Hochschule Luzern – Technik & Architektur

IGL.HS10

Stoffzusammenfassung

Luzern, 09.01.2011

Projekt	MEP bestehen ;)
Dokument	Stoffzusammenfassung
Schule	Hochschule Luzern Technik & Architektur
Modul	Informatikgrundlagen (IGL)
Ersteller	**Flavio De Roni** http://playground.mrf2thed.ch
Letzte Änderung	09.01.2011 15:17:00

Änderungshistorie

Version	Datum	Autor	Änderung	Status
0.0	18.12.2010	Flavio De Roni	Dokument erstellt	Fertig
0.1	18.12.2010	Flavio De Roni	Lernziele ergänzt	Fertig
0.2	19.12.2010	Flavio De Roni	Kapitel 5 Programmierung fertig gestellt	Fertig
0.3	22.12.2010	Flavio De Roni	Kapitel 2 Internet fertig gestellt	Fertig
0.4	22.12.2010	Flavio De Roni	Kapitel 6 Datenbanken fertig gestellt	Fertig
0.5	23.12.2010	Flavio De Roni	Kapitel 1 Einführung Informatik fertig gestellt.	Fertig
0.6	23.12.2010	Flavio De Roni	Kapitel 3 Teil Geschäftsprozesse fertig gestellt	Fertig
0.7	24.12.2010	Flavio De Roni	Kapitel 3 Teil ERP-Systeme fertig gestellt	Fertig
0.9	24.12.2010	Flavio De Roni	Kapitel 4 Rechnersysteme und Betriebssysteme fertig gestellt	Fertig
1.0	24.12.2010	Flavio De Roni	Version 1 erstellt	Fertig

Lernziele

gemäss Modulbeschreibung
(http://www.hslu.ch/download/t/t&a_bachelormaster/t_informatikgrundlagen.pdf) :

Informatik Einführung

- Teilgebiete der Informatik
- Grundlagen Geschäftsprozesse
- Internet

ERP-Systeme

- Informationstechnische Unterstützung von Geschäftsprozessen verstehen.
- Eigenschaften konkreter Enterprise Ressource Planning Systeme (ERP-Systeme) benennen können.
- Grundfunktionen eines ERP-Systems kennen.
- Vorgehen bei Planung und Einführung eines Informationssystems erklären können.

Rechnerarchitekturen

- Zahlensysteme, wichtige Zahlenformate und die Grundoperationen der Bool'schen Algebra anwenden können.
- Grundlegende Hardware-Architekturen von Computersystemen, sowie Schnittstellen und Peripheriegeräten bezeichnen und einordnen können.
- Das Funktionieren eines von Neumann-Rechners erklären können.
- Aufbau und Funktionen eines Betriebssystems kennen.
- Einsatzmöglichkeiten von Client/Server Systemen kennen.

Programmieren

- Das Konzept von Compilern und Interpretern erklären können.
- Konzepte moderner Hochsprachen verstehen und Anwendungsbereiche kennen.
- Konzepte der objektorientierten Programmierung kennen und an einfachen Beispielen anwenden können.
- Einfache Beispiele in Java implementieren können.

Datenbanken

- Funktion und Einsatz von Datenbanken verstehen.
- Einfache Datenbank Aufgaben lösen können (Entity-Relationship Modell, SQL).
- Transaktionskonzept auf Stufe Datenbank erklären können.
- Aufgaben zur Datenbank-Administration kennen.

Inhaltsverzeichnis

1 Einführung in die Informatik 7
1.1 Informatik 7
1.1.1 Teilgebiete der Informatik 7
1.1.2 Bedeutung/Zweck der IT im Unternehmen 8
1.1.3 Ziele der rechnergestützten Informationsverarbeitung 9
1.2 Prozess-, Funktions-, Datensicht 9
1.2.1 Prozesssicht (Wie?) 9
1.2.2 Funktionssicht (Womit?) 9
1.2.3 Datensicht (Was?) 9
1.3 Aufbau eines Computersystems 10
1.3.1 Hardware 10
1.3.2 Software 10
1.3.3 verteilte Applikation 10
1.3.4 digitale Daten 10
2 Internet 12
2.1 Grundlagen 12
2.1.1 Technologie-Konvergenz 12
2.1.2 Gliederung der Informationsinfrastruktur 12
2.1.3 Bedeutung des Internets für neue Geschäftsmodelle 13
2.1.4 Einbindung und Schwerpunkte von Informationssystemen in Aktivitäten von Firmen 13
2.1.5 Strategie und Schwerpunkte für die Informationsgesellschaft Schweiz 13
2.1.6 Mitspieler (Rollen) im Informations- und Kommunikationsgeschäft 14
2.1.7 Wichtige Parteien 14
2.2 Die Technik 15
2.2.1 Protokoll-Stapel 15
2.2.2 IP 16
2.2.3 TCP 16
2.2.4 UDP 17
2.2.5 DNS 17
2.2.6 HTTP 17
2.2.7 E-Mail 17
2.2.8 Routing 17
2.2.9 Schichtenmodell der Telekommunikation 18
2.3 Sicherheit 18
2.3.1 Arten von Sicherheit 18
2.3.2 Bedrohungen 19
2.3.3 Sicherheitsmassnahmen 19
2.3.4 Private-Key-Verfahren 19
2.3.5 Public-Key-Verfahren 20
2.3.6 Funktionsweise von Digitaler Signatur und Message Digest (Hash) 20
2.3.7 Security Policy 21
2.3.8 Steganografie 21
2.3.9 Massnahmen zur Computersicherheit 21
3 Geschäftsprozesse & ERP-Systeme 22
3.1 Geschäftsprozesse 22
3.1.1 grundlegende Prozesse eines Industrieunternehmens 22
3.1.2 wichtigste aktuelle Herausforderungen an die Kernprozesse 23
3.2 IT-Systeme zur Prozessunterstützung 23
3.2.1 kritische Leistungsmerkmale moderner IT-Systeme 23
3.2.2 historische Entwicklung des Geschäftsbedarfs nach IT-Unterstützung 24
3.3 Überblick Business-Software 24
3.3.1 Umfeld „Business Software“ 25
3.3.2 wichtige Begriffe aus dem Bereich Business Software 26
3.3.3 Marktüberblick & Einblick in marktgängige ERP-Systeme 27
3.4 ERP-Systeme 27
3.4.1 Bestandteile eines ERP-Systems 27
4 Rechnerarchitekturen 29
4.1 Zahlensysteme und Logik 29

4.1.1 digital vs. analog ... 29
4.1.2 der Begriff: Bit ... 29
4.1.3 Zahlensysteme ... 29
4.1.4 2er-Komplement ... 30
4.1.5 Gleitkommazahlen (IEEE754-Format) ... 30
4.1.6 bool'sche Operationen ... 31
4.1.7 Addierer und FlipFlop ... 31
4.2 Rechnersysteme ... 32
4.2.1 Von-Neumann und Harvard-Architektur ... 32
4.2.2 Funktion einer CPU ... 32
4.2.3 Verarbeitungsablauf auf einem Von-Neumann-Rechner ... 33
4.2.4 Lesen und Schreiben einer Speicherzelle ... 33
4.2.5 Moor'sches Gesetz ... 33
4.2.6 Speicherhierarchie ... 33
4.3 Betriebssysteme ... 34
4.3.1 Architekturmodell ... 34
4.3.2 Aufgaben und Funktionsweise ... 34
4.3.3 Prozess, Prozesszustände ... 34
4.3.4 Aufgabe eines Schedulers ... 35
4.3.5 Hauptaufgaben des Speichermanagements ... 35
4.3.6 Konzept der virtuellen Adressen ... 36
4.3.7 grundsätzlicher Aufbau eines Filesystems ... 36
4.3.8 programmierter IO vs. Interrupt IO ... 36
4.3.9 Dateioperationen inkl. Funktion ... 37
4.3.10 Funktionsweise einer verketteten Liste ... 37
4.3.11 Client-Server-Konzept in einer Dreistufen-Architektur ... 37
5 Programmierung ... 38
5.1 Grundlagen ... 38
5.1.1 Programmiersprachen und Programmierparadigma ... 38
5.1.2 Java-Compiler (javac) ... 39
5.1.3 Java Virtual Machine (JVM) ... 39
5.1.4 Entwicklungszyklus Editieren – Kompilieren – Ausführen – Testen ... 39
5.1.5 Unterschied .java und .class Dateien ... 40
5.2 Objektorientierte Programmierung (OOP) ... 41
5.2.1 Klasse vs. Objekt ... 41
5.2.2 Aufbau einer Klasse ... 41
5.2.3 Operatoren für float / int und Stringkonkatenation ... 42
5.2.4 Datentypen ... 43
5.2.5 der Begriff: Interface (WAS vs. WIE) ... 43
5.2.6 Schleifen und Fallunterscheidungen (Erklärung und Code) ... 44
5.2.7 Zinsberechnung (Code) ... 44
5.2.8 einfache Verarbeitung an gegebenem Array (Code) ... 46
6 Datenbanken ... 47
6.1 Entity-Relationship-Model (ERM) ... 47
6.1.1 Kardinalitäten-Angabe ... 47
6.2 Tabellendefinition (CREATE TABLE) ... 48
6.2.1 Code ... 48
6.2.2 weitere Befehle (INSERT, UPDATE, DELETE, DROP…) ... 48
6.2.3 Data-Definition und Data-Manipulation-Language ... 49
6.3 Datenbank-Abfragen (SELECT) ... 49
6.4 Transaktionskonzept auf Stufe Datenbank ... 50

1 Einführung in die Informatik

Lernziele
• Sie wissen, was Informatik ist und kennen die Bedeutung für Unternehmen. • Sie kennen die die Teilgebiete der Informatik und können diese erläutern • Sie können den Zweck der IT im Unternehmen aufzeigen und erklären • Sie kenne die drei Hauptsichten – Prozesse, Funktionen und Daten – der IT und können diese erläutern und kennen Beispiele • Sie kennen den grundsätzlichen Aufbau (Hardware) eines Computersystems • Sie kennen den grundsätzlichen Aufbau (Schichtung der Software) auf einem Computersystem und den Aufbau einer verteilten Applikation • Sie wissen, was digitale Daten sind, kennen Beispiele und können die Art und Weise der Speicherung erläutern

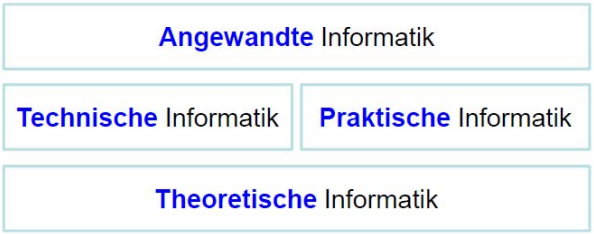
1.1 Informatik

Informatik ist eine Wortneubildung aus den beiden Begriffen **Infor**mation und Auto**matik** und bedeutet das systematische Verarbeiten von Informationen, insbesondere die automatische Verarbeitung mithilfe von Rechenanlagen.

Informatik umfasst allgemein die automatisierte Informationsverarbeitung in Natur, Technik und Gesellschaft. Informatik ist nicht sichtbar, steuert aber alles.

„In der Informatik geht es genauso wenig nur um Computer wie in der Astronomie um Teleskope!“ (Edsger W. Dijkstra, 1930 – 2002)

1.1.1 Teilgebiete der Informatik



1.1.1.1 Theoretische Informatik

→ ca. 90% Mathematik

Basis für die technische, praktische und angewandte Informatik.
Sie beinhaltet:

- Automatentheorie und formale Sprachen
- Berechenbarkeitstheorie
- Komplexitätstheorie

1.1.1.2 Technische Informatik

→ Hardware

Die technische Informatik befasst sich mit Hardware als Grundlage für die Informatik:

- Aufbau von Computer- und Speichersystemen

- Mikroprozessorentechnik
- Rechnerarchitekturen
- Netzwerken und Rechnerkommunikation

1.1.1.3 Praktische Informatik

→ Software

Die praktische Informatik befasst sich mit der Software (Programmen) als Grundlage für die Informatik.

- Programmiersprachen, Compiler, Programmierung
- Algorithmen und Datenstrukturen
- Betriebssysteme (z.B. Windows, Linux etc.)
- Datenbanken

1.1.1.4 Angewandte Informatik

→ Anwendungen

Die angewandte Informatik befasst den Informatikanwendungen mit Software-Applikationen und Hardware-Produkten. Folgende Themenbereiche gehören dazu:

- Wirtschaftliche, kommerzielle Applikationen
  - Banken-Software
  - Buchhaltungs-Software
  - ERP-Systeme
  - Büro-Software (z.B. MS Office)
  - HW-Produkte (z.B. Mobiles)
- Wirtschaftsinformatik
  - befasst sich mit der Planung, Entwicklung und dem Betrieb von Informationssystemen, vor allem in Dienstleistungsunternehmen
- Technisch-wissenschaftliche Applikationen.
  - Simulationen
  - Steuerungen (Verkehr, Strom-Netz etc.)
- Computervisualistik
  - beschäftigt sich mit Bilderzeugung, Bildverarbeitung und Bildgestaltung in den Bereichen Computergraphik, Spiele und Simulationen.

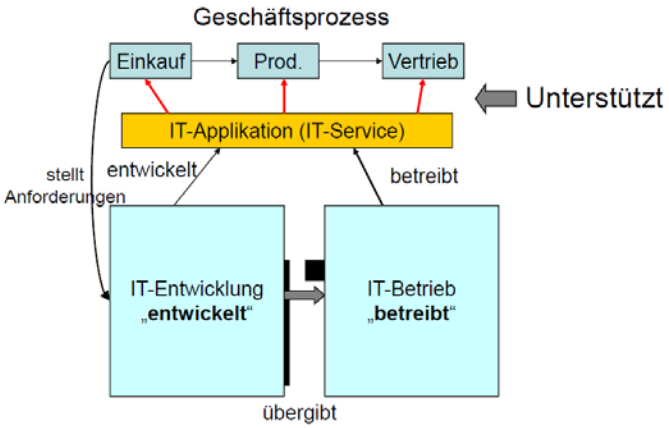
Interdisziplinäre Gebiete
Im Bereich der angewandten Informatik gibt es wichtige interdisziplinäre Gebiete:

- Künstliche Intelligenz: „keine“ Algorithmen, sondern der Computer wird zur Lösungsfindung befähigt. → Expertensysteme, Robotik
- Computerlinguistik: Verarbeitung natürlicher Sprache mit dem Computer (Informatik und Sprachwissenschaft)
- Bioinformatik: befasst sich mit Entschlüsselung des Erbgutes von Lebewesen (Informatik und Biologie)

1.1.2 Bedeutung/Zweck der IT im Unternehmen

Die Hauptaufgabe der IT besteht darin,

- Geschäftsprozesse zu unterstützen (Automation)
- IT-Applikationen zu entwickeln und zu betreiben



1.1.3 Ziele der rechnergestützten Informationsverarbeitung

- Rationalisierung = Kosteneinsparungen
- Bewältigung großer Datenmengen
- Beschleunigung von Geschäftsprozessen
- Verbesserung von Qualität und Service
- Unterstützung der Planung, Steuerung und Kontrolle
- Umfangreiche, komplizierte Berechnungen (OR)
- Ermöglichung neuer Organisationsformen
  - E-Mail, Groupware, Work Flow Management
  - Elektronischer Datenaustausch (EDI), virtuelle Unternehmen
- Strategische Wettbewerbsvorteile

1.2 Prozess-, Funktions-, Datensicht

1.2.1 Prozesssicht (Wie?)

Die Prozesssicht zeigt die zu automatisierenden **Abläufe** im Unternehmen.

Ein Prozess ist ein allgemeiner Ablauf mehrerer Abschnitte, bei denen es sich um Aufgaben, Ausführungen, Arbeitsschritte oder Ähnliches handeln kann. Zwischen diesen Prozessabschnitten bestehen bestimmte Abhängigkeiten.

1.2.2 Funktionssicht (Womit?)

Die Funktionssicht definiert notwendige Funktionen **der Applikationen (SW) & Systeme (HW)** und ihre Verbindung zu den Prozessen (Use Cases).

1.2.3 Datensicht (Was?)

Die Datensicht beschreibt die **Informationen**, welche das Unternehmen für das Abwickeln von Geschäften benötigt (ER-Modelle).

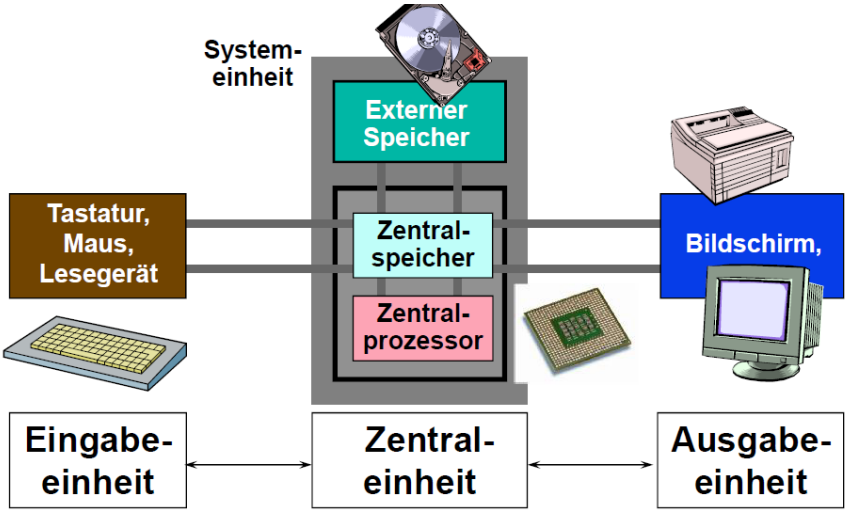
Die Datensicht

- hat die Definition der Daten in einem Informationssystem zum Gegenstand
- beschreibt die digitalen Repräsentationen der Objekte des zu beschreibenden Realitätsausschnittes

Darstellungs- bzw. Realisierungsform:

- auf Ebene des Fachkonzepts → konzeptionelle Datenmodelle
- auf ebene des IT-Konzepts → relationale Datenbanksysteme

1.3 Aufbau eines Computersystems



1.3.1 Hardware

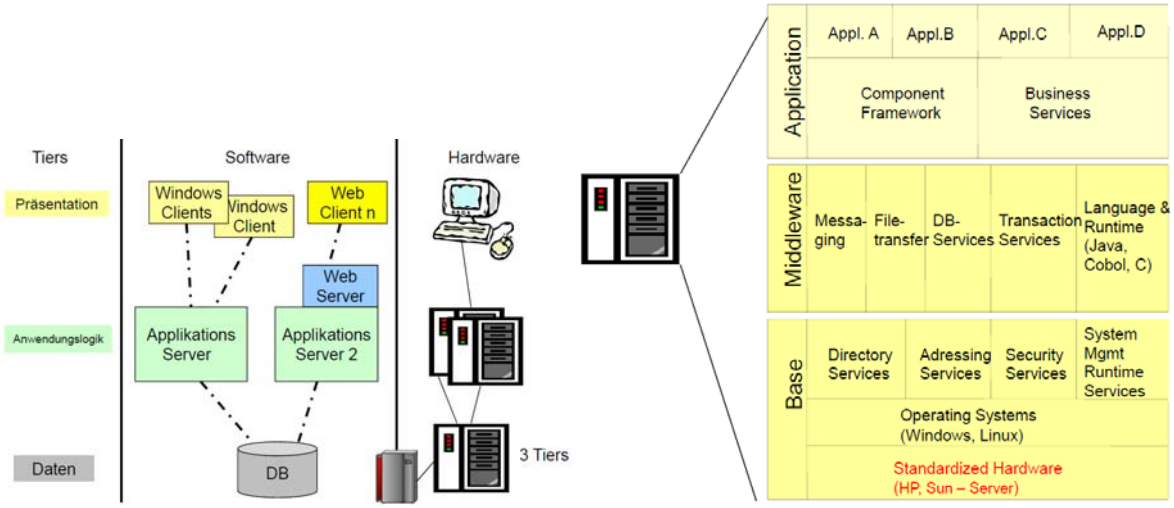
Sammelbegriff für die informationstechnischen Geräte

1.3.2 Software

Sammelbegriff für die Programme

- Systemsoftware
- Entwicklungssoftware
- Anwendungssoftware

1.3.3 verteilte Applikation



1.3.4 digitale Daten

- Repräsentation durch Zeichen

  Ein Zeichen ist ein Element aus einer zur Darstellung von Information vereinbarten endlichen Menge von verschiedenen Elementen, dem Zeichenvorrat

- Beispiele: Buchstaben, Ziffern, Interpunktionszeichen, Steuerzeichen, Farbpunkte von Bildern, akustische Signale

- Zeichen werden bei der maschinellen Verarbeitung durch elektrische Impulsfolgen, magnetisierte Positionen auf Datenträgern usw. dargestellt

Rechnerintern werden die digitalen Daten durch 0 (binäre Null) und 1 (binäre Eins) dargestellt ➔ Bit (Binary Digit)

Die zwei verschiedenen Zeichen werden dabei durch Zustände unterschiedlicher Stromspannung realisiert.

2 Internet

Lernziele

- Sie können den Begriff Technologie-Konvergenz mit praktischen Beispielen unterlegen.
- Sie kennen eine Gliederung der Informationsinfrastruktur.
- Sie können die Bedeutung des Internets für neue Geschäftsmodelle erläutern.
- Sie können die Einbindung und Schwerpunkte von Informationssystemen in Aktivitäten von Firmen erklären.
- Sie kennen Strategie und Schwerpunkte für die Informationsgesellschaft Schweiz.
- Sie kennen die Begriffe Standardisierungsgremien, Fachverbände und Foren und können je ein konkretes Beispiel nennen.
- Sie kennen die beteiligten Mitspieler (Rollen) im Informations- und Kommunikationsgeschäft.

Wer die Information und Kommunikation kontrolliert, kontrolliert die Welt!

Hauptgründe/Motivation:

- Stimulation der Ökonomie
- gesteigerte Wettbewerbsfähigkeit
- mehr Lebensqualität

2.1 Grundlagen

2.1.1 Technologie-Konvergenz

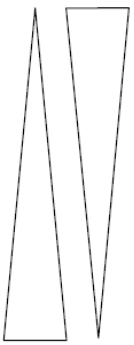
Gründe für die rasante Entwicklung in den letzten 50 Jahren:

- Fortschritte in der Technologie
- Innovation im Computerdesign

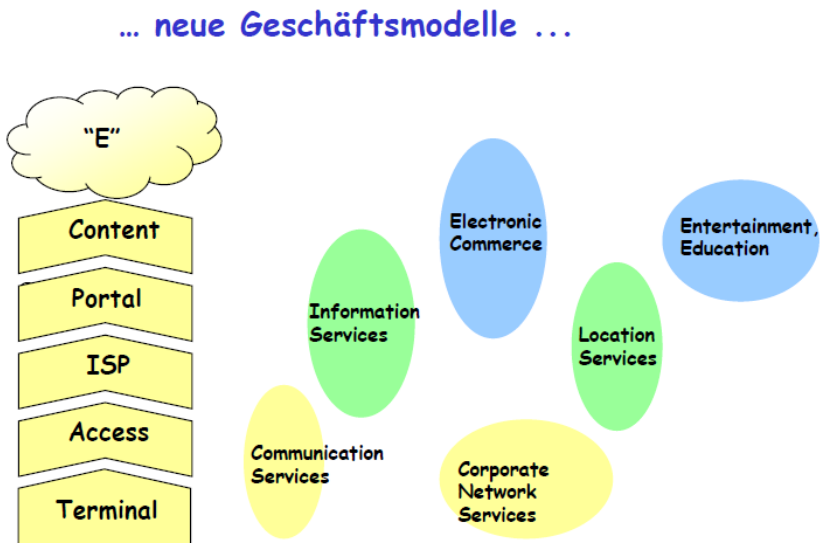
➔ völlig neue Lösungen möglich

Nachrichtentechnik (Codierung), Elektronik (Schalter, Verstärker…) und Informatik (PC, Programme,…) sind **konvergiert**.

2.1.2 Gliederung der Informationsinfrastruktur

<table>
<tr><td>Informations-gesellschaft</td><td colspan="2">Sozialer Hintergrund
- Soziales System
- Kulturelle Traditionen
- Kundenbedürfnisse</td><td rowspan="2">Dienste</td><td rowspan="4">Bedürfnisse
 
Leistungser-bringung</td></tr>
<tr><td>Informations-anwendungen</td><td colspan="2">Dienste
- Mehrwertdienste
- Basisdienste</td></tr>
<tr><td>Informations-verarbeitung</td><td>Verarbeitung
- Kundenschnittstelle
- Servertechnik</td><td>Speicherung
- Datenbanken
- Suchalgorithmen</td><td rowspan="2">Informations- und Kommunikations technologie (IKT)</td></tr>
<tr><td>Informations-übermittlung</td><td colspan="2">Informationstransfer
- Netzwerk Technologie
- Telekommunikationstechnik, Nachrichtenübertragung</td></tr>
</table>

2.1.3 Bedeutung des Internets für neue Geschäftsmodelle



z.B. „Wireless Portal“: Die Kontrolle über den Kundenzugriff wird entscheidend.

2.1.4 Einbindung und Schwerpunkte von Informationssystemen in Aktivitäten von Firmen

z.B. e-Business/e-Commerce:

- Wertschöpfungskette verbessern
- kleinere Lager, Produktion besser planbar

ERP → Optimierung der Geschäftsprozesse

2.1.5 Strategie und Schwerpunkte für die Informationsgesellschaft Schweiz

Ziele der aktualisierten Strategie aus dem Jahre 2006:

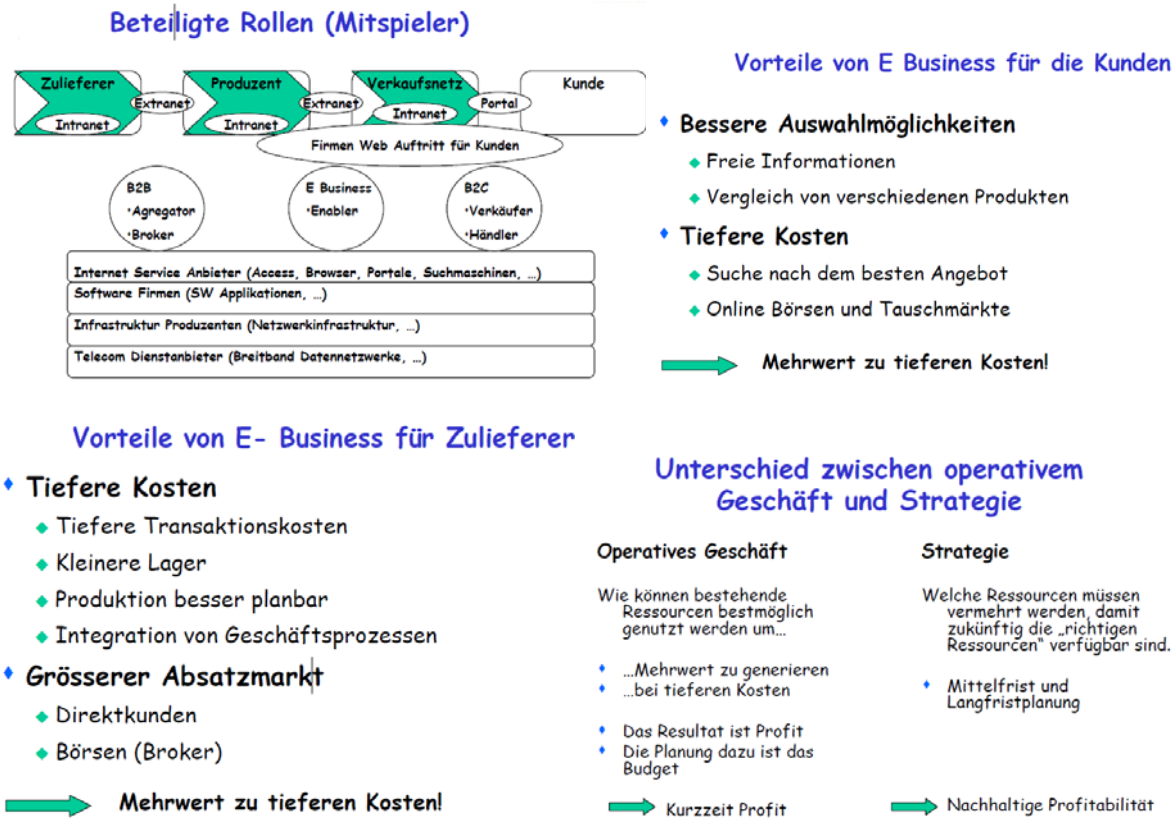
- Wohlstand vermehren
- Nachhaltigkeit sichern

→ Voraussetzung dafür ist die Bereitstellung von und Zugang zu Wissen und Informationen durch nachhaltige Bildungs- und Forschungsaktivitäten sowie kompatible Datenformate oder Kommunikationsprotokolle.

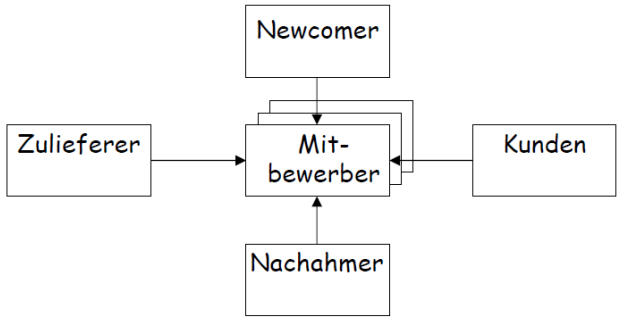
Grundsätze der Strategie

- Grundversorgung
- Vertrauen
- Zugang für alle
- Befähigung
- Föderalismus
- Zusammenarbeit
- Internationales

2.1.6 Mitspieler (Rollen) im Informations- und Kommunikationsgeschäft



Wer hat welchen Einfluss?



2.1.7 Wichtige Parteien

2.1.7.1 Standardisierungsgremien

Standardisierungsgremien sind für die Normierung verantwortlich.

Global

- International Organisation for Standardisation (ISO)

Europa

- European Telecommunications Standards Institute (ETSI)

National

- Schweizerische Normenvereinigung (SNV)
- Deutsches Institut für Normung (DIN)

2.1.7.2 Fachverbände

International

- Institute of Electrical and Electronics Engineers (IEEE)
- Internet Engineering Task Force (IETF)

National

- Schweizerischer Elektrotechnischer Verein (electrosuisse)

2.1.7.3 Interessensgemeinschaften/Foren

Verbunde von Firmen und anderen wirtschaftsorientierten Interessensgruppen mit dem Ziel

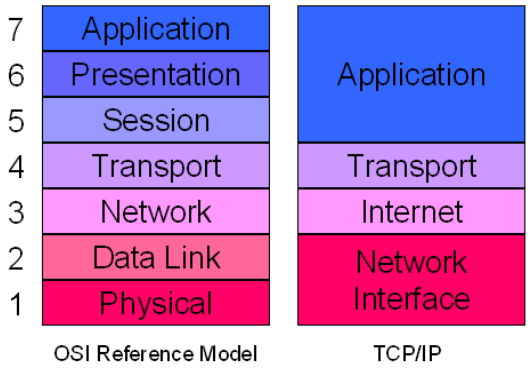
- Inputs für die Standardisierungsgremien zu liefern
- Tatsachen (Quasi-Standards) zu schaffen, bevor die Arbeit der Standardisierungsgremien abgeschlossen ist
- Interoperabilität zwischen Produkten der einzelnen Mitglieder sicherzustellen

2.2 Die Technik

Lernziele
• Sie kennen den prinzipiellen Aufbau der Internet-Protokolle (Protokollstapel, IP, TCP, UDP, Applikation, DNS, Adressierung, Routing, E-Mail, http) • Sie können die prinzipielle Funktionsweise des Schichtenmodells der Telekommunikation mittels eines anschaulichen Beispiels erläutern.

2.2.1 Protokoll-Stapel

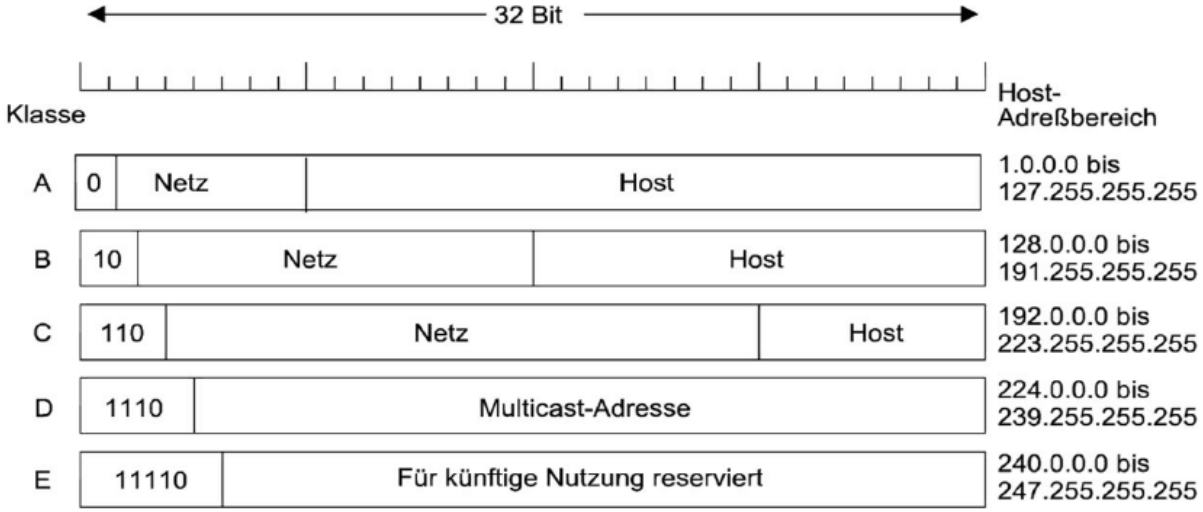
TCP-Layer	OSI-Layer	TCP-Layer-Name	Protokolle	Beschreibung
4	**5-7**	**Application**	Telnet, FTP, SMTP, DNS, Games	„alles andere“ bzw. die Anwendung
3	**4**	**Transport**	TCP. UDP	Transport von einer Anwendung auf einem Computer zu Anwendung auf anderem Computer
2	**3**	**Internet**	IP, ICMP	Adressierung und Pfadfindung durch ein Netzwerk bestehend aus vielen durch Router verbundene Netzwerke
1	**1-2**	**Network Access**	ADSL, WLAN, WAN, LAN	Medium, z.B. Kabel, Radiowellen etc. und Interface zwischen Computer und Medium (Netzwerkkarte)



2.2.2 IP

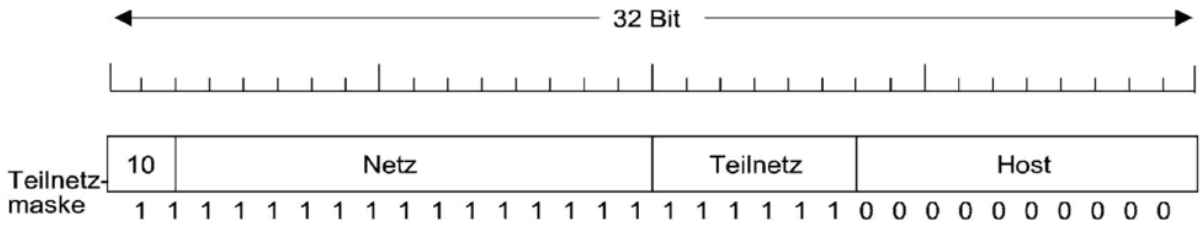
2.2.2.1 Netzwerkklassen

- führende Bits legen die Klasse fest
- Klasse bestimmt Grenze zwischen Präfix und Suffix
    - Präfix: Provider
    - Suffix: Netzwerkadministrator



2.2.2.2 Subnetting

Mittels Manipulation der Netzwerkmaske lässt sich ein grosses Netzwerk in viele kleine Teil-Netzwerke zerlegen (Hostteil weiter unterteilen). Dies nennt man Subnetting.



2.2.2.3 Paketaufbau

Paketkopf (Header)

- enthält Zieladresse
- feste Feldgrösse
- Art der Nutzlast (Protocol)

Nutzlast (Payload)

- variable Grösse bis 63 KB
- keine minimale Grösse

2.2.3 TCP

- Zuverlässige, bidirektionale Verbindung
- Von Anwendung (Source-Port) zu Anwendung (Destination-Port)
- Zuverlässigkeit wird durch geregelten Verbindungsaufbau und Bestätigungsmechanismus erreicht (Acknowledgment)

- Three-Way-Handshake
- Jedes Datagram ist mit Sequenznummer versehen
    - somit können verloren gegangene Datagramme nachgefragt oder
    - nicht in der richtigen Reihenfolge eingetroffene Datagramme geordnet werden

2.2.4 UDP

- Ebenfalls ein Transport Protokoll
- Unzuverlässige Verbindung (wie IP), aber von Anwendung (Source-Port) bist zum Ziel (Destination-Port)
- Kein Three-Way-Handshake zum Auf- und Abbau
- Benötigt dazu:
    - minimaler Overhead
    - minimale Rechenzeit
    - minimaler Kommunikationsaufwand

➔ Ergibt Performancegewinn

2.2.5 DNS

- DNS ist ein verteiltes, hierarchisches System zur Konvertierung von Rechnernamen (URIs) in IP –Adressen.
- „hosts“ Datei ist die einfachste Variante des DNS
- DNS-Datenbank weist eine in Zonen aufgeteilte baumförmige Struktur auf
- Zonen werden durch verschiedene DNS Server abgedeckt
- „Resolver“ Programm auf jedem Computersystem startet Abfrage nach der IP für eine gesuchte URI (nslookup)
- Verwendet dazu UDP (TCP nur für den Zonentransfer)
- Serverseitig Port 53

2.2.6 HTTP

- einfaches Request-Response Protokoll
- aktuelle Version 1.1
- TCP, Port 80
- Ermöglicht persistente Verbindungen, d.h. Verbindung wird über mehrere Requests aufrechterhalten

2.2.7 E-Mail

- Unterschiedliches Protokoll zum Versenden bzw. Empfangen:
- SMTP →Versenden von Mails
- POP3/IMAP4 → Empfangen von Mails

2.2.8 Routing

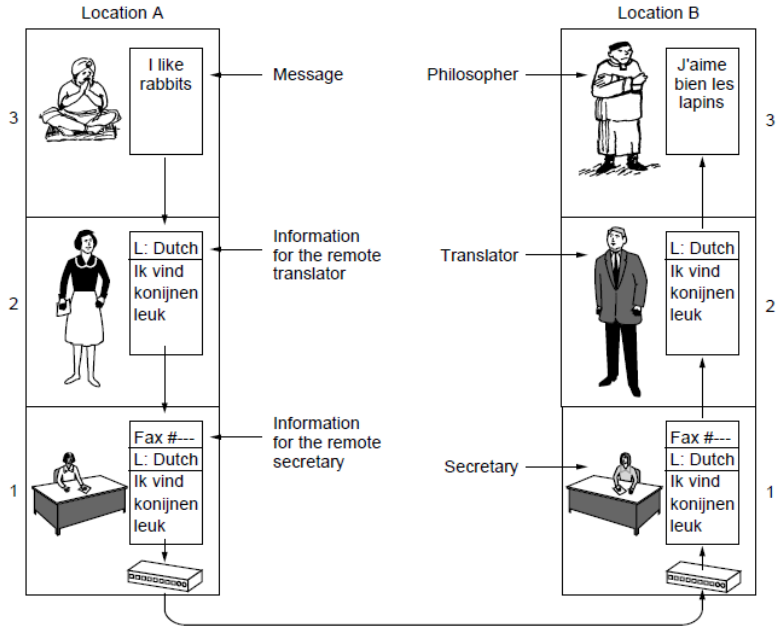
Der Router

- verbindet Netzwerke
- weiss anhand der Routingtabelle, wohin er ein Paket weiterleiten muss

Routingtabelle kann von Hand (Static Routing) oder automatisch (Dynamic Routing) erzeugt werden. Die Routingtabelle enthält:

- Zielnetzwerk
- Zielmaske
- Schnittstelle/Next Hop

2.2.9 Schichtenmodell der Telekommunikation



2.3 Sicherheit

Lernziele

- Sie können verschiedene Arten von Sicherheit in Netzwerken angeben.
- Sie sind sich der unterschiedlichen Bedrohungen der Sicherheit bewusst und können diese beschreiben.
- Sie kennen verschiedene Sicherheitsmassnahmen und deren Nutzen zur Abwehr von Bedrohungen.
- Sie können den Ablauf des Private-Key-Verfahrens erklären.
- Sie können den Ablauf des Public-Key-Verfahrens erklären.
- Sie können die Funktionsweise einer Digitalen Signatur und eines Message Digests (Hash) erklären.
- Sie können eine Security Policy für ein kleines Unternehmen erstellen.
- Sie können (einfache) Regeln für eine Firewall interpretieren.
- Sie kennen den Nutzen und die Anwendung von Steganografie.
- Sie können in einer Fallstudie adäquate, technische, logische und organisatorische Massnahmen zur Computersicherheit vorschlagen.

2.3.1 Arten von Sicherheit

Geheimhaltung, Vertraulichkeit

- Schutz von Daten vor Zugriff durch Unberechtigte

Authentifizierung

- Überprüfung der Person bzw. des Zugriffs

Nichtabstreitbarkeit, Verbindlichkeit

- Rechtskräftige digitale Unterschrift

Integrität

Unverfälschtheit, Vollständigkeit, Vertraulichkeit von Nachrichten oder Daten

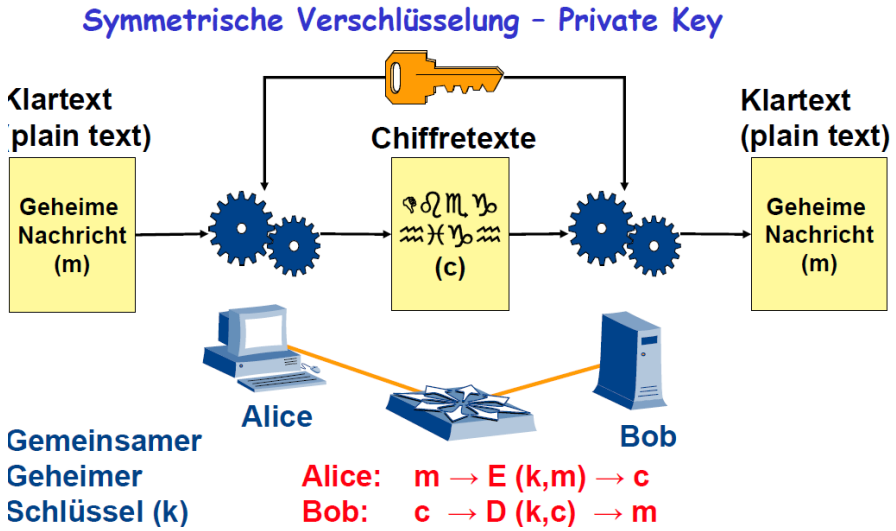
2.3.2 Bedrohungen

Bedrohung	Beschreibung
DoS (Denial of Service)	Angriff auf einen Server. Ziel: Ein oder mehrere Dienste arbeitsunfähig machen
MitM (Men in the Middle)	Sich logisch zwischen zwei Kommunikationspartner schalten. Datenverkehr einsehen und manipulieren
Replay-Attacke	Verbindung aufzeichnen und späteres abspielen (inkl. Passwort) → heute werden darum Sitzungs-Token verwendet
Sniffer/Spionagesoftware	Datenverkehr auslesen, anzeigen, auswerten → PW-Spionage
Trojaner, Viren Phishing	Schadsoftware Vorspiegelung falscher Webseiten, um PW auszuspionieren

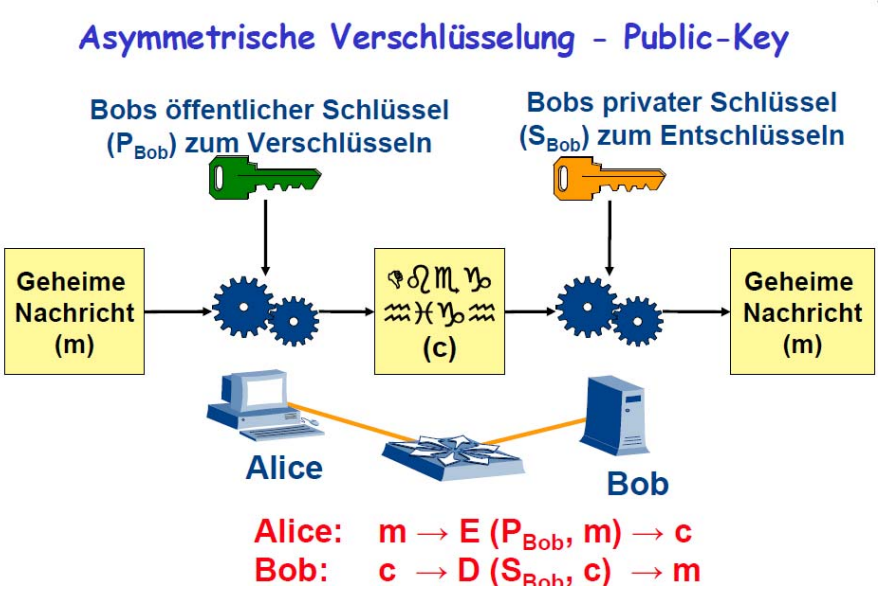
2.3.3 Sicherheitsmassnahmen

- Schutz gegen Zugriff von innen und aussen (Firewall)
- Verschlüsselung (Kryptographie)
- Kryptoanalyse: Wie aufwändig ist das Knacken des Schlüssels? → Bewertung
- Einspielen von Software-Updates
- Anti-Viren-Software inkl. aktuelle Virendefinitionen
- Datensicherung (RAID, Backup)
- Authentifizierung und Integrität → SecurID, RADIUS, Authentifizierungsprotokolle)
- Verschlüsselungsverfahren und-algorithmen, und Protokolle: IPSec, SSL, SSH

2.3.4 Private-Key-Verfahren



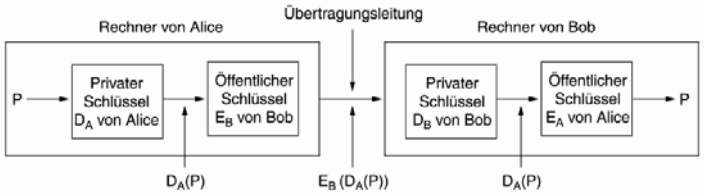
2.3.5 Public-Key-Verfahren



2.3.6 Funktionsweise von Digitaler Signatur und Message Digest (Hash)

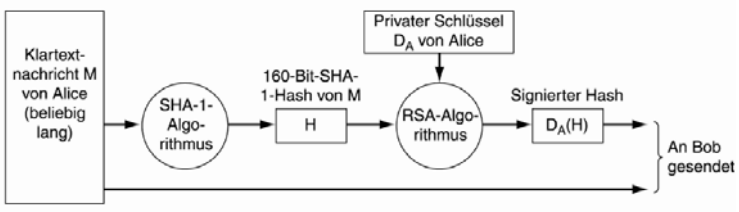
2.3.6.1 Digitale Signatur

- Anforderungen:
  - Empfänger kann Identität des Senders überprüfen
  - Sender kann Inhalt der Nachricht nicht leugnen
  - Empfänger kann Nachricht nicht selbst generiert haben.
- Signatur mit öffentlichem Schlüssel:



2.3.6.2 Message Digest

Beispiel: SHA-1



- Bob berechnet auch Hash-Wert H
- Bob entschlüsselt signierten Hash mit öffentlichem Schlüssel von Alice
- Falls beide H identisch, dann: Nachricht gültig

2.3.7 Security Policy

Eine Security Policy:

- ist ein Regelwerk
- beschreibt den Umgang mit Sicherheit unter ganzheitlichen Gesichtspunkten
- muss regelmässig überprüft werden

Eine Security Policy definiert:

- zur Verfügung gestellte Dienste → Internet, Intranet
- Nutzer dieser Dienste → intern und extern
- technische Massnahmen → verwendete Technologie, regelmässige Funktionsüberprüfung

2.3.8 Steganografie

Verbergen von Informationen in einer Datei, z.B. versteckter Urheberrechtskennzeichnung.

2.3.9 Massnahmen zur Computersicherheit

einige Beispiele…

2.3.9.1 Risiken

technisch

- kein Backup Server
- keine Redundanz vorhanden (Server)

menschlich

- Sabotage durch Mitarbeiter
- schlechte Anwendung von Software (besuchen infizierter Websites)

organisatorisch

- lokale Adminrechte von Benutzern
- kein Security/Safety-Konzept
- kein IT-Konzept

rechtlich

- Verlust von wichtigen Daten (10 Jahre Aufbewahrungspflicht von Projektunterlagen)
- Verlust von Kundendaten (Veröffentlichung von privaten Daten der Kunden)

2.3.9.2 Massnahmen

- Einrichten einer Firewall zwischen Mail-, Web- und Proxyserver und dem Internet
- Definition, d.h. Festlegung eindeutiger Firewallregeln.
- Einrichten einer Firewall zwischen den drei im Internet stehenden Servern und dem Firmennetz, d.h. Bilden einer "demilitarisierten Zone" (DMZ)
- Optional: Verlagerung des Mailservers in das Firmennetz und und Installation eines Mailproxies in der DMZ
- Einführung eines Backup Systems für die relevanten Server.
- Einführung eines Virenschutzes auf allen Servern und Rechnern.
- Entfernung der lokalen administrativen Rechte und Vergabe von eingeschränkten Berechtigungen
- Installation von Software nur durch die Administratoren.
- Schulung der Administratoren.
- Ausbildung eines 2. Administrators.
- Einführung einer Spam-Filterung.
- Einführung von Contentfilterung auf dem \'i/ebserver.
- IT Sicherheitsanweisungen der Geschäftsleitung zur Nutzung der IT an alle Mitarbeiter. (Verbot privater Nutzung, …)

3 Geschäftsprozesse & ERP-Systeme

Lernziele

Geschäftsprozesse

- Sie können die grundlegenden Prozesse eines Industrieunternehmens erläutern und in die drei Hauptkategorien einteilen.
- Sie kennen die wichtigsten aktuellen Herausforderungen an die Kernprozesse und können anhand eines Beispiels darstellen, wie aktuelle Lösungsversuche aussehen

IT-Systeme zur Prozessunterstützung

- Sie kennen die kritischen Leistungsmerkmale moderner IT-Systeme und können erläutern, wie diese eine breite und tiefe Unterstützung von Geschäftsprozessen ermöglichen.
- Sie können die historische Entwicklung des Geschäftsbedarfs nach IT-Unterstützung am Beispiel einer Bank und dieser kritischen Leistungsmerkmale darlegen.

Business-Software

- Sie haben einen Überblick über das Umfeld „Business Software“, ERP u.a.
- Sie kennen die wichtigsten Begriffe aus dem Bereich Business Software und können diese in der ERP-Landschaft einordnen.
- Sie haben einen Überblick über die wichtigsten Bestandteile eines ERP-Systems.
- Sie haben einen groben Marktüberblick und einen ersten Einblick in marktgängige ERP-Systeme.

ERP-Systeme

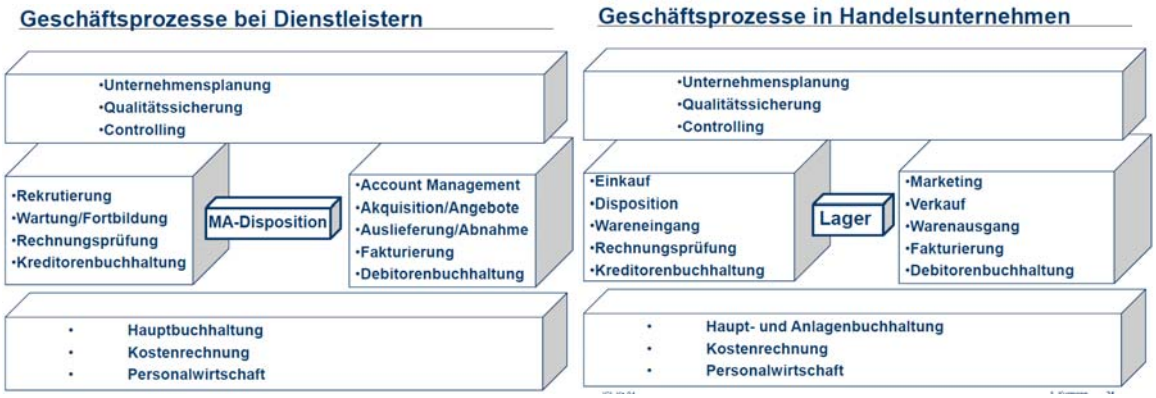
- Sie kennen die einzelnen Bestandteile eines ERP-Systems und deren Funktionen
- Sie haben vertieften Einblick in EIN beispielhaftes ERP-System.

3.1 Geschäftsprozesse

Ein Geschäftsprozess ist ein Ablauf von Aktivitäten, die der Erzeugung eines Produktes/einer Dienstleistung dienen und eine Wertschöpfung (Gewinn) erzielen. Er wird durch ein oder mehrere Ereignisse gestartet und durch ein oder mehrere Ereignisse abgeschlossen. Es liegt eine Organisationsstruktur zu Grunde.

3.1.1 grundlegende Prozesse eines Industrieunternehmens

Managementprozesse	Kernprozesse	Unterstützungsprozesse
▪ Strategieplanung ▪ Qualitätsmanagement ▪ Controlling	▪ Innovation ▪ Vertrieb ▪ Auftragsabwicklung ▪ Service	▪ Personalmanagement ▪ Finanzmanagement ▪ Ressourcen-management ▪ IT-Management



3.1.2 wichtigste aktuelle Herausforderungen an die Kernprozesse

in Handelsunternehmen:

- Globalisierung/Konsolidierung (Mergers & Acquisitions) → Preisdruck
- exzellent informiert, anspruchsvolle, selektive Kunden → Preisdruck, Notwendigkeit schneller & kompetenter Beratung
- Konsolidierung Supply-Chains
- Neue Geschäftsmodelle (Vertikalisierung, Branchenüberschreitung etc)
- Produktivität/Effizienz der Prozesse, Liefergeschwindigkeit
- Informationsflut, Integration, Schlüsseltechnologien (RFID, Self-Checkout etc)
- Online-Shopping
- Individueller Konsum, flüchtige Märkte
- „grüne Produktion“
- Sicherheit (Transportwege etc.)

Herausforderungen Kernprozesse

- Die zentralen Herausforderungen:
  1. **Kostendruck** durch Konkurrenz (Globalisierung als potenzierte Konkurrenz)
     - → Zwang zur *operativen Exzellenz*
     - → Zwang zur *Zusammenarbeit* (sogar mit Feinden)
  2. **Individualisierung** Abnehmerseite
     - → Zwang zu *Flexibilität* und *Kompetenz*
  3. **Zunahme Internetnutzung**
     - → Zwang zur *Mobilität angebotener Dienste*
- Das gilt sowohl für den Handel als auch für Dienstleister und das produzierende Gewerbe

Herausforderungen Kernprozesse

- Zwei Geschäftsprozess-Beispiele für
  - operative Exzellenz und
  - Zusammenarbeit:
  1. **Wareneingang** (firmeninterne operative Exzellenz)
  2. **Supply Chain Management** (firmenübergreifende operative Exzellenz und Zusammenarbeit)

3.2 IT-Systeme zur Prozessunterstützung

3.2.1 kritische Leistungsmerkmale moderner IT-Systeme

Die Formelprogrammierung ist die Minimalunterstützung seitens der IT:

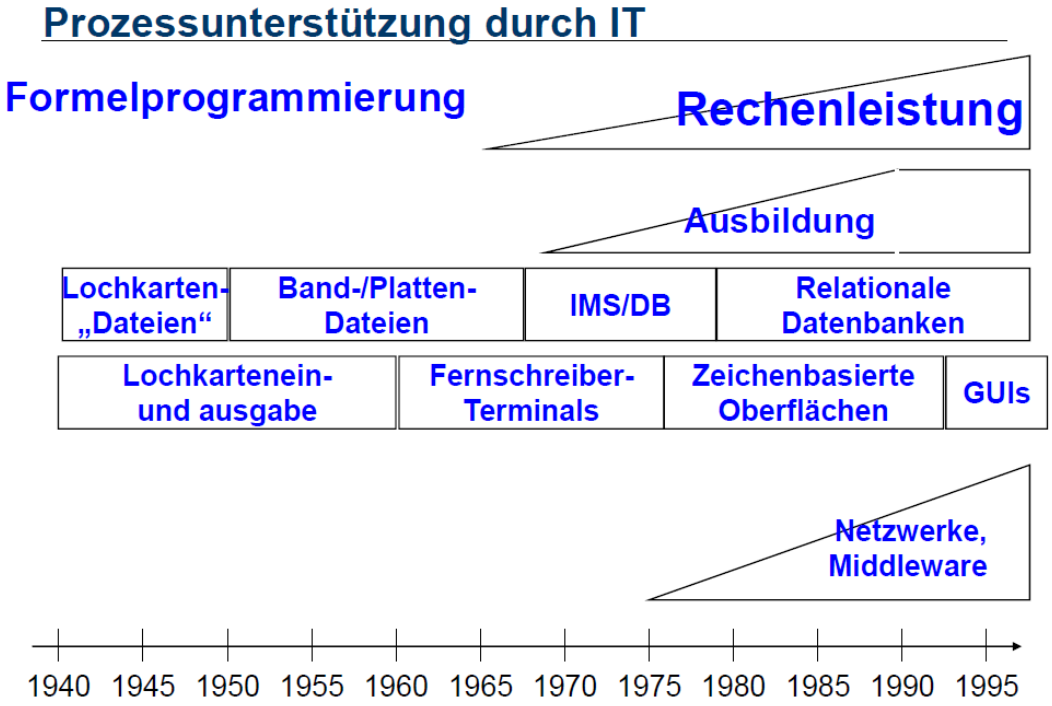
- Zahlreiche Medienbrüche
- Kerndaten anderweitig abgelegt
- Keine automatische Verarbeitung grosser Datenmengen möglich
- Nicht IT-spezifisch: Abakus, Rechenschieber, ...

zusätzlich notwendig zur Abdeckung der Geschäftsvorfälle sind:

- Noch viel mehr Rechenleistung

- Ausgebildete Informatiker
- Ein echtes Datenbanksystem
- Standardisierte Netzwerkverbindungen und Middleware
- Online-Betrieb, Benutzeroberflächen
- Mehrbenutzerfähigkeit

3.2.2 historische Entwicklung des Geschäftsbedarfs nach IT-Unterstützung



3.3 Überblick Business-Software

Wertschöpfung entsteht dann, wenn der Verkaufswert höher ist als die Summe der Einstandskosten aller Produktionsfaktoren.

Preis – Kosten = Gewinnspanne ➔ Wertschöpfung

Business Software (auch ERP-Systeme) ist eine integrierte betriebswirtschaftliche Standardsoftware zur elektronischen Unterstützung von Unternehmensprozessen.

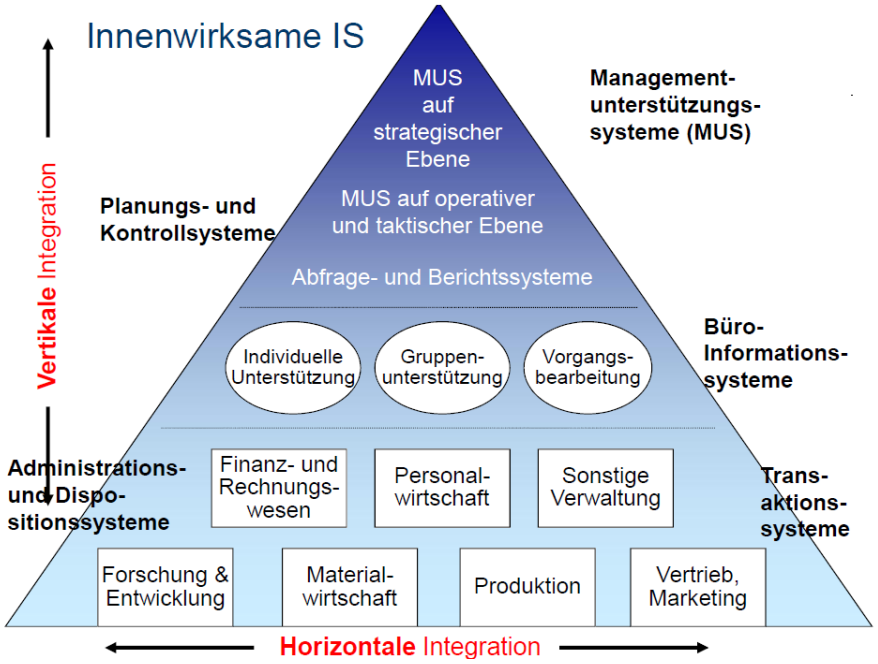
Standardsoftware

- Auf Allgemeingültigkeit und mehrfache Nutzung bei unterschiedlichen Anwendern ausgelegt
- Vorteile
  - Kostengünstigkeit
  - Zeitersparnis
  - „reiferes“ Produkt
  - Besserer Support
  - Prozess Know How des Herstellers
  - Zukunftssicherheit

Individualsoftware

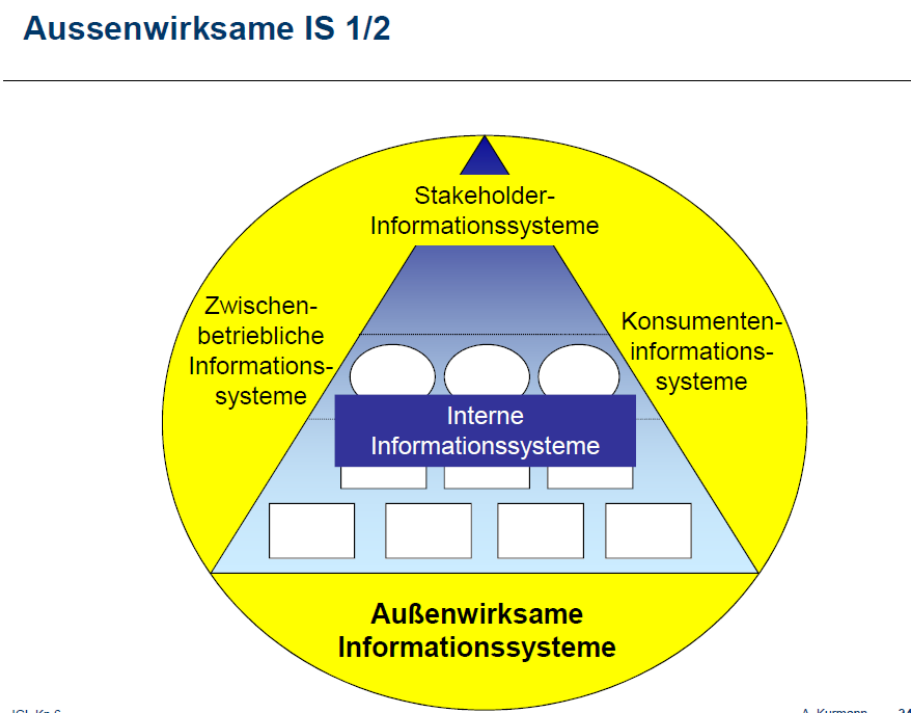
- Eigens für einen Anwendungsfall erstellt und an ein spezifisches Umfeld angepasst
- Vorteile
  - Ausrichtung auf die spezifischen Bedürfnisse eines Betriebs
  - Betrieb erwirbt i.R. die alleinigen Rechte am Quellprogramm sowie der Dokumentation

3.3.1 Umfeld „Business Software“



horizontale Integration:
Integration der Prozesse (rechts nach links, links nach rechts) → Modularer Aufbau

vertikale Integration:
Aggregation der Informationen



3.3.2 wichtige Begriffe aus dem Bereich Business Software

3.3.2.1 E-Business

E-Business umfasst die Unterstützung der Beziehungen und Prozesse eines Unternehmens mit seinen Geschäftspartnern, Kunden und Mitarbeitenden durch elektronische Medien. Der Einsatz von Internettechnologien spielt dabei eine zentrale Rolle.

3.3.2.2 E-Procurement

E-Procurement ist die elektronische Unterstützung der Beschaffungsprozesse (Einkauf) eines Unternehmens. Unterstützt wird der Einkauf sowohl von direkten Gütern (Güter, die in das Produkt eingehen) wie auch von indirekten Gütern (Verbrauchsmaterial).

3.3.2.3 E-Commerce

E-Commerce ist die elektronische Unterstützung der Absatzprozesse (Verkauf) eines Unternehmens, die klassischerweise in die Kernphasen Anbahnung, Vereinbarung und Abwicklung. Die zentrale E-Commerce-Applikation ist der E-Shop.

3.3.2.4 ERP

Ein ERP-System ist die zentrale Unternehmenssoftware für das so genannte "Enterprise Resource Planning". ERP-Systeme decken Basisfunktionen wie das Rechnungswesen, das Personalmanagement oder die Offerten- und Kundenverwaltung ab und verwalten unternehmensrelevante Finanz-, Personal-, Kunden und Produktdaten.

3.3.2.5 E-Shop

E-Shops sind Verkaufssysteme, mit denen im Internet Waren zum Kauf bereitgestellt werden. Sie bieten Funktionen wie die Suche im Produktkatalog, das Anlegen eines Warenkorbs und das Absenden einer Bestellung.

3.3.2.6 Portal

Ein Portal ist eine Webapplikation, die einen strukturierten Einstieg in eine Reihe von Applikationen oder Informationsangeboten bietet. Eine spezielle Form stellen Unternehmensportale (Enterprise Portals) als Werkzeuge für den Einstieg am Arbeitsplatz dar.

3.3.2.7 CRM – Customer Relationship Management

CRM ist das systematische, strategische Management von Kundenbeziehungen. Im Zentrum stehen die Gewinnung von Kunden, die Erweiterung des Kaufvolumens pro Kunde, die Stärkung der Kundenbindung und die Rückgewinnung von abgesprungenen Kunden.

3.3.2.8 SCM – Supply Chain Management

Supply Chain Management (SCM) ist das integrierte Management der gesamten Wertschöpfungskette vom Einkauf über die Verarbeitung, den Verkauf bis zur Entsorgung bzw. zum Recycling.

3.3.2.9 PPS – Produktionsplanung und -Steuerung

Produktionsplanung und –steuerungssystem. Integriertes Systems für die Steurung und Planung der Produktion oder Assembly.

3.3.2.10 Sell-Side / Buy-Side

Von Sell-Side- und Buy-Side-Lösungen spricht man bei E-Shops im B2B-Bereich. Sie unterstützen sowohl die Beschaffung als auch den Verkauf im Internet. Bei einer Buy-Side-Lösung werden die Einkaufssoftware und der überwiegende Teil des Produktkatalogs vom Käufer betrieben. Bei einer Sell-Side-Lösung werden sowohl die Einkaufssoftware als auch

der Produktkatalog vom Lieferanten zur Verfügung gestellt. Die grosse Zahl von B2B-E-Shops verfolgen den Sell-Side-Ansatz (Beispiel: Arp Datacon, www.arp.ch).

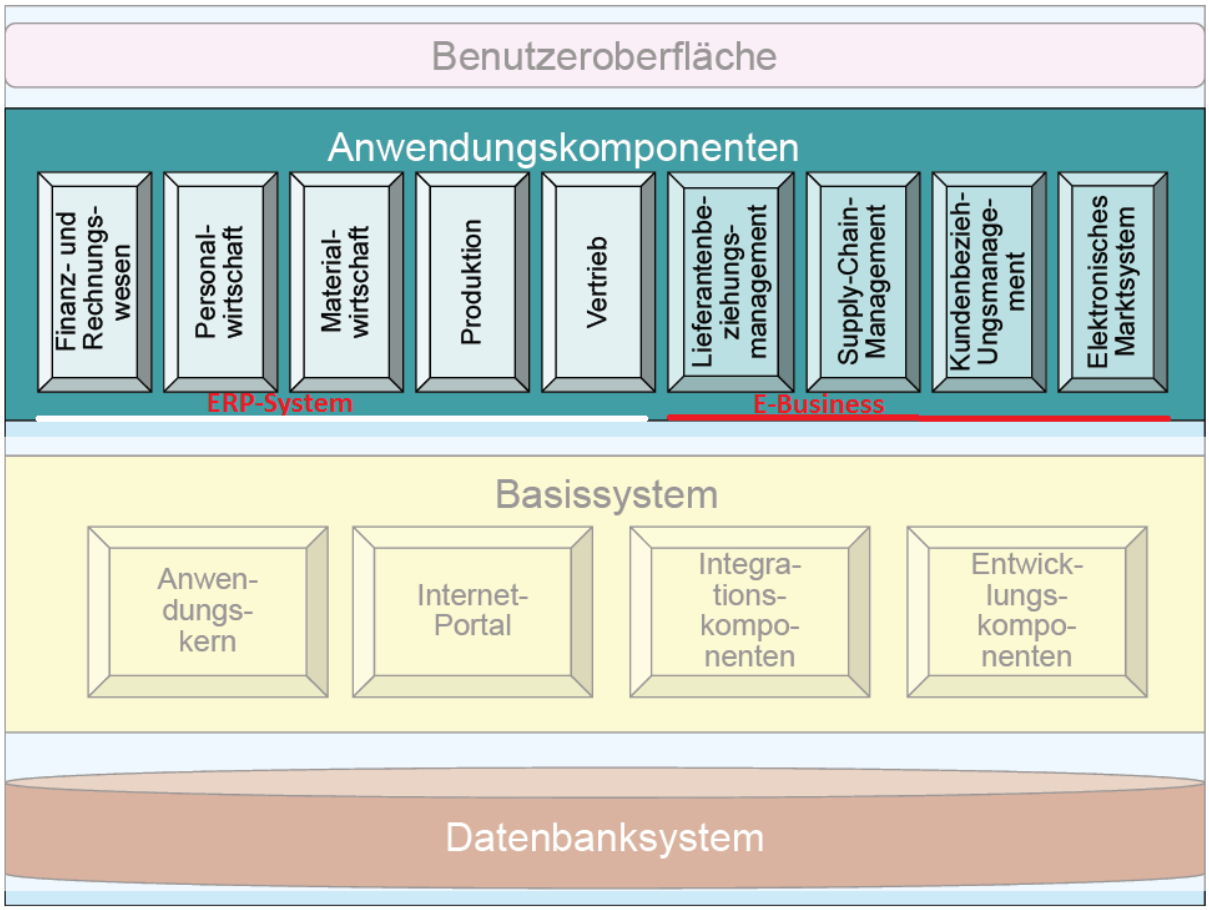
3.3.3 Marktüberblick & Einblick in marktgängige ERP-Systeme

Führende Softwarehersteller weltweit:

- **SAP**
- Oracle Applications
  - PeopleSoft
    - J.D. Edwards
- Microsoft Business Solutions
- Sage Group
- Open-Source-ERP-Systeme

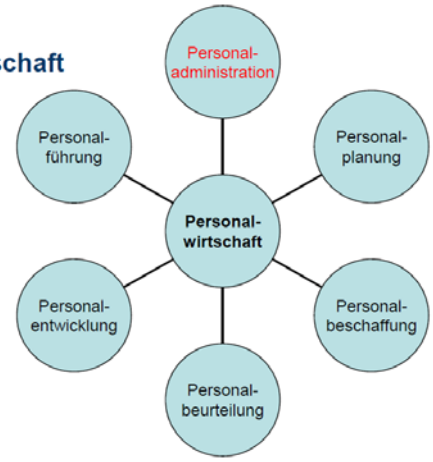
3.4 ERP-Systeme

3.4.1 Bestandteile eines ERP-Systems

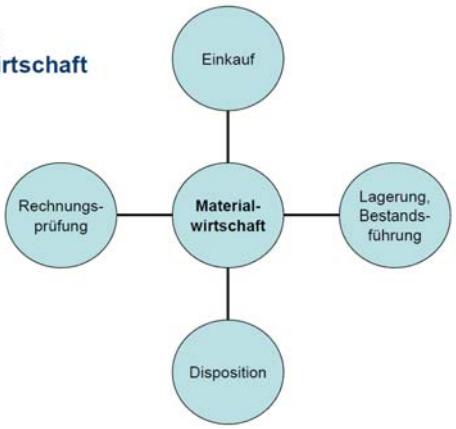


Übersicht Finanz- und Rechnungswesen

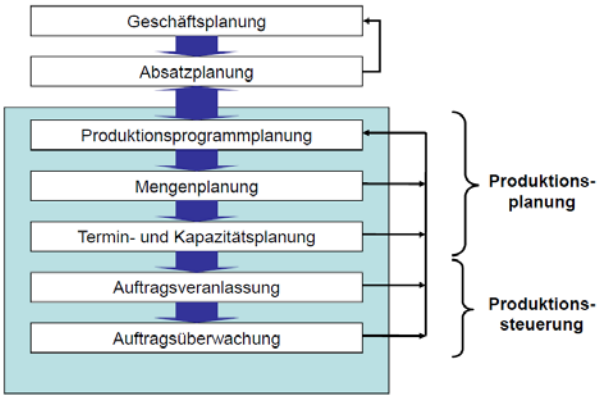
Übersicht Personalwirtschaft



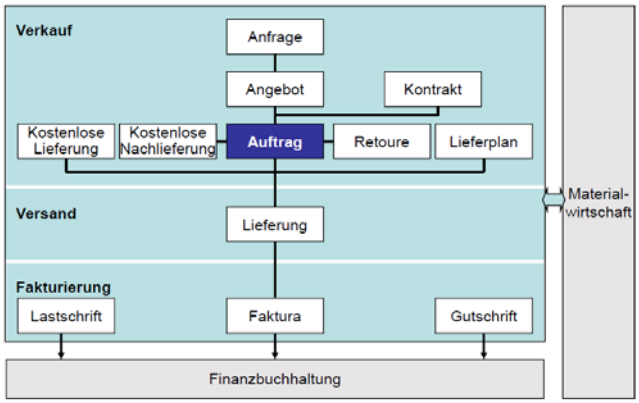
Übersicht Materialwirtschaft



Übersicht Produktionsplanung und -steuerung



Vertriebsabwicklung in SAP



4 Rechnerarchitekturen

4.1 Zahlensysteme und Logik

Lernziele
• Sie Verstehen den Begriff „digital“ und „analog“. • Sie kennen den Begriff „Bit“ als „Atom“ der digitalen Datenverarbeitung. • Sie kennen den Begriff „Bit“ als Mass für Information. • Sie verstehen die prinzipielle Gleichheit von Zahlen, Bildern, Musik und Texten auf der Bit-Ebene. • Sie kennen das duale, oktale und hexadezimale Zahlensystem. • Sie können mit Binärzahlen im 2er-Komplement rechnen. • Sie können die Haupteigenschaften von Gleitkommazahlen benennen. • Sie können mit Gleitkommazahlen im IEEE-754 Format rechnen. • Sie kennen die bool’schen Operationen NOT, AND, OR und ihre Kombinationen auswendig. • Sie können das DeMorgan’sche Gesetz anwenden. • Sie kennen auf konzeptioneller Stufe die Komponenten Addierer und FlipFlop.

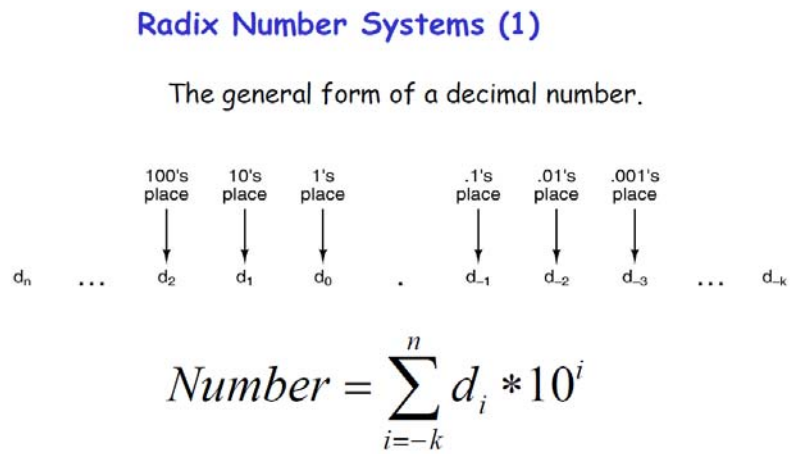
4.1.1 digital vs. analog

Zahlen, Bilder, Musik, Texte usw. werden in der digitalen Welt als Bitstrom (Folge von 0 und 1) dargestellt.

4.1.2 der Begriff: Bit

Bit = Binary Digit
kennt 2 Zustände (0 oder 1)

4.1.3 Zahlensysteme

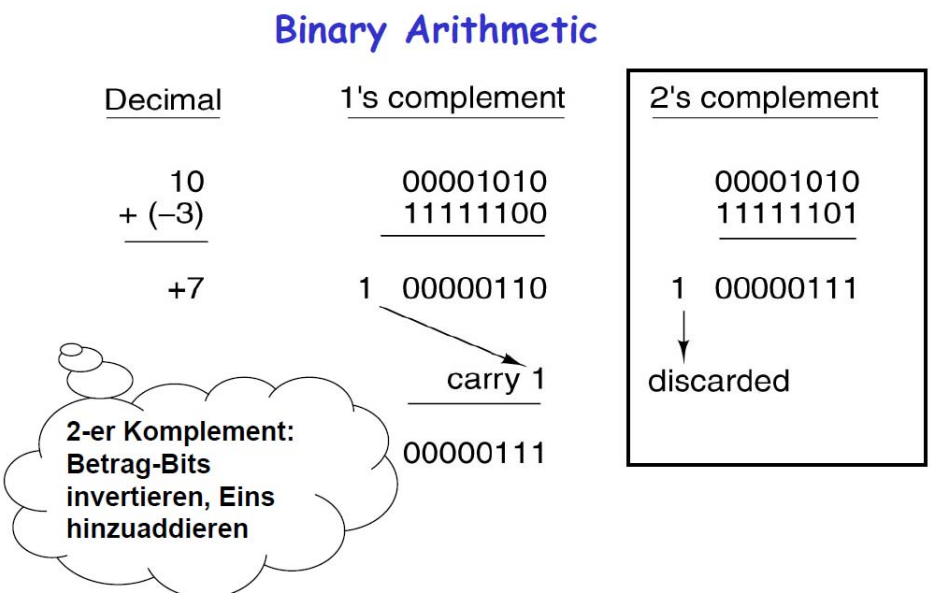


4.1.3.1 dual → Basis 2

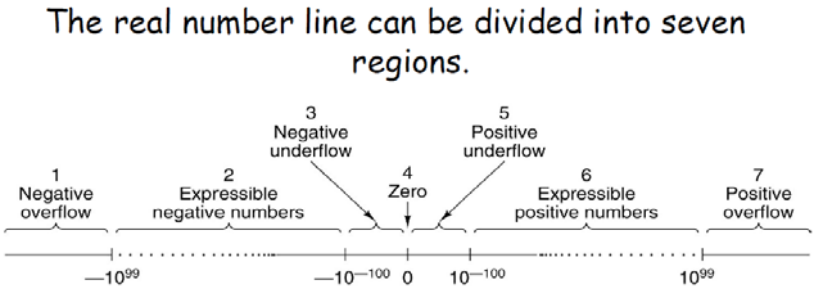
4.1.3.2 oktal → Basis 8

4.1.3.3 hexadezimal → Basis 16

4.1.4 2er-Komplement



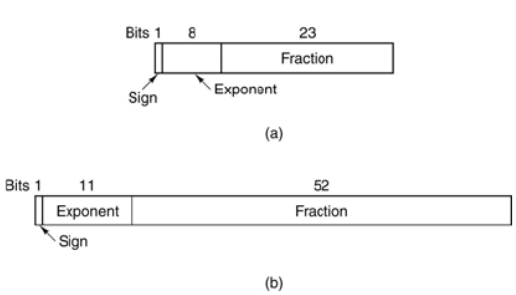
4.1.5 Gleitkommazahlen (IEEE754-Format)



4.1.5.1 Eigenschaften

- Must separate range from precision
- Use scientific notation $n = f \times 10^e$
  - f is the fraction or mantissa
  - e is the exponent (a positive or negative integer)
- grosser Bereich darstellbar
- nicht exakt darstellbar → begrenzte Anzahl Bits
- Es kann Rechenfehler geben.

4.1.5.2 IEEE754-Format



Der IEEE-Standard 754 definiert drei Formate: *einfache Genauigkeit (32 Bits)*, *doppelte Genauigkeit (64 Bits)* und erweiterte Genauigkeit (80 Bits). Das Format mit erweiterter Genauigkeit dient der Reduktion von Rundungsfehlern. Es wird vorwiegend in Gleitkommaarithmetik-Einheiten benutzt. Es wird hier nicht weiter behandelt. Die beiden Formate für einfache und doppelte Genauigkeit nutzen die Basis-2 für Mantissen und die Überschuss-Notation für Exponenten. Die entsprechenden primitiven Datentype in Java heissen *float* und *double*.

https://elearning.hslu.ch/ilias/repository.php?ref_id=1259154&cmd=sendfile

4.1.6 bool'sche Operationen

4.1.6.1 NOT

A	Z
0	**1**
1	**0**

NOT Ein Öffner:

4.1.6.2 AND

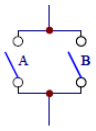
A	B	Z
0	0	**0**
0	1	**0**
1	0	**0**
1	1	**1**

AND Zwei Schliesser in Serie:

4.1.6.3 OR

A	B	Z
0	0	**0**
0	1	**1**
1	0	**1**
1	1	**1**

OR Zwei Schliesser parallel:



4.1.6.4 DeMorgan'sches Gesetz

Gesetze:

Kommutativgesetz: $AB = BA$ und $A + B = B + A$

Assoziativgesetz: $A(BC) = (AB)C$ und $A + (B + C) = (A + B) + C$, etc.

Distributivgesetz: $A(B + C) = AB + AC$ und
$A + BC = (A + B)(A + C)$ → **gilt nur im Dualsystem !!**

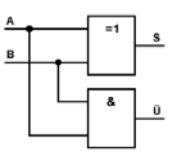
Umwandlung nach DeMorgan:

$\overline{A + B} = \overline{A}\,\overline{B}$ und $\overline{AB} = \overline{A} + \overline{B}$, etc.

4.1.7 Addierer und FlipFlop

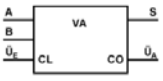
Addierer

Halb-Addierer



B	A	Ü (2¹)	S (2⁰)
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

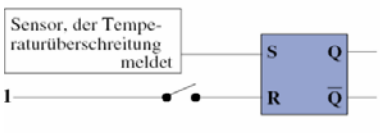
Voll-Addierer

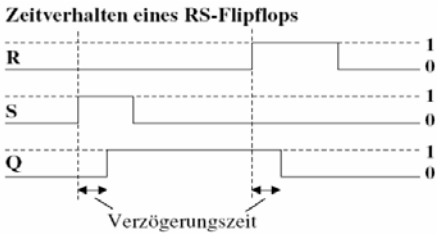


Ü_E	B	A	Ü_A	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Anwendung für Flip Flop

- Warnlampe wird eingeschaltet,
  wenn Sensor Temperatur-Überschreitung am S-Eingang signalisiert
- Rücksetzen (Ausschalten) der Lampe muss manuell über R-Eingang erfolgen



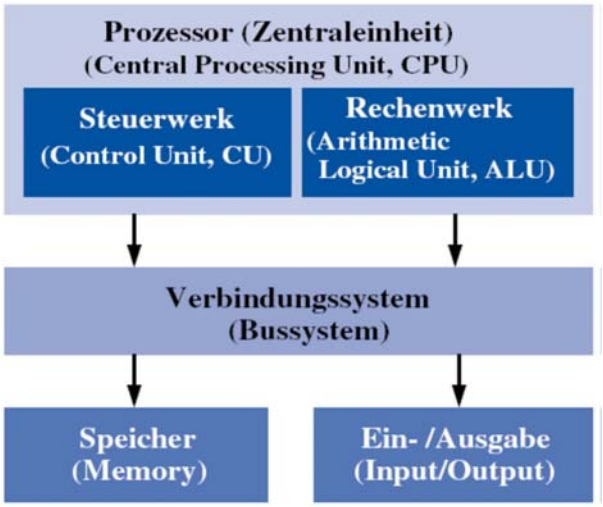


4.2 Rechnersysteme

Lernziele
• Sie kennen zwei verschiedene Rechnerarchitekturen. • Sie können die Funktion einer CPU beschreiben. • Sie können den Verarbeitungsablauf auf einem Von-Neumann Rechner erklären. • Sie können das Lesen und Schreiben einer Speicherzelle detailliert beschreiben unter Verwendung der Begriffe Adressbus und Datenbus. • Sie können ein gegebenes „Holzcomputer“ Programm analysieren. • Sie können ein gegebenes „Holzcomputer“ Programm mittels vorhandener Befehlstabelle von Hand ins Maschinenprogramm assemblieren. • Sie kennen das Moor’sche Gesetz. • Sie können eine Speicherhierarchie (Zugriffszeit, Datenmengen) beschreiben.

4.2.1 Von-Neumann und Harvard-Architektur

Von-Neumann	Harvard
▪ gemeinsames Memory für Daten und Befehle ▪ Jede Speicherzelle kann adressiert / referenziert werden ▪ Besteht aus n Speicherzellen ▪ Klassische von Neumann Architektur (nach Flynn) SISD: Single Instruction - Single Data Stream	▪ Befehlsspeicher physikalisch vom Datenspeicher getrennt ▪ mehrere Rechenwerke parallel mit Daten und Befehlen füllen ▪ Befehle und zugehörige Daten in einem Taktzyklus ins Rechenwerk laden ▪ Von-Neumann-Architektur braucht mindestens zwei Taktzyklen



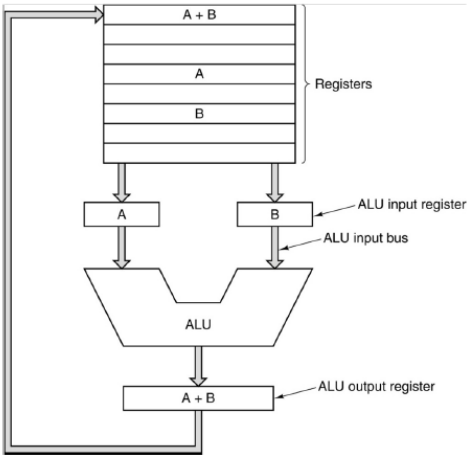
4.2.2 Funktion einer CPU

Die CPU muss

- ein paar Befehle der Reihe nach ausführen können
- ein bisschen rechnen können (Speicherinhalt als Zahl interpretieren)
- auf den Inhalt des Speichers reagieren

4.2.3 Verarbeitungsablauf auf einem Von-Neumann-Rechner

CPU Organization: Datenpfad



- Fetch next instruction from memory into instr. register
- Change program counter to point to next instruction
- Determine type of instruction just fetched
- If instruction uses word in memory, determine where. Fetch word, if needed, into CPU register
- Execute the instruction
- Go to step 1 to begin executing following instruction

4.2.4 Lesen und Schreiben einer Speicherzelle

4.2.4.1 Lesen

Die CPU

- gibt Adresse der Funktionsgruppe ROM und der Speicherzelle auf den Adressbus
- aktiviert die Steuerleitung READ, adressierte Speicherzelle gibt Inhalt auf den Datenbus
- übernimmt Daten vom Datenbus und deaktiviert die Steuerleitung READ

4.2.4.2 Schreiben

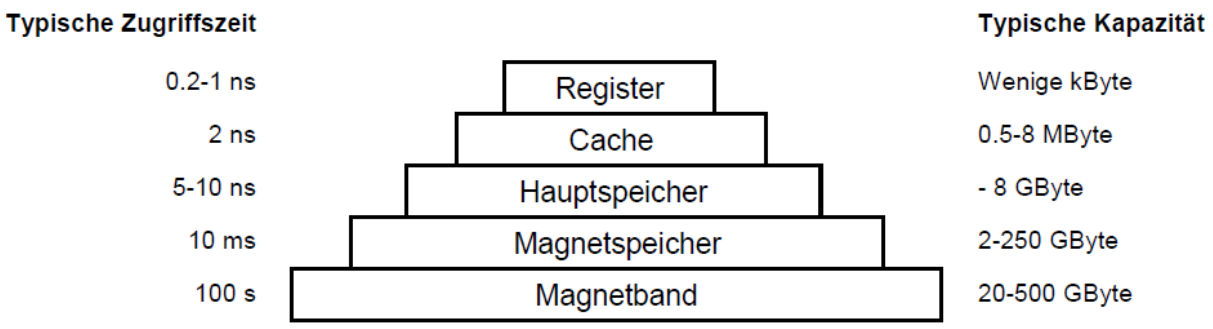
Die CPU

- gibt Adresse der Funktionsgruppe OUT auf den Adressbus
- gibt die Daten auf den Datenbus
- aktiviert die Steuerleitung WRITE, Out nimmt die Daten vom Datenbus
- deaktiviert die Steuerleitung WRITE

4.2.5 Moor'sches Gesetz

Das Moor'sche Gesetz besagt, dass sich die Packungsdichte der Transistoren auf einem Mikroprozessor in etwa alle 18 Monate verdoppelt.

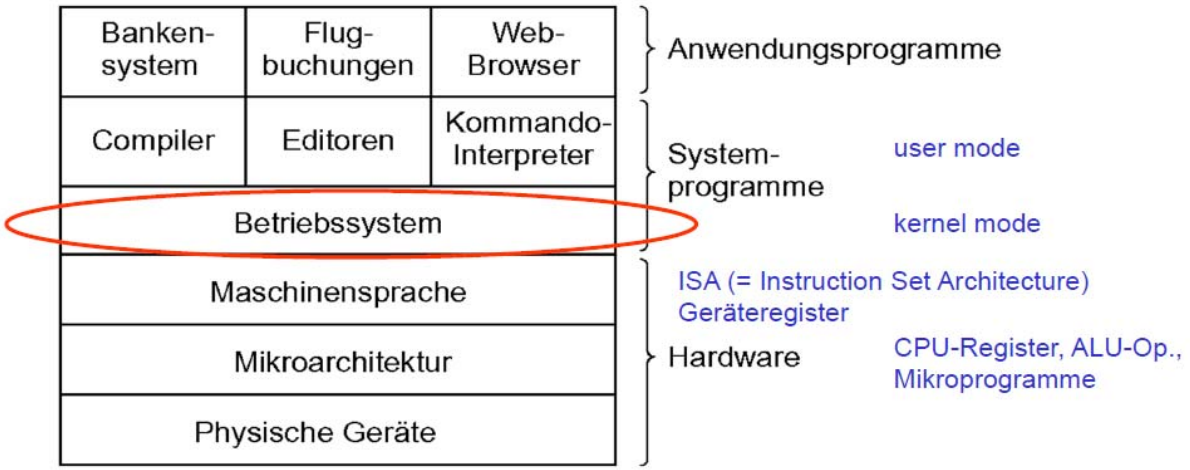
4.2.6 Speicherhierarchie



4.3 Betriebssysteme

Lernziele
• Sie können das Betriebssystem ins Architekturmodell richtig einordnen. • Sie kennen Aufgaben und Funktionsprinzipien eines Betriebssystems. • Sie können den Begriff Prozess und die Prozesszustände anschaulich erklären. • Sie kennen die Aufgaben eines Schedulers. • Sie können die Hauptaufgaben des Speichermanagements beschreiben. • Sie verstehen das Konzept der virtuellen Adressen. • Sie kennen den grundsätzlichen Aufbau eines Filesystems. • Sie können den Unterschied zwischen programmiertem IO und Interrupt IO erklären. • Sie können 6 Dateioperationen benennen und ihre jeweilige Funktion beschreiben. • Sie können die Funktionsweise einer verketteten Liste mit einer Skizze erklären. • Sie können das Client-Server-Konzept in einer Dreistufen-Architektur erklären.

4.3.1 Architekturmodell



4.3.2 Aufgaben und Funktionsweise

Betriebssysteme realisieren eine Softwareschicht zwischen Anwendungen und Hardware, die sich mit der Verwaltung der Hardwarekomponenten beschäftigt und für die Anwendungen einfachere Schnittstellen bereitstellt.

- Die Schnittstelle zum Betriebssystem stellt aus Anwendungssicht zusätzliche (komfortablere) Operationen (u.a. auch für Geräte) zur Verfügung.
  Damit kann man es als eine erweiterte oder virtuelle Maschine auffassen, die mächtiger ist als die eigentliche durch Hardware realisierte Maschine.

- Das Betriebssystem verwaltet die komplexe Struktur der Hardwarekomponenten und teilt die Ressourcen (Prozessor, Speicher, Geräte usw.) in einer kontrollierten Art und Weise den Anwendungen zu.

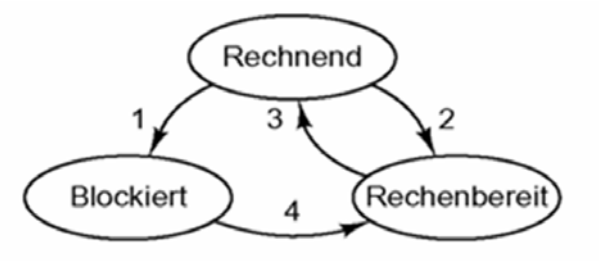
4.3.3 Prozess, Prozesszustände

- Ein Prozess ist ein sich in Ausführung befindliches Programm (Abstraktion), z.B. Word
- Ein Prozess kann andere Prozesse kreieren oder beenden

- Prozesse können mit anderen Prozessen kommunizieren
- Zum Prozess gehört ein Adressraum (Programm- und Datenbereich, Stack) und Registerinhalte.

- Rechner können viele Dinge quasi "gleichzeitig" machen, bei 1-Prozessor-Maschinen läuft aber zu jeder Zeit immer nur ein Prozess auf der CPU.
- Pseudoparallelität: Das Hin- und Herwechseln zwischen Prozessen heisst Context-Switching.
- Ein Schedulingalgorithmus bestimmt, wann welcher Prozess (weiter) bearbeitet wird.

Prozesszustände



4.3.4 Aufgabe eines Schedulers

„Bewerben“ sich mehrere Prozesse um die CPU, trifft der Scheduler die Entscheidung, welcher dieser Prozesse in den Zustand „rechnend“ versetzt wird.

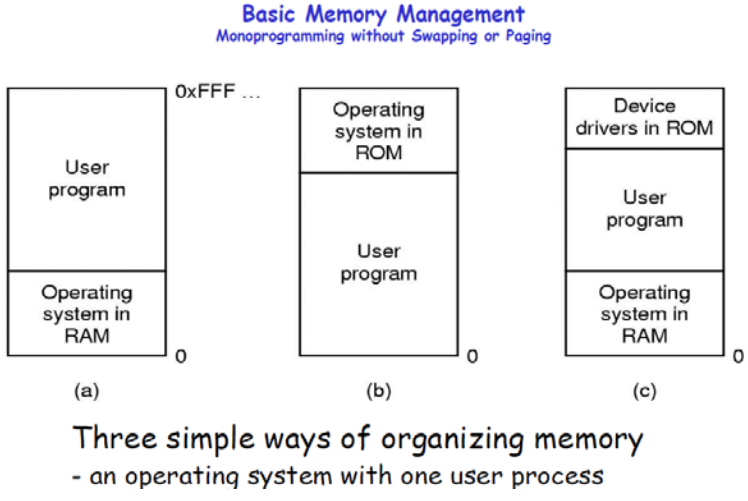
Nonpreemptive Scheduling:

- Ein einmal für den Zustand rechnend ausgewählter Prozess rechnet solange, bis er blockiert oder selbst die CPU freigibt.

Preemptive Scheduling:

- Der Scheduler suspendiert einen rechnenden Prozess nach einer bestimmten Zeitdauer
- Voraussetzung: Zeit-Interrupts durch die Hardware-Uhr

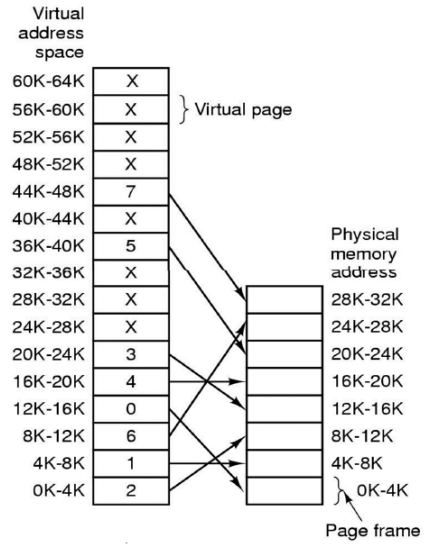
4.3.5 Hauptaufgaben des Speichermanagements



4.3.6 Konzept der virtuellen Adressen

Paging

The relation between **virtual** addresses and **physical** memory addresses is given by a page table



4.3.7 grundsätzlicher Aufbau eines Filesystems

4.3.7.1 Anforderungen an Dateisysteme

- Verwaltung grosser Datenmengen
- Persistenz
- Paralleler Zugriff

4.3.7.2 Implementierung

- Eine Festplatte ist im allgemeinen in Partitionen aufgeteilt.
- Im Master Boot Record (Sektor 0) der Festplatte befindet sich Code, der vom BIOS angestossen wird und der auf eine bestimmte Partition zugreift.
- Im Bootblock der Partition befindet sich der Code zum Starten des OS.
- Im nachfolgenden Superblock befindet sich grundlegende Information zum Dateisystem.
- Die freien Blocks eines Dateisystems werden durch Zeigerlisten oder Bitmaps verwaltet.

4.3.8 programmierter IO vs. Interrupt IO

Interrupts (effiziente Art)

- Treiber gibt die Kontrolle sofort wieder zurück,
- die aufrufende Anwendung wird aber vom OS blockiert
- andere Aufgaben können bis zum Interrupt des Gerätecontrollers ausgeführt werden
- beim Interrupt suspendiert die CPU ihre momentan ausgeführte Aktivität und ruft den (im Gerätetreiber enthaltenen) Interrupthandler auf, der dann z.B. Daten vom Gerätecontroller übernimmt
- spezielle Behandlung von sich überlagernden Interrupts

4.3.9 Dateioperationen inkl. Funktion

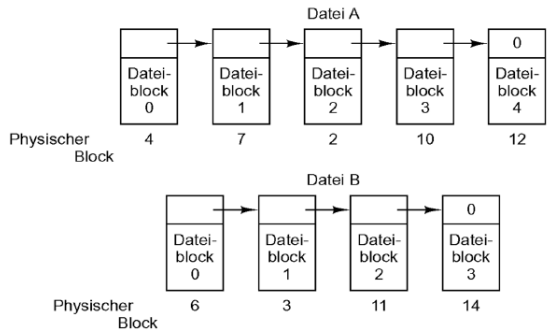
Dateioperationen

- Create: Anlegen der leeren Datei, Setzen von Attributen
- Delete: Freigabe des Platzes auf der Platte
- Open: Lesen der Attribute und Festplattenadressen
- Close: Löschen des internen Speichers für Verwaltungsinformation
- Read: Lesen einer Anzahl Bytes ab der momentanen Position in bereitgestellten Puffer
- Write: Schreiben ab der momentanen Position
- Append: Schreiben am Ende
- Seek: Dateizeigerpositionierung
- Get/Set: Attributzugriff bzw. -änderung
- Rename: Namensänderung

Verzeichnisoperationen

- Create: Anlegen eines leeren Verzeichnisses (nur . und ..)
- Delete: Freigabe des Platzes auf der Platte (nur wenn leer)
- Opendir: Öffnen der Verzeichnisdatei
- Closedir: Löschen des internen Speichers für Verwaltungsinformation
- Readdir: Lesen des nächsten Dateieintrags
- Rename: Namensänderung
- Link: Erzeugen eines symbolischen Dateieintrags auf eine Datei an anderer Stelle (i-node Zähler dieser Datei erhöhen)
- Unlink: Löschen eines Dateieintrages oder Verweises

4.3.10 Funktionsweise einer verketteten Liste

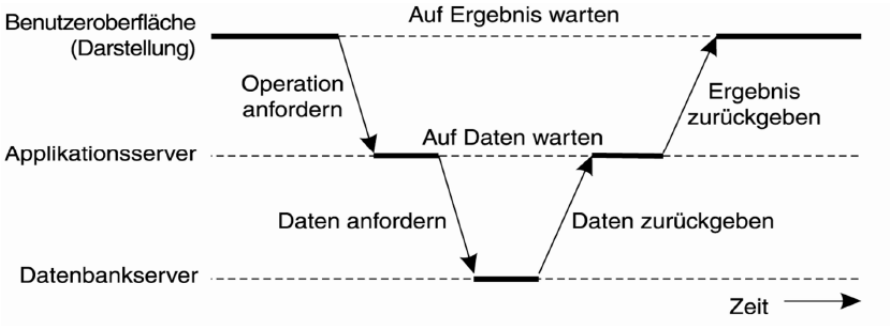


jeder Block kann verwendet werden

zur Identifizierung genügt der 1. Block

Random Access auf Block n erfordert Zugriff auf die Blöcke 1 bis n-1

4.3.11 Client-Server-Konzept in einer Dreistufen-Architektur



5 Programmierung

5.1 Grundlagen

Lernziele
• Sie kennen mindestens drei Programmiersprachen und können das zugehörige Programmierparadigma beschreiben. • Sie können die Funktion eines Java-Compilers (javac) erklären. • Sie können die Funktion einer Java Virtual Machine (JVM) erklären- • Sie verstehen den Entwicklungszyklus Editieren – Kompilieren – Ausführen – Testen. • Sie können den Entwicklungszyklus Editieren – Kompilieren – Ausführen – Testen mit einer Skizze darstellen- • Sie können den Unterschied zwischen .java und .class Dateien erklären.

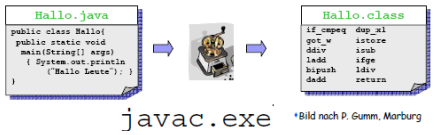
5.1.1 Programmiersprachen und Programmierparadigma

Programmiersprache	Programmier-Paradigma
C • kompiliert • prozedural imperativ, Funktionen, Pointer • typisiert • statisch gebunden • unsicher	**Imperatives Programmier-Paradigma** • Folge von Anweisungen, streng sequentiell abgearbeitet • beschreiben einer Variablen mit einem Wert • Funktionen und Prozeduren
C++ • kompiliert • hybrider Ansatz: prozedural + objektorientiert mit Klassen und Methoden, imperativ, Funktionen, Zeiger & Referenzen, Objekte und Klassen, Vererbung, Überladen, Fehlerbehandlung • typisiert • statisch und dynamisch gebunden • unsicher	**Objektorientiertes Programmier-Paradigma** • Klassen mit Vererbung und Polymorphie • „state of the art“
C# • objektorientiert • interpretiert (wie Java), benötigt .NET Laufzeitumgebung, Common Language Runtime (CLR) • Modularisierung durch Compilation Units • Value-Types (primitive Datentypen wie int, short… → Stack) und Reference-Types (String, Arrays, Objekte → Heap) • Klassen, Interfaces, Delegates • Pointer • Fehlerbehandlung & Exceptions • Multithreading	**Objektorientiertes Programmier-Paradigma** • Klassen mit Vererbung und Polymorphie • „state of the art“

Programmiersprache	Programmier-Paradigma
Java • objektorientiert, ohne prozedurale Elemente mit umfangreicher Klassenbibliothek • interpretiert und kompiliert, plattformunabhängiger Zwischencode (JVM, Hotspotcompiler, Just in Time Compilation) • streng typisiert • statisch und dynamisch gebunden • sicher • imperative Aspekte, Referenzen, Objekte und Klassen, Generische Klassen & Methoden, Vererbung, Fehlerbehandlung • Metadaten	**Objektorientiertes Programmier-Paradigma** • Klassen mit Vererbung und Polymorphie • „state of the art“
PROLOG	**Logisches Programmier-Paradigma** • Datenbasis mit Fakten und Regeln • Fakten sind wahre Aussagen („Axiome“) • Fakten werden in Form von Prädikate angegeben • Die **Problemformulierung**, nicht der Lösungsalgorithmus stehen im Vordergrund
LISP List Processing	**Funktionales Programmier-Paradigma** • Funktionsdefinitionen und Aufrufe • keine Wertzuweisungen **Anwendungen:** künstliche Intelligenz, Compilerbau, Computeralgebra

5.1.2 Java-Compiler (javac)

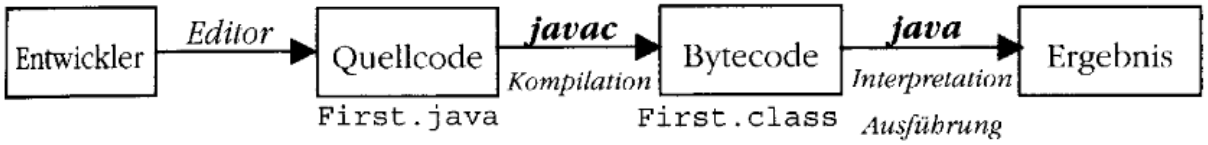
Aus einer Textdatei mit der Endung .java erzeugt der Compiler eine Datei mit gleichem Namen und der Endung .class (Bytecode)



5.1.3 Java Virtual Machine (JVM)

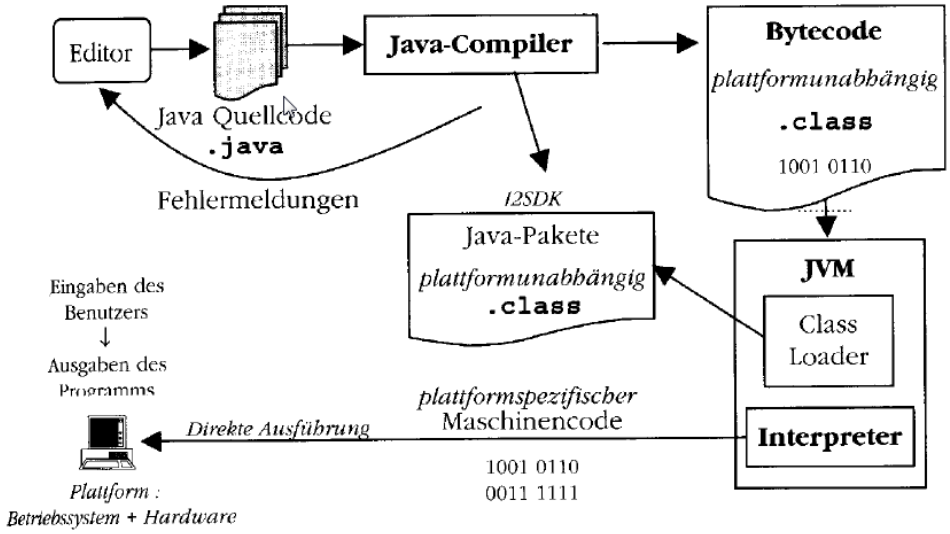
Die Java Virtuel Machine (JVM) interpretiert den Bytecode, d.h. führt den Bytecode aus.

5.1.4 Entwicklungszyklus Editieren – Kompilieren – Ausführen – Testen



Compilieren	Jeder Befehl in einer höheren Programmiersprache geschriebenem Programm wird in eine entsprechende Folge von Maschinenbefehlen übersetzt und in einer Datei gespeichert. Ausgeführt wird das Maschinenprogramm. → gesamtes Programm wird zuerst übersetzt, anschliessend hat man 2 Programme

Interpretieren	Zuerst werden die Befehle der Hochsprache in ein (Pseudo-)Maschinenprogramm übersetzt (compiliert) und abgespeichert. Anschliessend wird dieses (Pseudo)-Maschinenprogramm bei der Ausführung vom Interpreter interpretiert, d.h. ausgeführt



5.1.5 Unterschied .java und .class Dateien

.java	Java-Quellcode
.class	enthält den ausführbaren Bytecode für die Java Virtual Machine (JVM)

5.2 Objektorientierte Programmierung (OOP)

Lernziele

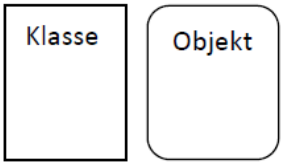
- Sie können den Unterschied zwischen einer Klasse und einem Objekt erklären (Hauptziel).
- Sie kennen den prinzipiellen Aufbau einer Klasse.
- Sie wissen, wozu Instanzvariablen, Methoden und Konstruktoren dienen.
- Sie können die Operatoren +, -, * und / für int und float Variablen anwenden.
- Sie können eine Stringkonkatenation mit „+“ ausführen.
- Sie verwenden System.out.println() in ihren Methoden.
- Sie verstehen, warum es verschiedene Datentypen gibt.
- Sie kennen die Datentypen int und float (vgl. Rechnersysteme).
- Sie können vergleichende Bedingungen formulieren.
- Sie kennen den Begriff „Interface“ (WAS versus WIE).
- Sie kennen die Zugriffsmodifizierer private und public.
- Sie können einfache Schleifen und Fallunterscheidungen programmieren (if, while, for, do).
- Sie können Klassenrahmen anhand einer gegebenen Spezifikation erstellen.
- Sie können eine Zinsberechnung ausführen.
- Sie können an einem gegebenen Array eine einfache Verarbeitung ausführen.

5.2.1 Klasse vs. Objekt

Klasse:
Bauplan für ein Objekt → z.B. Bauplan für ein Auto

Objekt:
Sind Instanzen der Klasse, Abbildungen aus der realen Welt → z.B. roter Ferrari

Von einer Klasse können mehrere Instanzen erzeugt werden.



5.2.2 Aufbau einer Klasse

```
public class Test {

      //1. Instanzvariablen, Attribute, Membervariablen
      //2. Konstruktor
      //3. Methoden

}
```

1. Eigenschaften eines Objekts
2. Initialisierung eines Objekts
3. Verhalten eines Objekts

5.2.2.1 Instanzvariablen, Methoden, Konstruktoren

Instanzvariablen → halten die Eigenschaften eines Objektes fest

- existieren zeitlebens eines Objektes
- sind innerhalb der ganzen Klasse sichtbar/ansprechbar
- besitzt entsprechenden Datentyp¨
- Sie erfordern zur Laufzeit für jedes Objekt Speicherplatz auf dem sogenannten Heap

Methoden → realisiert das Verhalten von Objekten

- Abstraktion
- Wiederverwendung
- Zugriff bzw. Kommunikation/Interaktion unter Objekten ermöglichen
- Signatur gemäss Java-Spezifikation: Methodenname inkl. aller Parameter

Konstruktor → ermöglicht beim Erzeugen das ordentliche Initialisieren eines Objektes

- wird beim Erzeugen des Objektes automatisch aufgerufen
- besitzt keine Parameter
- kein Rückgabewert
- Klasse kann mehrere Konstruktoren besitzen

5.2.2.2 Zugriffsmodifizierer

private

- sichtbar lediglich in der Klasse selbst (z.B. für Attribute)
- für abgeleitete Klassen und für Benutzer von Instanzen verdeckt
- private Elemente immer dann verwenden, wenn implementierungsabhängige Details zu verstecken sind (auch in abgeleiteten Klassen nicht sichtbar)

public

- sichtbar in der Klasse selbst (also in ihren Methoden), in Methoden abgeleiteter Klassen und für den Benutzer von Instanzen der Klasse
- public Elemente bilden die für alle sichtbaren Teile einer Klassendefinition

protected

- Der Zugriff ist auf die enthaltende Klasse oder die von der enthaltenden Klasse abgeleiteten Typen begrenzt.

5.2.3 Operatoren für float / int und Stringkonkatenation

```
int zahl1=5;
int zahl2 = 2;
float kommazahl = 2.0f;
float ergebnis;

ergebnis = zahl1/kommazahl = 2.5
ergebnis = zahl1/zahl2 = 2.0
ergebnis = (float)zahl1/zahl2 = 2.5
ergebnis = (float)(zahl1/zahl2) = 2.0
ergebnis = zahl1/(int)zahl2 = 2.0
```

```
Ausgabe
zahl1/kommazahl = 2.5
zahl1/zahl2 = 2.0
(float)zahl1/zahl2 = 2.5
(float)(zahl1/zahl2) = 2.0
zahl1/(int)zahl2 = 2.0
```

Stringkonkatenation

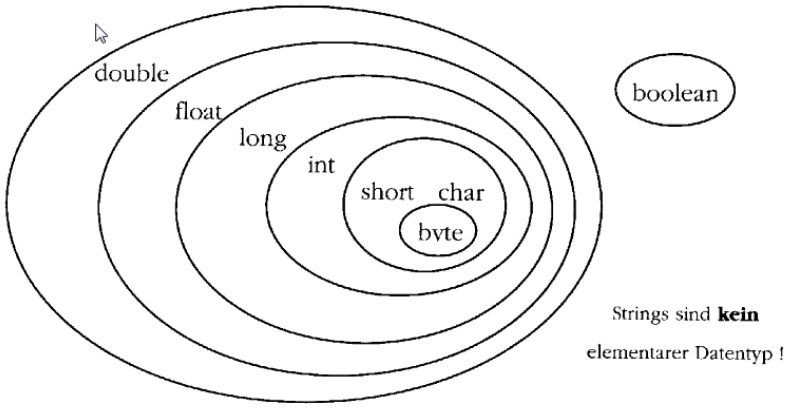
```
String neuesWort;
neuesWort = “Weihnacht“ + “s“ + “mann“;
```

Ausgabe in Konsole

```
System.out.println(„Dies wird ausgedruckt“);
```

5.2.4 Datentypen

Wertebereiche!



Computer speichert Bitmuster (0 und 1) ab. Der Datentyp legt dabei fest, wie dieses Bitmuster interpretiert wird und welche Operationen möglich sind.

implizites Casting: kommaZahl = ganzeZahl;

explizites Casting: kommaZahl = (float) ganze Zahl;

int	4 bytes = 32 bits	-2147483648 bis 2147483647
float	4 bytes = 32 bits	± 3.40282347E±38

5.2.5 der Begriff: Interface (WAS vs. WIE)

JAVADOC

für die Dokumentation des nach aussen sichtbaren Interfaces (was), nicht der Implementation (wie)

Programmierer muss nur wissen, welche Methoden eine Klasse anbietet (WAS), die Implementierungsdetails (WIE) muss er nicht kennen.

Interface (OOP)

Eine Schnittstelle (engl. interface) dient in der objektorientierten Programmierung der Vereinbarung gemeinsamer Signaturen von Methoden, welche in unterschiedlichen Klassen implementiert werden. Die Schnittstelle gibt dabei an, welche Methoden vorhanden sind oder vorhanden sein müssen.

5.2.6 Schleifen und Fallunterscheidungen (Erklärung und Code)

	Eigenschaften	Code

if	• Viele Aktionen erfolgen in Abhängigkeit von Bedingungen	`if(day>=1 && day<=31)`
		`{`
		`  return true;`
		`}`
		`else`
		`{`
		`  return false;`
		`}`

while	• Anzahl Schleifendurchläufe unbekannt
	• eventuell gar kein Schleifendurchlauf
	Bsp. „Zeichenweises Einlesen einer Datei“
		`int i = 1;`
		
		`while (i <= numberOfHellos) {`
		`  System.out.println(hello+"while-Schleife");`
		`  i = i + 1;`
		`}`

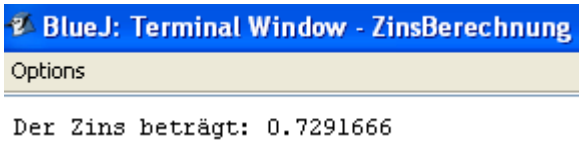
do-while	• Anzahl Schleifendurchläufe unbekannt
	• mindestens 1 Schleifendurchlauf
		`int i = 1;`
		
		`do`
		`{`
		`  System.out.println(hello+"do-Schleife");`
		`  i++;`
		`}while(i<=3);`

for	• Anzahl Schleifendurchläufe bekannt oder im Voraus berechenbar (eventuell erst zur Laufzeit)
	• eventuell überhaupt kein Schleifendurchlauf
	Bsp: „Fakultät berechnen“
		`for(int i = 1;i<=3;i++)`
		`{`
		`  System.out.println(hello+"for-Schleife");`
		`}`

5.2.7 Zinsberechnung (Code)

```
/**
 * Klasse für die Zinsberechnung
 * @author Flavio De Roni
 * @version V 1     08-NOV-2010
 * @version V 1.1   23-NOV-2010 (Gruppe Alfa)
 */
public class ZinsBerechnung
{
    private EinAusZahlung[] journal = new EinAusZahlung [10];
    private float rate = 0.125f;
    /**
     * Constructor for objects of class ZinsBerechnung
     */
    public ZinsBerechnung()
    {
        // initialise instance variables
        journal [0] = new EinAusZahlung(01, 02, 2010,  +500);
        journal [1] = new EinAusZahlung(01, 04, 2010,  -200);
        journal [2] = new EinAusZahlung(01, 04, 2010,  +700);
        journal [3] = new EinAusZahlung(01, 05, 2010,  -500);
        journal [4] = new EinAusZahlung(01, 06, 2010, +1000);
        journal [5] = new EinAusZahlung(01, 06, 2010,  -500);
        journal [6] = new EinAusZahlung(01, 06, 2010,  -500);
        journal [7] = new EinAusZahlung(01, 07, 2010,  +500);
        journal [8] = new EinAusZahlung(01, 9,  2010, +1000);
        journal [9] = new EinAusZahlung(01, 10, 2010, -2000);
```

```
    }

    /**
     * Methode zum Berechnen des Zinses
     *
     * @param  journal   Ein-/Auszahlungen
     * @return zins
     */
    public float berechneZins(EinAusZahlung[] journal)
    {
        //Zum festhalten des Saldo/Zins für ein Jahr (12 Monate)
        float[] zins = new float[12];
        float[] kontostand = new float[12];

        int m = 0;
        //Wertveränderung pro Monat bestimmen
        for(int i=0;i<journal.length;i++)
        {
            m = journal[i].getMonth();
            kontostand[m-1] = kontostand[m-1] + journal[i].getUmsatz();
        }
        //Saldo Ende Monat und Zins berechnen
        for(int i=0; i<kontostand.length; i++)
        {
            if(i!=0)
            {
               kontostand[i] = kontostand[i] + kontostand[i-1]
            }
            tot_zins += getZinsforPeriod(kontostand[i],30);
        }

        return tot_zins;

    }

    private float getZinsforPeriod(float kapital, int days)
    {
        return (kapital * rate * days)/(100*360);
    }

    public void testCaseZinsRechnung()
    {
        float zins;
        zins = berechneZins(journal);
        System.out.println("Der Zins beträgt: " + zins);
    }

}
```



5.2.8 einfache Verarbeitung an gegebenem Array (Code)

Beispiel vom Testat 3 (HS.2010):
Schreiben Sie Java-Anweisungen, die prüfen, ob der Kontostand während des zeitlichen Ablaufs einmal negativ war (Gegeben ist Array konto, welcher die zeitliche Abfolge der Transaktionen darstellt). Ihr Programmstück soll nach der Prüfung eine der beiden folgenden Ausgaben machen:

- „Konto war immer gedeckt“ oder
- „Konto war nicht immer gedeckt“

```
int n = 5; //Anzahl der Transaktionen
int[] konto = new int[n];
int anz;

//Der Array konto und die Variable n enthalten bereits Werte
//Nachfolgend soll ihre Lösung stehen:


//Lösung
boolean gedeckt = true;
int kontostand = 0;

for(int i=0;i<konto.length;i++)
{
     kontostand += konto[i];
     if(kontostand<0)
     {
          gedeckt = false;
          break;
     }
}

if(gedeckt)
{
     System.out.println(“Konto war immer gedeckt.”);
}
else
{
     System.out.println(“Konto war nicht immer gedeckt.”);
}
```

6 Datenbanken

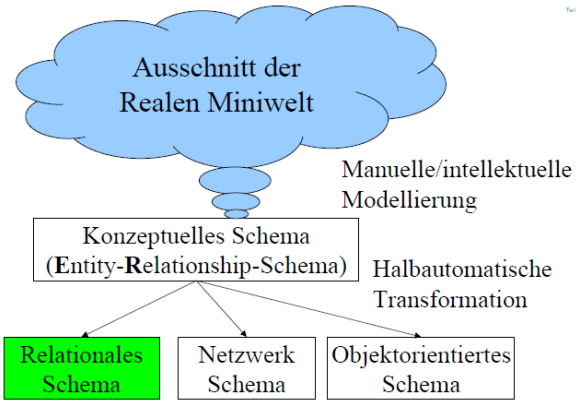
Lernziele
• Sie können ein (einfaches) Datenmodell nach dem Entity-Relationship Modell erstellen. • Sie können eine (einfache) DB kreieren (Tabellen-Definitionen). • Sie können einfache DB-Abfragen mit SQL formulieren (Select-statement). • Sie können das Transaktionskonzept auf Stufe Datenbank erklären/ nutzen.

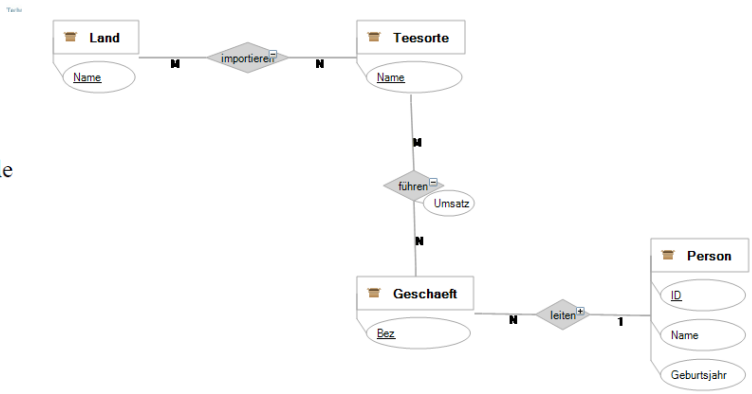
Was soll ein DBMS leisten/vermeiden:

- Redundanz und Inkonsistenz
- Beschränkte Zugriffsmöglichkeiten
- Probleme beim Mehrbenutzerbetrieb
- Verlust von Daten
- Integritätsverletzung
- Sicherheitsprobleme
- hohe Entwicklungskosten für Anwendungsprogramme

6.1 Entity-Relationship-Model (ERM)

Das Entity-Relationship-Modell, kurz ER-Modell oder ERM (deutsch wörtlich Gegenstand-Beziehung-Modell, diese Bezeichnung ist aber nicht gebräuchlich) dient dazu, im Rahmen der semantischen Datenmodellierung einen Ausschnitt der realen Welt zu beschreiben.



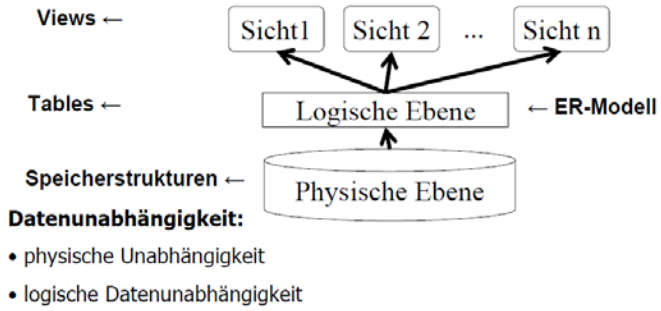


6.1.1 Kardinalitäten-Angabe

- c (can): 0 bis 1; Achtung: auch als 1 dargestellt
- 1 (one oder eins): genau 1;
  Achtung: auch mit Bedeutung 0 bis 1
- mc (multiple can): 0 bis viele; auch: M,N,P
- m (multiple): 1 bis viele

Wichtig: Bei n:n-Beziehungen gibt es eine eigene Tabelle!

6.2 Tabellendefinition (CREATE TABLE)



Keine Redundanz: wichtig für einfache Aufrechterhaltung der Datenintegrität (keine Widersprüche), Verstoss nur aus wichtigen Performance-Gründen zulässig!

6.2.1 Code

```
CREATE TABLE alfa01_Person(
ID INTEGER NOT NULL PRIMARY KEY,
Name VARCHAR(50) NOT NULL
);

CREATE TABLE alfa01_Auftrag(
ID INTEGER NOT NULL PRIMARY KEY,
Datum DATETIME NOT NULL,
FK_Parent INTEGER NULL,
FOREIGN KEY(FK_Parent) REFERENCES alfa01_Auftrag(ID)
   ON UPDATE CASCADE
   ON DELETE CASCADE
);

CREATE TABLE alfa01_AuftragsBearbeitung(
ID INTEGER NOT NULL PRIMARY KEY,
FK_Person INTEGER NOT NULL,
FK_Auftrag INTEGER NOT NULL,
FOREIGN KEY(FK_Person) REFERENCES alfa01_Person(ID)
   ON DELETE CASCADE,
FOREIGN KEY(FK_Auftrag) REFERENCES alfa01_Auftrag(ID)
   ON UPDATE CASCADE
   ON DELETE CASCADE
);
```

6.2.2 weitere Befehle (INSERT, UPDATE, DELETE, DROP…)

6.2.2.1 INSERT → Zeile einfügen

INSERT INTO <tableName>(<Attribut-Liste>)
VALUES(<Werte-liste>)

6.2.2.2 UPDATE → Daten ändern

update <tableName> set <AttributName> = <Ausdruck> {,<AttributName> = <Ausdruck> }
[where …]

6.2.2.3 DELETE → Daten löschen

Delete from <tableName> [where …]

6.2.2.4 DROP → Struktur und Inhalt löschen

drop database/table <name>

6.2.3 Data-Definition und Data-Manipulation-Language

CREATE gehört zur DDL (Data Definition Language)
INSERT, UPDATE, DELETE gehört zur DML (Data Manipulation Language)

6.3 Datenbank-Abfragen (SELECT)

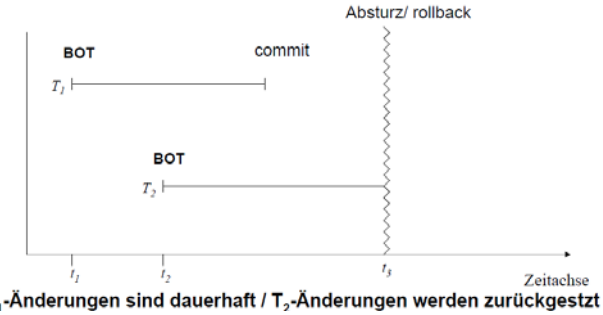
Hier folgen verschiedene SELECTs. Zum Nachvollziehen empfiehlt es sich die Befehle mit MySQL-Connect und der e-lab-DB durchzuspielen:

```
SELECT raum from tks.professoren where name = "Curie";

SELECT vorlnr as 'VorlesungsNummer' FROM tks.hoeren WHERE matrnr = 29120;

SELECT vorlnr, titel from tks.vorlesungen ORDER BY titel ASC;

SELECT titel, gelesenvon as 'PersNr' FROM tks.vorlesungen ORDER BY gelesenvon ASC,
titel DESC;

SELECT assistenten.name, professoren.name as 'BOSS' FROM tks.assistenten,
tks.professoren WHERE assistenten.boss = professoren.persnr;

SELECT assistenten.name, professoren.name as 'BOSS' FROM tks.assistenten JOIN
tks.professoren on assistenten.boss = professoren.persnr;

SELECT p.*, a.name as 'Assistent', a.fachgebiet from tks.professoren p LEFT JOIN
tks.assistenten a on p.persnr = a.boss;

SELECT DISTINCT p.Name as 'NameProfessorvonVorlesungStudentCarnap' FROM
(tks.studenten s JOIN tks.hoeren h on s.matrnr = h.matrnr)
JOIN tks.vorlesungen v on h.vorlnr = v.vorlnr JOIN tks.professoren p on p.persnr =
v.gelesenvon
WHERE s.name = 'Carnap';

select p.Name from tks.professoren p where not exists ( select * from assistenten a where
a.boss =  .persnr );

SELECT s.name FROM tks.pruefen p JOIN tks.studenten s ON p.matrnr = s.matrnr group by
p.note having max(p.note);


SELECT s.name FROM tks.pruefen p JOIN tks.studenten s ON p.matrnr = s.matrnr group by
p.note having min(p.note);
```

6.4 Transaktionskonzept auf Stufe Datenbank

- Atomicity (Atomarität)
  Alles oder nichts
- Consistency (Konsistenz)
  Konsistenter Zustand -> konsistenter Zustand
- Isolation
  gleichzeitige Benutzer „stören" einander nicht
- Durability (Dauerhaftigkeit)
  Erfolgreiche Änderungen gehen nicht verloren

Transaktions-Szenario



T_1-Änderungen sind dauerhaft / T_2-Änderungen werden zurückgestzt

Begin/end of transaction

- **begin of transaction (BOT)**: Mit diesem Befehl wird der Beginn einer Befehlsfolge (Transaktion) gekennzeichnet.
- **commit**: Hierdurch wird die Beendigung der Transaktion eingeleitet. Alle Änderungen der Datenbasis werden durch diesen Befehl festgeschrieben, d.h. sie werden dauerhaft in die Datenbank eingebaut.
- **rollback**: Dieser Befehl führt zu einem Selbstabbruch der Transaktion. Das Datenbanksystem muss sicherstellen, dass die Datenbasis wieder in den Zustand zurückgesetzt wird, der vor Beginn der Transaktionsausführung existiert.

- Transaktionen ermöglichen gutes Recovery
- Log-Protokoll für Transaktionen (before & after-image)