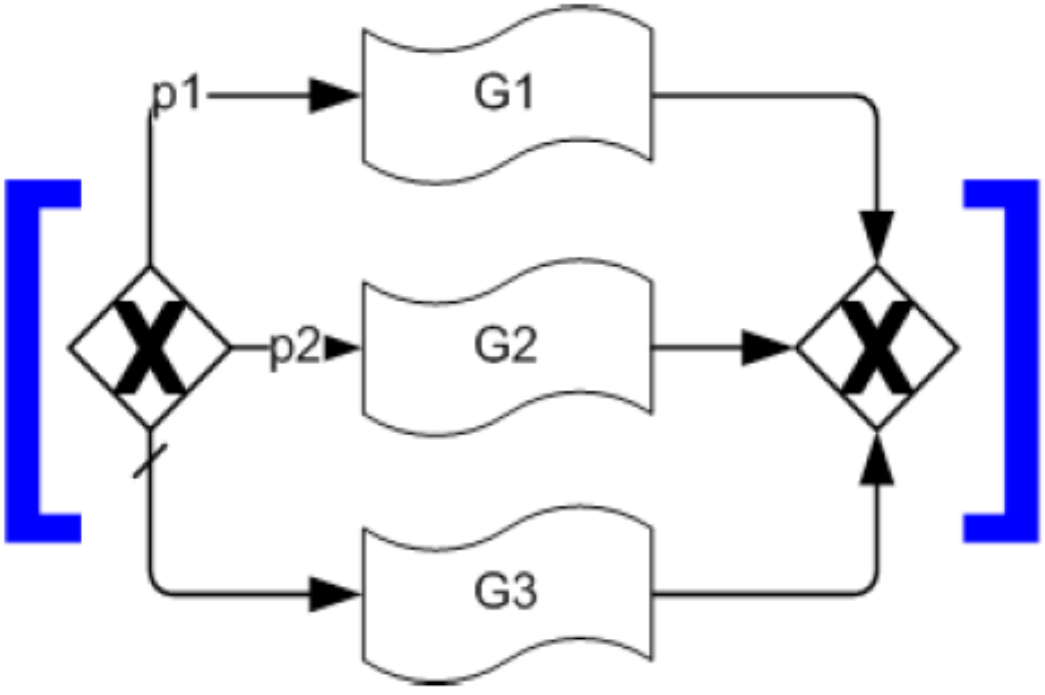
Abbildung - Sequenzen



=

<sequence>
[G1]
[G2]
[G3]
</sequence>

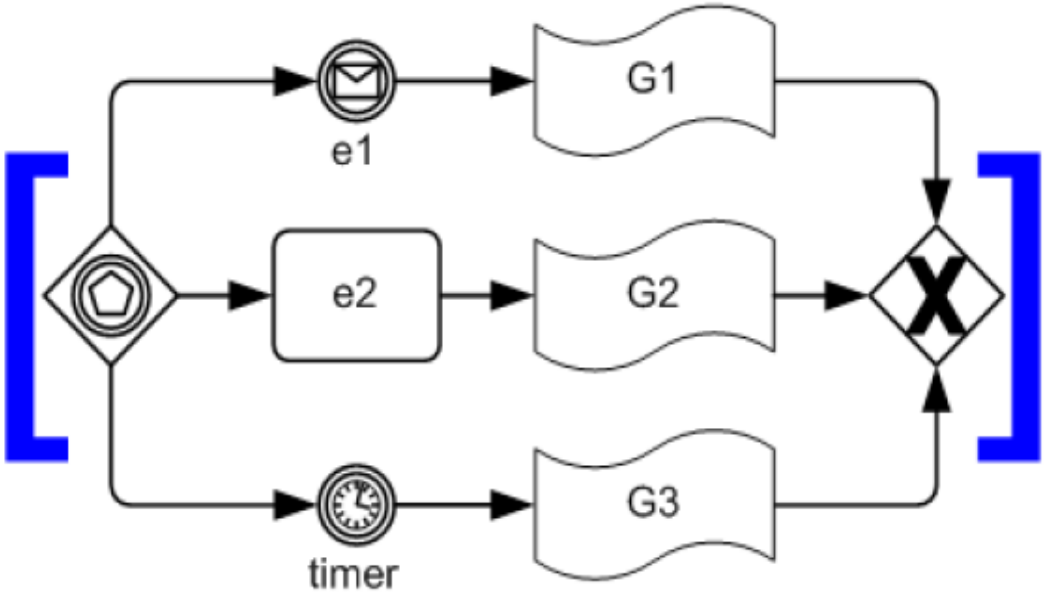
Abbildung - If-Then-Else



=

```
<if><condition>[p1]</condition>
  [G1]
<elseif><condition>[p2]</condition>
  [G2]
</elseif>
<else>
  [G3]
</else>
</if>
```

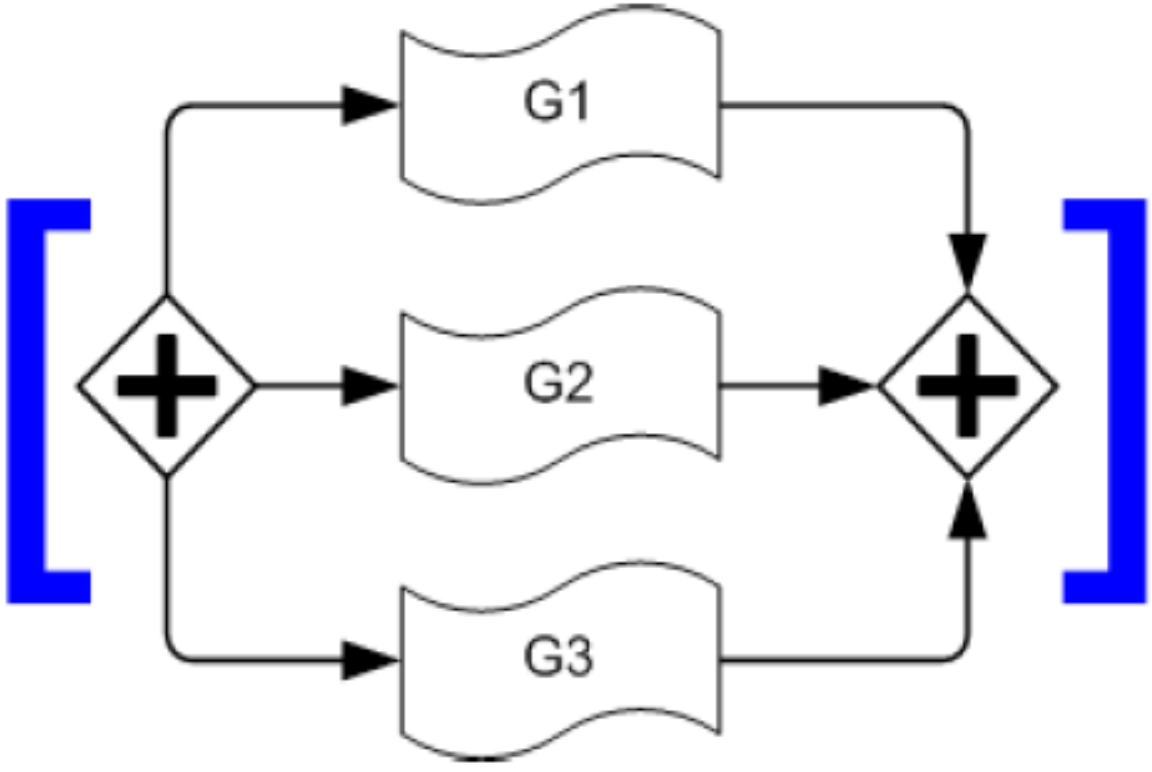
Abbildung - Wahl



=

```
<pick createInstance="[instantiate? 'yes':'no']">
  <onMessage partnerLink="[e1-operation-interface]"
      operation="[e1-operation]">
    [G1]
  </onMessage>
  <onMessage partnerLink="[e2-operation-interface]"
      operation="[e2-operation]">
    [G2]
  </onMessage>
  <onAlarm>
    [timer-spec]
    [G3]
  </onAlarm>
</pick>
```

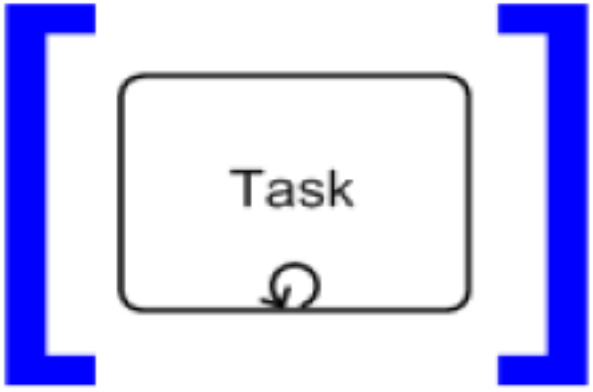
Abbildung - Parallel



=

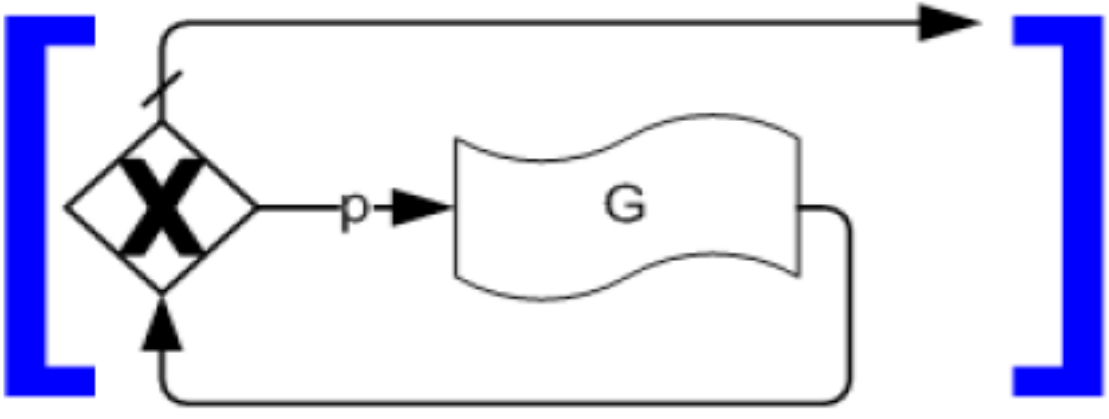
```
<flow>
  [G1]
  [G2]
  [G3]
</flow>
```

Abbildung - While-Schleifen



=

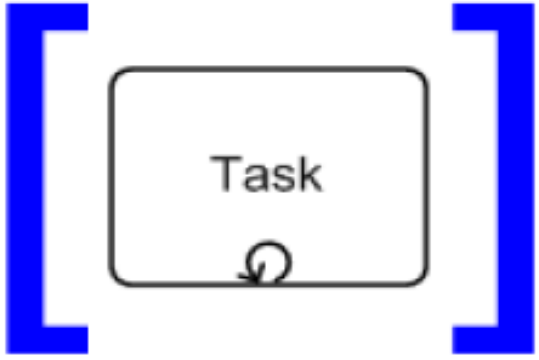
```
<while>
  <condition>[p]</condition>
  [Task]
</while>
```



=

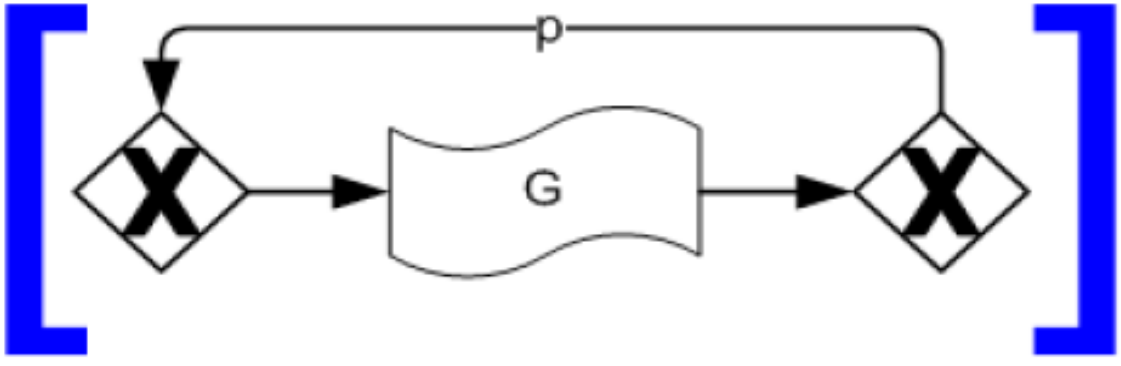
```
<while>
  <condition>[p]</condition>
  [G]
</while>
```

Abbildung - Until-Schleifen



=

```
<repeatUntil>
  [Task]
  <condition>[not p]</condition>
</repeatUntil>
```



=

```
<repeatUntil>
  [G]
  <condition>[not p]</condition>
</repeatUntil>
```

Zusammenfassung

- Lesson learned:
  - Lebenszyklus von Workflows
  - Zusammenhang Modellierung und Ausführung
  - Workflow-Aspekte
    - Begriffe und Architekturansätze
  - Integration von WfMS in Anwendungssysteme
  - Referenzmodell der WfMC
  - Überführung von BPMN nach BPEL
  - WfMS als SOA

Anhang

Abbildungsdaten

[D1] = `<variable name="[D1-name]" type="[D1-structureDefinition]"/>`

```
<inputSet name="iset">
  <dataInput name="input1">
    <structureDefinition structure="type1"/>
  </dataInput>
  ...
</inputSet>
```

=

```
<wsdl:message name="[iset-name]">
  <part name="[input1-name]" type="[type1]"/>
  ...
</wsdl:message>
```

```
<outputSet name="oset">
  <dataOutput name="output1">
    <structureDefinition structure="type3"/>
  </dataOutput>
  ...
</outputSet>
```

=

```
<wsdl:message name="[oset-name]">
  <part name="[output1-name]" type="[type3]"/>
  ...
</wsdl:message>
```

```
<property id="P1-id"
  structureRef="P1-structureDefinition"/>
```

=

```
<variable name="[{container-name}.P1-id]"
    type="[P1-structureDefinition]"/>
```

Nachrichten und Interfaces einer Abbildung


```
[ <Message name="msg-name">
    <StructureDefinition typeLanguage=
          "http://www.w3.org/2001/XMLSchema">
      xmlSchema
    </StructureDefinition>
  </Message> ]
```

=

```
<wsdl:message name="[msg-name]">
  [xmlSchema]
</wsdl:message>
```

```
[ <Interface name="if-name">
    <Operations>
      <Operation name="op1-name">
        <inMessageRef ref="msg1i-name"/>
        <outMessageRef ref="msg1o-name"/>
        <errorRef ref="error1a-name"/>
        ...
      </Operation>
      ...
    </Operations>
  </Interface> ]
```

=

```
<wsdl:portType name="[if-name]">
  <operation name="[op1-name]">
    <wsdl:input message="[msg1i-name]" />
    <wsdl:output message="[msg1o-name]" />
    <wsdl:fault name="[error1a-faultname]"
      message="[error1a-name]" />
    ...
  </operation>
  ...
</wsdl:portType>
```

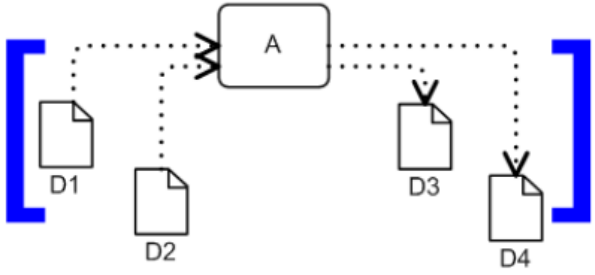
Abbildungskorrelationen

```
<KeyBasedCorrelationSet name="c-set">
  <Key name="k-name1" type="k-type1"
        messageRef="msg-name1">
    <MessageKeyExpression
        expressionLanguage="lang1">
      expr1
    </MessageKeyExpression>
  </Key>
  ...
  <Key name="k-nameN" />
    ...
</KeyBasedCorrelationSet>
```

[ ] =

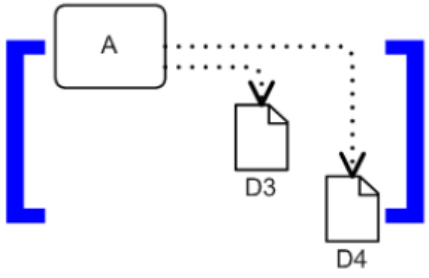
```
<vprop:property name="[k-name1]" type="[k-type1]"/>
...
<vprop:property name="[k-nameN]" />

<vprop:propertyAlias propertyName="[kName1]"
    messageType="[msg-name1]"
    part="[expr1-part]">
  <vprop:query queryLanguage="[lang1]">
    [expr1]
  </vprop:query>
</vprop:propertyAlias>
...
<vprop:propertyAlias propertyName="[kNameN]" />

<correlationSets>
  <correlationSet name="[c-set]"
      properties="[k-name1] ... [k-nameN]"/>
  ...
</correlationSets>
```

Task einer Abbildung Eingabe/Ausgabe



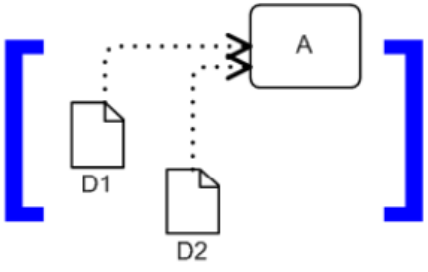
=

```
<invoke ...>
  <toParts>
    <toPart part="[dataInput1-name]"
      fromVariable="[D1-name]"/>
    <toPart part="[dataInput2-name]"
      fromVariable="[D2-name]"/>
  </toParts>
  <fromParts>
    <fromPart part="[dataOutput1-name]"
      toVariable="[D3-name]"/>
    <fromPart part="[dataOutput2-name]"
      toVariable="[D4-name]"/>
  </fromParts>
</invoke>
```



=

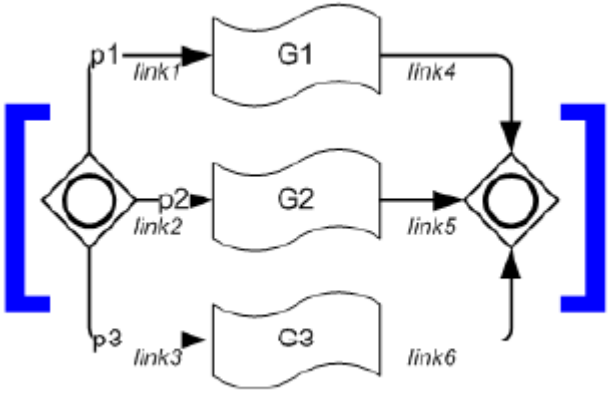
```
<receive>
  <fromParts>
    <fromPart part="[dataOutput1-name]"
      toVariable="[D3-name]"/>
    <fromPart part="[dataOutput2-name]"
      toVariable="[D4-name]"/>
  </fromParts>
</receive>
```



=

```
<invoke>
  <toParts>
    <toPart part="[dataInput1-name]"
      fromVariable="[D1-name]"/>
    <toPart part="[dataInput2-name]"
      fromVariable="[D2-name]"/>
  </toParts>
</invoke>
```

Abbildung - Fluss



=

```
<flow>
  <links>
    <link name="[link1]"/>
    ...
    <link name="[link6]"/>
  </links>

  <empty>
    <sources>
      <source linkName="[link1]">
        <transitionCondition>[p1])</transitionCondition>
      </source>
      <source linkName="[link2]">
        <transitionCondition>[p2])</transitionCondition>
      </source>
      <source linkName="[link3]">
        <transitionCondition>[p3])</transitionCondition>
      </source>
    </sources>
  </empty>
```

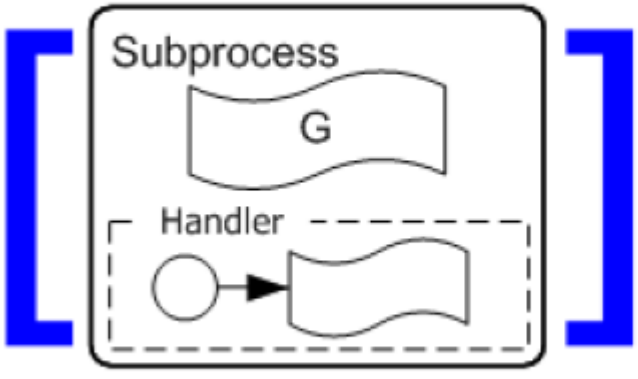
```
<flow>
  <targets><target linkName="[link1]"/></targets>
  <sources><source linkName="[link4]"/></sources>
  [G1]
</flow>

<flow>
  <targets><target linkName="[link2]"/></targets>
  <sources><source linkName="[link5]"/></sources>
  [G2]
</flow>

<flow>
  <targets><target linkName="[link3]"/></targets>
  <sources><source linkName="[link6]"/></sources>
  [G3]
</flow>

<empty>
  <targets>
    <target linkName="[link4]"/>
    <target linkName="[link5]"/>
    <target linkName="[link6]"/>
  </targets>
</empty>
</flow>
```

Abbildung - Multi-Instanz-Schleifen-Charakteristika

=

```
<variable name="[counter]" type="xsd:integer"/>
...
<forEach counterName="[counter]" parallel="[isSequential? 'no':'yes']">
  <startCounterValue>1</startCounterValue>
  <finalCounterValue>[condition]</finalCounterValue>
  <scope>
    [Task]
  </scope>
</forEach>
```

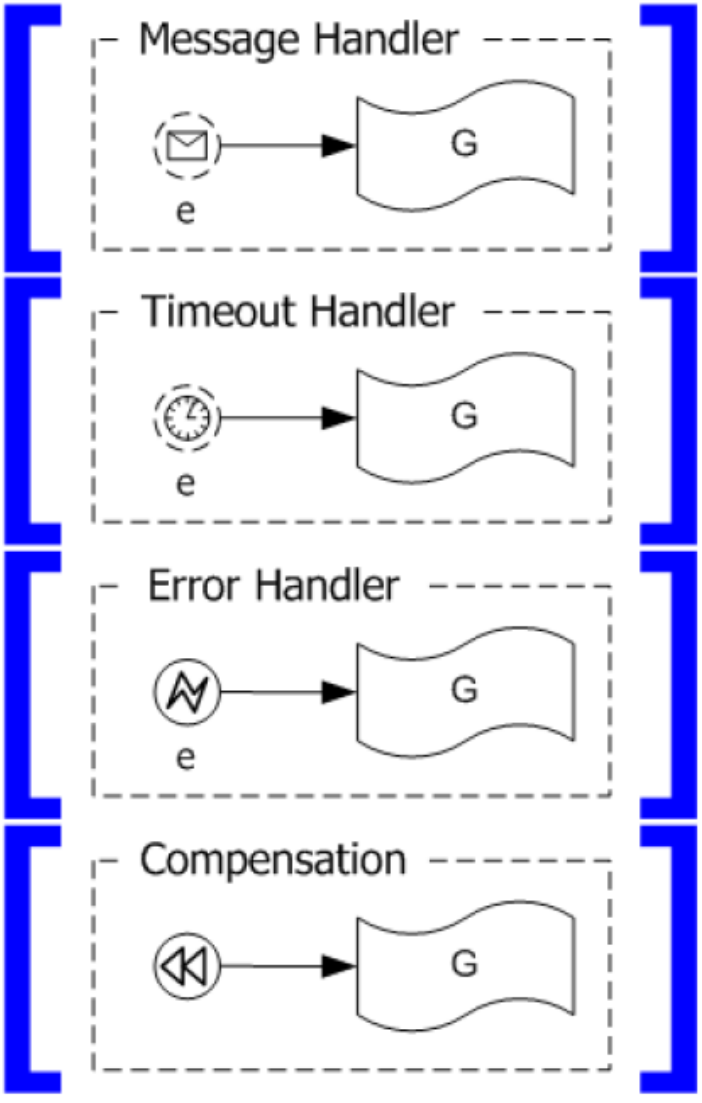
Abbildung - Subprozesse



=

```
<scope name="[Subprocess-name]">
  <partnerLinks>
    [ {serviceRefs} ]
  </partnerLinks>
  <variables>
    [ {dataObjects} UNION {properties} ]
  </variables>
  <correlationSets>
    [ {correlations} ]
  </correlationSets>
  [Handler]
  [G]
</scope>
```

Abbildung - Steuerung




```
<eventHandlers>
  <onEvent partnerLink="[e-operation-interface]"
      operation="[e-operation]">
    <scope>[G]</scope>
  </onEvent>
</eventHandlers>
```

=

```
<eventHandlers>
  <onAlarm>[timer-spec]
    <scope>[G]</scope>
  </onAlarm>
</eventHandlers>
```

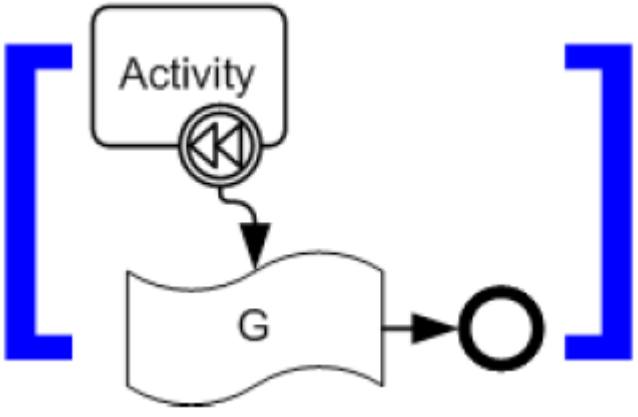
=

```
<faultHandlers>
  <catch faultName="[e-error]">
    [G]
  </catch>
</faultHandlers>
```

=

```
<compensationHandler>
  [G]
</compensationHandler>
```

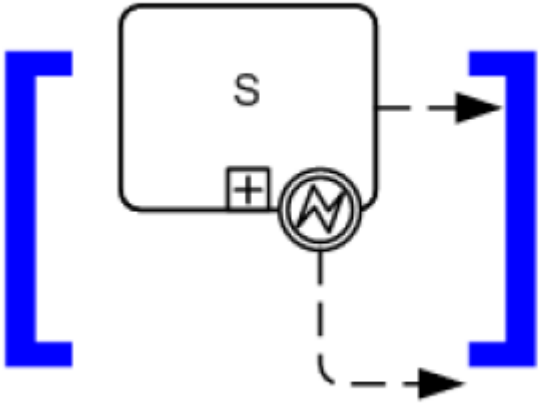
Abbildung – Rand-Events (1/4)



=

```
<scope name="[Activity-name]">
 <compensationHandler>
   [G]
 </compensationHandler>
 [Activity]
</scope>
```

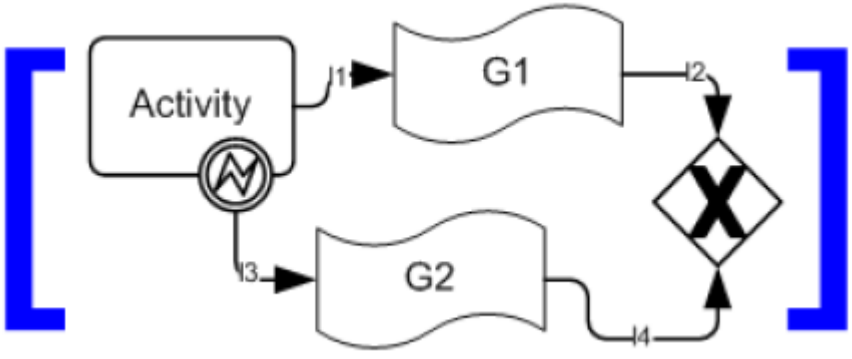
Abbildung – Rand-Events (2/4)



=

```
<scope>
  <faultHandlers>
    <faultHandler>
      <catch faultName="...">
        <empty>
          <sources><source linkName="faultLink"/></sources>
        </empty>
      </catch>
    </faultHandler>
  </faultHandlers>
  [S]
</scope>
```

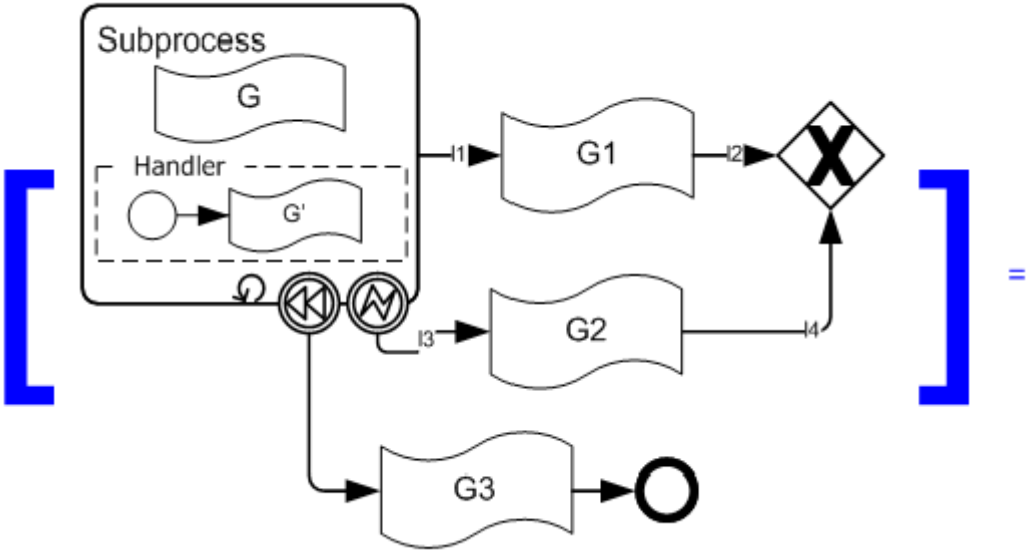
Abbildung – Rand-Events (3/4)



=

```
<flow>
 <links>
   <link name="[l1]"/>
   ...
   <link name="[l4]"/>
 </links>
 <scope>
   <sources><source linkName="[l1]"/></sources>
   <faultHandlers>
     <catch faultName="[e-error]">
       <empty>
         <sources><source linkName="[l3]"/></sources>
       </empty>
     </catch>
   </faultHandlers>
   [Activity]
 </scope>
 <flow>
   <targets><target linkName="[l1]"/></targets>
   <sources><source linkName="[l2]"/></sources>
   [G1]
 </flow>
 <flow>
   <targets><target linkName="[l3]"/></targets>
   <sources><source linkName="[l4]"/></sources>
   [G2]
 </flow>
 <empty>
   <sources><source linkName="[l2]"/>
   <source linkName="[l4]"/></sources>
 </empty>
</flow>
```

Abbildung – Rand-Events (4/4)




```
<flow>
  <links>
    <link name="[l1]"/>
    ...
    <link name="[l4]"/>
  </links>
  <scope>
    <sources><source linkName="[l1]"/></sources>
    <faultHandlers>
      <catch faultName="[e-error]">
        <empty>
          <sources><source linkName="[l3]"/></sources>
        </empty>
      </catch>
    </faultHandlers>
    <compensationHandler>
      [G3]
    </compensationHandler>
    <while>
      <condition>[p]</condition>
      <scope>
        [Handler]
        [G]
      </scope>
    </while>
  </scope>
  <flow>
    <targets><target linkName="[l1]"/></targets>
    <sources><source linkName="[l2]"/></sources>
    [G1]
  </flow>
  <flow>
    <targets><target linkName="[l3]"/></targets>
    <sources><source linkName="[l4]"/></sources>
    [G2]
  </flow>
  <empty>
    <sources><source linkName="[l2]"/>
    <source linkName="[l4]"/></sources>
  </empty>
</flow>
```