SOA oder so

ESB für Querdenker

Produktlinien-Design mit ESB

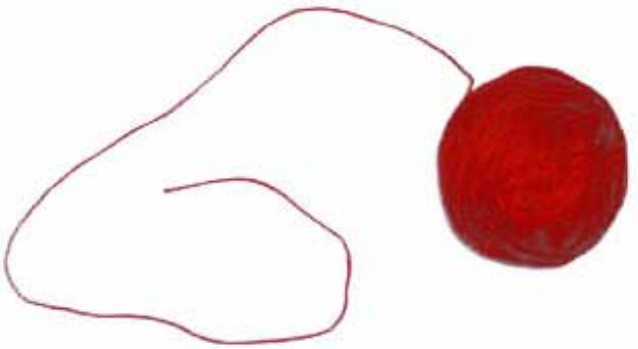
Peter Klotz

Chief Architekt
blue elephant systems GmbH

(peter.klotz@blue-elephant-systems.com)

Agenda

- SOA & ESB
- Apache Servicemix & JBI
- ESB-Architektur & Mechanismen
- Design-Beispiele
- Erfahrungen im Produkt-Einsatz
- Fragen

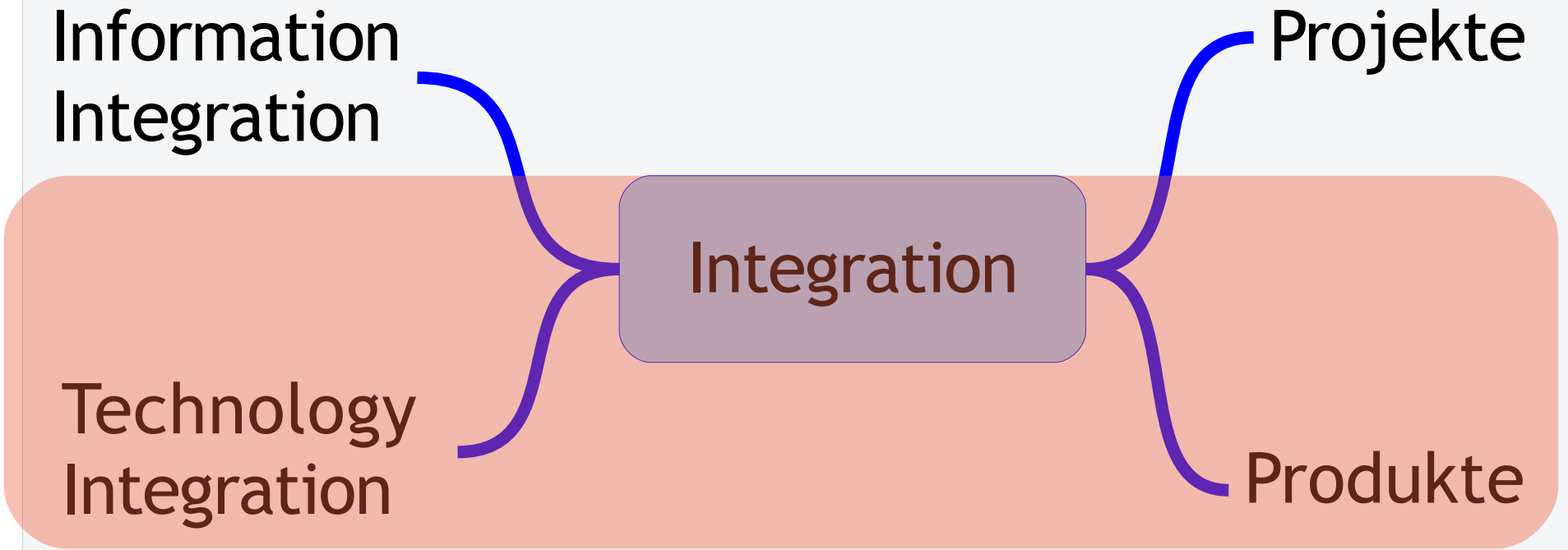


blue elephant systems

SOA

- Aufbau von Applikationen aus Services
- Interaktion über Protokolle statt Methoden (XML, SOAP, MOM)
- Schnittstellen Metadata (WSDL)
- Service Orchestrierung
- Loose Coupling
- SOA ist ein Architektur-Prinzip
- ESB als SOA Infrastruktur/Implementierung

Integration



Main und andere Ströme

Enterprise Service Bus

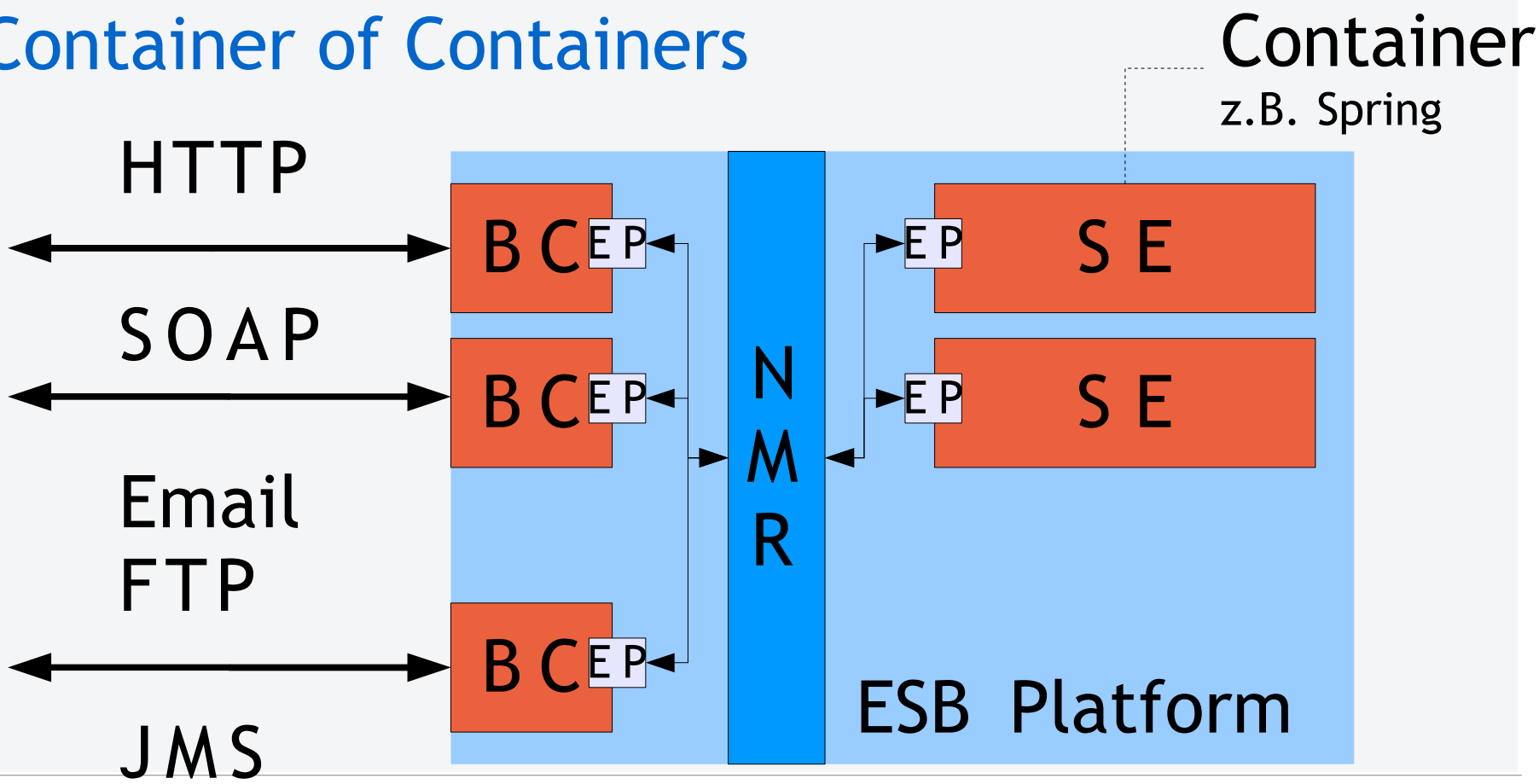
- **Invocation:** Synchrone and asynchrone Transport Protokolle
  Service Mapping (locating and binding)
- **Routing:** Addressierung, statisches/deterministisches, Content-based,
  Rules-based, Policy-based Routing
- **Mediation:** Adaptoren, Protokoll Transformation, Service Mapping
- **Messaging:** Message Processing, Message Transformation und Enhancement
- **Process Choreography:** Komplexe Business Prozesse
- **Service Orchestration:** Koordination mehrerer Implementation-Services
  als ein einziger aggregierter Service
- **Complex Event Processing:** Event Interpretation, Korrelation, Pattern-Matching
- **Quality of Service:** Security (Encryption und Signing), Zuverlässiger Transport,
  Transaction Management
- **Management:** Monitoring, Auditing, Logging, Metering, BAM

ESB Implementierungen

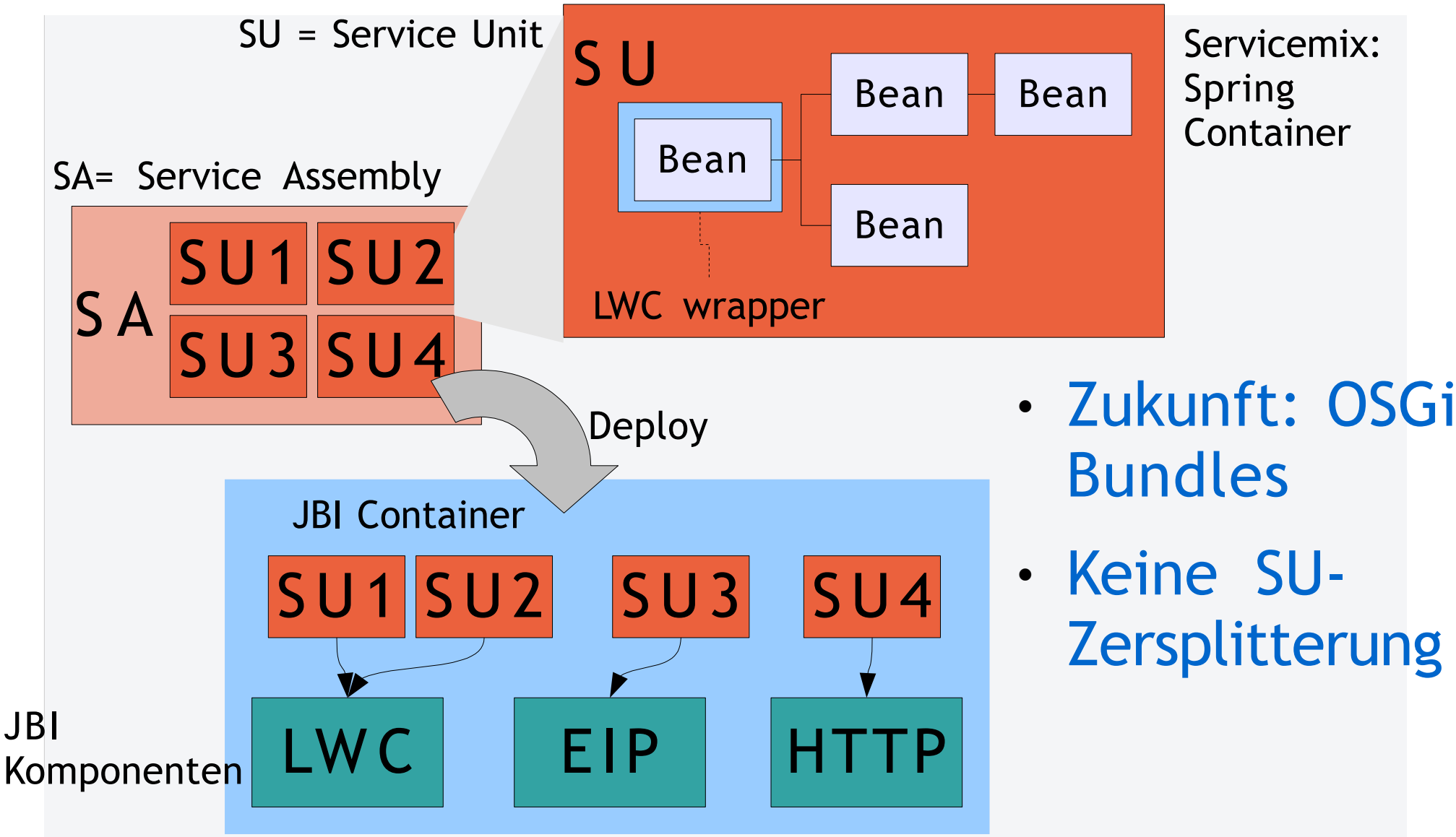
- Open-Source
  - Servicemix (www.servicemix.org)
  - OpenESB (open-esb.dev.java.net)
  - Mule (mule.mulesource.org)
  - JBoss ESB
  - …
- Kommerziell
  - BEA AquaLogic
  - Oracle ESB
  - …

Java Business Integration
JBI (JSR-208)

- Normalized Message Router (NMR)
- Binding Component und Service Engine
- Container of Containers



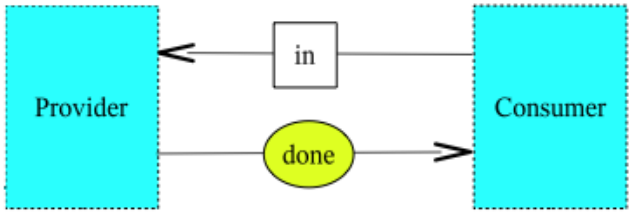
JBI Deployment



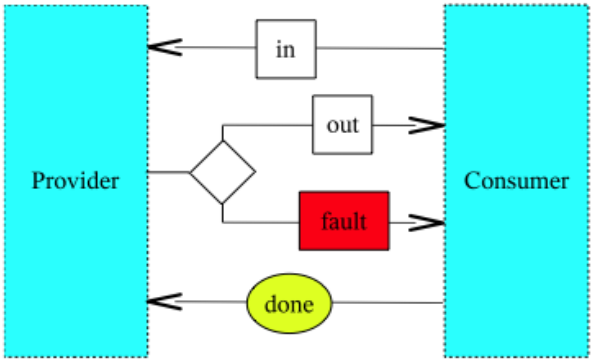
- Zukunft: OSGi Bundles
- Keine SU-Zersplitterung

Message Exchange Patterns

- JBI Message Exchange Patterns (MEP)
  - In-Out
  - In-Only
  - Robust-In-Only
  - In-Optional-Out

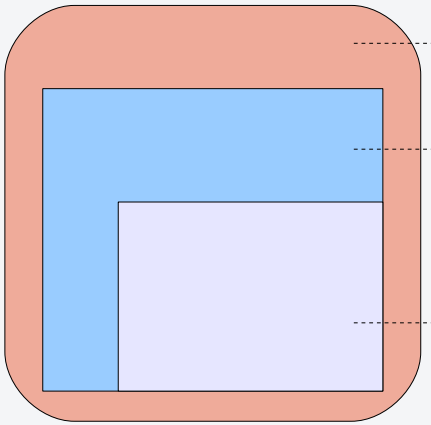


- Binding Components definieren MEP
- Fehler-anfällig
  - insb. In-Out async
  - done() vergessen

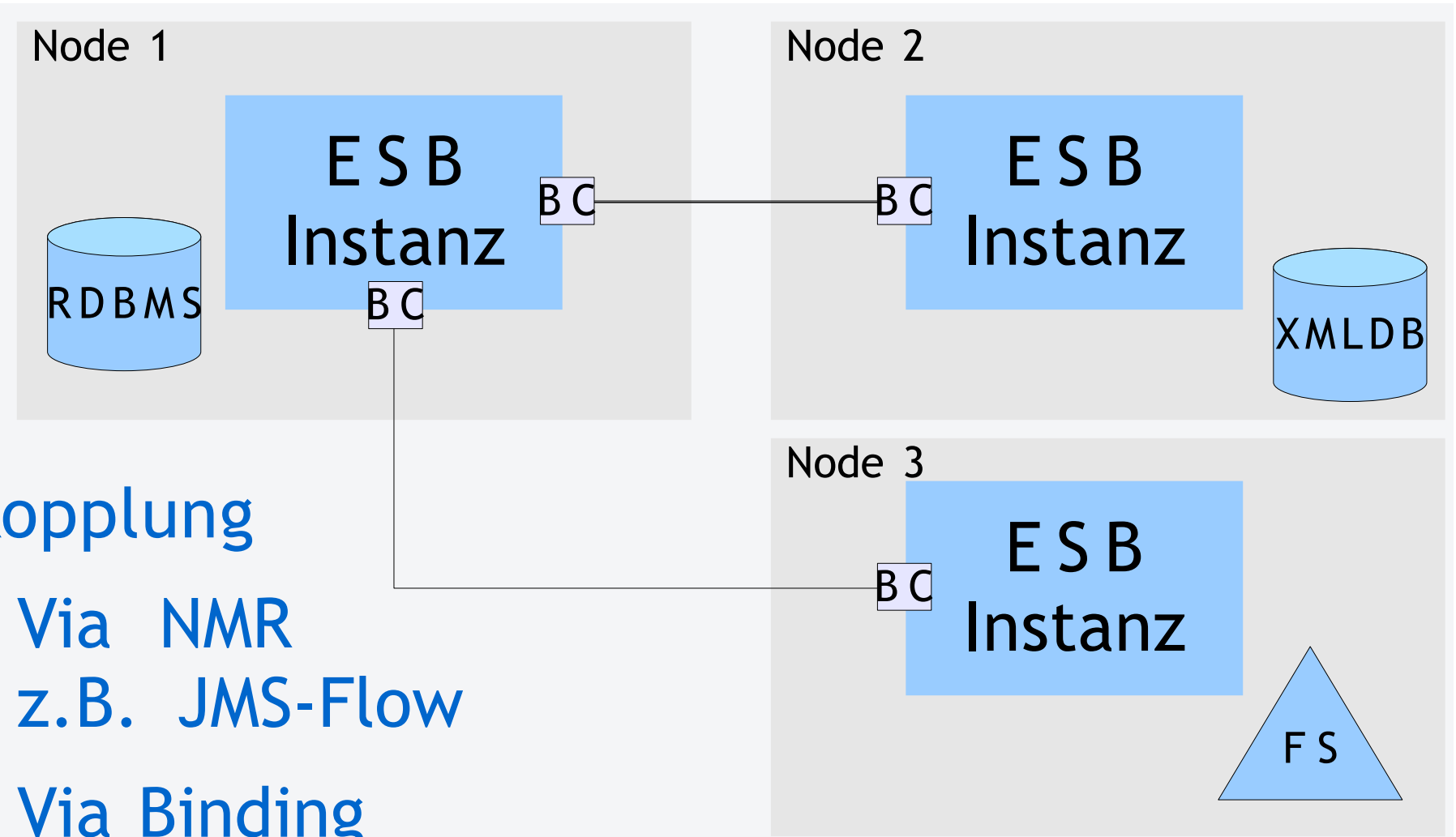


Messaging

- Payload: XML
- Attachments (MIME-Messages)
- Standardisierte Formate (z.B. WS-Notification)
- Addressierung
  - Protokoll-Header
    z.B. SOAP-Action,
    Properties, WS-A
  - Request/Response
    Envelope



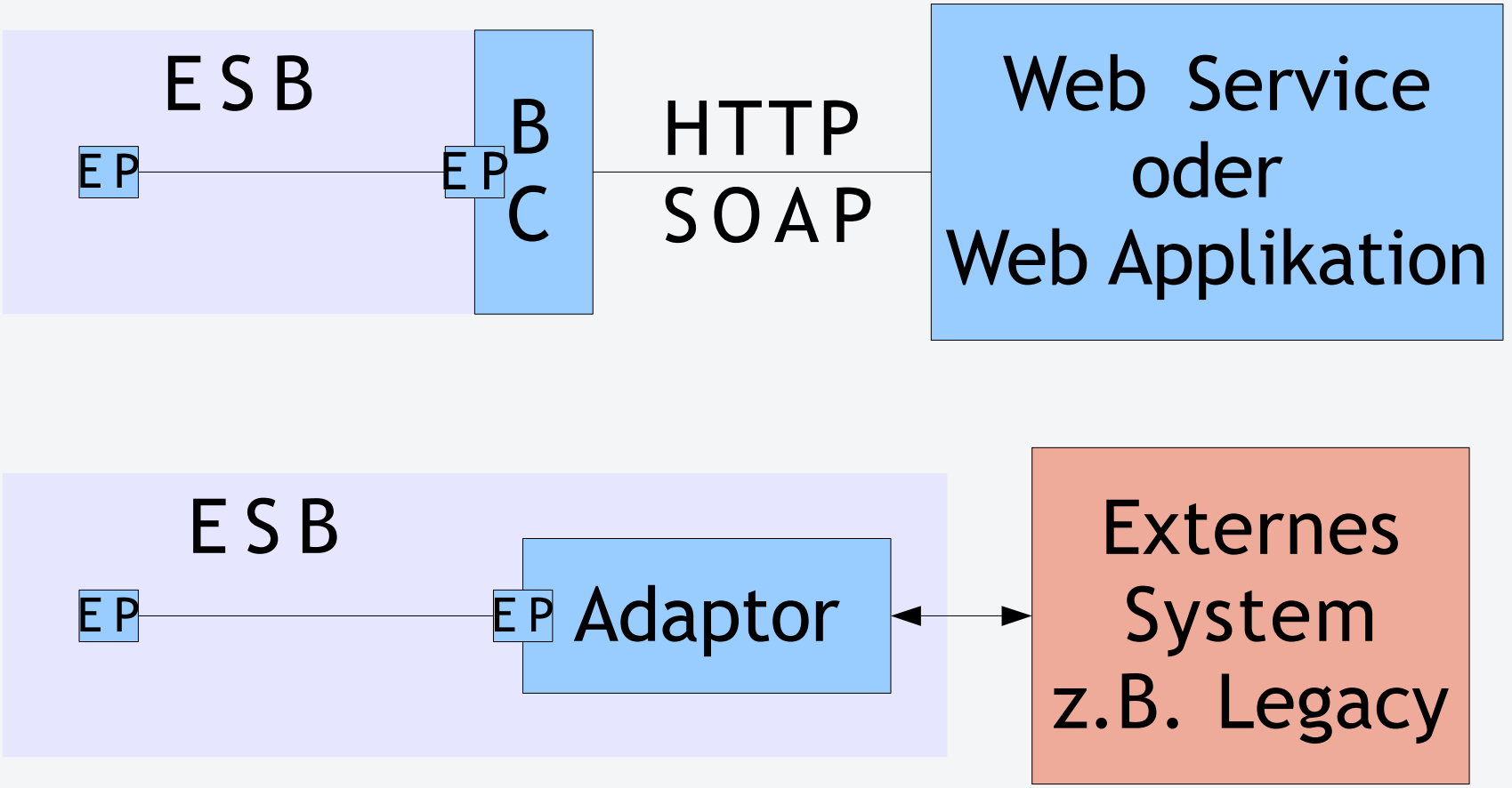
ESB-Verteilung



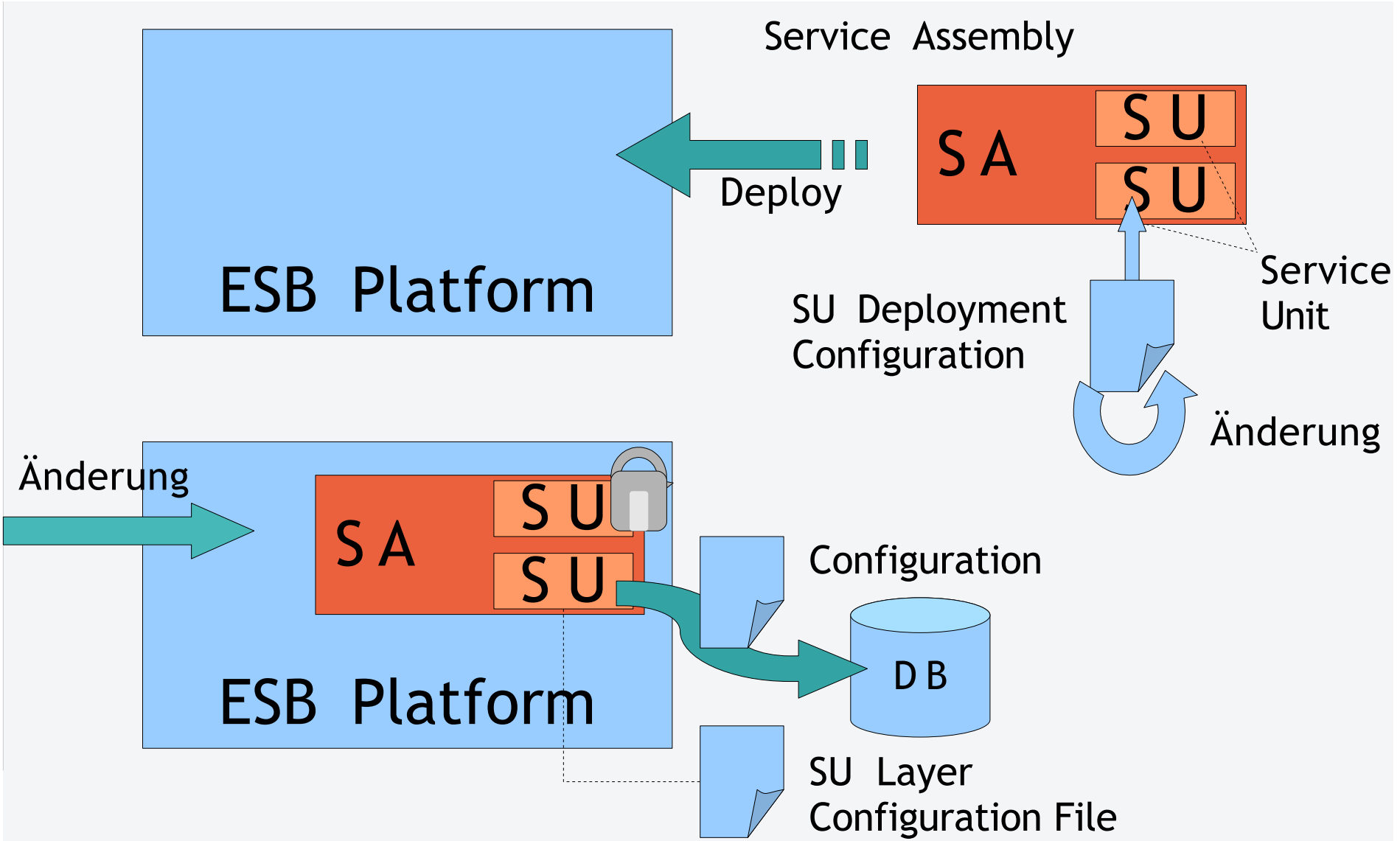
Kopplung

- Via NMR
  z.B. JMS-Flow
- Via Binding Components & Routing

Extern vs. Intern

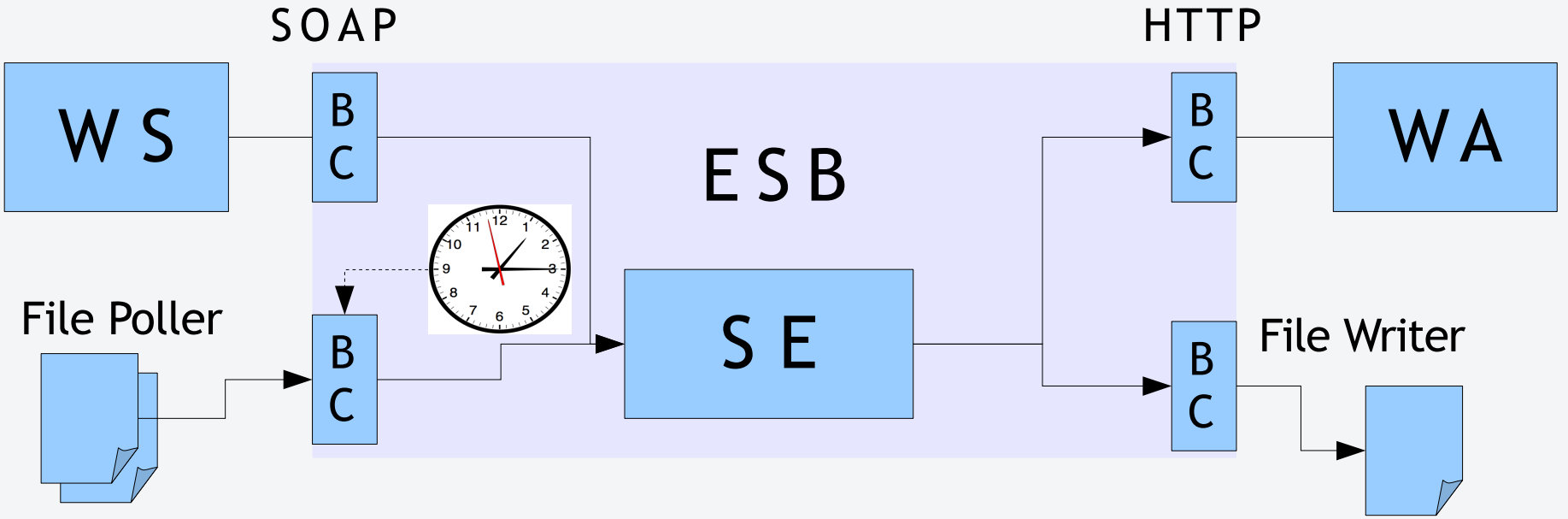


Deployment vs. CIs



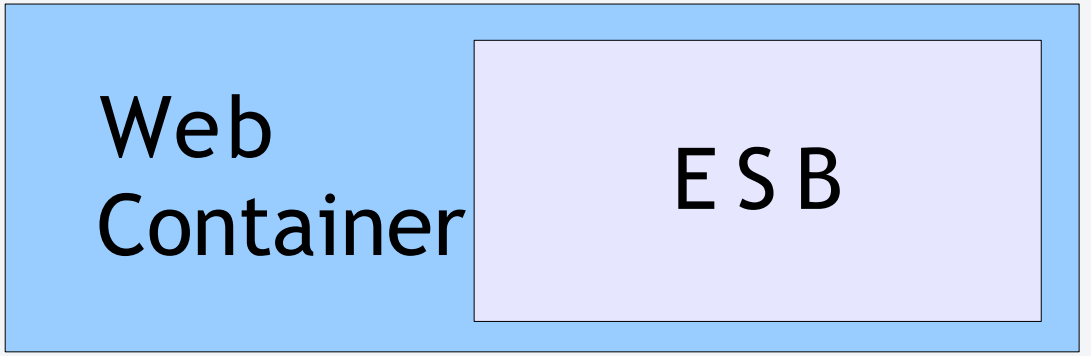
Realtime & Polling

- Realtime MOM
- Lokale Files oder FTP Polling und Writer

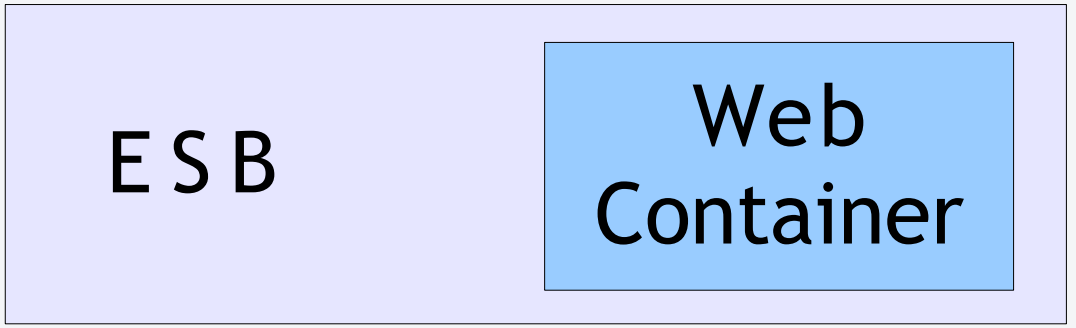


ESB & Web Container

Klassisch:
ESB in Web Container
z.B. JBoss

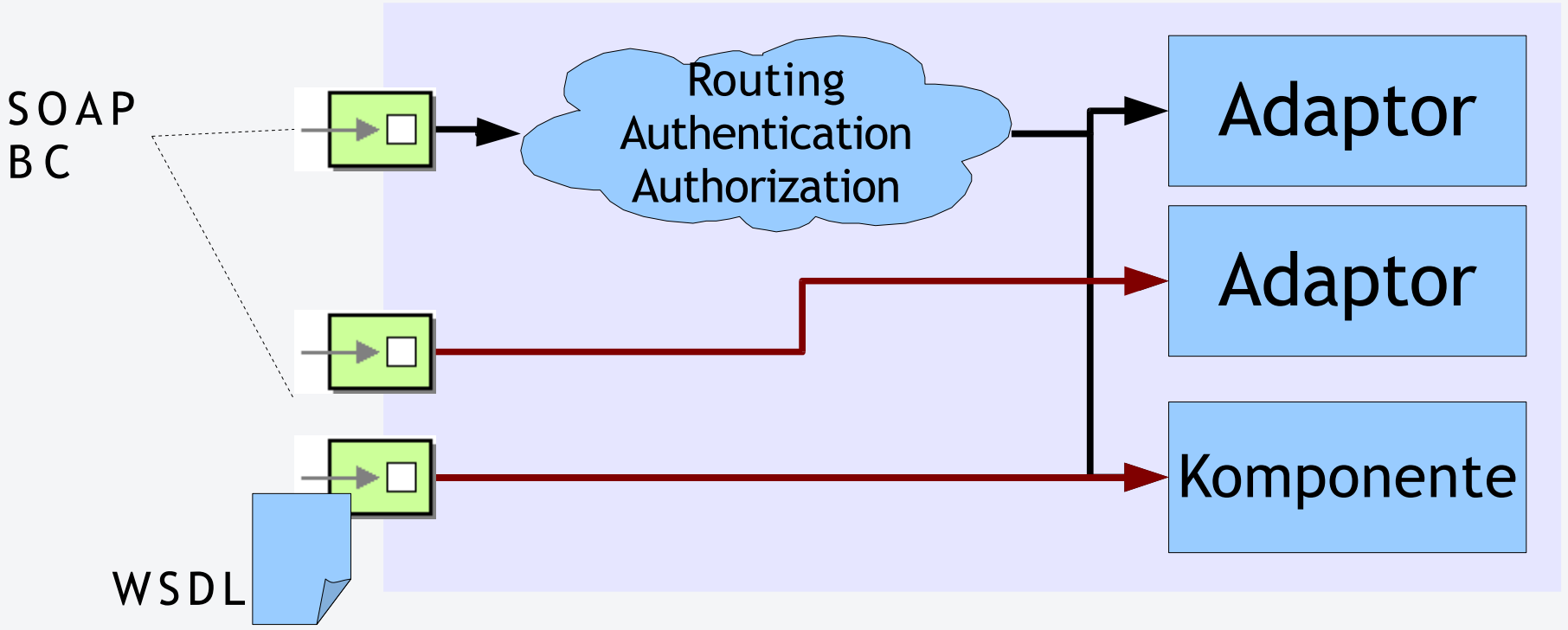


Anders:
Web Container in ESB
z.B. Jetty in SM



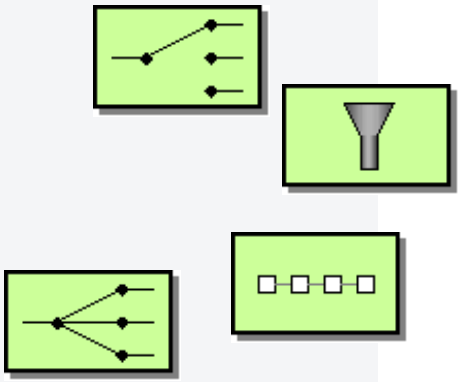
Web Services & ESB

- SOAP Binding Component
- WSDL Generierung
- ESB als WS vs. einzelne Komponenten als WS



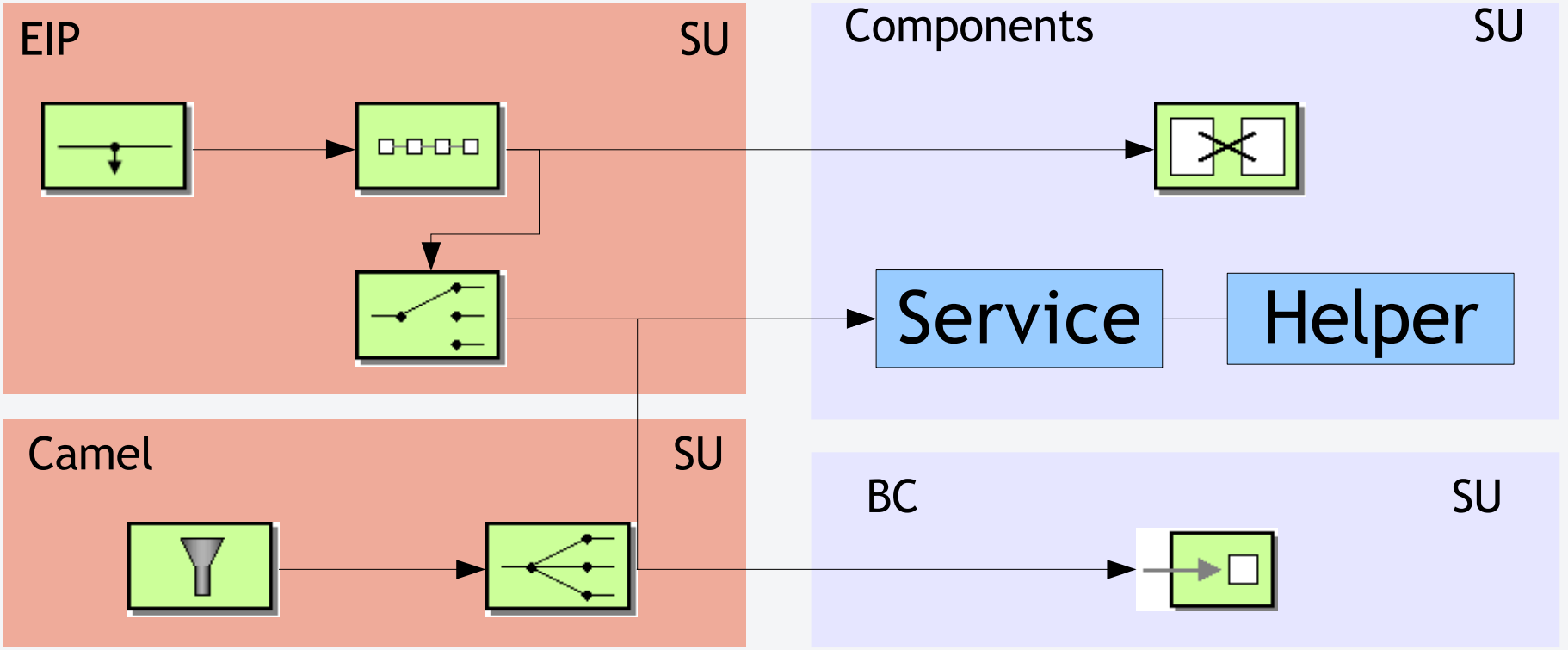
Orchestrierung/Routing

- *Orchestration describes the automated arrangement, coordination, and management of complex computer systems, middleware, and services.*
- Statisch
- Java-Code
- Scripting (Groovy)
- Enterprise Integration Patterns (EIP)
  (Servicemix EIP, Apache Camel)
- Rules (Drools)
- Business Prozesse (BPM)



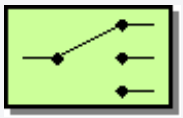
Routing

- Statisch: Routing an Komponenten
- EIP: Nur Routing
- BPEL: Routing + Daten Transformation

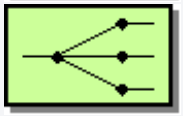


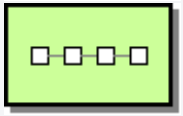
Enterprise Integration Patterns

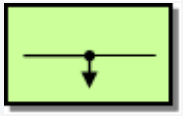
- http://www.enterpriseintegrationpatterns.com/
- Achtung: einige nur In-Out oder nur In-Only
- Vorsicht beim Einsatz mancher Patterns z.B. Content Enricher

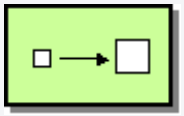

Content-based Router

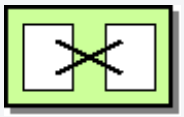

Message Filter

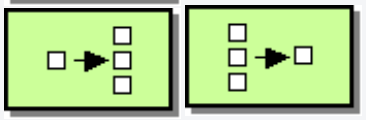

Recipient List


Routing Slip

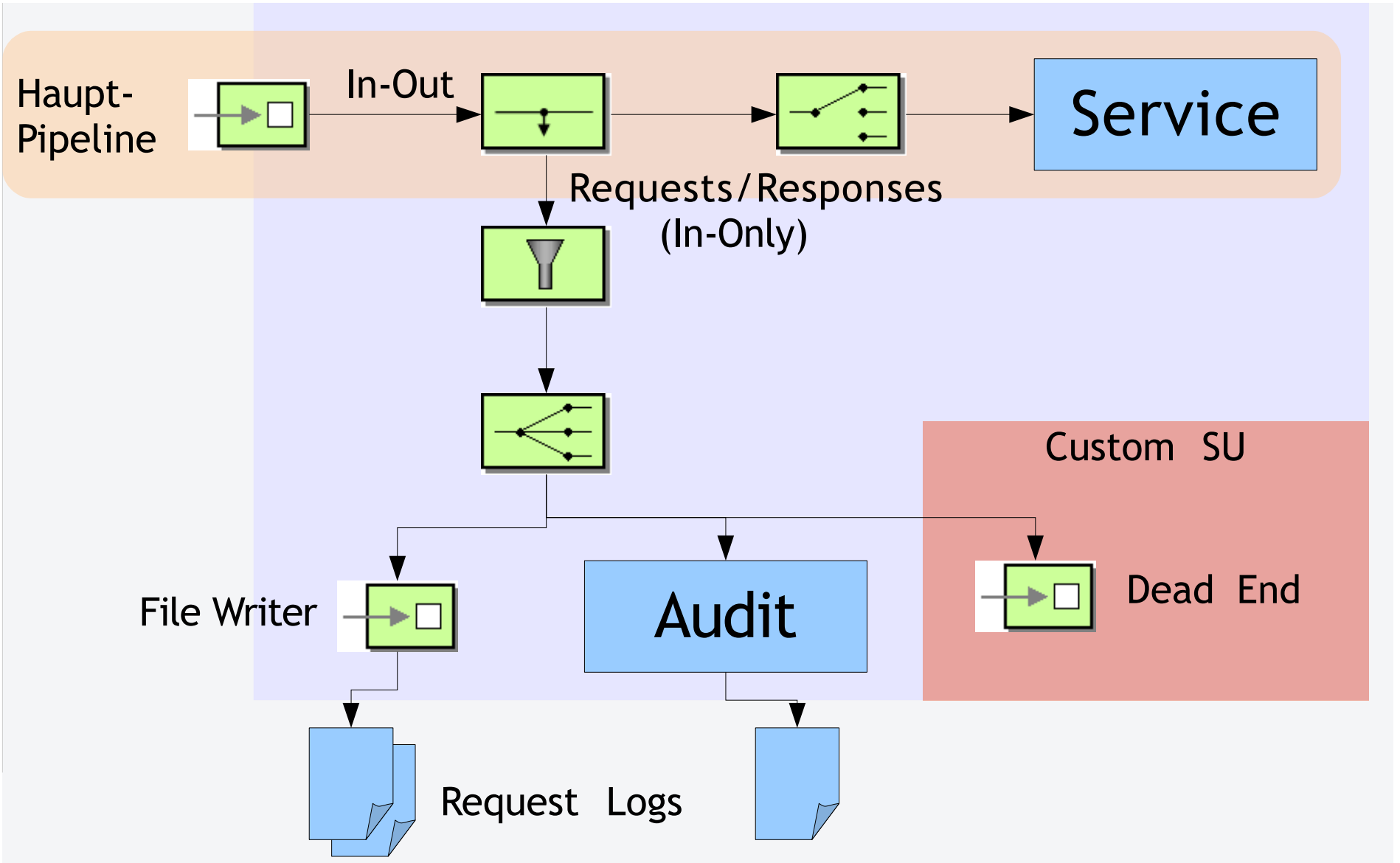

Wire Tap


Content Enricher


Transformer

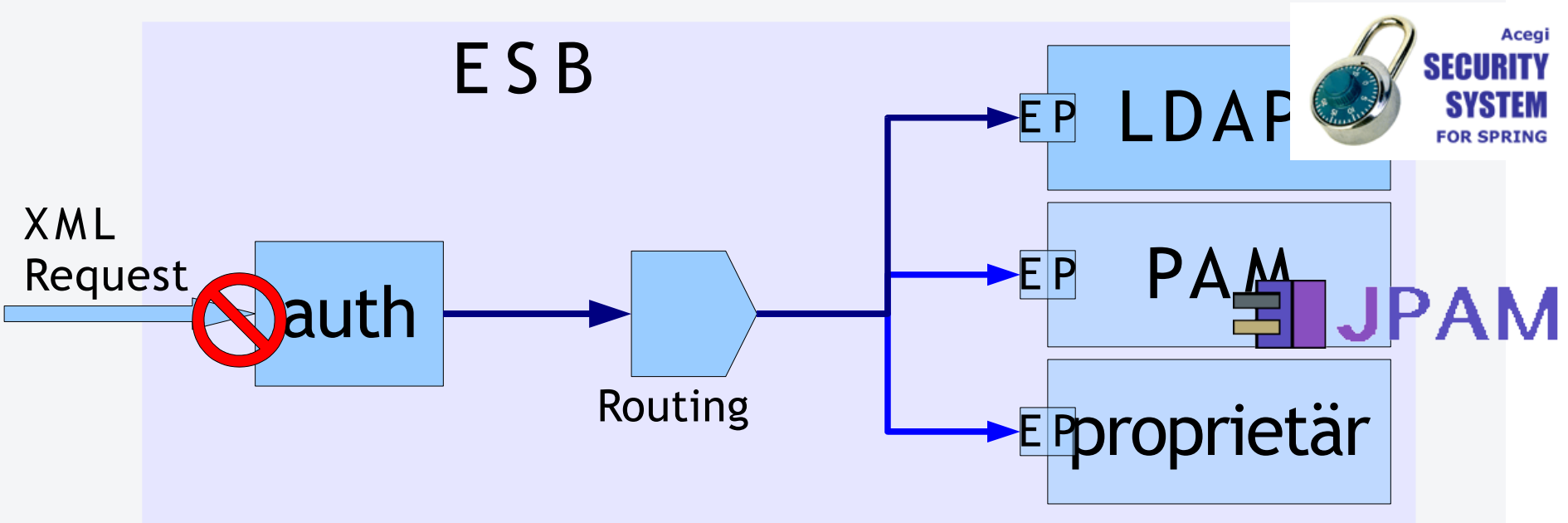

Splitter/Aggregator

Beispiel: Wire Tap



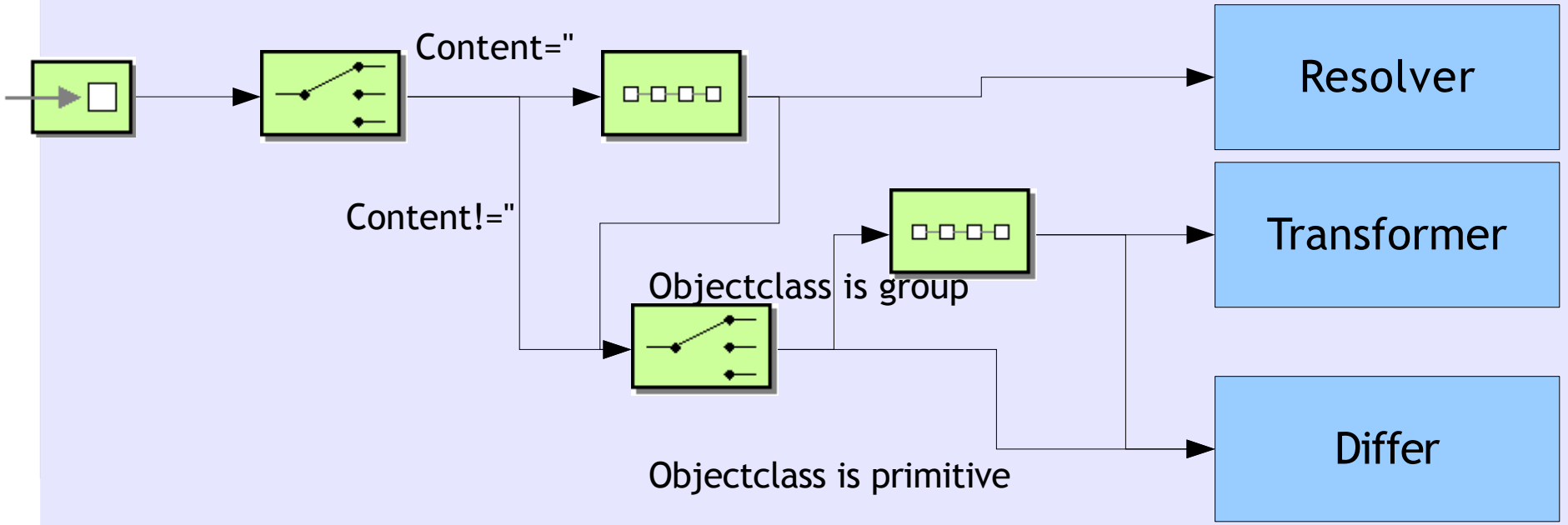
Authentifizierung

- Client-Authentifizierung nicht NUR für GUIs!
- Nutzung der ESB-Mechanismen
- Single Sign-On Frameworks versagen



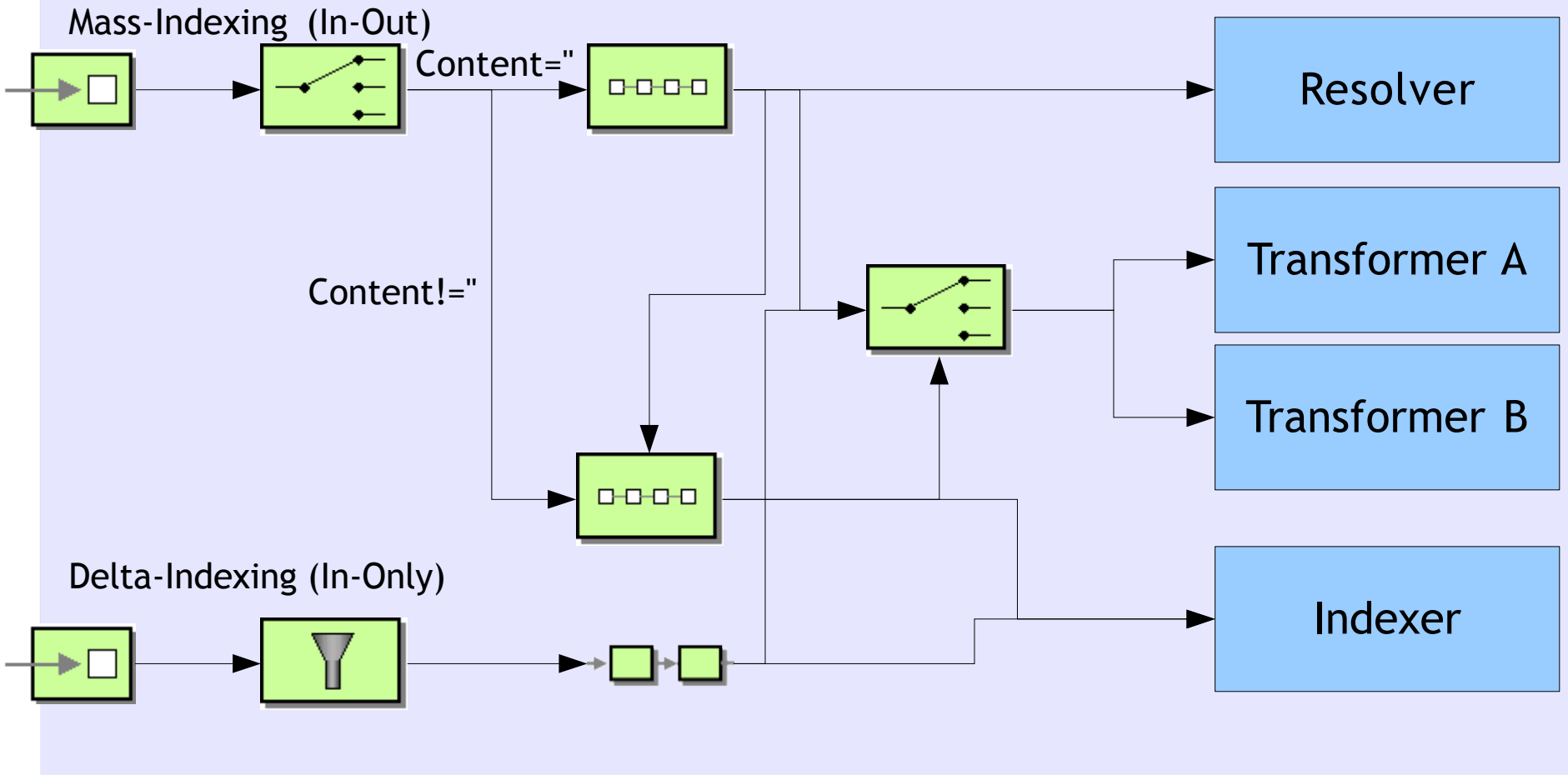
Programming in the Large

- Programm = Adaptors + Routing
- UNIX-Prinzip: 1 Tool pro Zweck
- Beispiel: Compare/Diff mit EIP



Beispiel II

- Beispiel: Indexing mit EIP



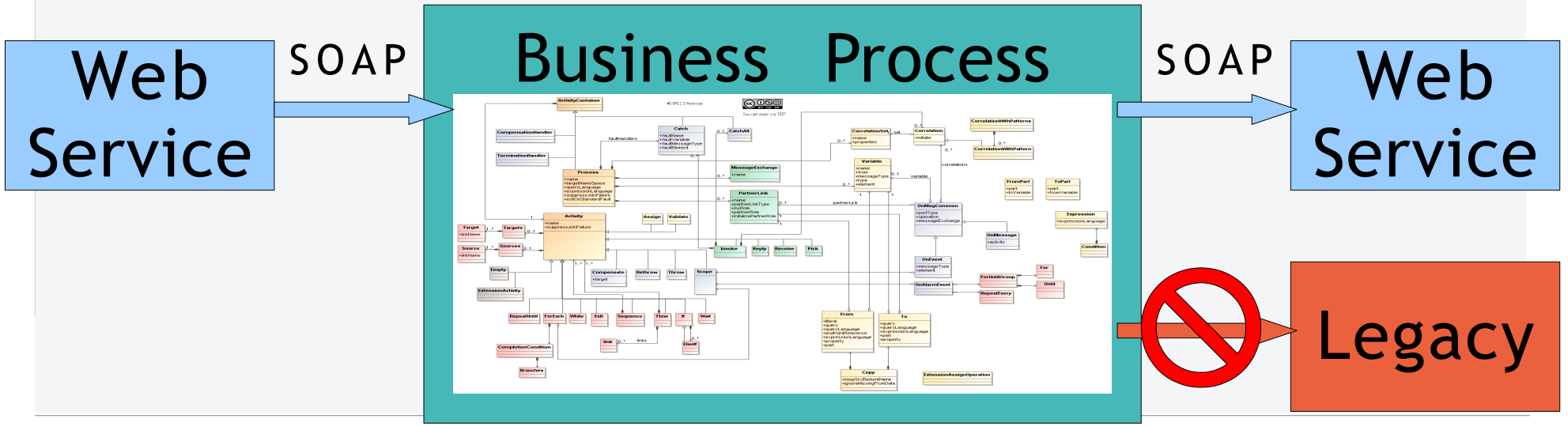
Scripting

- Script-Komponenten
  - Compilierung bei Deployment
  - Spring Script Support
    z.B. auto script reload

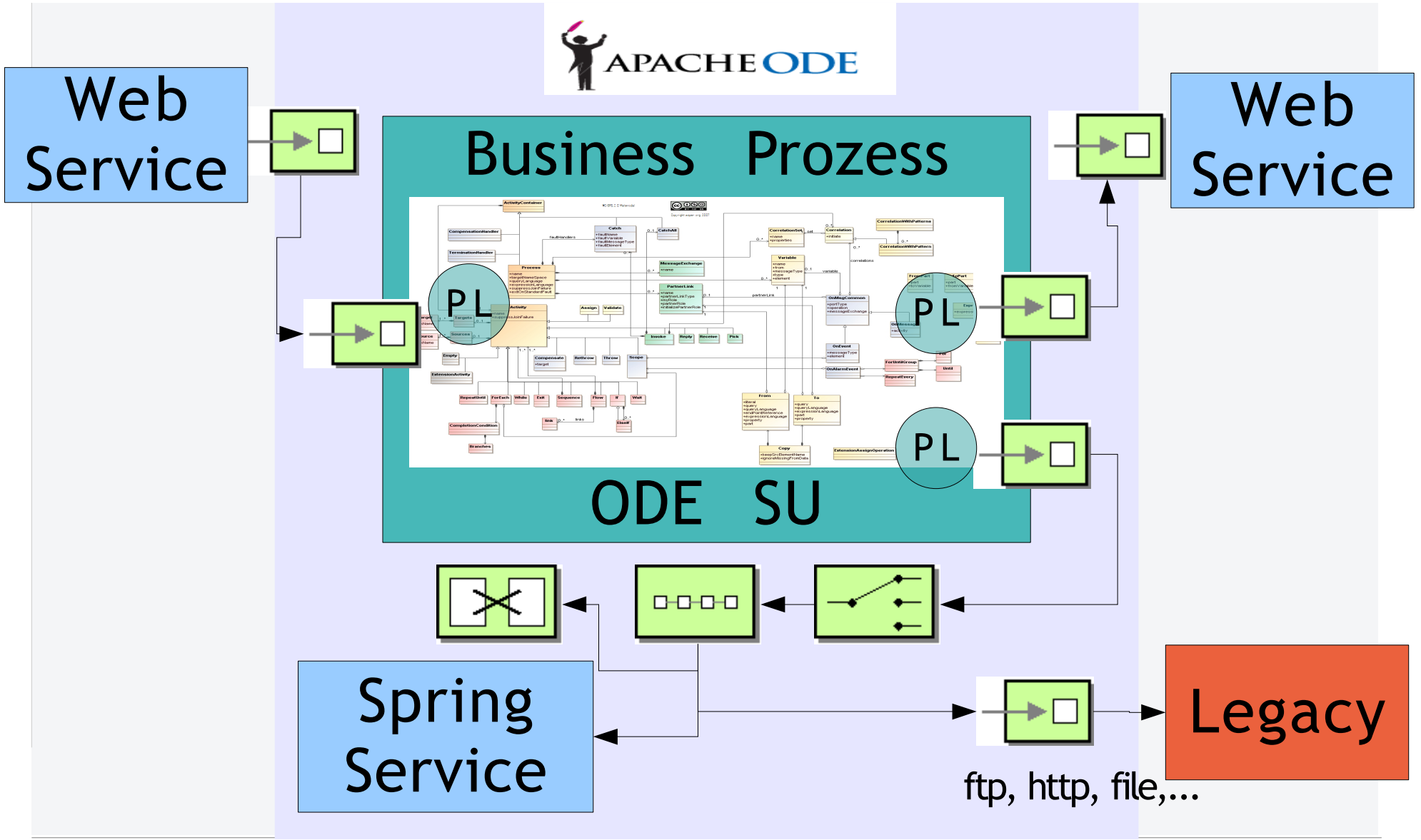
- Scripts als ESB-Client
- Scripting für Routing
- Groovy: XML Slurper, Builder, GPath
- JSR-223: JRuby, JavaScript
- Issues: Deployment Cycle, Debugging

BPM & ESB

- Reines WS-BPEL: Web Service Orchestrierung
- Standardisiert (vs. EIP)
- BPEL Editoren (BPMN)
- ESB-basiertes BPM:
  Orchestrierung ohne Einschränkungen

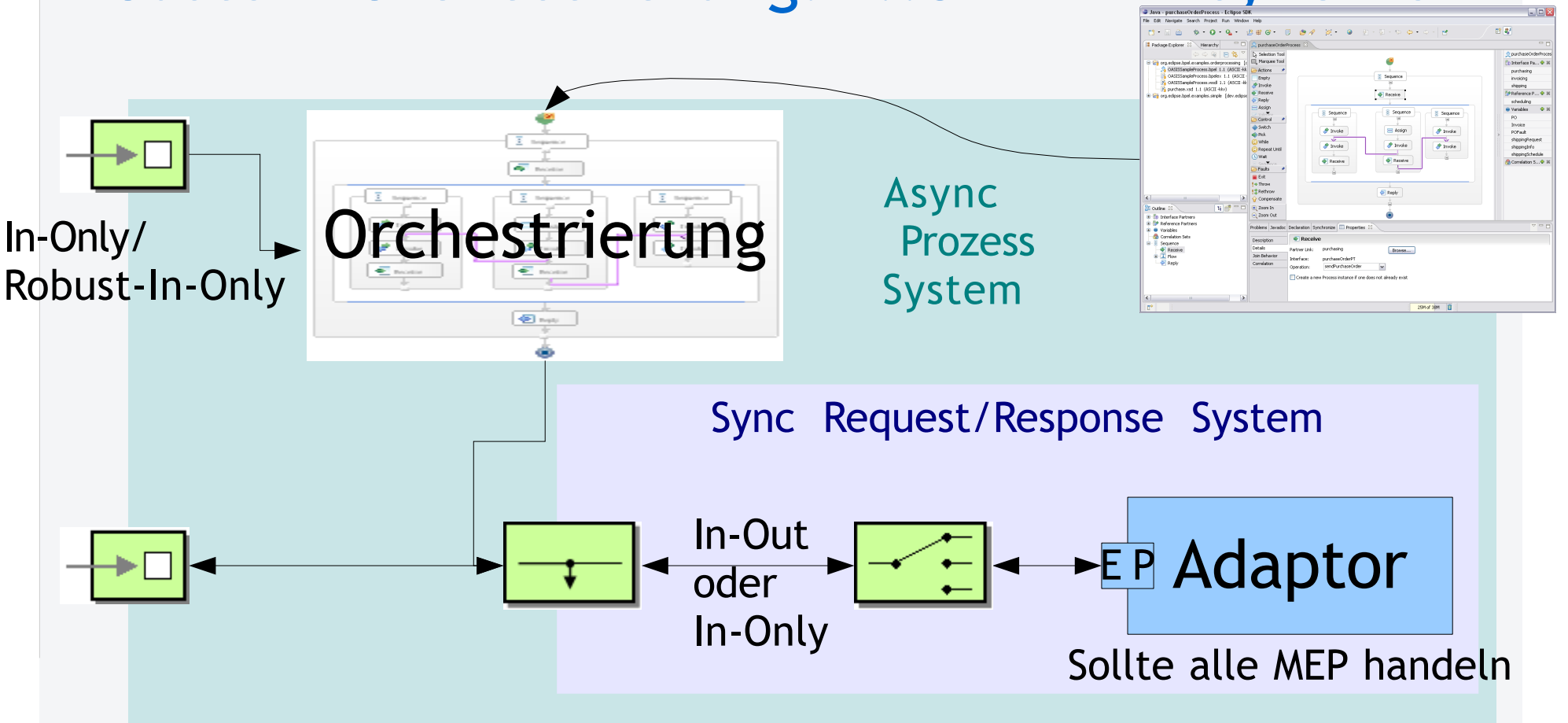


BPM mit ESB


Web Service
APACHE ODE
Business Prozess
PL
PL
PL
ODE SU
Web Service
Spring Service
Legacy
ftp, http, file,...

Sync & Async

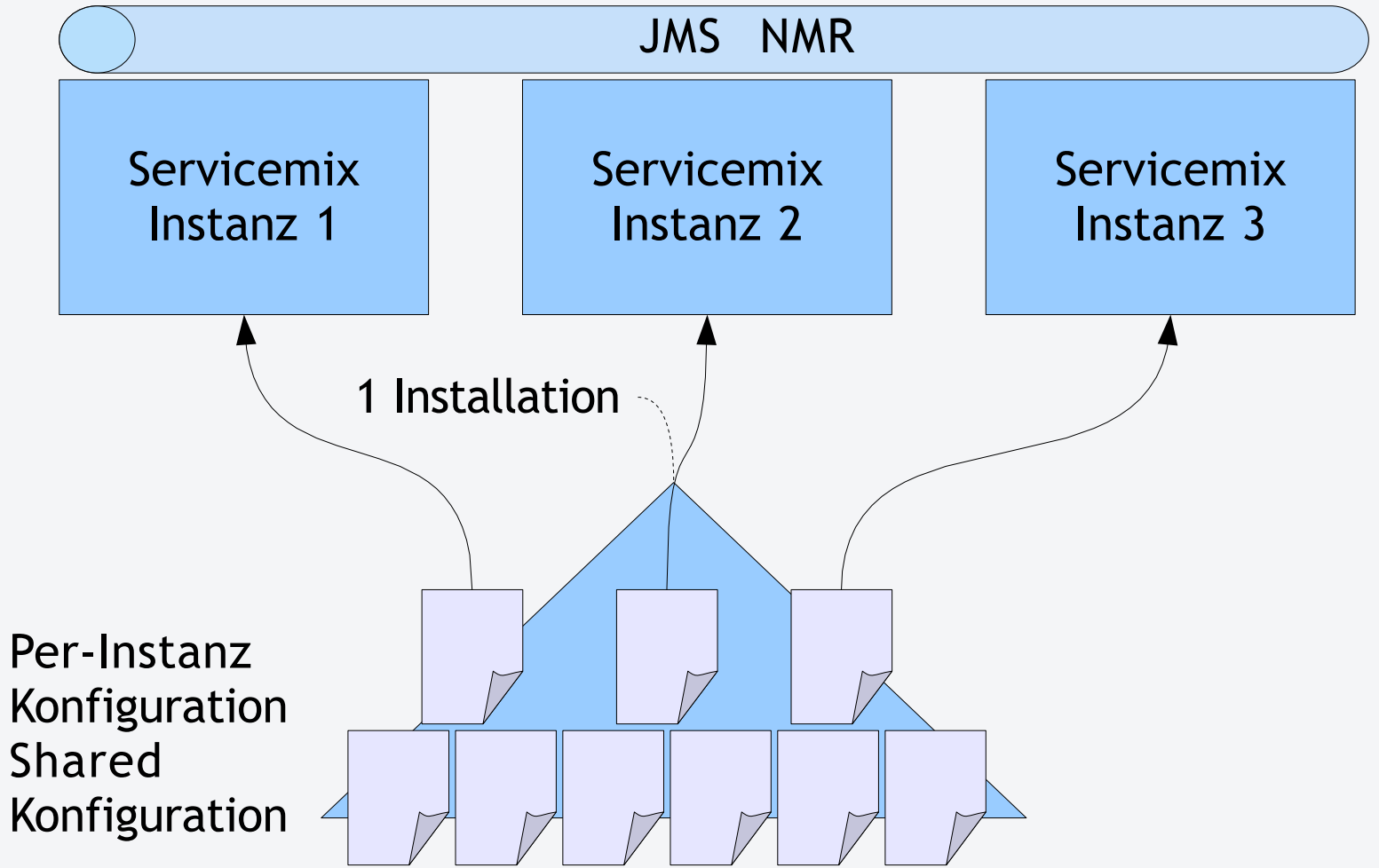
- Kernsystem: In-Out Synchron
- Custom-Orchestrierung: WS-BPEL Asynchron



Skalierbarkeit

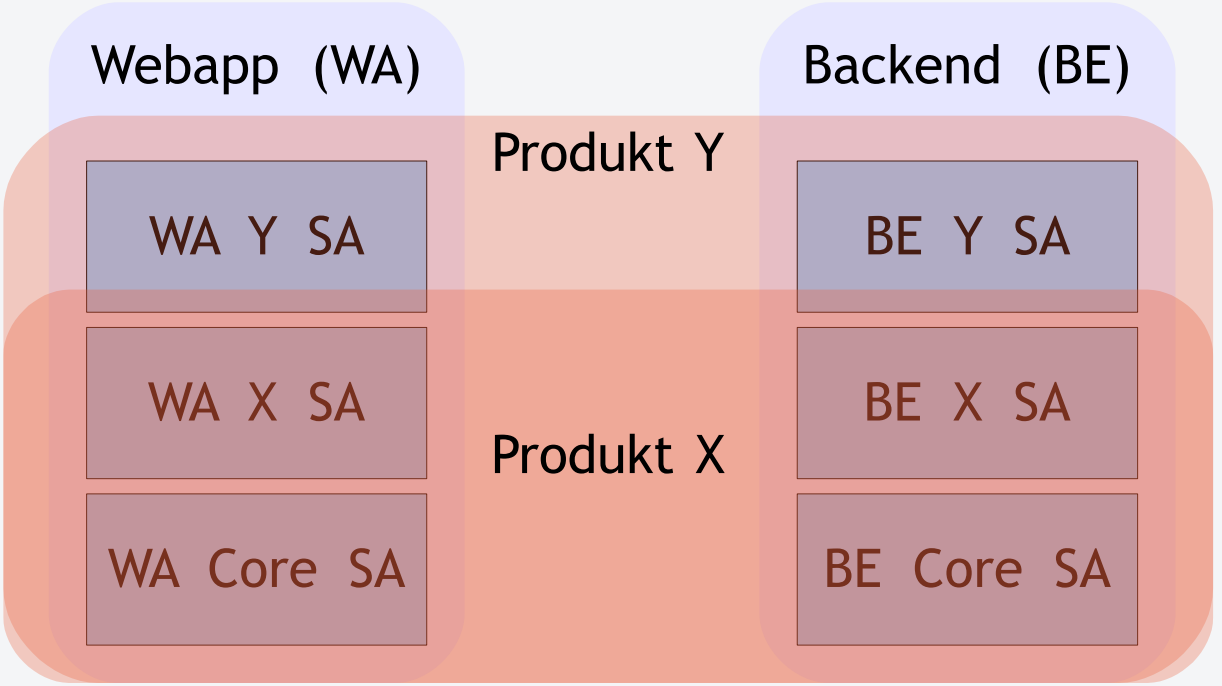
- Große XML Dokumente vermeiden
- Streaming (Stax, kein DOM)
- Software-Loadbalancing
  z.B. mehrere Jetty-SUs, mehrfache Pipelines
- Prozeß-Split
  - Pipeline Server
  - Compute Server
- Daemon-Wrapper für Auto-Restart

Multi-Instanz-Installation



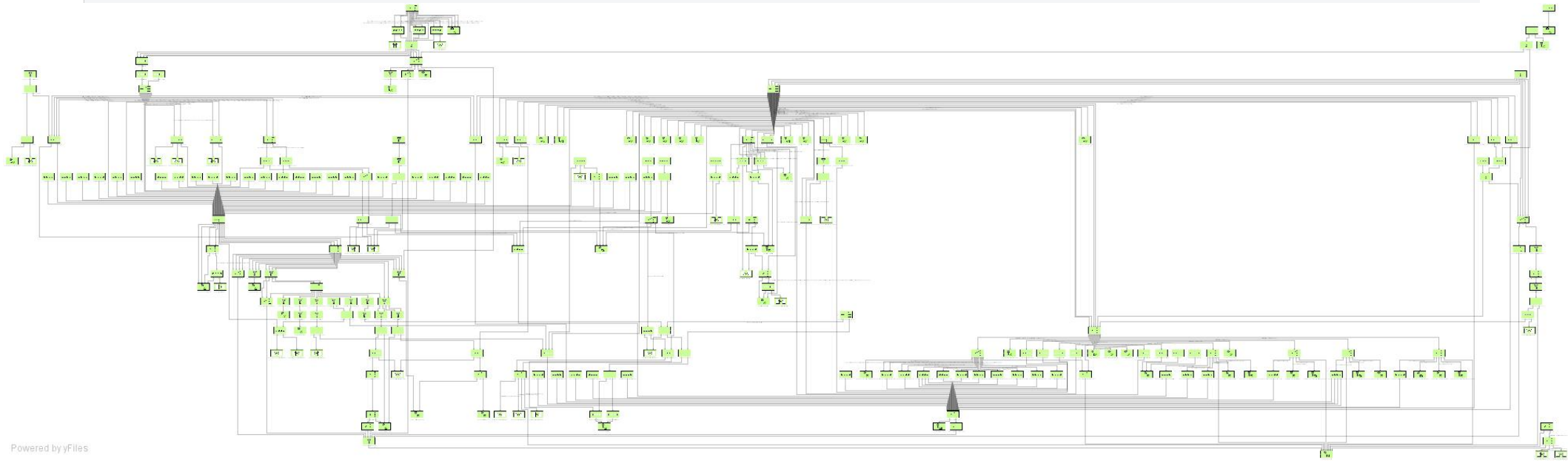
Produkt mit ESB

- Additive Service-Assemblies
  (Core + Produkt, Backend + Webapp)
- benowa, wanobe SAs
- Installer deployed Service-Assemblies



Fazit

- Geniale Modularisierung
- Lausige Kopplung
- Obacht auf Performance und Memory
- Komplexität



Fragen

Fragen?

blue elephant systems

Peter Klotz
Chief Architect

blue elephant systems GmbH
Wollgrasweg 49
D-70599 Stuttgart
(Germany)

phone: (+49) 711 45 10 17 677
fax: (+49) 711 45 10 17 573
mobile: (+49) 170 93 80 410

Peter.Klotz@blue-elephant-systems.com
http://www.blue-elephant-systems.com