

Service Oriented Architectures

Einführung in die Integration verschiedener
Anwendungssysteme

-

Problematik und allgemeine Architektur

Julia Weisheitel (WI5773)

Inhalt

- Überblick
- Probleme und Entscheidungen im Vorfeld
- Einführung einer Service Oriented Architecture

- **Überblick**

- Probleme und Entscheidungen im Vorfeld

- Einführung einer Service Oriented Architecture

Überblick

- **Ziel der Integration:** zwei oder mehr Anwendungen miteinander verbinden
 - Daten(bank)zugang
 - Geschäftslogik
 - Zusammenspiel in einem großen Ganzen

Überblick

- **Zwei Arten der Integration**
 - Reaktion auf unmittelbare Anforderungen
 - Integration generell vereinfachen
 - Nachteil: Standardisation steigert Auswirkungen, auch auf Kosten
 - Vorteil: Wiederverwendbarkeit/
Effizienzvergleich

Überblick

- **Zwei Arten der Lösung (prozessorientiert)**
 - Erweiterungen zu bestehenden Anwendungen
 - Unterstützung neuer Features
 - Veränderung von Geschäftsprozessen
 - Zugang zu fremden Daten oder Geschäftslogik
 - Integrationskanal nötig, Schwierigkeit plattformabhängig (heterogen oder homogen)

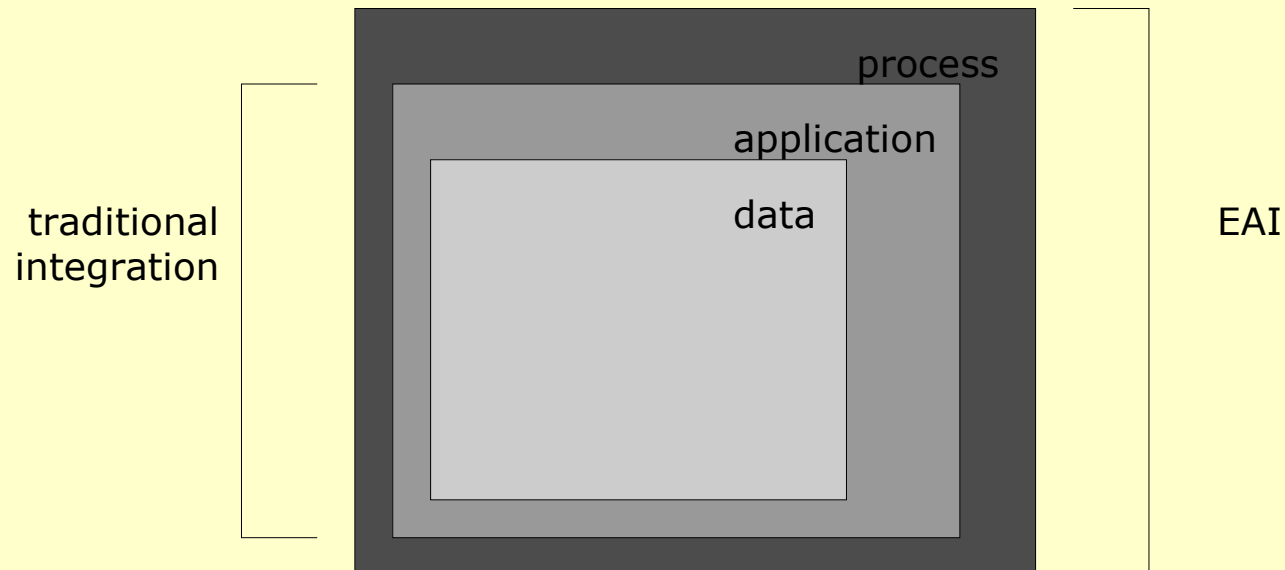
Überblick

- **Zwei Arten der Lösung (prozessorientiert)**
 - Neue Geschäftsprozesse
 - Überarbeitung bestehender Modelle führt zu komplettem Redesign, Zusammenschluss, Annäherung von Prozessen
 - Hohe Anforderungen an die technische Umgebung, die für Automation der Geschäftsfelder zur Verfügung steht

● Überblick

- **Integrationsebenen**

- Übergreifendes Arbeiten von Prozessen führt zu Kommunikationsbedarf

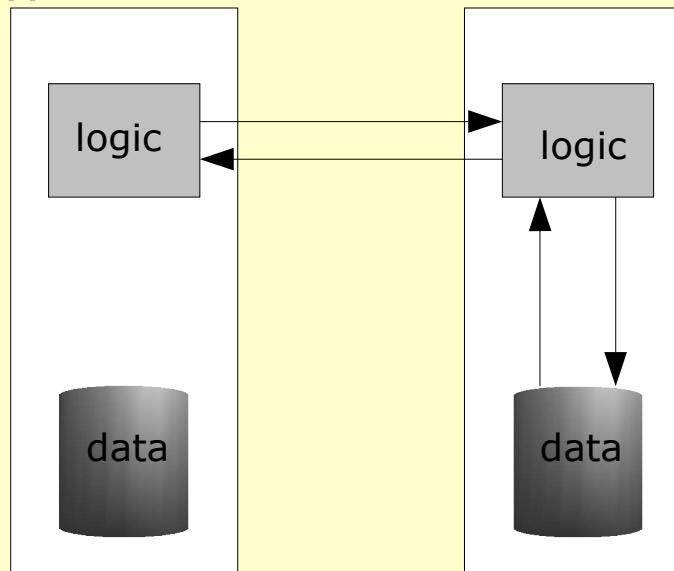


Überblick

- **Integration auf Datenebene**
 - Daten einer Anwendung ohne Geschäftslogik genutzt
 - Anwendung A hat direkten Zugriff auf die Datenbank von Anwendung B oder
 - Daten werden in die Datenbank von Anwendung A repliziert

● Überblick

- **Integration auf Anwendungsebene**
 - Anwendungslogik beider Anwendungen beteiligt
 - Punkt-zu-Punkt
 - Anwendungslogik von Anwendung B validiert z.B. Daten

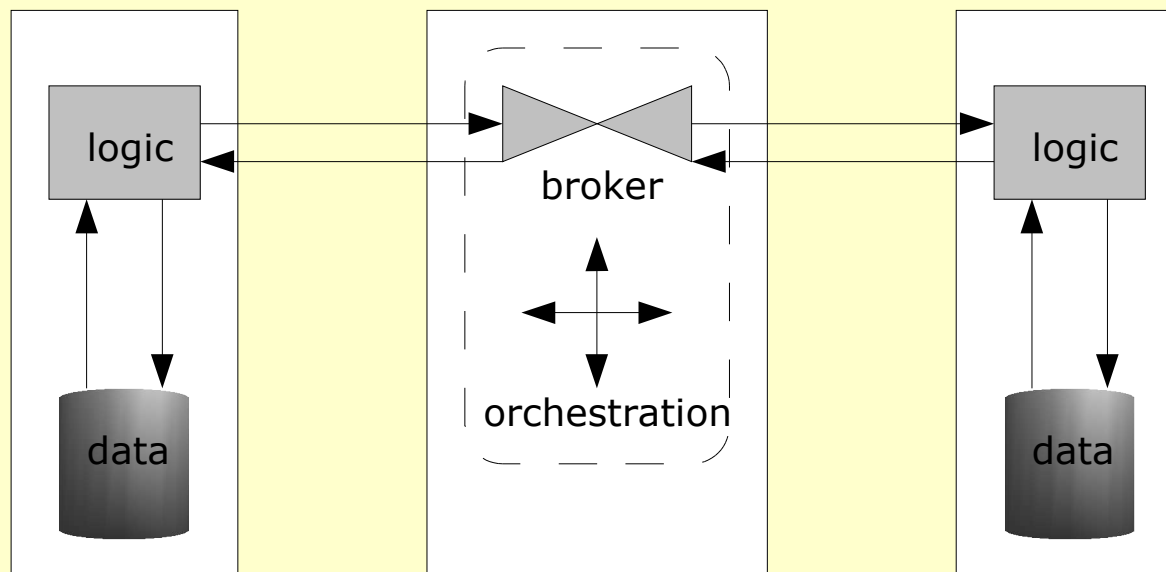


Überblick

- **Integration auf Prozessebene**
 - Schaffung eines neuen Prozesses zur Verbindung zweier oder mehrerer Prozesse
 - Traditionell: Punkt-zu-Punkt
 - Aktueller: EAI, spezielle Middleware (Messaging Framework)

● Überblick

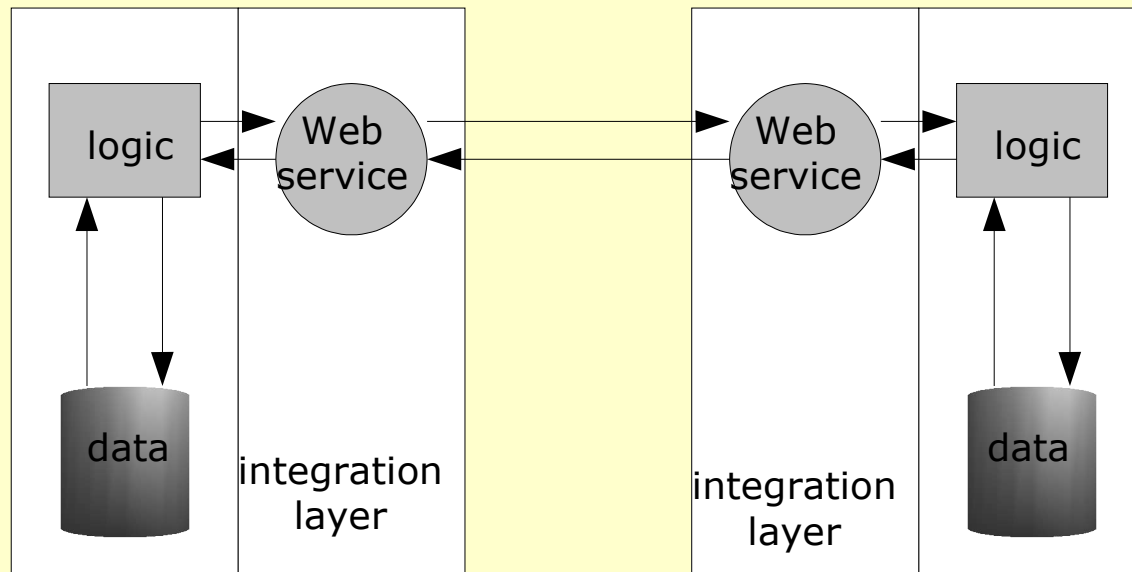
- Broker (Vermittler): zuständig für die Kommunikation
- Orchestration: Management und Ausführung neuer Prozesse



Überblick

- **Serviceorientierte Integration**
 - Web Services sind keine eigenständige Integrationslösung!
 - Neue Komponenten in einer EAI-Umgebung
 - EAI/Middleware weiterhin benötigt zur Integration
 - Web Services als Zugang zu fremden Daten und Geschäftslogik

Überblick



- Überblick
- **Probleme und Entscheidungen im Vorfeld**
- Einführung einer Service Oriented Architecture

Probleme und Entscheidungen im Vorfeld

Was ist Middleware?

- Vager Begriff
- Entstanden aus der Entwicklung von Mainframes hin zu verteilten Anwendungen
- Ermöglichte ursprünglich Kommunikation zwischen Teilen einer Software auf unterschiedlichen Servern
- Für transaktionsorientierten Datenaustausch zwischen verschiedenen Anwendungen

● Probleme und Entscheidungen im Vorfeld

Middleware vs. EAI

- EAI als modernere Form von Middleware
- Allgemeinerer Ansatz
- Mehr Möglichkeiten und Freiheiten
- Oftmals leichter anpassbar

● Probleme und Entscheidungen im Vorfeld

Verbreitete Kategorien zur Einordnung

- Datenzugang (Protokolle, APIs, Stored Procedures)
- Verteilte Anwendungen, entstanden aus Client-Server-Modell (RPC, Adapter, ORB)
- Message Queues, Nachrichten-Orientierung (MOM)
- Anwendungsserver, Transaktionsüberwachung (Webanwendungsserver, Fremdsystemserver)
- Integrations-Broker/Message-Broker (EAI)

Probleme und Entscheidungen im Vorfeld

Allgemeine Hinweise zur Auswahl eines passenden Produktes

- Weniger ist mehr...
- Auswahl an unterstützten Modellen
- Fortschritt statt Rückschritt
- Abbildung von Geschäftsprozessen
- Aufwand zur Integration
- Wartung/Administration
- Skalierbarkeit

Probleme und Entscheidungen im Vorfeld

Viele Wege führen zur Integration – welcher ist der Richtige?

- Aufbauen auf dem, was vorhanden ist
- Direkter Sprung in EAI

● Probleme und Entscheidungen im Vorfeld

Auf dem Weg zur EAI

- Je schneller, desto besser?
- Kaufen = unterwerfen?
- Integration ist teuer
- Integration kostet Zeit
- Integration ist ein Störfaktor
- Direkter vs. Indirekter Wert

● Probleme und Entscheidungen im Vorfeld

Der leichtere Weg

- Über mehrere Phasen
- Zuerst aufbauen auf Bestehendem
 - Kostengünstig
 - Weniger abrupte Änderungen
 - Bessere Planung/Reaktion möglich

- Überblick
- Probleme und Entscheidungen im Vorfeld
- **Einführung einer Service Oriented Architecture**

● Einführung einer Service Oriented Architecture

SOA und Web Services

- Gegenseitig beflügelt
- Integrationskosten als Grund für Frage nach standardisierter Datenaustauschplattform, unabhängig von Anbietern und Technologien
 - Schub für XML und damit auch Web Services

● Einführung einer Service Oriented Architecture

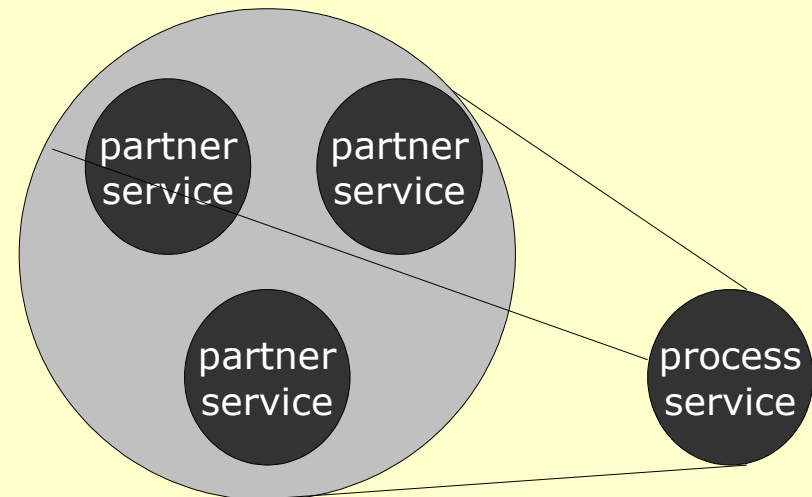
Service-Modelle für Enterprise-Integration Architekturen

- EAI ist prozessorientiert
 - Modelle gefragt, die verschiedene Aspekte des Geschäftsgeschehens abbilden und koordinieren können
- Hauptsächlich zwei Modelle
 - Prozessservices
 - Koordinationsservices (für Geschäftszwecke)

● Einführung einer Service Oriented Architecture

Prozessservices

- Kapselung aller nötigen Teilservices eines Geschäftsprozesses in einem dedizierten Prozess
- Gesamte Logik an einer Stelle verfügbar (Regeln, Ausnahmen, Bedingungen bezüglich des Workflows)
- Kapselung auf mehrere Ebenen erweiterbar



● Einführung einer Service Oriented Architecture

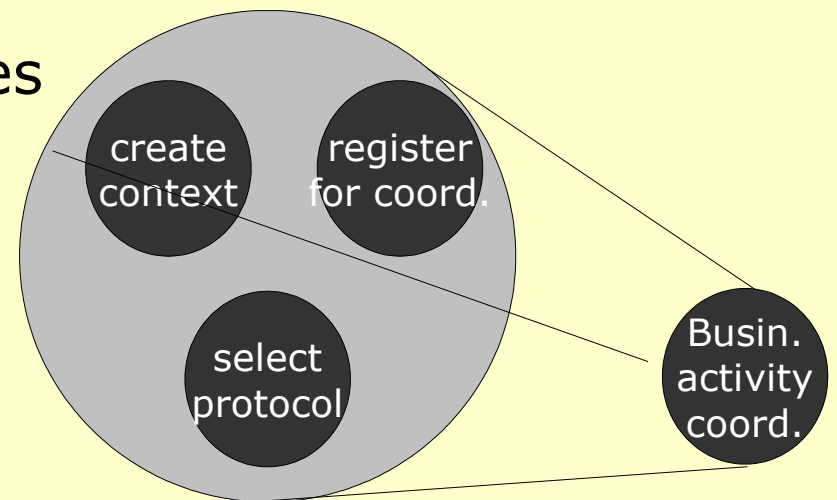
Charakteristika

- Voraussichtliches Nutzungsaufkommen: hoch bis sehr hoch, Service ist an jedem Schritt des Prozesses beteiligt
- Schnittstellendesign: BPEL4WS hat sich durchgesetzt; implementationsunabhängig durch Weglassen der Binding-Information im dazugehörigen WSDL-Dokument
- Umgebung: Orchestration Engine

● Einführung einer Service Oriented Architecture

Koordinationservices

- Eng verknüpft mit Transaktionen (WS-Coordination + WS-Transaction)
- Zwei Arten von Transaktionen: Atomare Transaktionen (ACID), Langzeittransaktionen
- Langzeittransaktionen zur Behandlung von Erfolgs- und Fehlerbedingungen, die den Ablauf des gesamten Prozesses beeinflussen



● Einführung einer Service Oriented Architecture

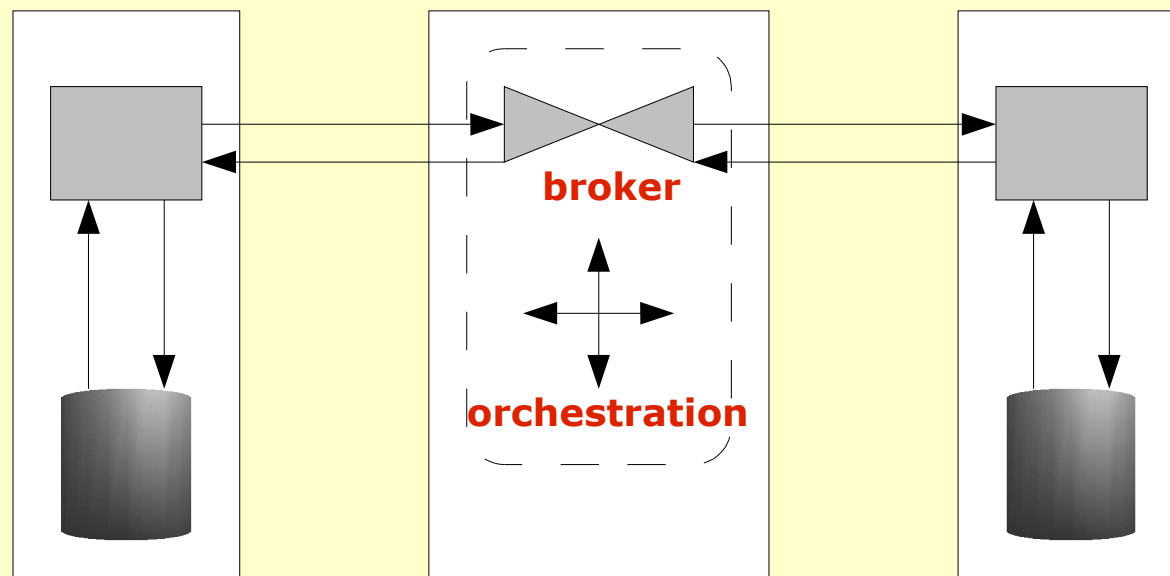
Charakteristika

- Voraussichtliches Nutzungsaufkommen: niedrig bis hoch, da Geschäftsaktivität wenige Minuten bis zu mehreren Tagen dauern kann; tatsächliche Bearbeitung meist sporadisch
- Schnittstellendesign: vordefiniert durch WS-Coordination (Aufbau und Schnittstelle nach außen festgelegt, aber erweiterbar)
- Umgebung: häufig im Rahmen von EAI-Servern genutzt

● Einführung einer Service Oriented Architecture

Grundlegende Architekturkomponenten der Enterprise Integration

- Kommunikation zwischen Anwendungen und Automation neuer Prozesse durch zwei Kernkomponenten



● Einführung einer Service Oriented Architecture

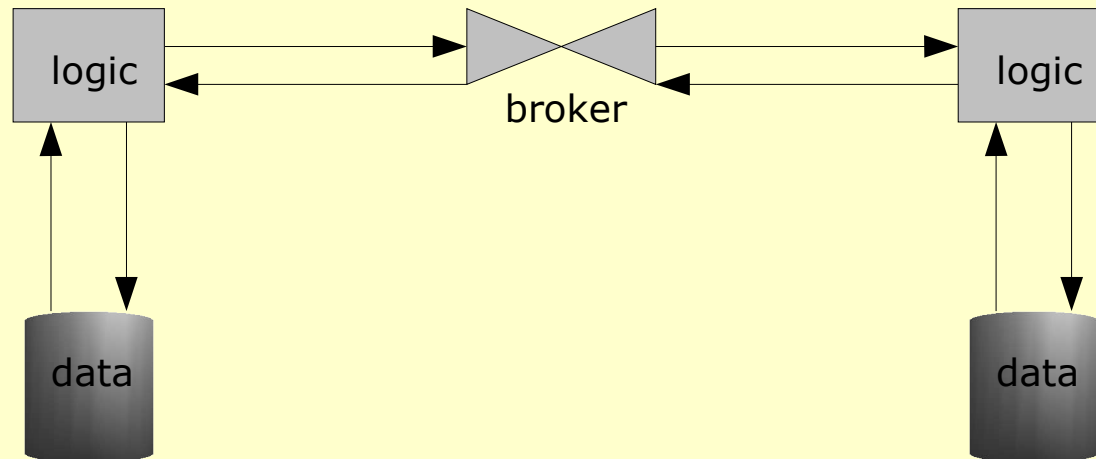
Broker

- Zuständig für viele Laufzeitfunktionen
 - Anspruchsvolle Datentransformationen
 - Dokumente aus verschiedenen Quellen zusammenführen
 - Daten einer Anwendung mit Daten einer anderen ergänzen
 - **Daten müssen immer in dem Format vorliegen, in dem sie erwartet werden**

● Einführung einer Service Oriented Architecture

Datenaustausch am Beispiel

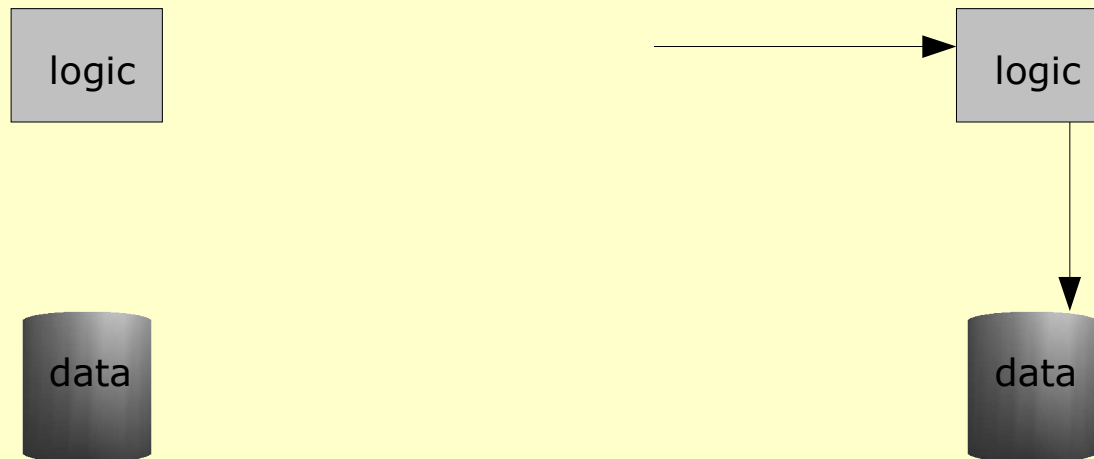
- Bereitstellung von Daten in fünf Schritten



● Einführung einer Service Oriented Architecture

Schritt 1

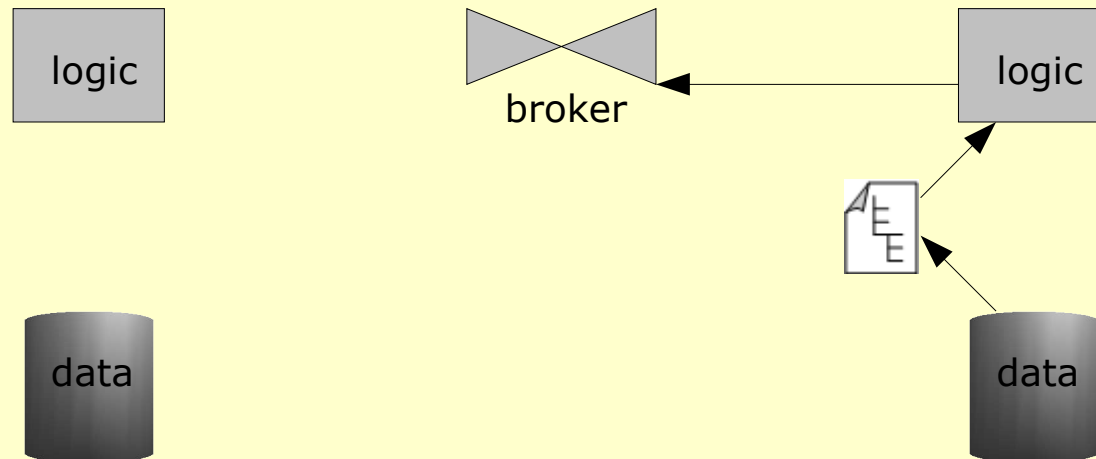
- Gewünschte Daten aus der Quelldatenbank ermitteln



● Einführung einer Service Oriented Architecture

Schritt 2

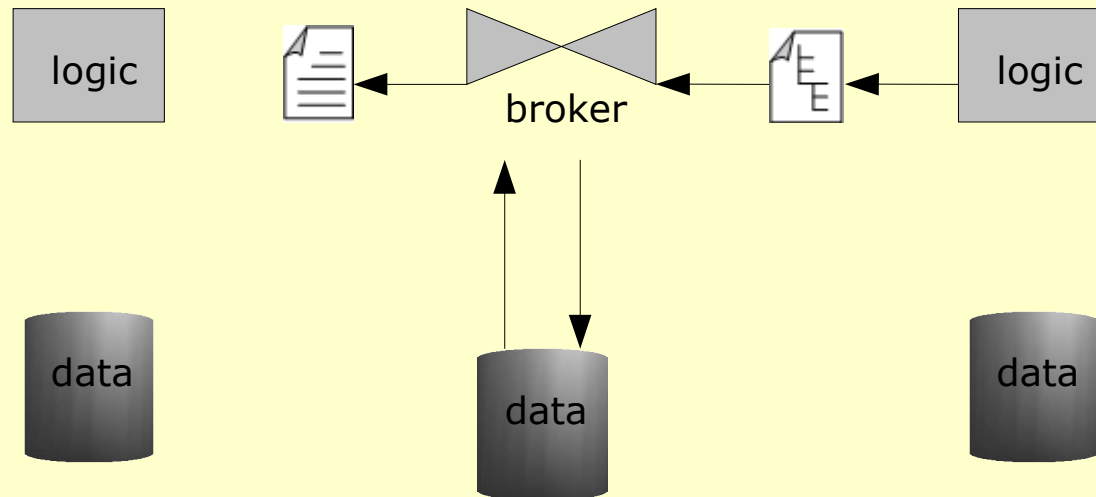
- Daten mit Schema validieren (direkt in der Datenbank oder durch Anwendungslogik)



● Einführung einer Service Oriented Architecture

Schritt 3

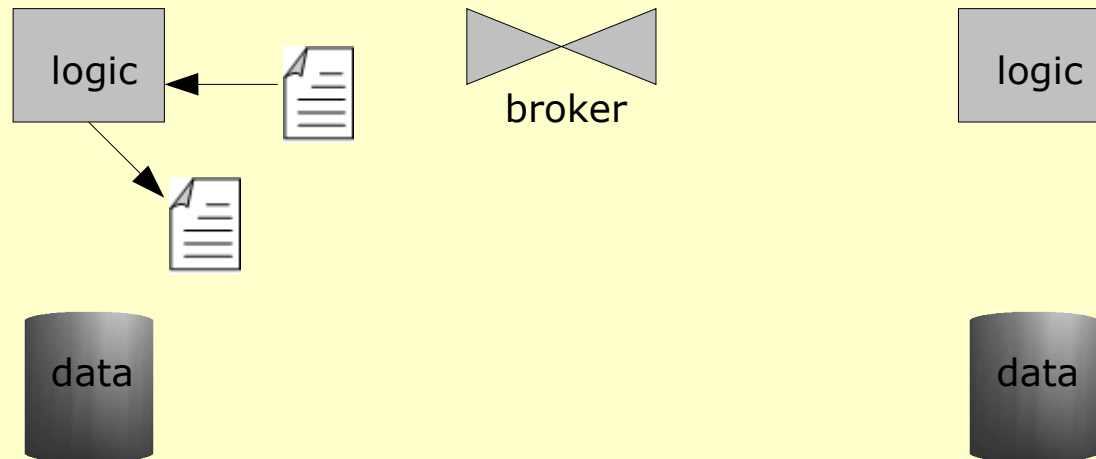
- Das Datenformat ändern



● Einführung einer Service Oriented Architecture

Schritt 4

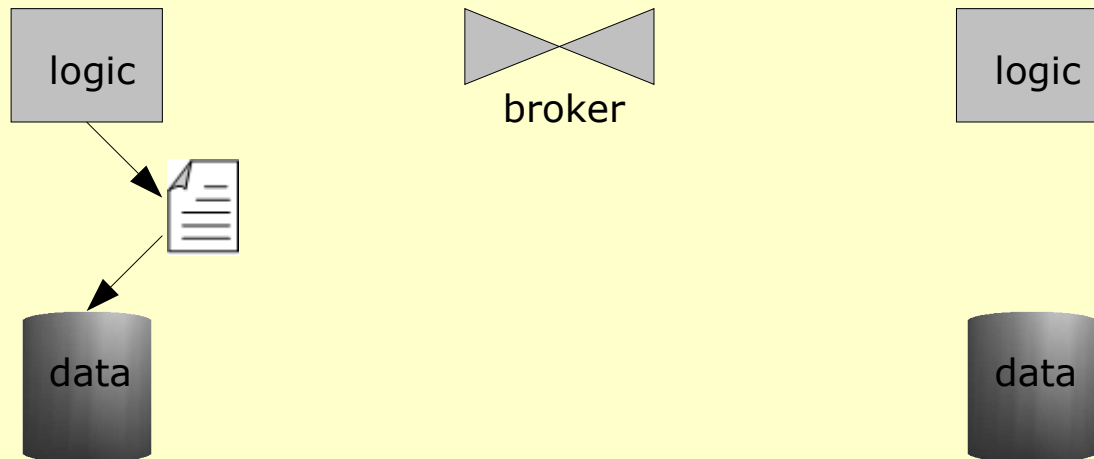
- Zieldaten mit Zielschema validieren



● Einführung einer Service Oriented Architecture

Schritt 5

- Daten in Zieldatenbank einfügen



● Einführung einer Service Oriented Architecture

Orchestration

- Zur Kapselung und Ausführung der Logik des Geschäftsprozesses
 - Integration mit anderen Anwendungen um Zusatzdaten zu ermitteln
 - Broker zur Manipulation der Daten nutzen
 - Bestehende Logik nutzen um ermittelte Daten weiter zu validieren
 - Daten im Fehlerfall zurückweisen

● Einführung einer Service Oriented Architecture

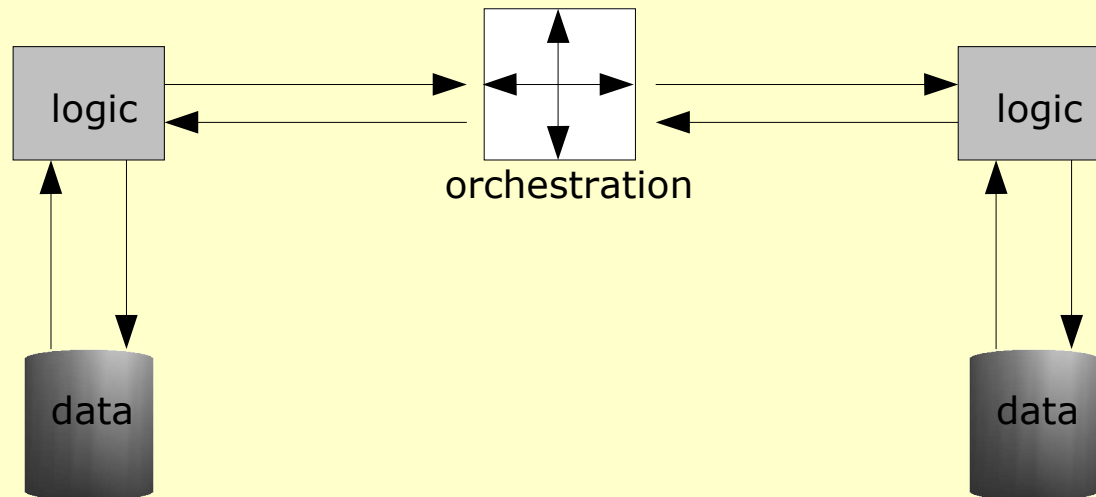
Orchestration

- ein-/ausgehende Nachrichten oft anspruchsvoller als herkömmliche SOAP-Messages, z.B. durch zusätzliche Routing-/Transaktionsinformationen
- Status für Nachrichten möglich: permanent, in Bearbeitung

● Einführung einer Service Oriented Architecture

Datenaustausch am Beispiel

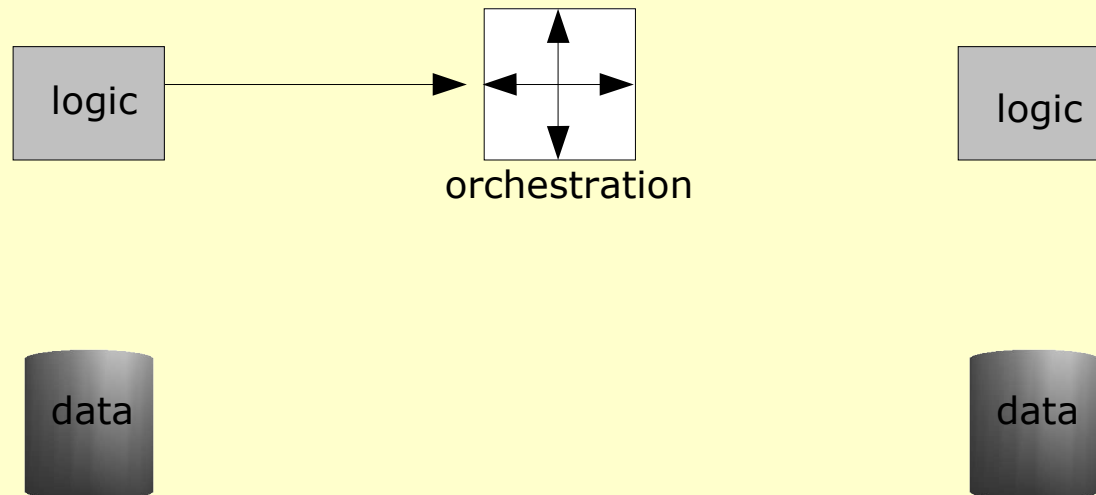
- Bereitstellung von Daten in fünf Schritten



● Einführung einer Service Oriented Architecture

Schritt 1

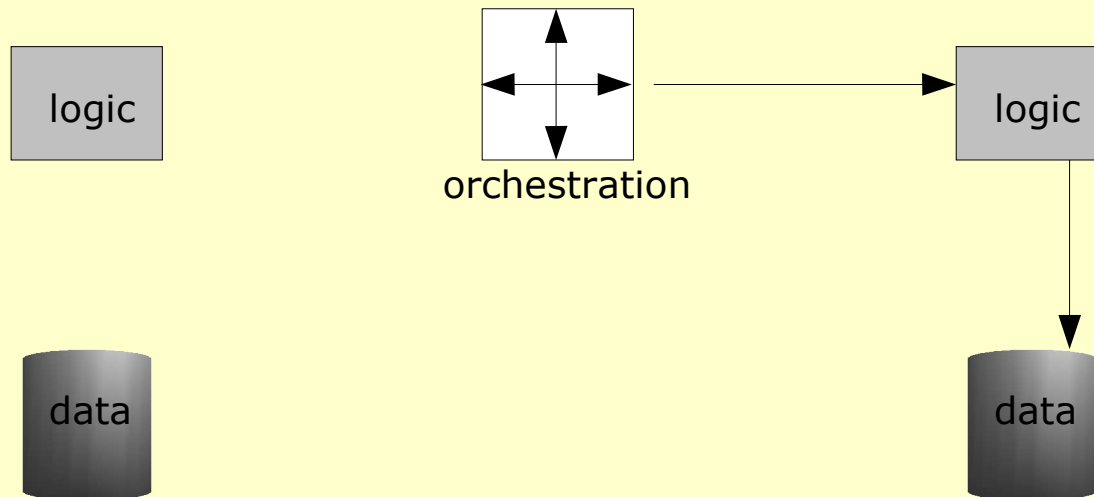
- Daten erfragen (A muss dabei von Bs Existenz nichts wissen, Zugang ist Prozess/Workflow)



● Einführung einer Service Oriented Architecture

Schritt 2

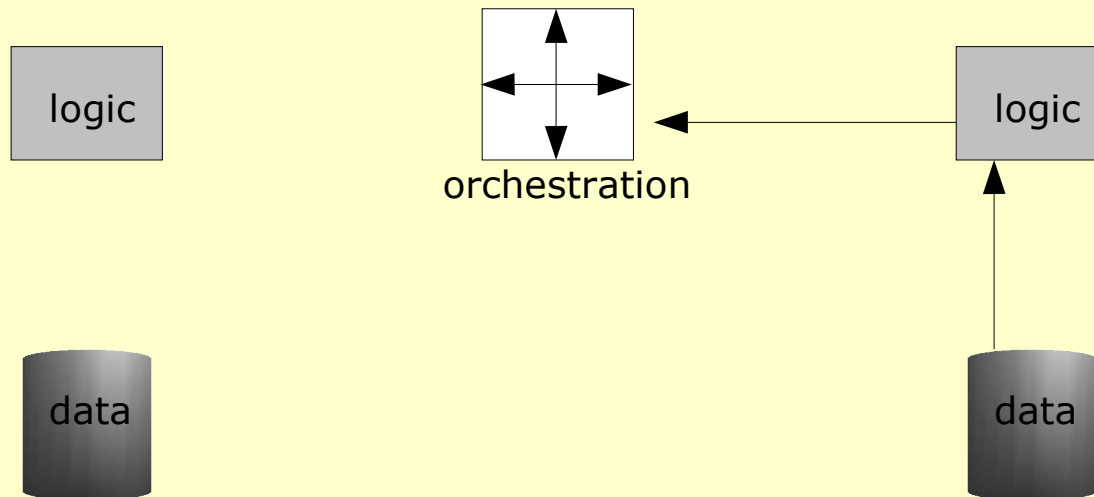
- Daten von Anwendung erfragen



● Einführung einer Service Oriented Architecture

Schritt 3

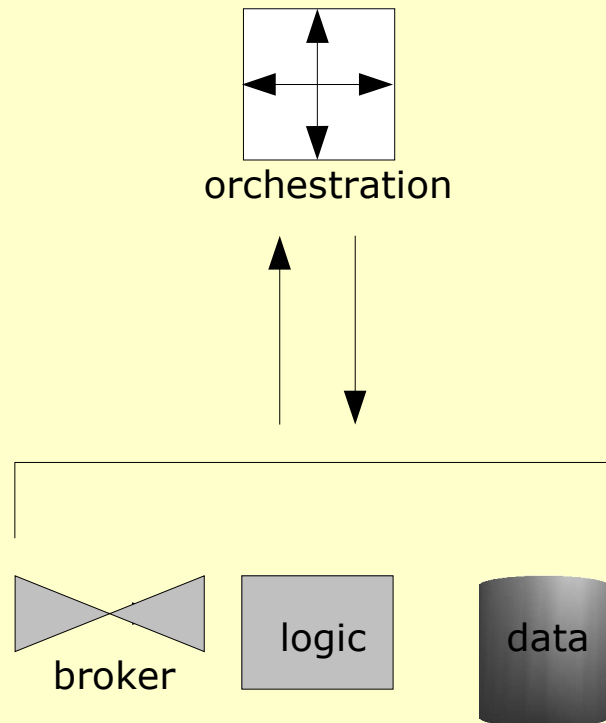
- Antwort an Orchestration



● Einführung einer Service Oriented Architecture

Schritt 4

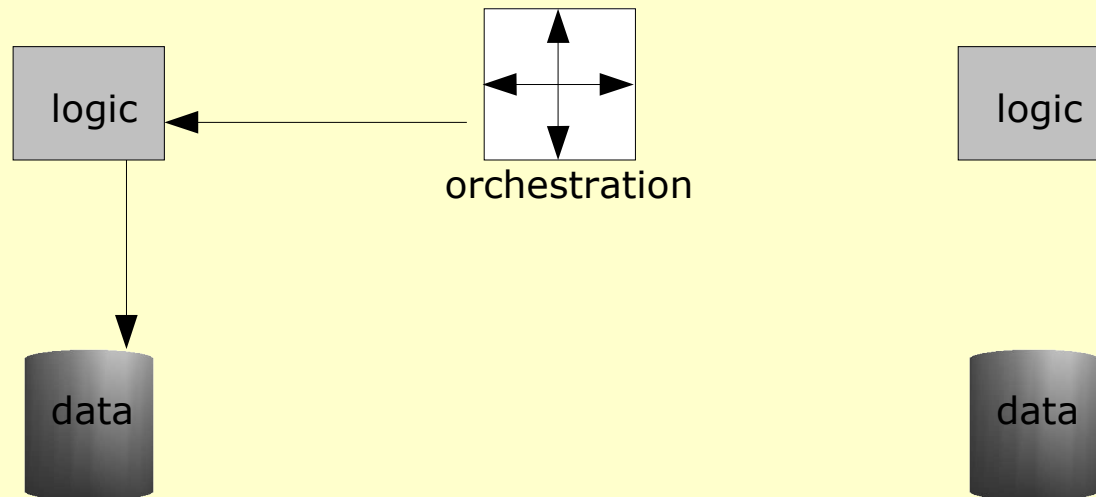
- Logik der Orchestration (entsprechend des Geschäftsprozesses), auch in Schritt 1 möglich



● Einführung einer Service Oriented Architecture

Schritt 5

- Daten an fragende Anwendung weiterleiten



● Einführung einer Service Oriented Architecture

Web Services und Enterprise Integration Architekturen

- EAI hauptsächlich asynchron, damit prädestiniert für XML
- WS teils integriert, zum Teil reine serviceorientierte Integrationsprodukte
- Brücke zwischen anbieterspezifischen EAI-Plattformen und plattformübergreifenden Bedürfnissen der Unternehmen

● Einführung einer Service Oriented Architecture

Drei Architekturmodelle in Überblick

- Hub and Spoke
- Messaging Bus
- Enterprise Service Bus (ESB)

● Einführung einer Service Oriented Architecture

Hub and Spoke

- Proprietäre Middleware fungiert als Hub
 - Zentralisierte Verarbeitung
 - Zentraler Server als Host der Integrationslogik
 - Kontrolliert Orchestration und Broker

● Einführung einer Service Oriented Architecture

Realisation

- Adapter
 - Einbindung vieler unterschiedlicher Anwendungen möglich
 - Verbindung mit Hub als Einstiegspunkt zu allen anderen Anwendungen, kein Wissen um physische Existenz nötig, logische Autonomie
 - Punkt-zu-Punkt-Verbindung von Anwendung zum Hub, beliebige Kardinalität von Anwendung zu Anwendung

● Einführung einer Service Oriented Architecture

Vor- und Nachteile

- Redundante Mehrfachverarbeitung/-speicherung von Daten minimiert
- Zentrale Lage aller Logik für bessere Wartbarkeit und Wiederverwendung
- Engpässe, da Hub an allen Vorgängen beteiligt (abhängig von Skalierbarkeit)
- Single Point of Failure
- teuer

● Einführung einer Service Oriented Architecture

Rolle von Web Services

- Als zusätzlicher Zugang zum Hub, um Anwendungen mit Service-Möglichkeit Anschluss zu geben, ohne (anzupassenden) Adapter nutzen zu müssen
- Vorsicht: Wartung/Administration dieser Zugänge schwierig

● Einführung einer Service Oriented Architecture

Messaging Bus

- = publish/subscribe-Modell = Informationsbus-Modell
- Konzept ähnlich Hub and Spoke: zentral gelegene Umgebung, individuelle Verbindung zu einzelnen Anwendungen
- Interne Struktur anders als Hub and Spoke

● Einführung einer Service Oriented Architecture

Realisation

- Information Pipeline mit Adaptern für ein-/ausgehende Nachrichten
- Anwendungen nehmen Rolle des Veröfentlichers oder Abonnenten ein
 - Jede Integrationsquelle als Veröfentlicher, andere Anwendungen Abonnent dieser Quelle
 - Nachricht als Broadcast an alle Abonnenten
- Transformation, Routing, Prozesslogik weiterhin durch Orchestration

● Einführung einer Service Oriented Architecture

Vor- und Nachteile

- Prozess stärker zerteilt
 - Weniger zentralisiert, weniger Engpässe
 - Intelligente Adapter können Teile der Aufgaben übernehmen
 - Schlechter wartbar

● Einführung einer Service Oriented Architecture

Rolle von Web Services

- Service-Integrationsschichten für standardisierte Kommunikation
 - Abstraktion der Adapter
 - Verarbeitungslast der Messaging Bus Server sinkt weiter

● Einführung einer Service Oriented Architecture

Enterprise Service Bus

- Architektur von Grund auf serviceorientiert designt
- Sämtliche EAI-Komponenten als Service/durch Service steuerbar

● Einführung einer Service Oriented Architecture

Realisation

- Integrationsplattform als dynische Hosting-Umgebung
- (proprietäre) Service-Container für (freie) Service-Komponenten

● Einführung einer Service Oriented Architecture

Vorteile

- Alle Services den eigenen Bedürfnissen entsprechend anpassbar/austauschbar
- Plattform austauschbar, ohne dass Logik verloren geht
- Bindung an einen Anbieter wird gelockert