Matthias Jauernig
03INB
http://www.linux-related.de

Leipzig, 05.07.06

Ausarbeitung “Informationssysteme“

1. Informationssysteme – Grundbegriffe

- Begriff Information: aus technischer Sicht (Informationstheorie), aus betriebswirtschaftlicher Sicht

  - Informationstheorie:
    - Information als Aneinanderreihung von Zeichen
    - Keine Aussage über die Bedeutung einer Information
    - Information = Beseitigung von Unsicherheit
  - Betriebswirtschaftliche Sicht:
    - Information ist immaterielles Gut, wird bei Nutzung nicht verbraucht
    - Können kostenadäquaten Wert haben
    - Wert hängt von Kontext und Zeit ab
    - Wertsteigerung durch Erweiterung und Verdichtung möglich
    - Durchsetzung von Eigentumsrechten schwierig

- Wirtschaftsgut vs. Information

Wirtschaftsgut	Information
Hohe Vervielfältigungskosten	Niedrige Vervielfältigungskosten
Wertverlust durch Gebrauch	Kein Wertverlust durch Gebrauch
Begrenzte Teilbarkeit, Wertverlust dabei	Beliebiger Teilbarkeit, kein Wertverlust
Wert im Markt bestimmbar	Wert nur schwer bestimmbar
Individueller Besitz	Vielfacher Besitz möglich

- Informationsystem: Definition (als Mensch-Maschine-System), Bestandteile

  - Def.: „IS sind Mensch-Maschine-Systeme, die menschliche und maschinelle Komponenten umfassen und zum Ziel der optimalen Bereitstellung von Information und Kommunikation nach wirtschaftlichen Kriterien eingesetzt werden.“
  - Rechnergestützte IS: Erfassung, Speicherung und Transformation von Information durch Einsatz von IT teilweise automatisiert
  - Bestandteile: Hardware, Betriebssystem, Datenbanksystem, Anwendungssysteme, Menschen

- Klassifikation von Informationssystemen, z.B. nach Verwendungszweck, funktionsbezogen vs. Unternehmensbezogen

  - Nach Verwendungszweck:
    - Administration: Speicherung von Massendaten
    - Dispositionssysteme: Entscheidungsunterstützung bei Routine-Vorgängen
    - Entscheidungsunterstützungssysteme: Vorbereitung von Entscheidungen auf mittlerer oder oberer Managementebene

- Funktionsbezogen:
  - IS für Produktion, Technik, Beschaffung, Absatz, Rechnungswesen, Verwaltung
- Unternehmensbezogen:
  - IS als integriertes Gesamtsystem der betrieblichen Informationsverarbeitung

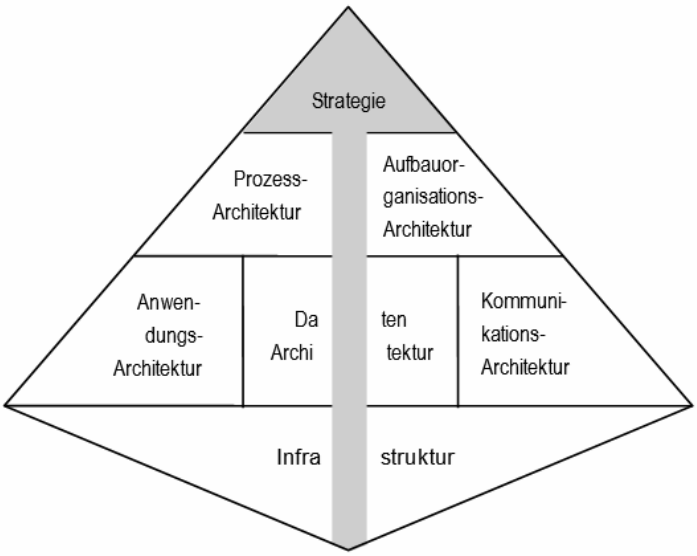
- Typen von Informationen im Informationssystem

  - Fakten
  - Texte: Titel/Abstracts, Volltexte
  - Bilder
  - Videos
  - Audio
  - Mischformen, Hypermedia

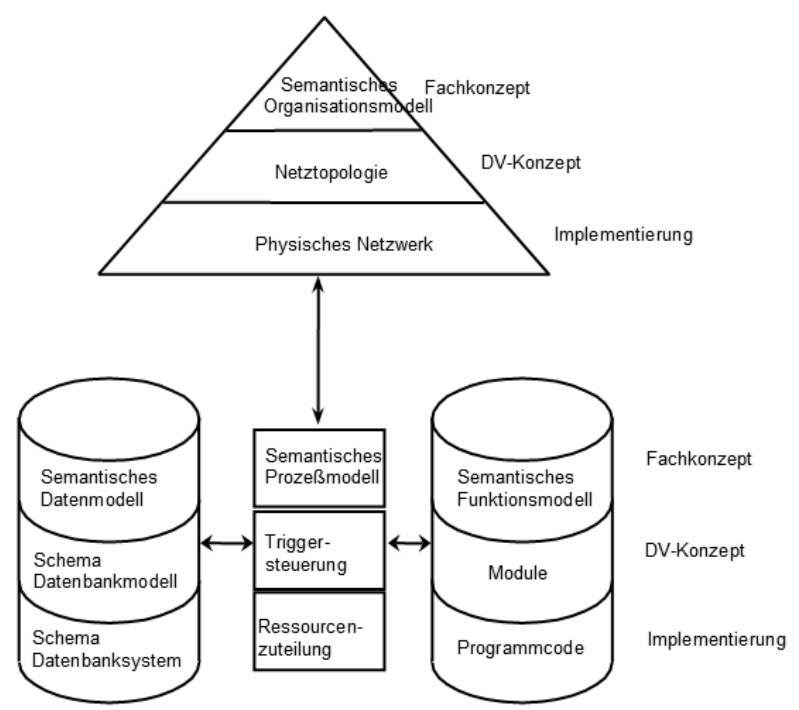
2. Informationsmanagement

- Architekturkonzepte: Zachmann, ISA-Konzept / Kreiselmodell (Krcmar), ARIS

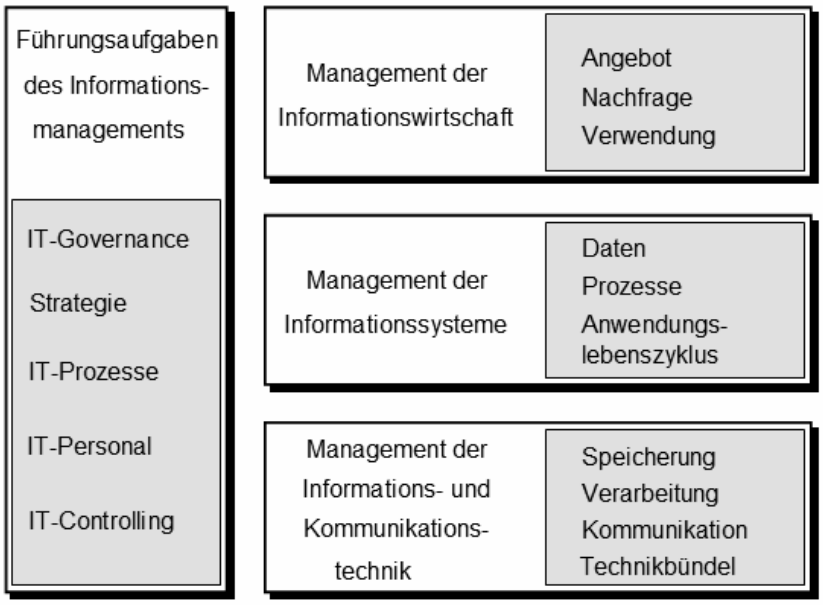
  - Zachmann – Unternehmensarchitektur:
    - 5 Perspektiven: Planner, Owner, Designer, Builder, Subcontractor
    - 6 Fragestellungen: Was, Wie, Wo, Wer, Wenn, Warum?
  - Krcmar – ISA-Konzept als Kreiselmodell:



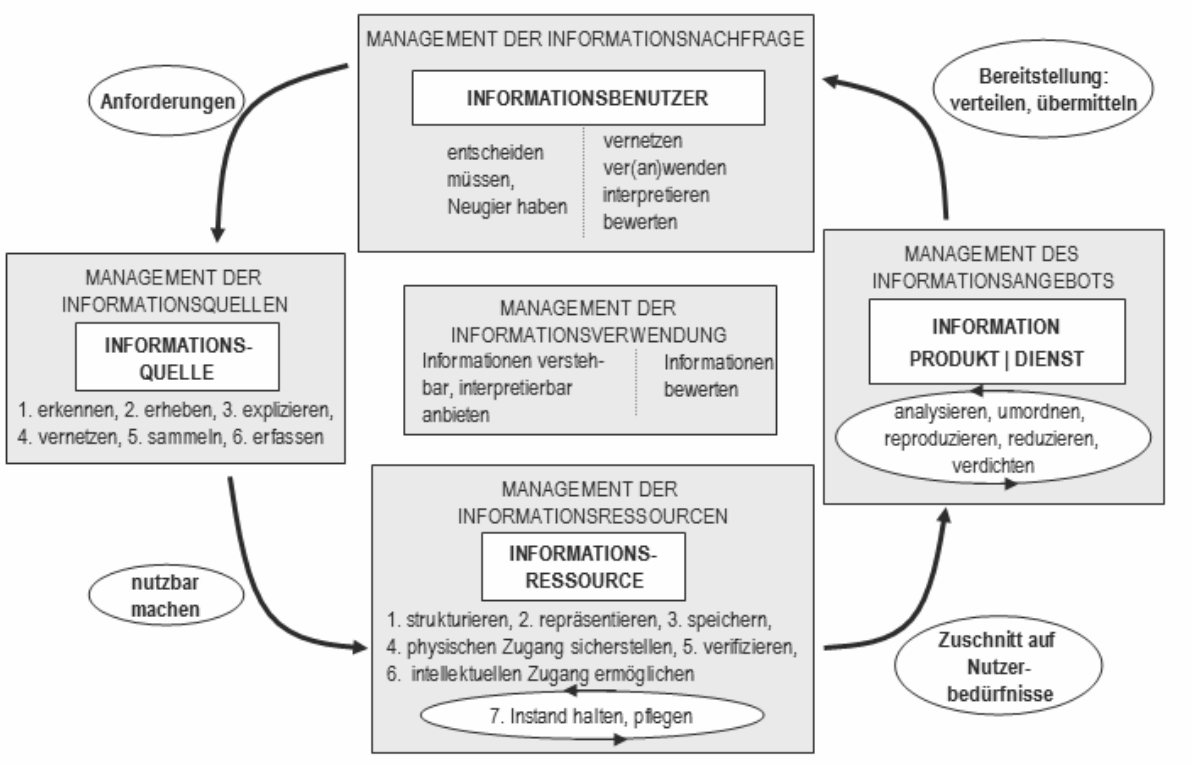
  - ARIS:



- Rahmenmodell des Informationsmanagement (Krcmar)



- Lebenszyklusmodell der Informationswirtschaft von (1) Management des Informations-bedarfs bis (5) Management der Informationsverwendung



- Weitere Aspekte:

  - Balanced Scorecard Methode
    - Ziel- und kennzahlenbezogene Management-Methode
    - Abbildung von Visionen und Strategie eines Unternehmens auf 4 Bereiche: Finanzen, Kunde, Interne Geschäftsprozesse, Lernen und Entwicklung
    - Vermeidung der einseitigen Ausrichtung auf finanzielle Aspekte
  - Informationsquellen
    - Bedarfsorientierte Angebotsgestaltung: Angebotsumfang, Ermitteln des Informationsbedarfs, Sammeln benötigter Informationen
    - Informationsangebot (intern/extern) abhängig vom Informationsnachfrager (intern/extern)
    - Klassifikation: nach Zugänglichkeit, Erscheinungshäufigkeit, Kosten, Medien
  - Informationsstrukturierung und –repräsentation (Thesaurus, Ontologie)
    - Ontologie:
      - formale Beschreibung der Semantik von Informationsobjekten
      - Beziehungen von Begriffen innerhalb eines Anwendungsbereichs
      - Klassifikation: Top-Level-O., Bereichs-O., Anwendungs-O.
    - Thesaurus:
      - Systematisch geordnetes Verzeichnis von Schlagwörtern eines bestimmten Fachgebiets
      - Beziehungen: Synonyme, Homonyme, Äquivalenzen
  - Datenqualität
    - Mehrdimensional, Messung sehr komplex
    - Konkurrierende Aspekte erfordern Kompromisse:
      - Genauigkeit
      - Vollständigkeit
      - Zeitbezogene Aspekte
      - Konsistenz
    - Ca. 15-20% der Datenwerte einer typischen Kunden-DB sind falsch

    - Schlechte Datenqualität führt zu Kundenbeschwerden, Vertrauensverlust, falsche Zielgruppen ➔ Image-Verlust
    - Vielzahl an kommerziellen und freien Tools verfügbar, kein „all-in-one“-Tool
- OLAP, OLTP, Portal
    - OLAP:
        - „Online Analytical Processing“
        - Analyse von Unternehmensdaten in Echtzeit
        - Zusammenfassung mehrdimensionaler Daten
        - Vergangenheitsorientiert
    - OLTP:
        - „Online Transaction Processing“
        - Gleichzeitige Verarbeitung von Transaktionen vieler Benutzer
        - Gegenwartsorientiert
    - Portal:
        - Zentraler Einstiegspunkt für einen Benutzer, um ihm Zugang zu weiteren Informationen zu gewähren
        - Unterscheidung in: Horizontale Portale (über versch. Themengebiete), Vertikale Portale (Themen- und Fachportale)
        - Funktionen: Personalisierung, Navigation, Workflow-Komponenten, Integration von Anwendungen von Informationsquellen

3. Management der Informationssysteme

3.1. Referenzmodelle

- Arten und Beispiele

    - Def.: Referenmodell ist Modell, welches allgemeingültigen Charakter haben soll, dient als Ausgangspunkt für unternehmensspezifische Modelle
    - Arten: Ebenen-/Architekturmodelle, Metamodelle
    - Beispiele. SAP, ISO/OSI
    - Metamodelle: UML, ARIS, MDA

- Rolle bei der Softwareeinführung

    - Alternative 1: künftige Systeme nur auf Basis des Referenzmodells spezifizieren; setzt voraus, dass alle Funktionalitäten beschrieben sind
    - Alternative 2: Vergleich eigener Modelle mit dem Referenzmodell der Software; Kompromiss aus Anpassung der Software und Anpassung des Unternehmens – Prozess-Priorität bestimmt die Entscheidung

- ARIS (Schwerpunkt: Sichtenkonzept)

    - Vier Sichten
    - Organisationssicht: welche Orga-Einheiten existieren?
    - Datensicht: welche Informationen sind relevant?
    - Funktionssicht: Welche Funktionen werden durchgeführt?
    - Steuerungssicht: Zusammenhang zwischen Daten, Funktionen und Orga-Einheiten (Prozesse)

3.2. Datenmanagement

- Aufgaben des Datenmanagement

  - Datenmodellierung
  - Datenadministration
  - Datentechnik
  - Datensicherheit / Datenschutz
  - Datenkonsistenz
  - Sicherung von Daten
  - Datenbezogene Benutzerservices

- Rolle von Datenbanksystemen

  - Rahmen für das Datenmanagement, elementarer Bestandteil
  - ???

- Data Dictionary Systeme

  - Enthalten Meta-Informationen über die im DBS enthaltenen Daten und Anwendungsprogramme
  - Dienen der Konsistenzüberwachung eines Datenbestandes
  - Überprüfung auf Redundanz- und Widerspruchsfreiheit

3.3. Prozessmanagement

- Aufgaben des Prozessmanagement

  - Business Process Reengineering: Gestaltung oder Reorganisation betrieblicher Abläufe
  - Optimierung der Prozesse zur Effizienzsteigerung innerhalb eines Unternehmens

- Definition Geschäftsprozess (Business Process)

  - Zielgerichtete Abfolge von Aufgaben, die von mehreren Organisationseinheiten ausgeführt werden
  - Erstellung von Leistungen entsprechend vorgegebener Prozessziele
  - Max. Detaillierungsgrad, wenn die Aufgaben je in einem Zug von einem Mitarbeiter an seinem Arbeitsplatz ausgeführt werden können

- Klassifikation von Geschäftsprozessen

  - Kerngeschäftsprozesse:
    - Hoher Wert für den Kunden
    - Von Kundenwunsch bis Auslieferung
    - Beispiele: Auftragsbearbeitung, Produktion
  - Unterstützungsprozess:
    - Geringer Wert für den Kunden
    - Nicht wettbewerbskritisch
    - Beispiele: FiBu, Kostenrechnung, Personalwesen

- Geschäftsprozessoptimierung

  - Reihung von Funktionen: sequentiell, parallel, wiederholte Ausführung, Verzweigung bei bestimmten Bedingungen → Durchlaufzeitverkürzung angestrebt

- Verwendung von Referenzmodellen überlegen

- Beurteilung von Prozessen

  - Ziel: Kundenzufriedenheit
  - Qualität des Prozesses: entspricht Ergebnis dem Ziel?
  - Durchlaufzeit: Durchschnittswerte, Minima/Maxima
  - Kosten: für einzelne Prozesselemente
  - Unterschiedliche Gewichtung dieser Kriterien

- Definition Workflow

  - Formal beschriebener, ganz oder teilweise automatisierter Geschäftsprozess
  - Einzelne Arbeitsschritte hierbei zur Ausführung durch Mitarbeiter oder Anwendungsprogramme vorgesehen

- Typen von Workflows

  - Allgemeiner Workflow: (z.B. Reisekostenabrechnung)
    - Vollständig strukturierbare Arbeitsabläufe
    - Viele wiederholbare Elemente
    - Arbeitsschritte im Voraus definiert
  - Fallbezogener Workflow: (z.B. Schadensbearbeitung, Reklamationen)
    - Nur teilweise wiederholbare Elemente
    - Benutzer kann/muss entscheidend mitwirken
    - Nicht alle Arbeitsabläufe vorweg definierbar
  - Ad-hoc Workflow: (z.B. Entwicklung eines Marketingkonzepts)
    - Kaum Wiederholungen
    - Benutzer müssen das meiste selbst regeln
    - Arbeitsschritte nicht im Voraus definierbar
    - Nicht modellierbar

- Geschäftsprozess vs. Workflow

Geschäftsprozess	Workflow
Ziel: Was ist zu tun?	Ziel: Wie ist es zu tun?
Konzeptionelle Ebene, Verbindung zur Geschäftsstrategie	Operative Ebene, Verbindung zur unterstützenden Technologie
Elementare Arbeitsschritte	Konkretisierung von Arbeitsschritten

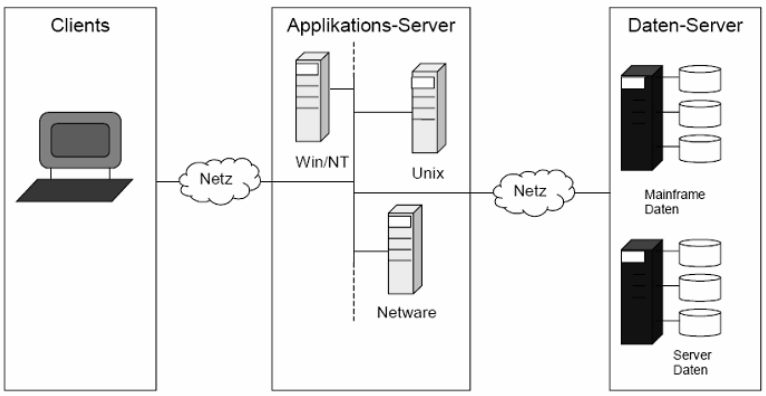
4. Verteilte Systeme, Client/Server-Systeme

- Definition Verteiltes System

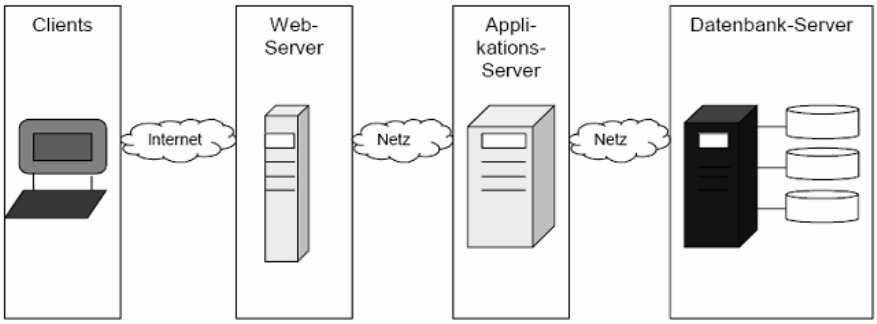
  - Menge von Funktionseinheiten oder Komponenten, die in Beziehung zueinander stehen und eine Funktion erbringen, die nicht durch die Komponenten allein erbracht wird.

- Technische Architektur eines Informationssystems (Tiers)

  - 3 Tier Modell:

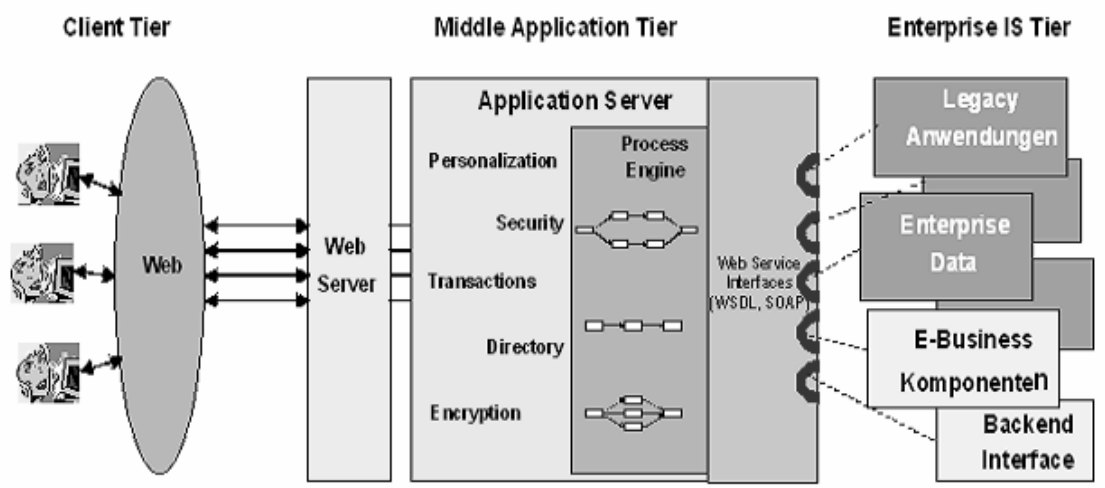


  - 4 Tier Modell:



- Middleware, Application Server

  - Architektur eines IS:



  - Middleware:
    - in der Mitte zwischen Betriebssystemschicht und darauf basierender Anwendungen
    - Softwareschicht auf Basis von Standard-Protokollen, die Transparenz in der Kommunikation verteilter Anwendungen bereitstellt
    - ermöglicht Zugriff auf lokale und entfernte Dienste

- Interaktionsarten in C/S-Systemen

- Synchrone Kommunikation: Client wartet auf Server-Antwort
- Zurückgestellte synchrone Kommunikation: Client arbeitet weiter, prüft periodisch auf Server-Antwort
- Asynchrone Kommunikation: Client arbeitet weiter, Server ruft Event bei Client auf wenn Antwort zu senden ist
- One-Way-Kommunikation: Client arbeitet weiter, erhält keine Rückantwort

- Fehlerbehandlung (z.B. at least once)

  - Fehler behandeln: Anforderung oder Rückantwort verloren, Server oder Client abgestürzt
  - At least once: Client hat Timeout, sendet erneut wenn abgelaufen und keine Antwort; Problem: Server bearbeitet gleiche Anfrage mglw. mehrfach
  - At most once: Client hat Timeout, sendet erneut wenn abgelaufen und keine Antwort; Server trägt Client-Requests in Request-Stack ein, erwartet ACK vom Client auf Rückantwort – verwirft Requests, die im Stack mehrfach vorhanden sind ohne ACK
  - Exactly once: Ausführung genau einmal, auch bei Systemfehlern; Request-Liste in stabilem Speicher halten, Plattenausfall darf sich nicht auswirken

- Serveraktivierung

  - Shared Server: Serverprozess für mehrere Services; explizit starten wenn nicht aktiv
  - Unshared Server: pro Service ein anderer Server
  - Per Request Server: Start eines Servers bei Service Request
  - Persistent Server: wie shared, aber Serverprozess dauerhaft aktiv

- Stateless vs. Stateful Server

  - Stateless Server: Client Requests erzeugen keinen neuen Server-Zustand, z.B. FTP, HTTP
  - Stateful Server: Client Request überführt Server in neuen Zustand, bei Absturz gehen allerdings Zustandsinformationen verloren

- Caching-Problem in C/S-Systemen

  - Problem: Daten in Cache/HS müssen auf Platte ausgelagert werden
  - Lösung: Ersetzungsalgorithmen:
    - LRU: „least recently used“, Stack-Prinzip
    - Dirty Bit: setzen, wenn Daten im Cache modifiziert; überschreiben wenn nicht gesetzt, sonst vorher auf Platte zurückschreiben

- Verteilte Prozesse

  - Nachteil von C/S-Systemen: verminderte Ausfallsicherheit, Flaschenhalsproblem durch zentralen Server
  - Verteilter Prozess: Server kann auch Client sein, zusätzliche Koordinaten notwendig

- Client-Server-Server

  - Ein oder mehrere Clients
  - Mehrere Server, Zwischen-Server erfüllt spezielle Aufgabe

- Intermediate Server: Proxy, Broker, Trader, Filter, Balancer, Koordinator, Agent

  - Proxy:
    - Stellvertreter für mehrere Server
    - versucht Client Anfragen selbst zu beantworten, speichert Server Replies ab
  - Broker:
    - Vermittler zwischen Clients und Servern
    - Hat Informationen über Services und Server, welche diese anbieten
    - Arten:
      - intermediate (forwarding): leitet Client-Anfragen zum Server, Server-Antwort zum Client
      - separat (handle-driven): gibt Service-Handle an Client zurück, Client kommuniziert direct mit Server
      - hybrid
    - zentraler Broker nachteilig in Bezug auf Ausfallsicherheit
  - Trader:
    - Auswahl des am besten geeigneten Servers
  - Filter:
    - Filter für Anfragen und Antworten
    - Analysiert und bearbeitet Client-Anfrage, sendet modifizierten Request an Server
    - Analysiert/bearbeitet Server-Reply, sendet dann an Client zurück
  - Balancer:
    - Verteilung der Arbeitslast unter mehreren Servern
    - Wählt bei Client-Anfrage unbelasteten Server aus
  - Koordinator:
    - Koordinierte Bearbeitung von Einzelservices
    - Client-Anfrage wird in Anfrage an mehrere Services zerlegt, diese werden an die Server verteilt
  - Agent:
    - Ermittlung von Serveranfragen aus der Client-Anfrage unter Einsatz von KI-Techniken
    - Iterativ: Ergebnis eines Servers wird ausgewertet, so nächster Server bestimmt
    - Parallel: Versende Requests an Server parallel

5. Kommunikationsmechanismen

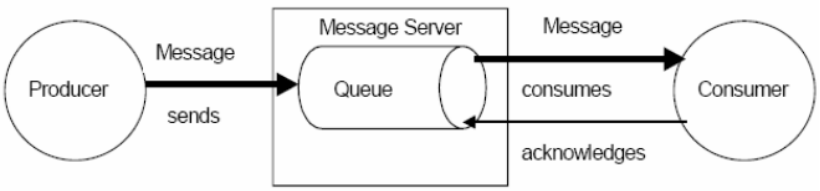
5.1. Nachrichtenbasierte Kommunikation

- Verbindungsorientiert vs. Verbindungslos

  - Verbindungsorientiert:
    - Über Stream-Sockets
    - Client macht connect() zum Server
    - Vorteil: Serverausfälle werden erkannt
    - Nachteil: dauert länger
  - Verbindungslos:
    - Über Datagramm-Sockets
    - Client macht kein connect(), sondern sendet ohne Verbindung an den Socket
    - Vorteil: schnelle Übertragung
    - Nachteil: „Schuss ins Blaue“ – man weiß nicht, ob der Server on ist

- Java Messaging Service (JMS)

  - Stellt zentralen Message-Server zur Verfügung
  - Vorteile: Client muss nicht auf Server-Antwort warten, lose Kopplung zwischen Sender und Empfänger
  - PTP, Publish/Subscriber
    - PTP: Clients (Producers) senden Nachrichten an virtuellen Kanal (Queue), diese hat nur einen Konsumenten



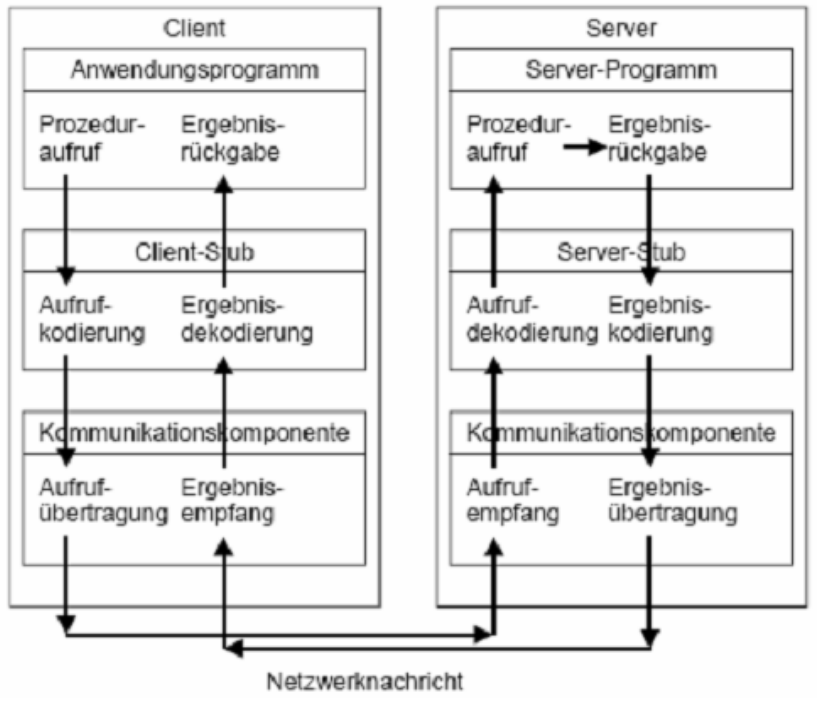
    - Pub/Sub: 1 Publisher sendet Nachricht an virtuellen Kanal (Topic), von dort aus wird die Nachricht an alle Subscriber übermittelt
  - Komponenten: Producer / Consumer, Destination, Listener
    - Producer: erzeugt die Nachricht
    - Consumer: empfäng und verarbeitet die Nachricht
    - Destination: hierüber spezifiziert Client den Bestimmungsort einer Nachricht
    - Listener: wartet auf das Eintreffen eines Ereignisses (event), d.h. auf das Reply eines Servers

5.2. Entfernte Prozeduraufrufe (RPC)

- Anwendung wird auf mehrere Rechner aufgeteilt
- Entfernte Prozeduraufrufe

- Komponenten- und Ablaufstruktur

  - Struktur:



- Parameter Marshalling

- Prozedurparameter werden in Nachricht verpackt und an den Server Stub gesendet

- XDR

  - eXternal Data Representation
  - Datenrepräsentationsstandard, von SUN entwickelt
  - maschinenunabhängig

- Identifikation und Binden der Aufrufpartner

  - Lokalisierung des Servers mit gesuchter Prozedur:
    - Statisches Binden: Durch direkte Angabe der Serveradresse
    - Dynamisches Binden: logischer Servername wird durch Broker (Binder) in physikalische Adresse umgewandelt
  - Bewertung:
    - Statisches Binden: unflexibel, abhängig von Netzwerkadressen
    - Dynamisches Binden: flexible Systemkonfiguration

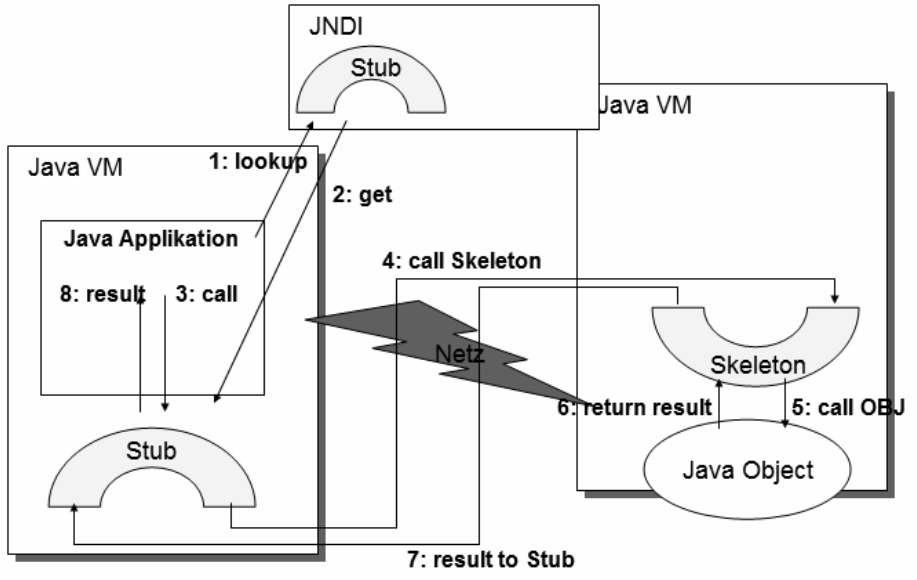
5.3. Objektbasierte Kommunikation

5.3.1. Java-RMI (Remote Method Invocation)

- Erstellung von verteilten Java-Anwendungen
- Java-Programm kann Remote-Objekt über Referenz aufrufen
- Sockets als Basis-Kommunikationsmechanismen

- Ablauf bei Kommunikation: Stub, Skeleton, JNDI

  - Überblick:



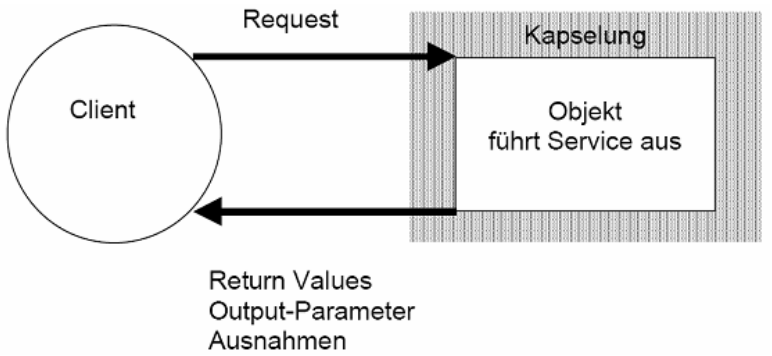
  - Ablauf:
    1. Client fragt nach Objektreferenz bei RMI-Registry (JNDI)
    2. Client erhält Objektreferenz auf Stub
    3. Client ruft Methode aus dem Stub auf
    4. Aufruf wird an Server übertragen, der eine Objekt-Deklaration besitzt, die dem Stub des Clients entspricht (Skeleton)
    5. über Skeleton wird das tatsächliche Java-Objekt angesprochen
    6. Resultat wird an Skeleton geliefert

7. Resultatübertragung von Skeleton an Stub
8. Resultatrückgabe vom Stub an die Java-Applikation

- Stub:
  - Für Client sichtbare Methoden eines aufzurufenden Objekts
  - Client erhält nach Lookup von JNDI eine Referenz auf einen Stub
- Skeleton:
  - Logische Abbildung des tatsächlichen Objektes auf dem Server, bietet Methoden nach außen an, versteckt tatsächliche Implementierungen
  - Entspricht Stub auf dem Client
- JNDI:
  - „Java Naming and Directory Interface“
  - Ablegen von Daten und Objektreferenzen anhand eines Namens
  - Registriert verteilte Objekte in einem Netzwerk, stellt sie Clients zur Verfügung

5.3.2. CORBA

- Client-Server-Prinzip
- Client sendet Anforderung für einen Service an einen Server
- Server besitzt Interface, welches in IDL spezifiziert ist (Interface Definition Language)
- Service wird ausgeführt, Ergebnis zurück an Client gesendet



- Wie arbeitet CORBA?

  - Koordination über ORB:
    - Liste von Objekt-Implementierungen
    - Vergabe von Objektreferenzen
    - Entgegennahme von Aufrufen vom Client und Transport/Übergabe der Aufrufe an Server
    - Entgegennahme von Ergebnissen und Transport/Rückgabe an Client

- Object Request Broker

  - Ermöglicht Kommunikation und Koordination zwischen beliebigen CORBA-Objekten
  - Semantik eines „Forwarding Broker“
  - Abschottung der Anwendung von Spezifika wie Hardware, Plattfor,, OS, Netzwerkprotokoll, Implementierungssprache etc.

- Interfaces (statisch vs. dynamisch), IDL

- Statische Interfaces:
  - IDL Stubs/Skeletons: beinhalten Funktionen, die zum Client-/Server-Programm hinzugebunden werden
- Dynamische Interfaces:
  - Dynamic Invocation Interface (DII): erlaubt Absetzen eines Objekt-Request zur Laufzeit; wenn Interface nicht bekannt → Anfrage an ORB
  - Dynamic Skeleton Interface (DSI): bindet zur Laufzeit Anfragen des ORB zu Objektimplementierung
- IDL:
- „Interface Definition Language“
- universelle Notation für Schnittstellen der Objektimplementierung
- rein deklarative Sprache, Untermenge von C++, erweitert um verteilte Konzepte
- sprachunabhängig, unterstützt C, C++, Java, …

- CORBA Services

  - Bieten gekapselte Funktionalität nach außen an
  - Lifecycle Services: Instanzenmanagement
  - Naming Service: Verwaltung von Objektnamen
  - Event Service: subscribe/unsubscribe für bestimmte Ereignisse
  - Concurrency Control Service: Lock Manager
  - Time Service: Zeit-Synchronisation
  - Transaction Service: 2-Phasen-Commit-Protokoll
  - Security Service: verteilte Objektsicherheit
  - Persistence Service: dauerhafte Speicherung auf Storage-Servern
  - Relationship Service: speichert Beziehungen zwischen Objekten
  - Externalization Service: E/A-Strom für Komponenten
  - Query Service: Datenbankanfragen
  - Properties Service: Assoziierung von benannten Werten zu Komponenten
  - Trader Service: erlaubt Objekten, ihren Service anzubieten
  - Collection Service: Interface für Collections

- Bewertung: RPC, RMI, CORBA

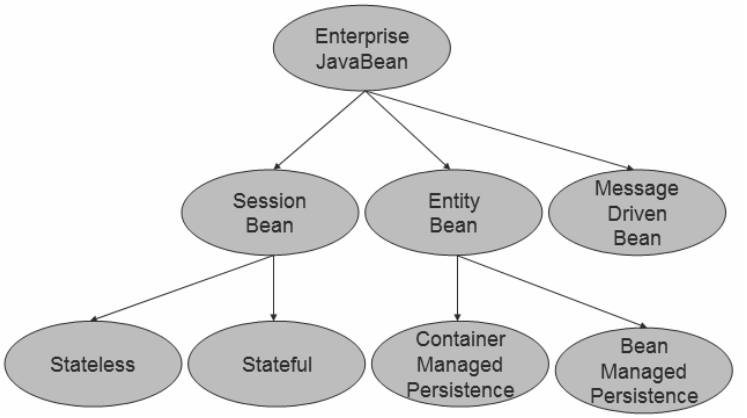
  - RPC:
    - Sprachenunabhängig: Stubs lassen sich für beliebige Sprachen generieren
    - Prozessorunabhängig: durch Konversion von RPC in XDR
    - Betriebssystemunabhängig: durch unterschiedliche RPC-Systeme für versch. OS
    - Aufwendig zu programmieren, einfache Aufrufe
  - RMI:
    - Basiert auf Java, java-gebunden
    - Plattformunabhängig, erfordert JVM (allerdings i.A. keine Einschränkung)
    - Nur entfernte Java-Objekte ansprechbar
    - Objektbasierte Kommunikation
  - CORBA:
    - Verbirgt Abhängigkeiten durch ORB
    - Sprachabhängige Interface Definition Language (IDL)
    - Objektbasierte Kommunikation

5.4. Komponentenbasierte Kommunikation

- Enterprise Java Beans (EJB)

- Vereinfachen Entwicklung mehrschichtiger verteilter Softwaresysteme in Java
- Komponentenarchitektur für Geschäftsanwendungen
- EJB-Applikationen sind skalierbar, mehrbenutzerfähig, serverunabhängig und transaktionsorientiert
- Keine Veränderung für Client bei Verteilung auf mehrere Server
- Unabhängige Entwicklung möglich durch Rollenkonzept
- Fokus auf Wiederverwendbarkeit und effizienter Nutzung

- Arten von Enterprise Java Beans: Entity Beans, Session Beans, Message-Driven Beans



  - Session Bean:
    - Bilden Vorgänge ab, die der Benutzer mit dem System durchführt
    - Implementieren eigentliche Geschäftslogik, greifen auf Entity-Beans zu
    - Lebensdauer maximal der Lebensdauer des Clients
    - Vorteil gegenüber eigener Implementierung: Netzlast gering
  - Entity Bean:
    - Modellieren dauerhafte Daten (Entitäten) eines Systems
    - Direkter Bezug zur DB
    - Können von mehreren Clients gemeinsam genutzt werden
  - Message Driven Bean:
    - Machen EJB-Systeme für asynchrone Kommunikation zugänglich
    - Werden nie direkt von der Anwendung angesprochen
    - Aktivierung über Container bei Eintreffen neuer Nachrichten

- EJB-Container, EJB-Server

  - EJB-Container:
    - Mittelschicht zwischen Bean-Klasse und Server
    - Verwaltet EJB-Objekte
    - Aufgaben: Transaktionen, Sicherheit, Nebenläufigkeit, Persistenz
  - EJB-Server:
    - Namensdienste
    - Arbeitet eng mit Container zusammen
    - Thread-/Prozess-Management
    - Lastverteilung
    - Ausfallsicherheit
    - Zugriff auf Ressourcen

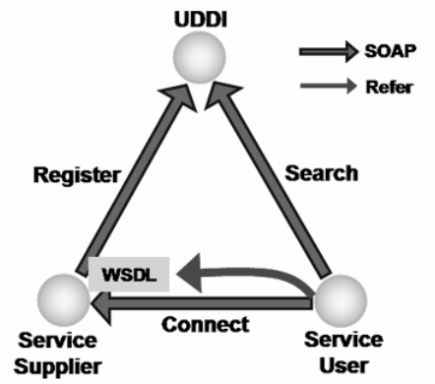
5.5. Web Services

- Software-System für Rechner-zu-Rechner-Interaktionen über ein Netzwerk

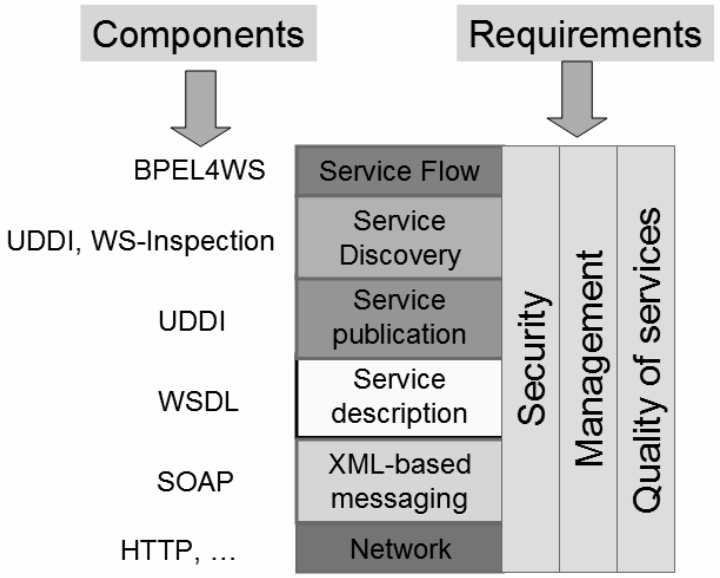
- Software, welche von anderer Software über Internet-Protokolle benutzt wird

- SOAP, WSDL, UDDI und deren Zusammenhang

  - SOAP: XML-basiertes Nachrichtenformat für verteilte Anwendungen und dessen Einbettung in ein Transportprotokoll, Austausch strukturierter und typisierter Daten
  - WSDL: XML-basierte Beschreibungssprache, um Web Services zu beschreiben
  - UDDI: Verzeichnisdienst für Web Services; spezifiziert standardisierte Verzeichnisstruktur für die Verwaltung von Metadaten von Web Services
  - Zusammenhang:



- WS-Stack

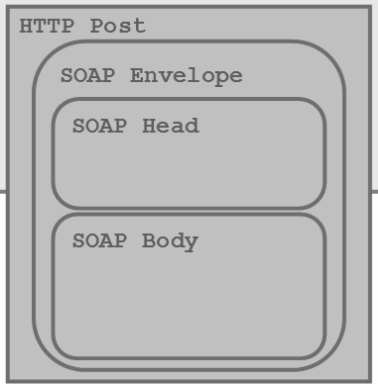


- Prinzipieller Aufbau einer SOAP-Nachricht

```
<soap:Envelope     soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">

<soap:Header>
        <!--Der Header ist optional -->
    </soap:Header>

    <soap:Body>
        <!-- Serialisierte Objektdaten -->
    </soap:Body>
</soap:Envelope>
```



- WSDL
  - Welche Datentypen gibt es?: types
    - <types /> Element
  - Welche Funktionalität besitzt der Dienst: message / interface
    - <message /> Element: abstrakte Definition der auszutauschenden Daten
    - <interface /> Element: Menge von Operationen
  - Wie wird kommuniziert? – SOAP binding
    - <binding /> Element: beschreibt Abbildung eines Interfaces auf konkretes Datenübertragungsprotokoll
  - Wo befindet sich der Dienst? – endpoint
    - <service /> Element: Sammlung zugehöriger Endpunkt

6. Geschäftsprozessmodellierung und –management (BPEL)

- Basiskonzepte und Basiselemente

  - Basiskonzepte:
    - Imperative, block-strukturierte Programmiersprache
    - Modellierung von Geschäftsprozessen
    - Standardisierte und semantisch klar definierte Sprachelemente
    - Programmcode als XML-Dokument
    - Plattformunabhängigkeit
    - Erweitert das WSDL-Konzept
  - Basiselemente:
    - Generisches Basiselement „Activity“
    - Elemente für Nachrichtenaustausch (receive und reply)
    - Aufruf von WSDL-Operatoren (invoke)
    - Basissteuerung der Reihenfolge durch sequence/switch/while/flow/pick
    - Prozessdatenspeicherung in Variablen möglich
    - Fehlerbehandlung (catch)

- Daten-Handling, Prozesszustand (Variable)

  - Jeder Prozess zustandsbehaftet
  - Zustand = Nachrichten + temporäre Daten
  - Speicherung instanzspezifischer Prozessdaten in Variablen

- Nachrichtenaustausch (receive, reply)

  - receive: blockierendes Warten des BusinessProcess bis Nachricht ankommt, Input wird in Prozessvariable eingelesen
  - reply: BusinessProcess kann Nachricht in Antwort eines receive senden

- Kontrollflusselemente

  - sequence, switch, while: steuern den sequentiellen Ablauf von Aktivitäten
  - flow: beschreibt Fluss parallel oder synchronisiert ablaufender Aktivitäten
  - pick: legt Aktivitäten fest, die aufgrund externer Ereignisse ausgeführt werden

- Bedeutung von Links

  - Zur Synchronisation von Aktivitäten
  - BusinessProcess blockiert solange, bis alle Bedingungen eines Links erfüllt sind
  - Graphentheoretisch: Verknüpfung zweier Aktivitäten in zeitlicher Abfolge, welche sich in unterschiedlichen Verschachtelungs-Ebenen befinden
  - Nötig, da Modellierung über sequence und flow hier nicht möglich ist

- Service-Aufrufe (invoke)

  - invoke: erlaubt einem BusinessProcess eine Operation auf einem Port des Kommunikationspartners zu starten

- Fehlerbehandlung

  - catch: Abfangen von Fehlern
  - catchAll: Default-Handler, wenn kein catch auf den geworfenen Fehler zutrifft
  - throw: Erzeugen/Werfen von Fehlermeldungen

- Kompensation

  - Im Fehlerfall sollen Operationen angewendet werden, welche die Effekte bereits abgeschlossener Aktivitäten möglichst minimieren („kompensieren“)
  - Vorteil: anwendungsorientierte Fehlerbehandlung
  - Nachteile: erheblicher Zusatzaufwand, kaskadierende Kompensation möglich

7. Integration heterogener Datenbanken

- Arten der Heterogenität von Informationssystemen

  - Heterogenität: Unterschiedlichkeit von miteinander verbundenen IS
  - Heterogenität überbrücken ist Kernaufgabe der Integration
  - Arten:
    - Technische Heterogenität
    - Datenmodellbasierte Heterogenität
    - Logische Heterogenität
      - Semantische Heterogenität

        - Schemabasierte Heterogenität
        - Strukturelle Heterogenität

- Autonomie, Arten von Autonomie

    - Grad, zu dem verschiedene DBMS unabhängig kooperieren können
    - Problem: Beschneidung von Kompetenzen und Verantwortung einzelner Systemverantwortlicher vonnöten
    - Arten:
        - Design-Autonomie
        - Ausführungsautonomie
        - Kooperationsautonomie
        - Unabhängigkeit bzgl. der Wahl des DBMS
        - Unabhängigkeit bzgl. der Wahl der Ablaufumgebung

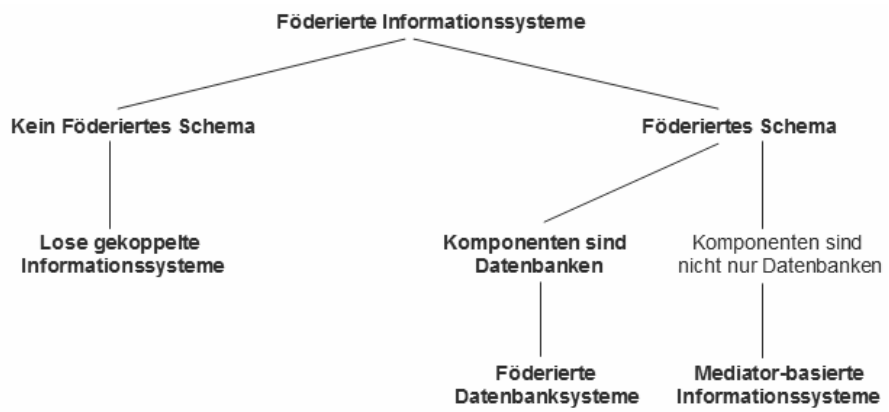
- Begriff der Föderation

    - Föderation ist Zusammenschluss von Systemen, bestimmt von Grad der verbleibenden Autonomie und Heterogenität der beteiligten Systeme

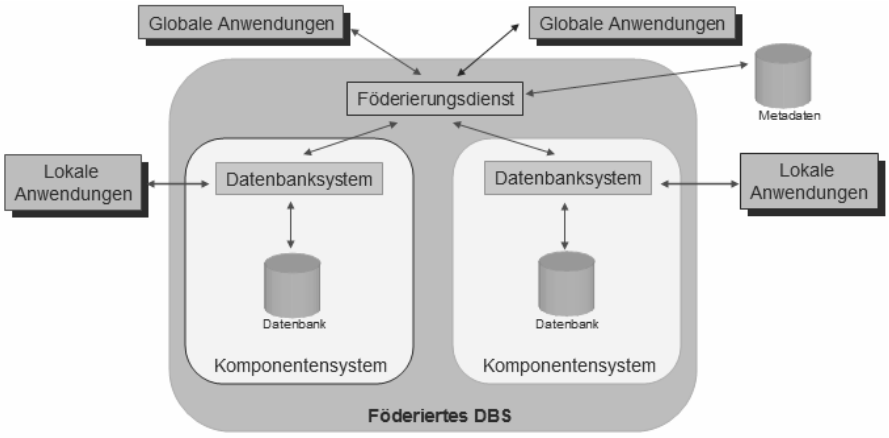
- virtuelle vs. materialisierte Integration

    - materialisiert:
        - Anwendung greift nur auf zentrale Datenbasis zu
        - Zentrale Datenbasis importiert periodisch die Daten der zugrunde liegenden Datenquellen
        - Nachteile: Redundanz, Konsistenzprobleme, Übertragungsprobleme, hohes Datenaufkommen und Transfervolumen
    - virtuell:
        - Anfragen werden an die einzelnen tatsächlichen Datenquellen verteilt
        - Nachteile: erfordert viel Wissen über alle zugrunde liegenden DBs, erfordert viel Arbeit beim Absetzen der einzelnen Anfragen und Zusammenführen der Ergebnisse

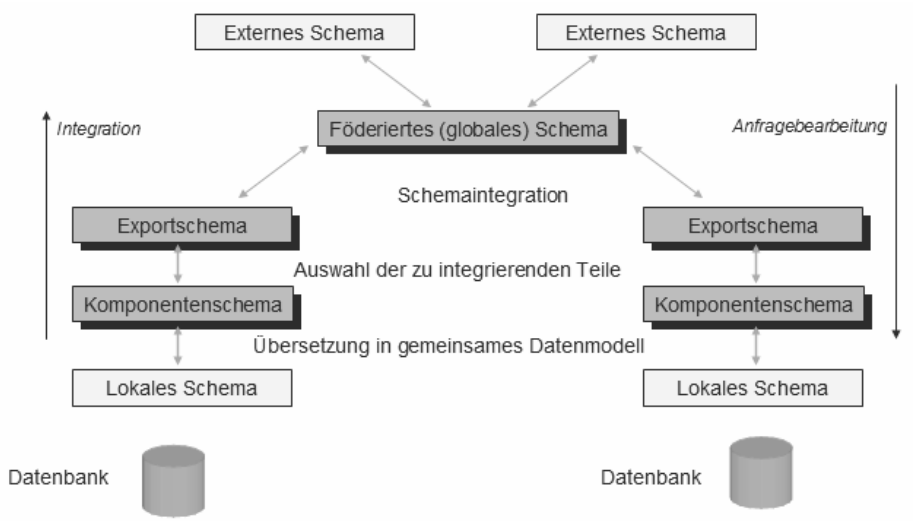
- Typen von föderierten Informationssystemen



- Systemarchitektur föderierter Informationssysteme



- 5-Ebenen-Schema-Architektur



- Klassifizierung von Integrationskonflikten: Datenmodell-K., Schema-K., Daten-K.

  - Datenmodell-Konflikte:
    - Unterschiedliche Semantik und Struktur von Datenmodellen
    - Vielzahl von Datenmodellen mit unterschiedlichen Modellierungskonstrukten: XML, ERD, Hierarchien, Objektrelational
    - Konstrukte von Datenmodellen werden oft falsch oder unvollständig verwendet
  - Schema-Konflikte:
    - Unterschiedliche Modellierung gleicher Sachverhalte
    - Strukturelle Konflikte: Modellierung (z.B. Relation vs. Attribut), Benennung (Relationen, Attribute), …
    - Tabellen-Konflikte: gleiche Namen, aber unterschiedliche Bedeutung; oder auch fehlende Attribute
    - Attribut-Konflikte: gleiche Namen, andere Bedeutung, andere Default-Werte, Datentypkonflikte, Konflikte bei Constraints
  - Daten-Konflikte:
    - Inkorrekte Einträge: Tippfehler, falsche Einträge durch Programmierfehler
    - Veraltete Einträge: vergessene Aktualisierungen, abweichende Aktualisierungszeitpunkte
    - Unterschiedliche Repräsentation von Werten: Datentypen, Schreibweisen, Genauigkeit, Skalierung
    - Behebung:

        - Explizite Werteabbildungen
        - Ähnlichkeitsmaße
        - Bevorzugung von Werten aus lokaler Quelle
        - Verwendung von Hintergrundwissen, z.B. Wörterbücher, Ontologien, Thesauri zur Behandlung von Homonymen/Synonymen
    - Beschreibungskonflikte:
        - Homonyme, Synonyme
        - Datentypen,
        - Skalierungen, Genauigkeiten
        - Integritätsbedingungen

- Semantische Aspekte der Integration

    - Problem ist unterschiedliche Bedeutung, Interpretation und Art der Nutzung von Daten
    - Annahme „gleiche Bezeichnung, gleiche Semantik“ gilt nicht
    - Abwägen, wann Informationsintegration möglich ist und wann nicht
    - Integration erfordert viel Handarbeit und Überblick

- Phasen des Integrationsprozesses (virtuelle vs. materialisierte Integration)

    - Integrationsprozess:
        - Bildung eines globalen Schemas
        - Generierung von Wrappern für jede Datenquelle
        - Daten bleiben vor Ort, Informationsquellen sind autonom
    - Phasen:
        1. Analyse der zu integrierenden Daten
        2. Transformation der Datenbankschemata in ein gemeinsames Datenmodell
        3. Feststellung der sich semantisch entsprechenden Daten
        4. Ableitung eines integrierten Schemas
        5. Integration der Daten

8. Anwendung der Konzepte auf Informationssysteme in der Praxis

- Anforderungen identifizieren und Technologien bewerten / vorschlagen