HOCHSCHULE
KONSTANZ
TECHNIK, WIRTSCHAFT
UND GESTALTUNG

Verteilte Systeme
Vorlesung 02

22. März 2017

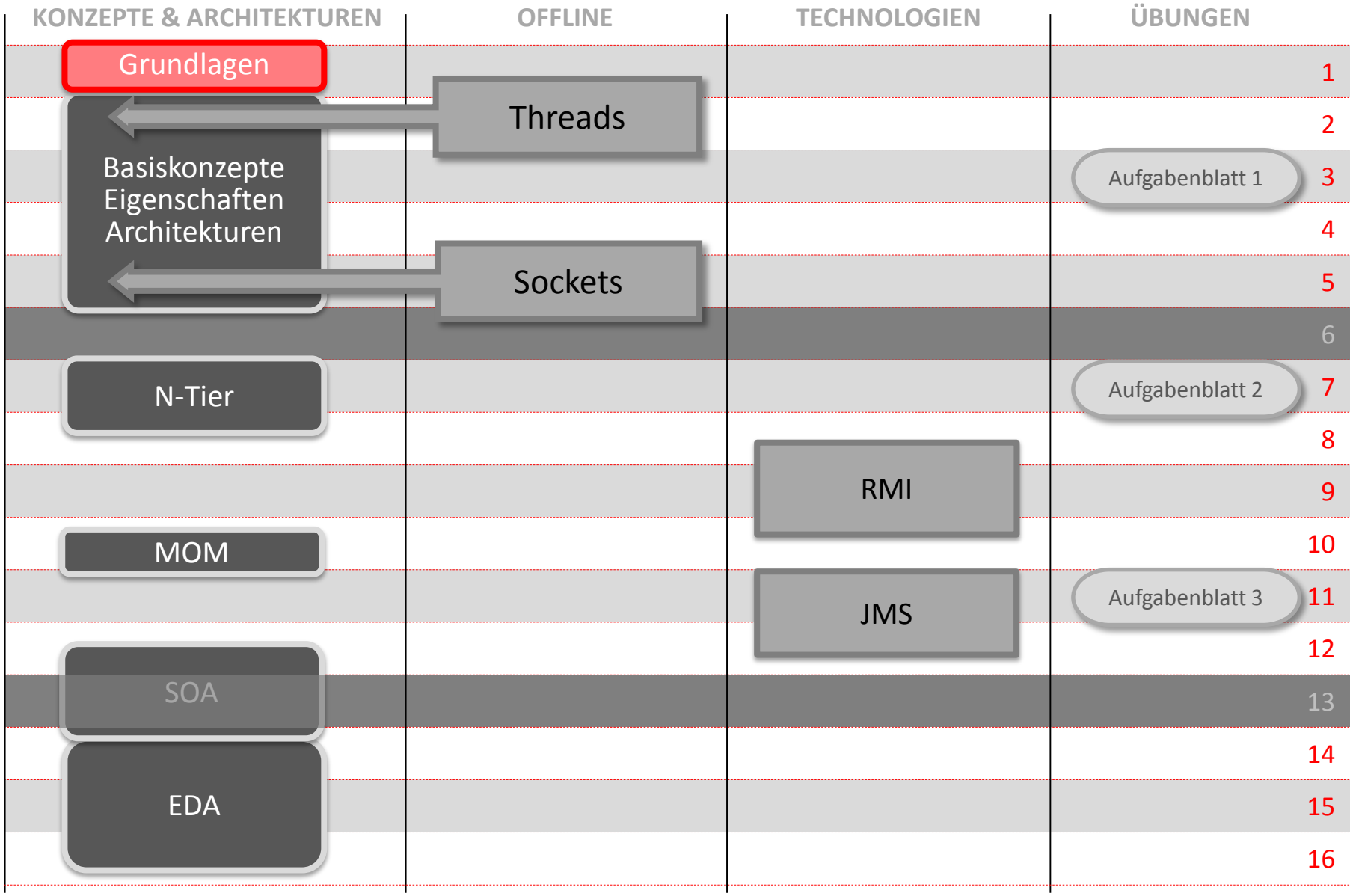
Administratives
Kapitel 1: Grundlagen
Kapitel 3: Basiskonzepte, Eigenschaften, Architekturen

Prof. Dr. Rainer Mueller

SS 2017

Hochschule Konstanz | Brauneggerstr. 55 | 78462 Konstanz | www.htwg-konstanz.de

Vorlesung: Übersicht



KAPITEL 1

Grundlagen

Übersicht

1.3.1 Konsequenzen aus der Definition

Essenz aus der Definition „Verteiltes System“:

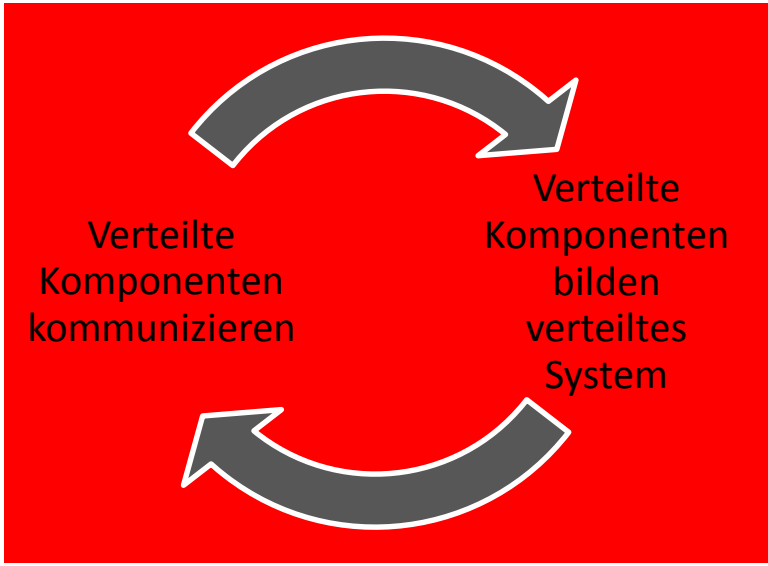
Komponenten auf vernetzen Rechnern kommunizieren (ausschließlich über Nachrichten)

NEBENLÄUFIGKEIT

- Nebenläufige Ausführung von Komponenten mit geteiltem Ressourcenzugriff
  → Synchronisierung der Komponenten erforderlich
  → Zusätzliche Ressourcen steigern Gesamtkapazität des verteilten Systems

FEHLENDE GLOBALE UHR

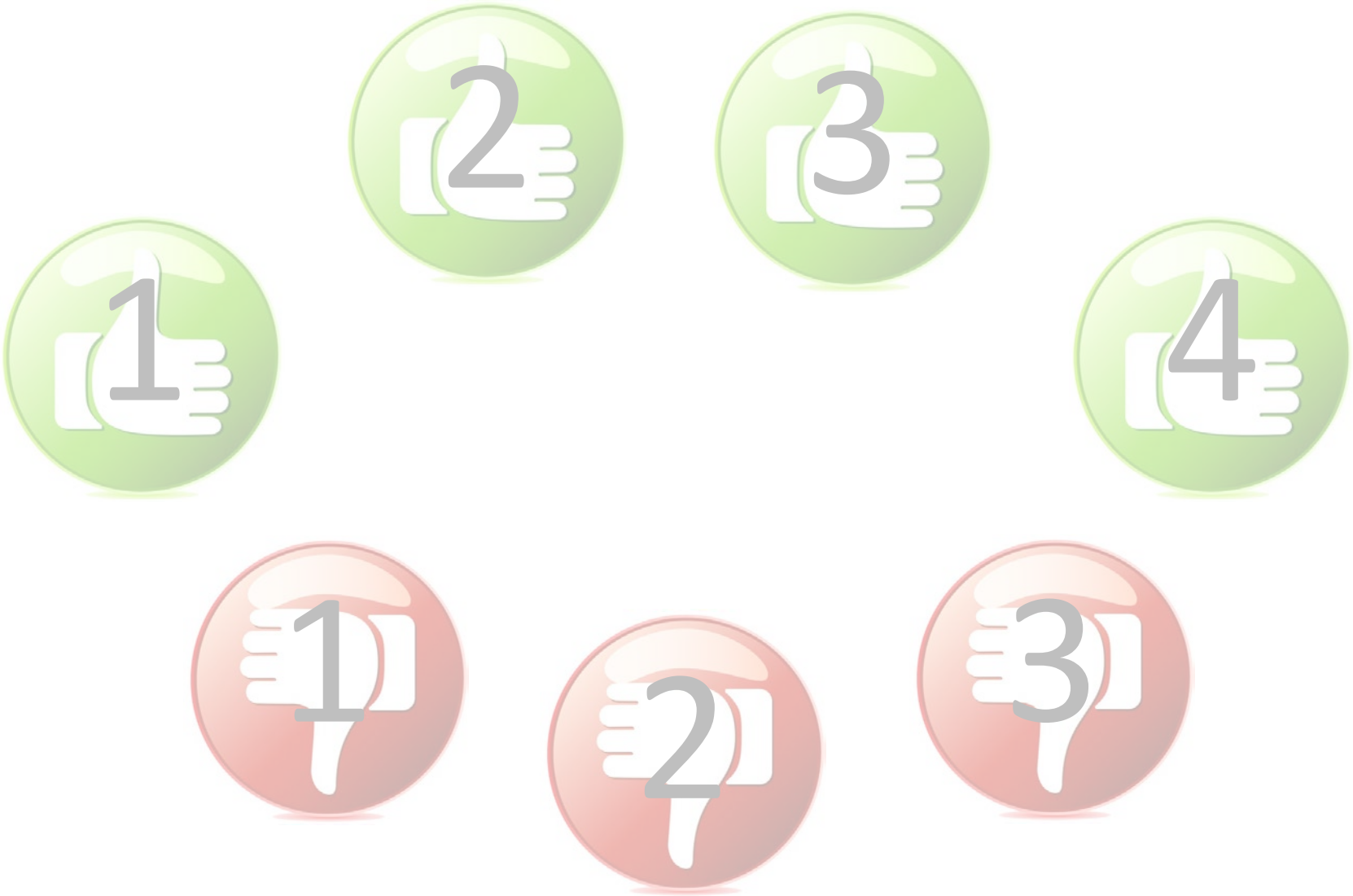
- Kein globales Zeitkonzept: Synchronisierung der Uhren auf Teilnehmersystemen schwierig
  → Koordination von Komponenten nur über Nachrichteninhalt, nicht über Nachrichtenzeitpunkt
  → Keine Koordination von Komponenten ohne Nachrichten (also z.B. nur durch Zeit)



KOMPONENTENAUSFALL

- Komponenten fallen unabhängig aus: Andere Komponenten funktionieren weiter
  → Komponenten bemerken Ausfall nicht und reagieren nicht darauf
  → Einzelkomponenten isoliert
  → Gesamtsystem wird langsamer oder fällt aus

1.3 Konsequenzen, Vor- und Nachteile



1.3.2 Unterstützung verteilter Aufgabenstellungen

AUFGABEN/FUNKTIONEN MIT GEOGRAPHISCHER VERTEILUNG

Aufgaben meist inhärent verteilt

- Videokonferenz-Systeme
- Internet-Telefonie
- Fahrzeugleit-Systeme (Autonomens Fahren)

AUFGABEN/FUNKTIONEN MIT ORGANISATORISCHER VERTEILUNG

- Systeme zur Unterstützung von verteilten Geschäftsprozessen
- Entfernter Zugriff auf zentralisierte Unternehmensinformationssysteme (*Beispiel: ERP-Systeme*)
- Verteilte Unternehmensinformationssysteme (*Beispiel: Information entsteht an unterschiedlichen Standorten*)

1.3.3 Robustheit, Zuverlässigkeit, Verfügbarkeit

- Verwendung von redundanten Datenkopien (verteilte Datenbanken)
- Zusätzliche Standby-Rechner
- Lastverteilung durch Rechner-Cluster

  → Kürzere Antwortzeiten durch Parallelisierung im Vergleich zu zentraler Lösung

verteilung Möglich

1.3.4 Unterstützung spezieller Hardware

GRÜNDE FÜR SPEZIELLE HARDWARE

- **Rechenleistung:** Hochleistungsrechner für rechenintensive Aufgaben
- **Daten mit eingeschränktem Zugriff:** Verwendung von vertraulichen Daten/Information mit Zugriffsschutz macht es sinn Server zu kaufen? oder macht es eher Sinn viele kleine zu Nutzen?
- **Kosten:** Ausnutzung günstigerer Hardware-/Rechen-/Softwareressourcen (→ Cloud Computing)
- **Wirtschaftlichkeit:** Zentrales leistungsstarkes Einzelsystem teurer, als viele verteilte leistungsschwache Systeme

1.3.5 Ausnutzung freier Rechenressourcen

GRID COMPUTING/DISTRIBUTED COMPUTING

- **@home-Projekte:** Simulation, Wetter, Pharma
- **seti@home** (Berkeley): **S**earch for **e**xtra**t**errestrial **i**ntelligence **at home**
  - > 1 Mio. teilnehmende Computer
  - Über 700 TeraFLOPS (Größenordnung eines Höchstleistungsrechners, Wert von 2009)
  - Seit 1999: 2,3 Mio. Jahre Rechenzeit
  - setiathome.ssl.berkeley.edu
- BOINC-Plattform (**B**erkeley **O**pen **I**nfrastructure for **N**etwork **C**omputing) für Projekte im Bereich Distributed Computing (Grid-Computing)
  - Plattform für projektweise Bereitstellung von ungenutzter (privater) Rechnerleistung

Startschuss in Stuttgart: Schnellster ziviler Supercomputer Europas eingeweiht

27. Februar 2012

Hermits Rechenleistung liegt im Petaflop-Bereich und dient Forschung, Entwicklung und Industrie.

Mit einem feierlichen Festakt wurde vergangenen Freitag am Höchstleistungsrechenzentrum Stuttgart (HLRS) der schnellste Supercomputer Deutschlands und zugleich schnellste zivil genutzte Rechner Europas offiziell für die Nutzer freigegeben. Bundesforschungsministerin Annette Schavan und Ministerpräsident Winfried Kretschmann gaben im Beisein des Rektors der Universität Stuttgart, Wolfram Ressel, den Startschuss für die Inbetriebnahme des Rechnersystems Hermit. Mit einer Leistung von mehr als 1 Petaflop pro Sekunde (1 Billiarde Rechenoperationen pro Sekunde) zählt es zu den leistungsfähigsten Supercomputern der Welt.

Quelle: www.nachrichten.de

Quelle: Badische Zeitung

1.3 Konsequenzen, Vor- und Nachteile

Distribution means Sharing

Verteilen bedeutet auch Teilen

Duden: **verteilen**

Wortart: **i** schwaches Verb

Häufigkeit: **i▮▮▮▯▯**

Synonyme

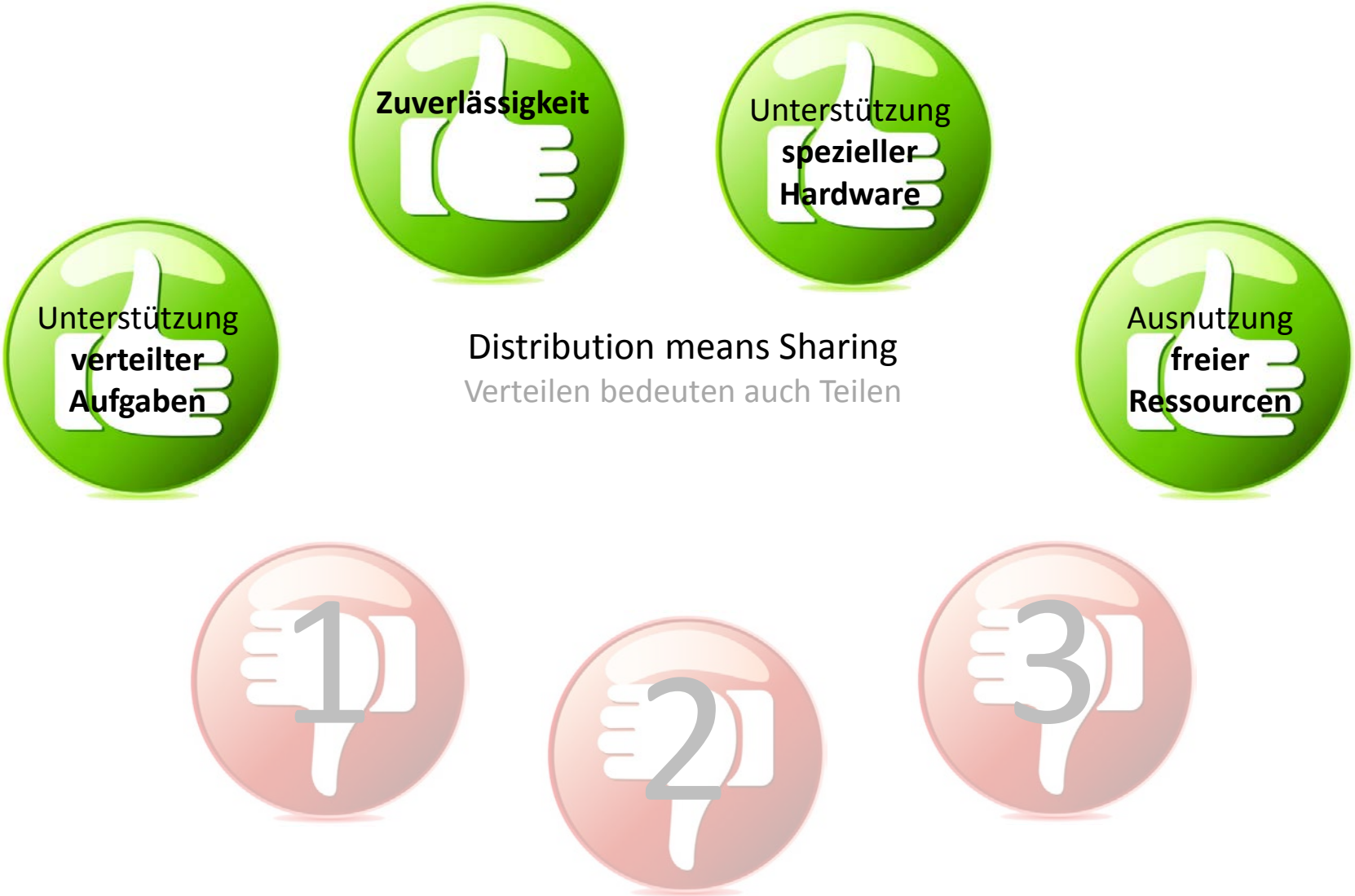
1. sich ausbreiten, auseinandergehen, auseinanderlaufen, ausschwärmen, sich verlaufen, sich zerstreuen
2. abgeben, ausgeben, aushändigen, austeilen, geben, vergeben; (bildungssprachlich) distribuieren
3. aufteilen, einteilen, zuteilen, zuweisen

Bedeutungsübersicht

1. [aufteilen und in einzelnen Anteilen, Portionen o. Ä.] an mehrere Personen vergeben, austeilen
2. aufteilen und in gleicher Menge oder Anzahl an verschiedene Stellen bringen, legen, stellen usw., irgendwo unterbringen

…

1.3 Konsequenzen, Vor- und Nachteile



Übersicht

1.3.6 Kosten für Kommunikation

GRÜNDE FÜR KOSTEN

- **Latenzzeit** (ggf. relevant für Echtzeit-Daten)
- **Jitter** (ggf. relevant für Echtzeit-Daten)
- **Netzbelastung**
- **Ausfall**

Nachteile: Kommunikation kostet (kein Geld aber Zeit)

Typische Klausurfrage
Jitter
-> schwankungen bei der Datenübertragung pro Zeiteinheit
-> Problemstellung bei Echtzeit Anwendungen (z.B. Aktienhandel)
-> Bei Ton/ Bild übertragen

1.3.7 Komplexität bei Entwicklung

	Vorlesung	Übung
ENTWURF		
• Verteilungsmodell: Geeignete Wahl	x	x
• Kommunikationsmechanismen	x	x
IMPLEMENTIERUNG		
• Neue Fehlerquellen: Ausbleiben der Antwort (Timer erforderlich); Unerwartete Nachricht		x
• Nebenläufigkeit: (Thread-)Synchronisierung für asynchr. Verarbeitung	x	x
TESTEN		
• Simulation entfernter Prozesse (inklusive Ausnahmenbehandlung)		x
• Integrationstest mit komplexen Setups der Komponenten		
• Vielzahl von möglichen Interaktionen, Fehlerfällen, Konfigurationen		x
FEHLERSUCHE/WARTUNG		
Fehlerreproduktion: Komplexe Interaktion vieler Prozesse erforderlich		x
Beispiel Breakpoints: Anhalten problematisch, wenn entfernte Prozesse mit Timern auf Antwort warten		

1.3.8 Komplexität bei Betrieb

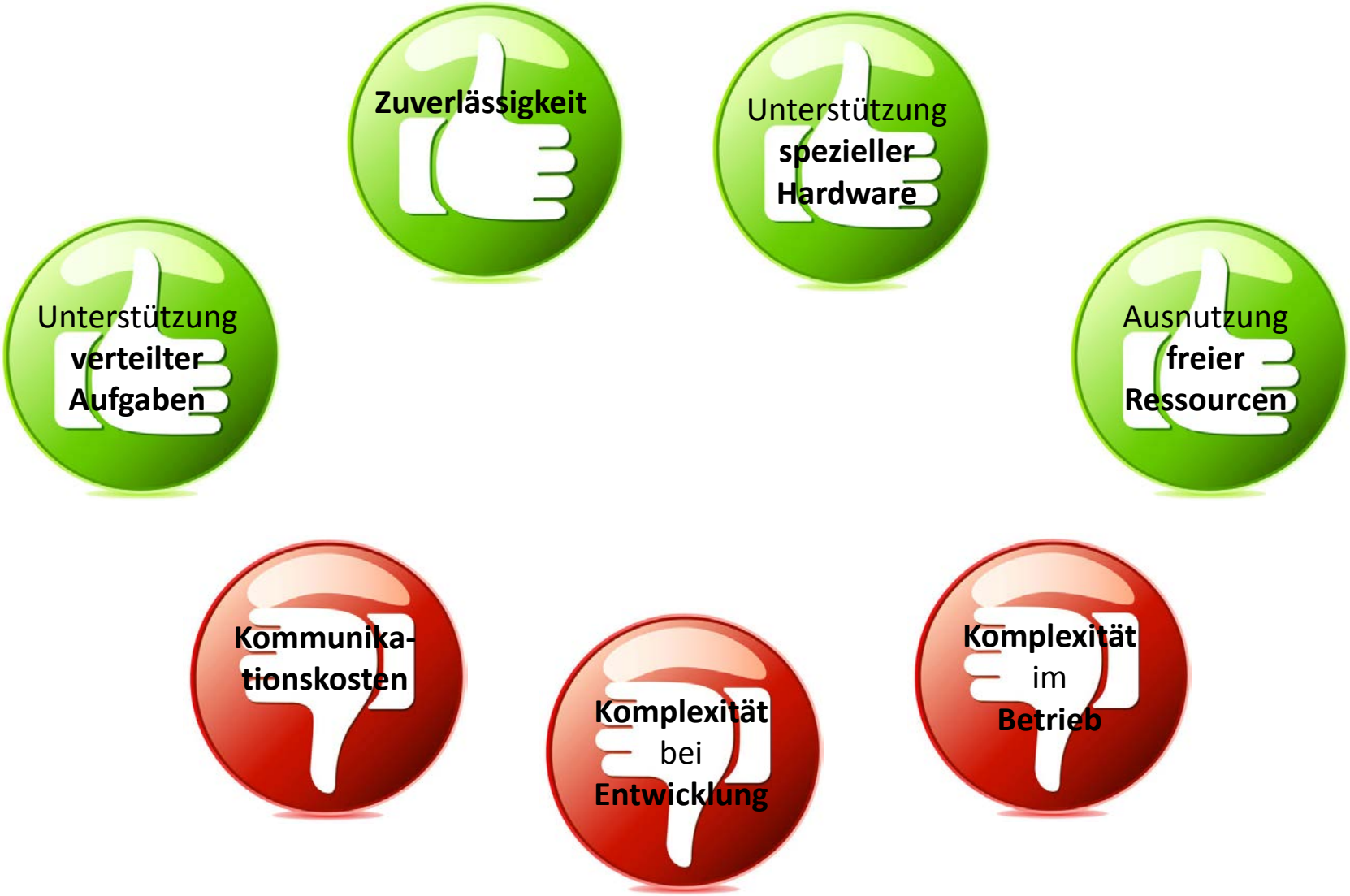
PROBLEME BEI BETRIEB

- Verfügbarkeit: Fehlende Reaktion, Absturz, undefinierter Zustand der Anwendung (Rechnerausfall?)
- Sicherheit: Komponenten und Kommunikationswege angreifbar

PROBLEME BEI VERSIONIERUNG UND KONFIGURATION

- Ggf. unterschiedliche Versionierungszyklen für einzelne Komponenten
  - Ggf. Abwärtskompatibilitäten erforderlich
  - Ggf. Austausch von Komponentenversionsnummern bei Kommunikation erforderlich

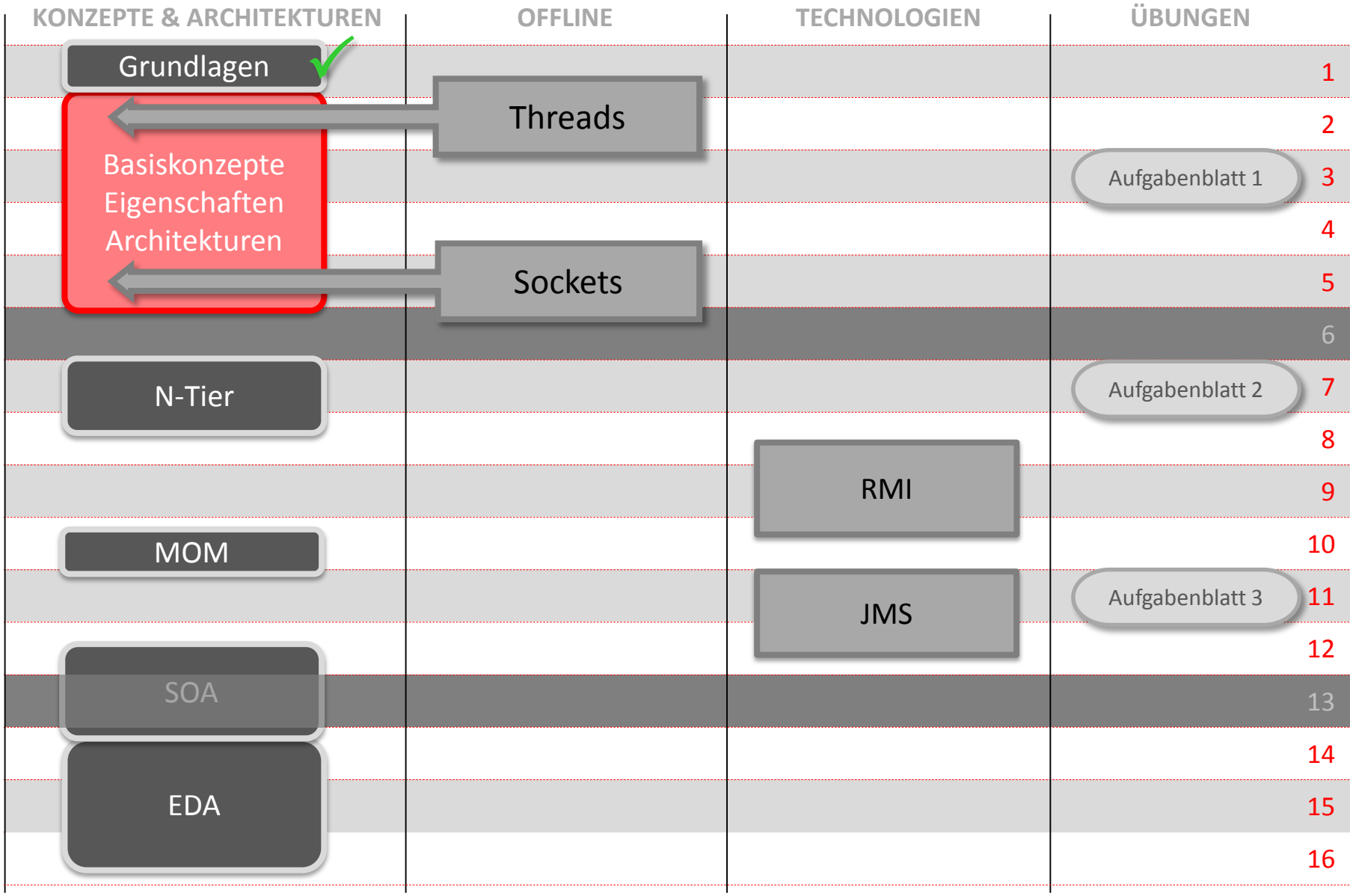
1.3 Konsequenzen, Vor- und Nachteile



Übersicht

- ✓ **1.1 MOTIVATION**
  - ✓ 1.1.1 Lokale Schranken
  - ✓ 1.1.2 Netzausbau
  - ✓ 1.1.3 Verteiltere Welt
  - ✓ 1.1.4 Technologischer Fortschritt
- ✓ **1.2 DEFINITION**
  - ✓ 1.2.1 Zitate
  - ✓ 1.2.2 Verteilt vs. zentralisiert
- ✓ **1.3 KONSEQUENZEN, VORTEILE UND NACHTEILE**
  - ✓ 1.3.1 Konsequenzen aus der Definition
  - ✓ 1.3.2 Unterstützung verteilter Aufgabenstellungen
  - ✓ 1.3.3 Robustheit, Zuverlässigkeit, Verfügbarkeit
  - ✓ 1.3.4 Unterstützung von Spezial-Hardware
  - ✓ 1.3.5 Ausnutzung freier Rechenressourcen
  - ✓ 1.3.6 Kosten für Kommunikation
  - ✓ 1.3.7 Komplexität bei Entwicklung
  - ✓ 1.3.8 Komplexität bei Betrieb

Vorlesung: Übersicht



KAPITEL 3

Basiskonzepte, Eigenschaften und Architekturen

K A P I T E L 3

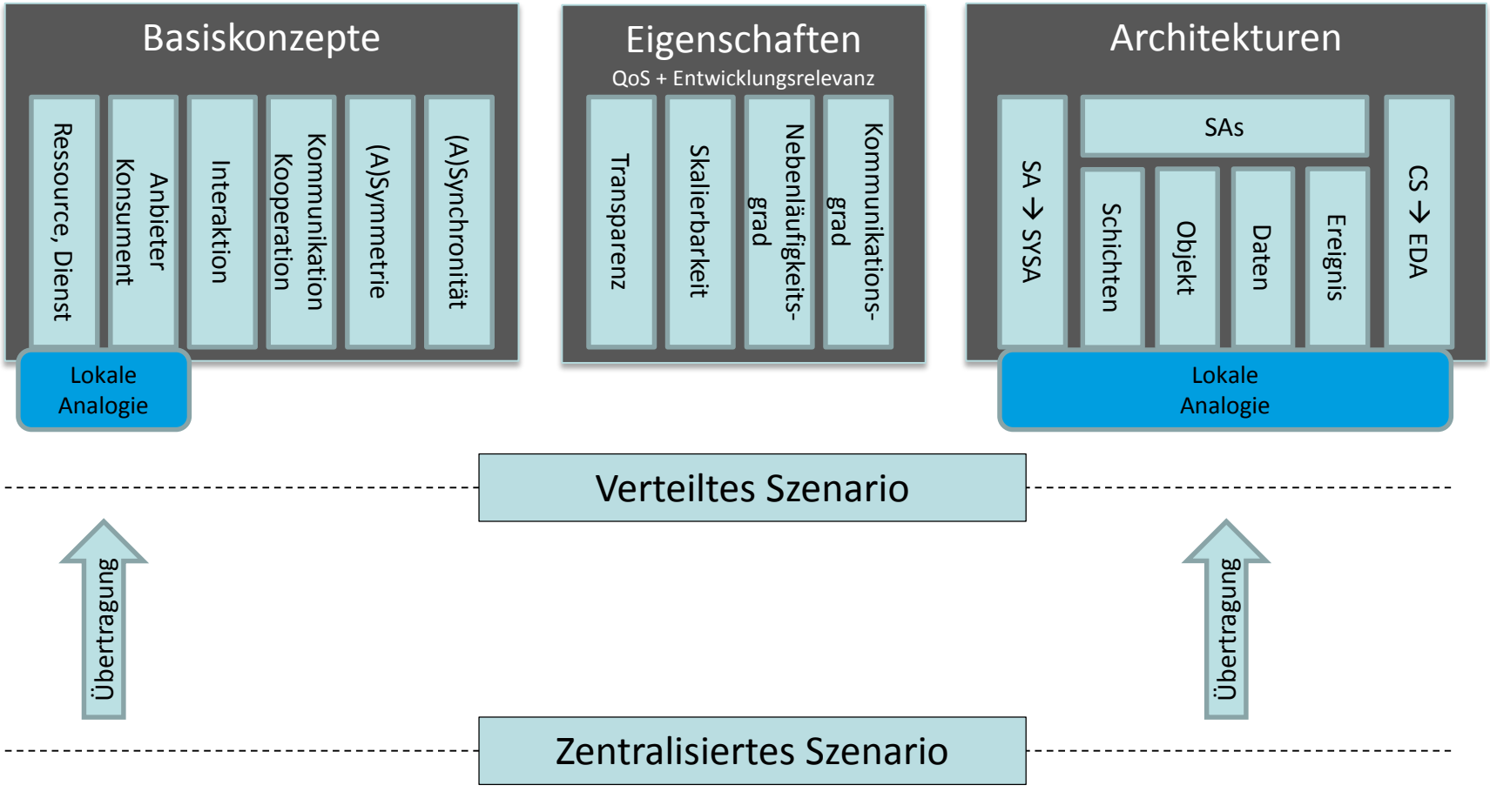
Übergeordnete Frage

Worin unterscheidet sich der verteilte Fall
vom zentralisierten Fall,
den wir gut kennen?

Übersicht

K A P I T E L 3

Vom zentralen Fall übertragbare Basiskonzepte, daraus ableitbare grundlegende Eigenschaften, erste teilweise auch zentral gültige Strukturen und Architekturelemente



Übersicht

K A P I T E L 3

Vom zentralen Fall übertragbare Basiskonzepte, daraus ableitbare grundlegende Eigenschaften, erste teilweise auch zentral gültige Strukturen und Architekturelemente

Übersicht

BASISKONZEPTE

3.1 RESSOURCEN UND DIENSTE

- 3.1.1 Prinzipien und Phasen für Verteilung
- 3.1.2 Ressource
- 3.1.3 Dienst
- 3.1.4 Anbieter und Konsument

3.2 CLIENT UND SERVER

- 3.2.1 Fundamentalverteilung: Client-Server
- 3.2.2 Netzwerkebene

3.3 INTERAKTIONSMODELLE

3.4 (A)SYNCHRONITÄT

- 3.4.1 Anfrage und Antwort
- 3.4.2 Multiple Anfragen

EIGENSCHAFTEN

3.4 KOHÄRENZ UND TRANSPARENZ

- 3.4.1 Definition
- 3.4.2 Transparenz von Verteilungseigenschaften
- 3.4.3 Bedeutung und Realisierbarkeit

3.5 SKALIERBARKEIT

- 3.5.1 Definition und Typen
- 3.5.2 Typ: Größe
- 3.5.3 Typ: Geographie
- 3.5.4 Typ: Administration

3.6 NEBENLÄUFIGKEITS-GRAD

- 3.6.1 Bedeutung und Konsequenz
- 3.6.2 Ausschlussverfahren
- 3.6.3 Grad und Auswirkung

3.7 KOMMUNIKATIONS-GRAD

- 3.7.1 Schalenmodell
- 3.7.2 Sockets
- 3.7.3 Nachrichten
- 3.7.4 Entfernter Prozeduraufruf
- 3.7.5 Entfernter Methodenaufruf
- 3.7.6 Runtime, Dienste, Komponenten

ARCHITEKTUREN

3.8 SOFTWARE- UND SYSTEM-ARCHITEKTUR

- 3.8.1 Software-Architektur

Übersicht

- 3.8.2 System-Architektur
- 3.8.3 Software-Architektur → Systemarchitektur

3.9 ZENTRALISIERTE SOFTWARE-ARCHITEKTUREN

- 3.9.1 Architekturstile
- 3.9.2 Zugehörige System-Architekturen

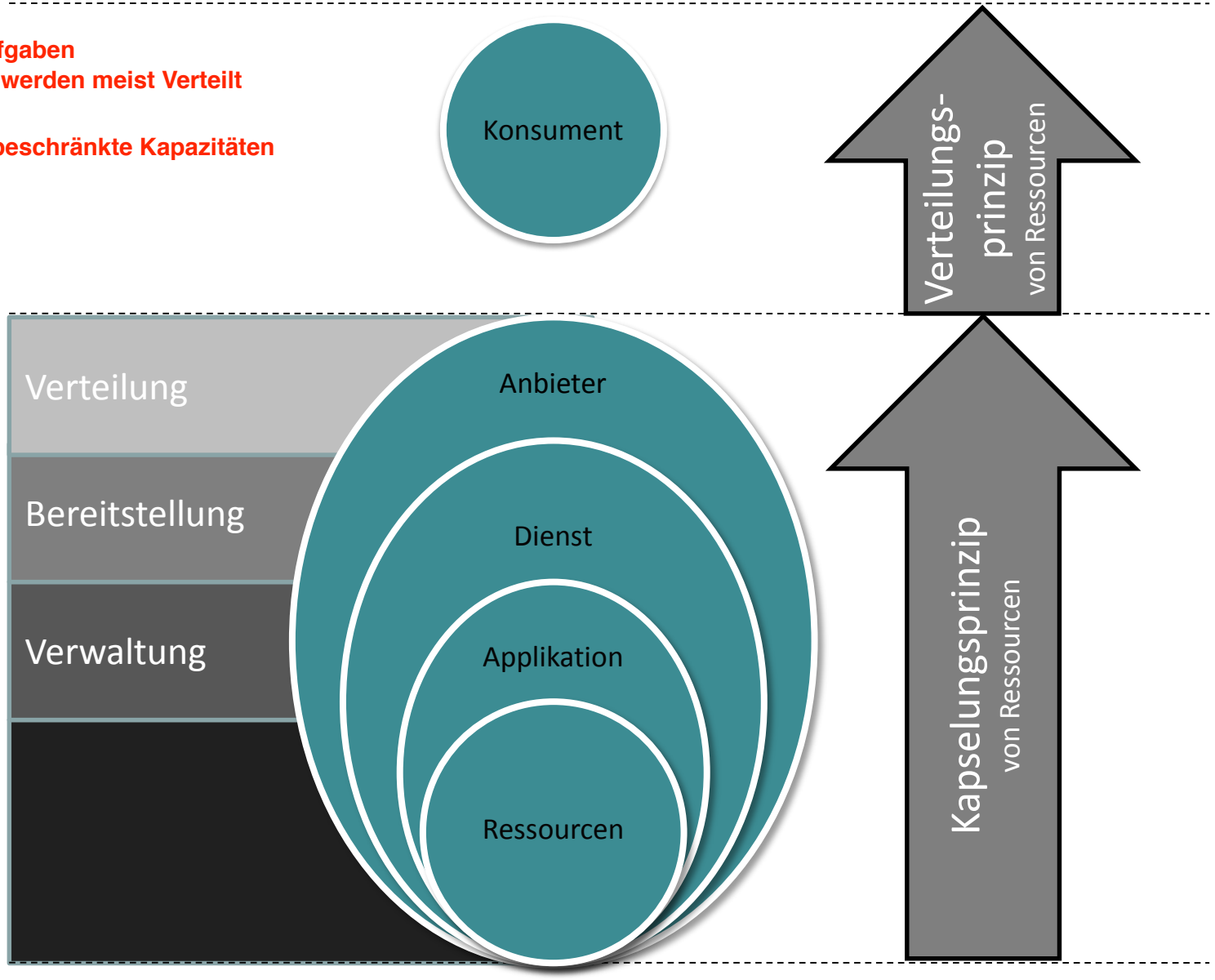
3.10 ROLLENAUFLÖSUNG VON CLIENT-SERVER

- 3.10.1 Funktionen eines VSYS und Client-Server
- 3.10.2 Client-Server: Aufgabenverteilung
- 3.10.3 Von Client-Server zu Mehrschicht-Architekturen
- 3.10.4 Vom Server zum Service
- 3.10.5 Von der Web-Anwendung zur RIA
- 3.10.6 Von der Dienstverteilung zur Komponentenverteilung
- 3.10.7 Vertauschbare Client-/Server-Rollen
- 3.10.8 Schrittweise Aufweichung des C-/S-Prinzips

3.1.1 Prinzipien und Phasen für Verteilung

-> Verteilen von Aufgaben
-> Überkapazitäten werden meist Verteilt

Ressourcen sind beschränkte Kapazitäten

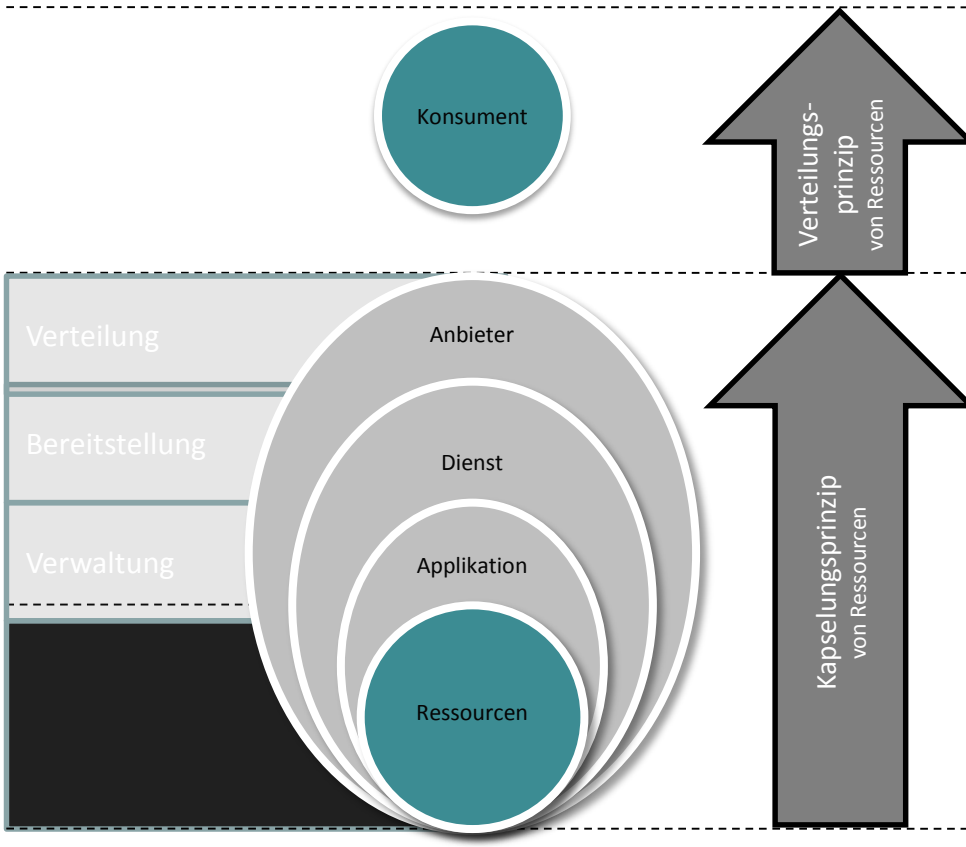


3.1.1 Ressourcen

RESSOURCE: DEFINITION

- **Ressource**: Beliebige physikalische oder virtuelle Komponente (Objekt) eines Rechnersystems, welche nur in begrenzter Form zur Verfügung steht

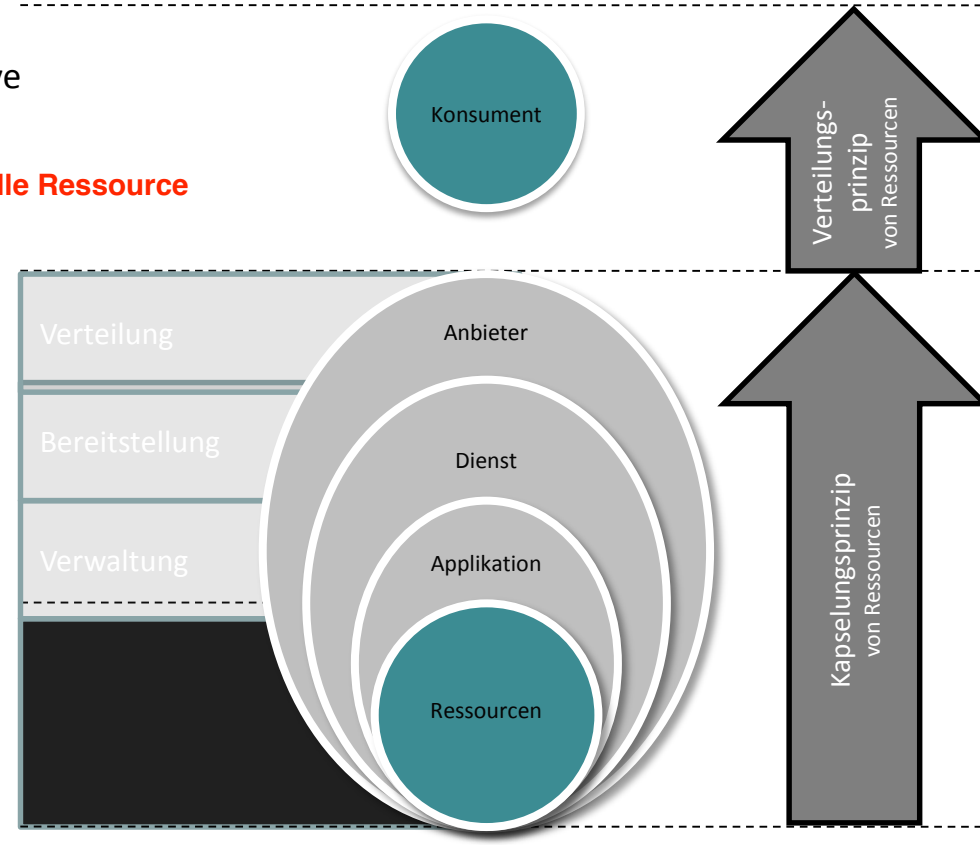
- Synonyme: System-Ressource, Betriebsmittel
- Abgrenzung: Ressource bei der Software-Entwicklung
  - Daten, Dateien oder Codebestandteile, die vom algorithmischen Kern (dem eigentlichen Quelltext) der Software verwendet werden, aber nicht Bestandteil des Quelltexts sind.



3.1.1 Ressourcen

RESSOURCE: BEISPIELE

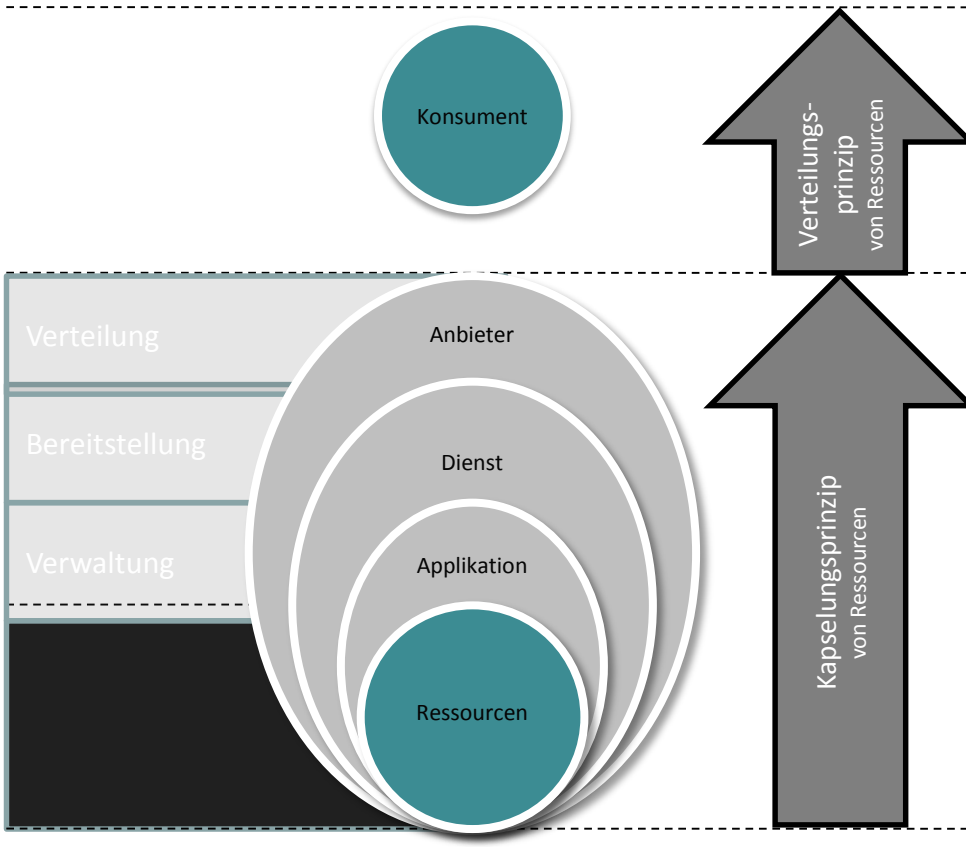
- Physisch: Hardware-Betriebsmittel **physische Ressource**
  - Prozessoren (CPU, E/A-Prozessoren): Zeitl. Aufteilung, Nebenläufige Nutzung
  - Speicher (Hauptspeicher, Sekundärspeicher, Festplatten): Räumliche Aufteilung
  - E/A-Geräte (Monitor/Tastatur, Maus, Netzwerkkarten, Drucker, CD/DVD): Exklusive Nutzung (meistens)
- Virtuell: Software-Betriebsmittel **virtuelle Ressource**
  - Speicher (Festplatten-Partitionen, Dateisysteme, Dateien, Datenbanken)
  - Systemprogramme (Gerätetreiber, DLLs, Dienstprogramme, Systemkommandos)
- Funktional: Daten, Datensätze, Information, Funktionen **funktionelle Ressource z.B Ressource**
  - Mail, Drucker-Spooling
  - Daten: Dateien, Datenbanken, Web-Seiten
  - Funktionen: Suchen, Nachschlagen, Umrechnen
- Keine Ressourcen im engeren Sinne: Web Services, Portlets, Widgets, Beans, etc.



3.1.1 Ressourcen

PRINZIPIEN DES GEMEINSAMEN ZUGRIFFS

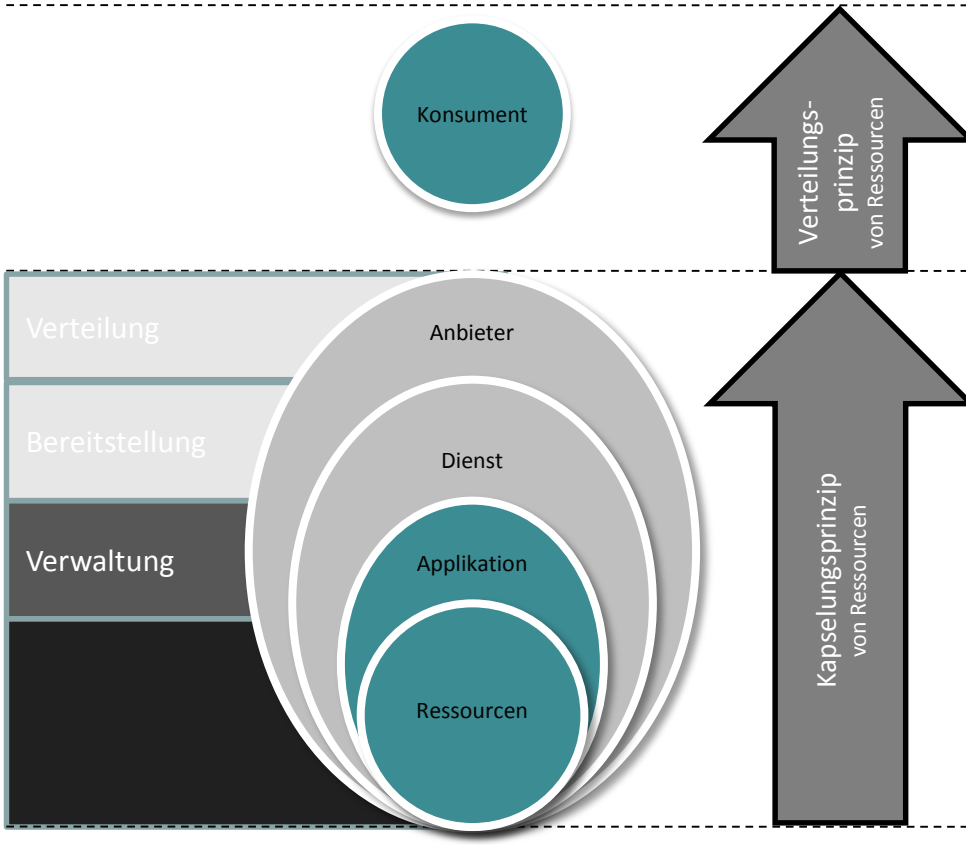
- Gültigkeitsbereich *Wo? Von wo?*
  - Ressourcen für einen (nahezu) uneingeschränkten Gültigkeitsbereich
    - *Beispiel: Web-Seiten*
  - Ressourcen für einen sehr eingeschränkten Gültigkeitsbereich
    - *Beispiel: Dokumente eines unternehmensinternen CMS (Content Management System); nur im Intranet bzw. mit VPN verfügbar*
- Anwenderkreis *Wer?*
  - Alle Anwender
    - *Beispiel: Web-Anwender der Welt*
  - Sehr eingeschränkter Anwenderkreis
    - *Beispiel: Angemeldete Anwender eines unternehmensinternen CMS*



3.1.3 Ressourcen

RESSOURCEN-VERWALTUNG

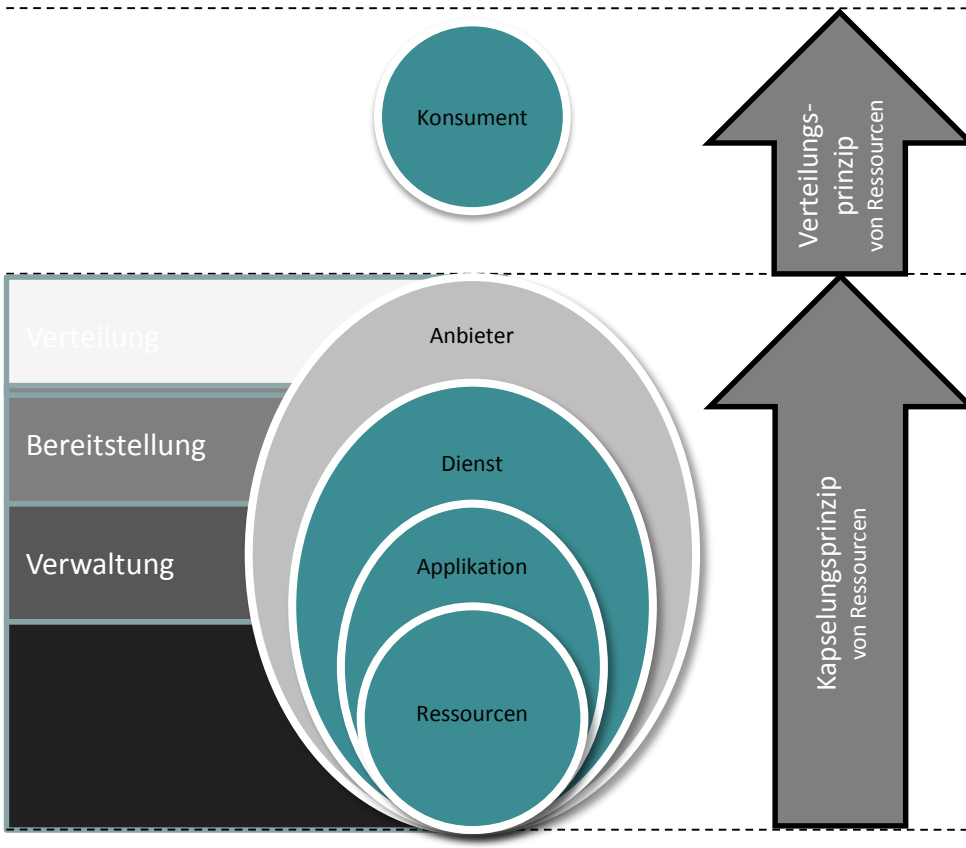
- Wer verwaltet Ressourcen?
  - Hardware/Firmware/Mikroprogramme: Nur indirekt
  - Betriebssystem/Treiber: Ja, auf einem hardware-nahen Niveau als Schnittstelle
    - Nur für lokale Applikationen
  - Applikationen: Ja, für Anwender
    - Nur für lokalen Einsatz
  - *Beispiele: Spezifische Software für Peripherie-Geräte od. Sekundärspeicher, lokale Benutzerverwaltung*



3.1.3 Dienste

RESSOURCEN-BEREITSTELLUNG

- Definition Dienste
  - Dienste (Services) sind Applikationen auf einem Rechner
  - Dienste verwalten verwandte Ressourcen
  - Dienste stellen Ressourcen (entfernten) Prozessen zur Verfügung
- *Beispiele auf verschiedensten Ebenen*
  - *Dateidienst fuer Dateien/Dateisysteme (NFS), Druckdienste (Druckerspooler)*
  - *Zahlungsdienst (Paypal, giroPay)*
  - *Web: Interaktive Seiten über CGI, Servlets, JSP , etc. → Web-Formulare, Suchmaschinen, Web Services*
- Zugriff auf Dienste: Dienstschnittstelle (Contract) als Menge exportierter Operationen
  - *Beispiel: Lesen, schreiben, löschen bei Dateidienst*



3.1.4 Anbieter und Konsument

DIENST-KAPSELUNG DURCH ANBIETER

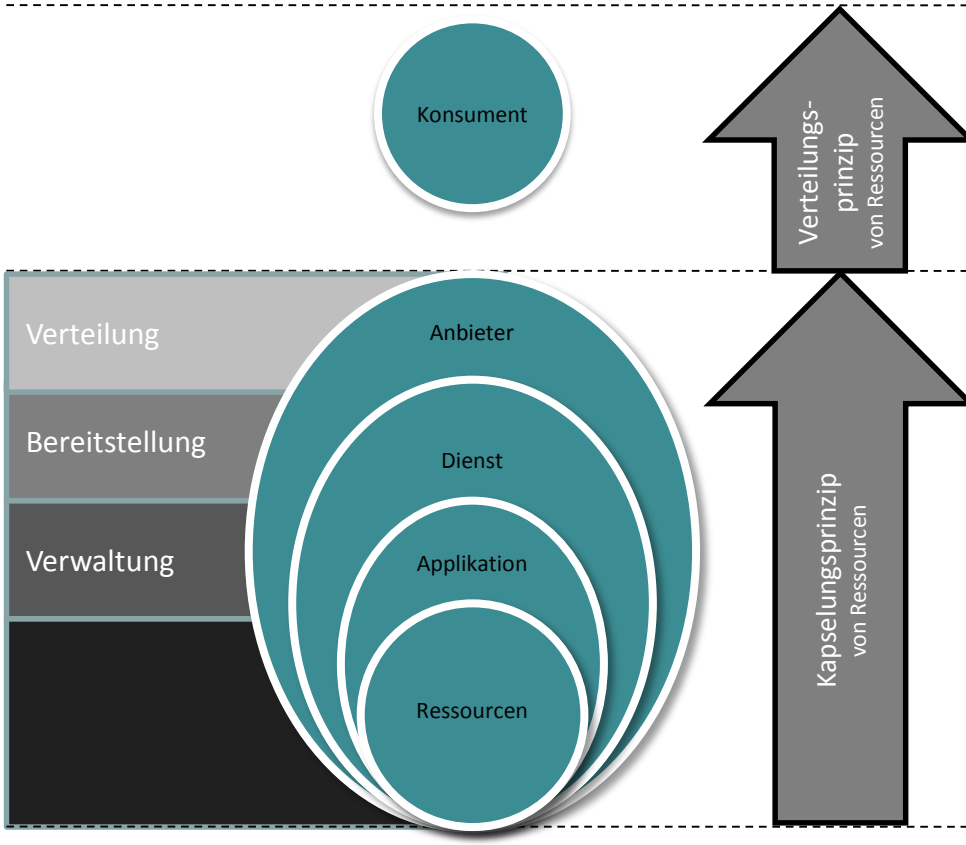
- Applikation bietet einen oder mehrere Dienste an

  → Dienst-Anbieter

DIENST-NUTZUNG DURCH KONSUMENT

- Applikation nutzt einen oder mehrere Dienste von einem oder mehrere Anbietern

  → Dienst-Konsument



Übersicht

BASISKONZEPTE

3.1 RESSOURCEN UND DIENSTE

- 3.1.1 Prinzipien und Phasen für Verteilung
- 3.1.2 Ressource
- 3.1.3 Dienst
- 3.1.4 Anbieter und Konsument

3.2 CLIENT UND SERVER

- 3.2.1 Fundamentalverteilung: Client-Server
- 3.2.2 Netzwerkebene

3.3 INTERAKTIONSMODELLE

3.4 (A)SYNCHRONITÄT

- 3.4.1 Anfrage und Antwort
- 3.4.2 Multiple Anfragen

EIGENSCHAFTEN

3.4 KOHÄRENZ UND TRANSPARENZ

- 3.4.1 Definition
- 3.4.2 Transparenz von Verteilungseigenschaften
- 3.4.3 Bedeutung und Realisierbarkeit

3.5 SKALIERBARKEIT

- 3.5.1 Definition und Typen
- 3.5.2 Typ: Größe
- 3.5.3 Typ: Geographie
- 3.5.4 Typ: Administration

3.6 NEBENLÄUFIGKEITS-GRAD

- 3.6.1 Bedeutung und Konsequenz
- 3.6.2 Ausschlussverfahren
- 3.6.3 Grad und Auswirkung

3.7 KOMMUNIKATIONS-GRAD

- 3.7.1 Schalenmodell
- 3.7.2 Sockets
- 3.7.3 Nachrichten
- 3.7.4 Entfernter Prozeduraufruf
- 3.7.5 Entfernter Methodenaufruf
- 3.7.6 Runtime, Dienste, Komponenten

ARCHITEKTUREN

3.8 SOFTWARE- UND SYSTEM-ARCHITEKTUR

- 3.8.1 Software-Architektur

3.2.1 Fundamentalverteilung: Client-Server

GRUNDLAGEN

- Definition
  - **Anbieter** (**Server**): Bietet Funktionalität (Ressourcen) in Form eines od. mehrerer **Dienste(s)** an
  - **Konsument** (**Client**): Nutzt den/die Dienst(e) eines Servers
- Begriffsklärung
  - Server
    - Hardware/Rechner/OS: Ja, aber nicht im Sinne verteilter Systeme
    - Applikationen: Ja, stellen Zugriff auf einen oder mehrere Dienste für externe Applikationen/Dienste zur Verfügung
  - Client
    - Hardware/Rechner/OS: Ja, aber nicht im Sinne verteilter Systeme
    - Applikation: Ja, nimmt Dienste eines oder mehrerer Server in Anspruch
- Beispiele
  - Server: *Web-Server à la Apache, IIS; Datenbankmanagementsystem, etc.*
  - Client: *Web-Browser à la Firefox, Opera, Safari, Chrome, IE*
- Anwendung
  - Reinform: Unabhängiges Client-Server-System
  - Übliche Verwendung: Bestandteil komplexer Systemarchitekturen
- Einordnung/Bewertung
  - Grundlegendstes Konzept der Verteilung von Systemen oder Aufgaben
  - Sehr viele, aber nicht alle VSYSe basieren auf Client-Server-Prinzip

3.2.1 Fundamentalverteilung: Client-Server

UNTERSCHIEDE ZWISCHEN CLIENT UND SERVER

- Clients sind aktiv und Server passiv
  - Clients initiieren die Kommunikation
  - Server warten auf Clients
- Laufzeit
  - Server sind dauerhaft verfügbar
  - Clients existieren idR nur zur Laufzeit der zugehörigen Applikationen
  - *Beispiel: E-Mail-Programm, FTP-Client, Remote Desktop Verbindung, X Window System*
- Rollen von Client und Server können wechseln

Rollen: Verantwortlichkeit/Verpflichtungen/Einschränkungen/Rechten

Impliziert Rechte und Pflichten

TECHNOLOGIEN FÜR CLIENT-SERVER AUF UNTERSCHIEDLICHEN NETZWERK-EBENEN

Nr.	Schichten		Einordnung		Einheiten	Protokoll-Beispiele	Netzwerk-Komponenten
	OSI	DoD	System	Verbindung			
7	Anwendung	Anwendung	Anwendung	Ende-zu-Ende (Multihop)	Daten	FTP, LDAP, NFS, SMTP, IMAP, DHCP, DB LOCK (2PL, TL, TO, SGT)	Gateway Content-Switch
6	Darstellung					HTTP(S), MIME, SSL, TLS, XDR	
5	Sitzung					NetBIOS, Named Pipes, RTP, etc.	
4	Transport	Transport	Transport		Segmente	TCP UDP SCTP SPX	
3	Vermittlung	Vermittlung			Pakete	IP IPsec IPX ICMP IGMP	Router Layer 3-Switch
2	Sicherung	Netzzugriff		Punkt-zu-Punkt	Rahmen	Ethernet FDDI Token Ring	Bridge Switch
1	Bitübertragung				Bits		Repeater Hub

Sockets, RPC, RMI

Konzeptuelle Ausrichtung auf Client-Server

Stark — Schwach

Übersicht

BASISKONZEPTE

3.1 RESSOURCEN UND DIENSTE

- 3.1.1 Prinzipien und Phasen für Verteilung
- 3.1.2 Ressource
- 3.1.3 Dienst
- 3.1.4 Anbieter und Konsument

3.2 CLIENT UND SERVER

- 3.2.1 Fundamentalverteilung: Client-Server
- 3.2.2 Netzwerkebene

3.3 INTERAKTIONSMODELLE

3.4 (A)SYNCHRONITÄT

- 3.4.1 Anfrage und Antwort
- 3.4.2 Multiple Anfragen

EIGENSCHAFTEN

3.4 KOHÄRENZ UND TRANSPARENZ

- 3.4.1 Definition
- 3.4.2 Transparenz von Verteilungseigenschaften
- 3.4.3 Bedeutung und Realisierbarkeit

3.5 SKALIERBARKEIT

- 3.5.1 Definition und Typen
- 3.5.2 Typ: Größe
- 3.5.3 Typ: Geographie
- 3.5.4 Typ: Administration

3.6 NEBENLÄUFIGKEITS-GRAD

- 3.6.1 Bedeutung und Konsequenz
- 3.6.2 Ausschlussverfahren
- 3.6.3 Grad und Auswirkung

3.7 KOMMUNIKATIONS-GRAD

- 3.7.1 Schalenmodell
- 3.7.2 Sockets
- 3.7.3 Nachrichten
- 3.7.4 Entfernter Prozeduraufruf
- 3.7.5 Entfernter Methodenaufruf
- 3.7.6 Runtime, Dienste, Komponenten

ARCHITEKTUREN

3.8 SOFTWARE- UND SYSTEM-ARCHITEKTUR

- 3.8.1 Software-Architektur

INTERAKTION DER PROZESSE

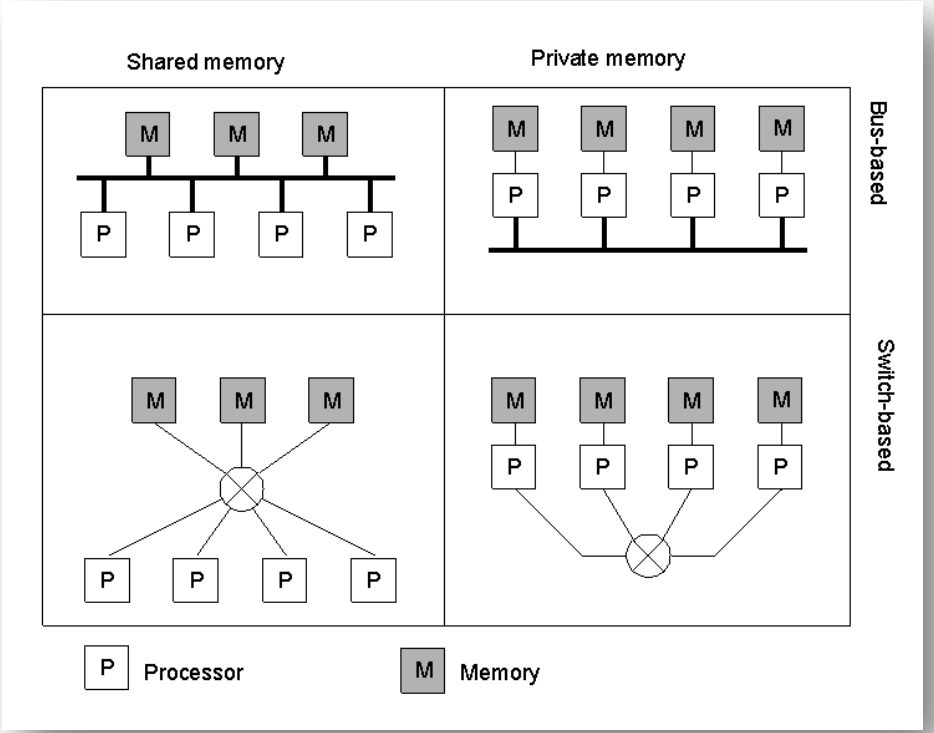
- *A distributed system is one in which components located at networked computers communicate and coordinate their actions only by passing messages.*

  Quelle: Coulouris et al., Distributed Systems – Concepts and Design

- Warum Interaktion?

  → Realisierung der verteilten Funktionalität durch Interaktion der Prozesse/Rechner

- Grundsätzliche Konzepte
  - Kooperation und Kommunikation
  - Symmetrische und asymmetrische Interaktion



Interaktion durch gemeinsame Daten

Quelle: A. S. Tanenbaum, Verteilte Systeme

SYMMETRIE ODER ASYMMETRIE IN DER INTERAKTION

- **Symmetrie:** Komponenten oder -Partner haben gleichwertige Interaktionsrollen **gleiche Rollen**

  Beispiele: Client-Server → Nein; Peer-To-Peer → Bedingt
  - Erweiterung der Definition: Komponenten oder Partner haben gleichwertigen Ressourcenzugriff

3.3 Interaktionsmodelle

… SYMMETRIE ODER ASYMMETRIE IN DER INTERAKTION

- Erweiterung der Defintion: …

Beispiele:

Client-Server → Nein

Peer-To-Peer → Meistens

- **Asymmetrie:** Komponenten haben keine gleichwertigen Interaktionsrollen

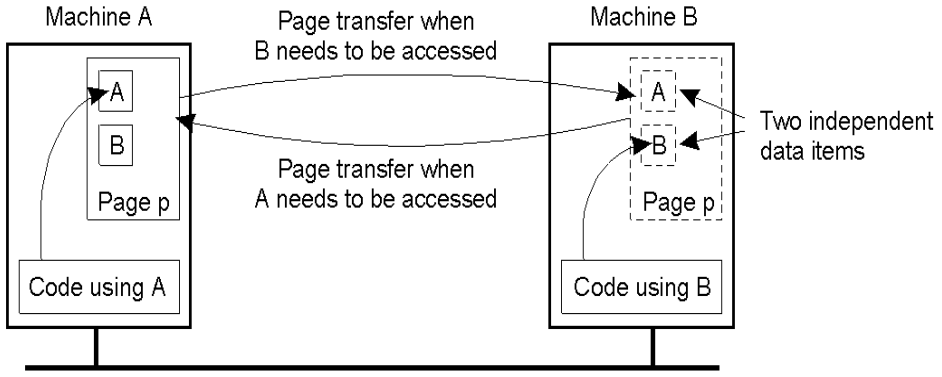
unterschiedliche Rollen

Rollenwechesel bei Peer-to-Peer

KOOPERATION

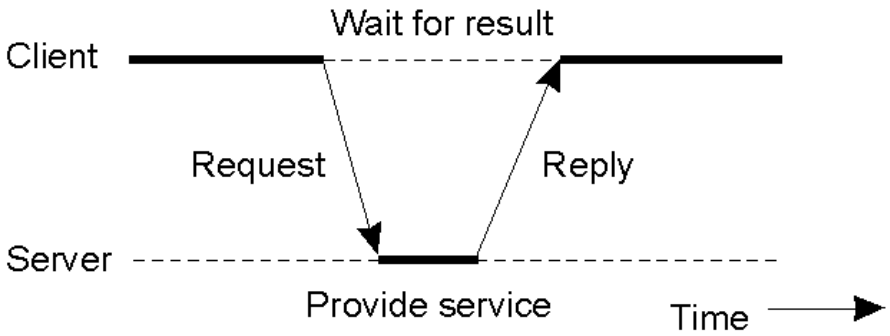
Beispiel gemeinsames bearbeiten von Daten

- Interaktion auf Basis gemeinsamer Ressourcen/Daten
  Beispiel: Shared Memory
- IdR. symmetrische Interaktion, da alle Prozesse gleichberechtigt beim Zugriff auf Daten
- Verwendung: Häufig nur bei lokalen, auf Prozesse verteilten Anwendungen
- Selten: Reale verteilte Kooperation
  Bsp.: Linda, JavaSpaces
- Häufig: Verteilte Kooperation durch Kommunikation



Kooperation durch Kommunikation (Page transfer)

Quelle: A. S. Tanenbaum Verteilte Systeme



Asymmetrische Kommunikation

Quelle: A. S. Tanenbaum Verteilte Systeme

3.3 Interaktionsmodelle

KOMMUNIKATION

- Häufig asymmetrische Interaktion, da
  - Anfrager-Antworter-Beziehung
    - Noch allgemeiner: Nachrichtenmuster (Message Patterns)
  - Ggf. mehrere Anfrager pro Antworter
- Übliches Interaktionsmodell für verteilte Systeme

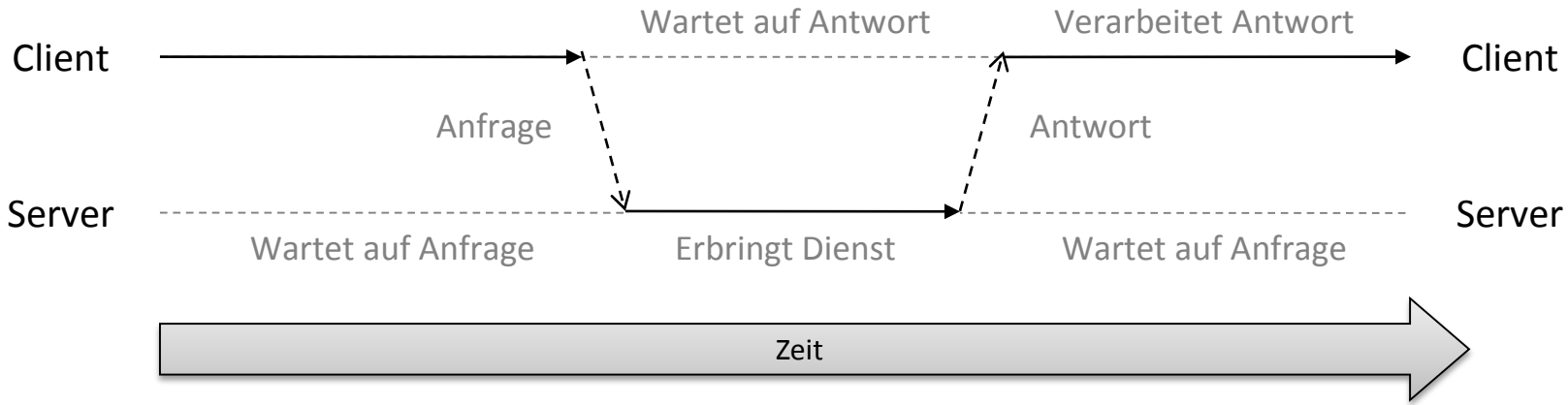
FAZIT

- Kommunikation üblich, Kooperation eher in Ausnahmefällen
- Interaktion idR. asymmetrisch
- Kooperation durch Kommunikation denkbar

3.4.1 Anfrage und Antwort

REQUEST-RESPONSE-MODELL

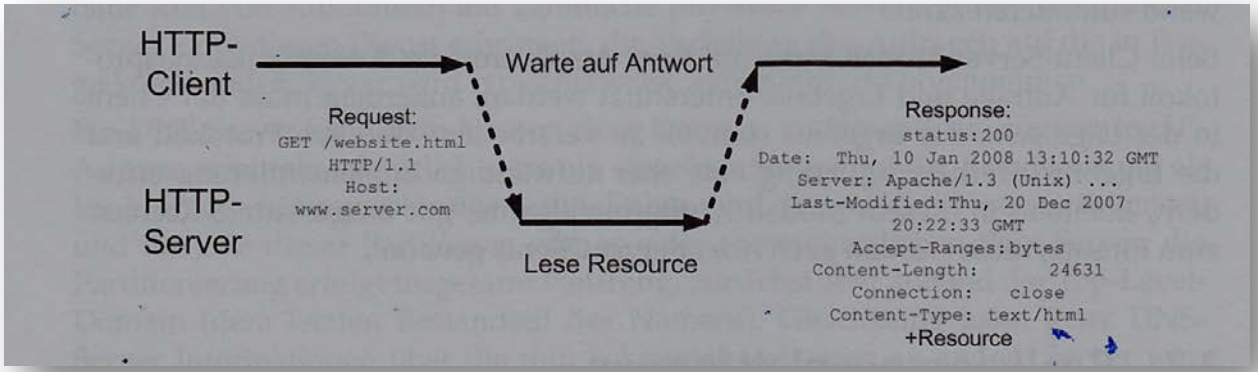
- Server wartet auf Anfrage
- Client initiiert Kommunikation → Anfrage (Request)
- Server empfängt und bearbeitet Anfrage
- Server sendet Antwort → Antwort (Response)



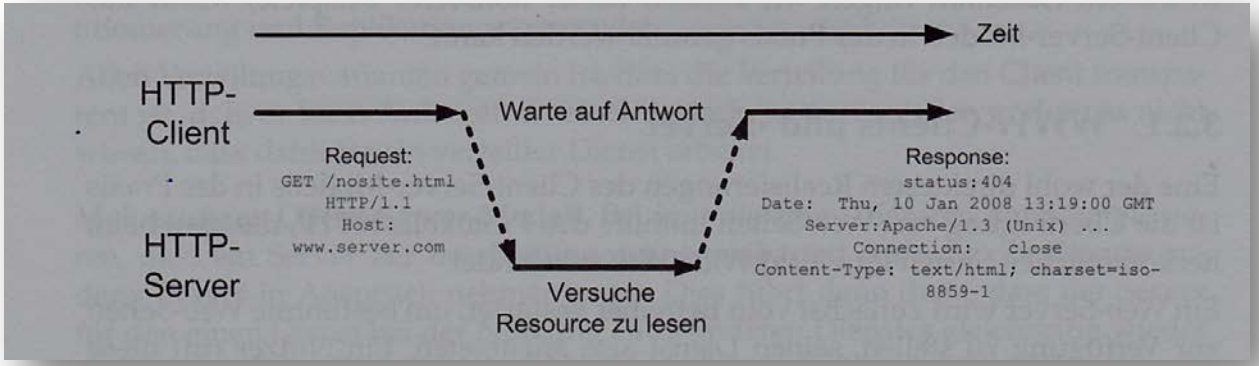
- **Synchron**: Client wartet auf Antwort
- **Asynchron**: Client wartet nicht und wird vom Server über Verfügbarkeit der Antwort benachrichtigt

3.4.1 Anfrage und Antwort

- Ressourcen: Web-Seiten
- Anfragen und Antworten
  - URL (Uniform Resource Locator): Server-Adresse[+Port]+Ressourcenbezeichnung
  - Browser: URL → HTTP-GET-Request
    - Optional: Umsetzung des DNS-Namens in IP-Adresse
  - Server: Prüfen und ggf. Senden der Ressourcen → HTTP-Response



Erfolgreiche Kommunikation



Erfolglose Kommunikation

Quelle:
Dunkel et al.
Systemarchitekturen
f. vert. Systeme

3.4.1 Anfrage und Antwort

- Request-Response-Zyklus gemäß HTTP zustandslos

  → Zustandsbehaftete Kommunikation (Bsp.: Session) erfordert Protokoll-Zusätze

- Weitere HTTP-Anfragen neben GET
  - POST: Übermittlung von Daten an den Server
  - HEAD: Wie GET, aber ohne eigentliche Übermittlung der Ressource
  - PUT: Übermittlung einer Ressource an den Server
  - DELETE: Löschen einer Ressource auf dem Server → Verwendung bei REST oder WEBDAV
  - TRACE: Server liefert eigene Anfrage zurück → Wurde eigene Anfrage verändert (Debugging)
  - OPTIONS: Liste mit unterstützten Methoden und Features
  - CONNECT: Verwendung in SSL-Tunnels
- Authentifizierung
  - Server: Statuscode „401 Unauthorized“ + Header mit WWW-Authenticate
  - Client: Header mit WWW-Authenticate und User + Passwort (Base64-codiert) an Server

3.4.2 Multiple Anfrage

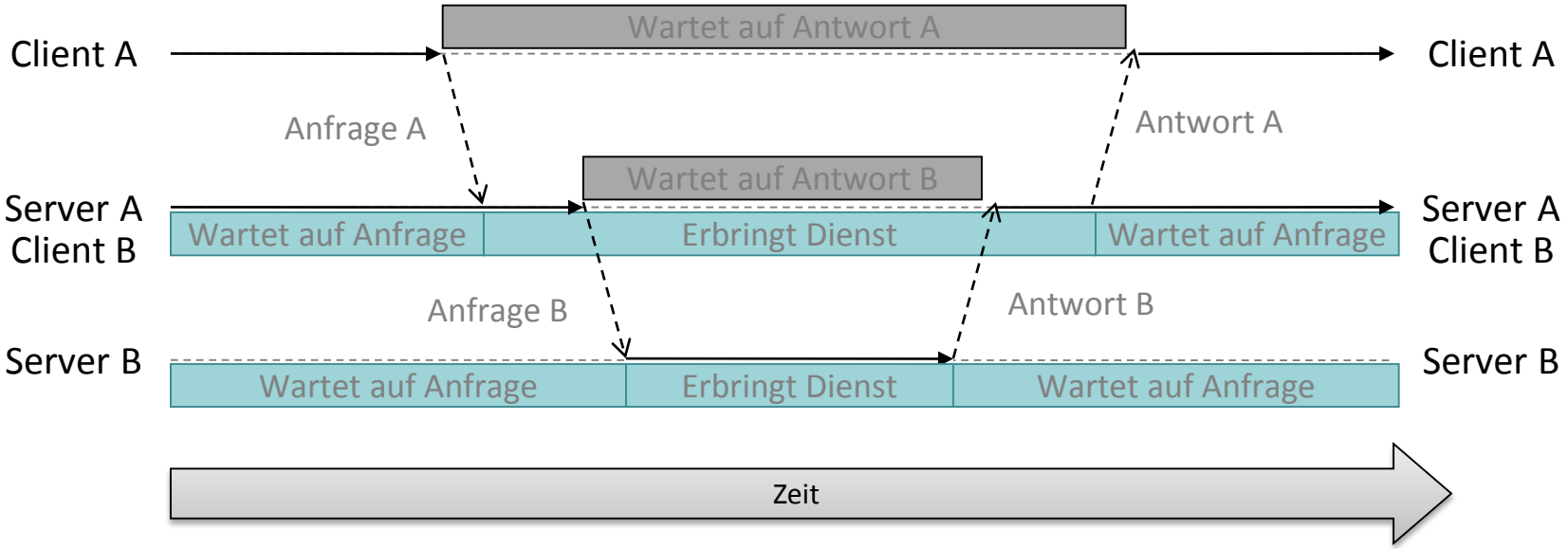
MULTI-REQUEST UND MULTI-CLIENT

- Mehrere Request-Response-Paare pro Kommunikation → Evtl. Verbindungskontext erforderlich
- Multi-Client
  - Angebot eines Dienstes für nur einen Client idR nicht sinnvoll
  - Server bedient mehrere Clients gleichzeitig (ggf. mit Verbindungskontext pro Client)

3.4.2 Multiple Anfrage

DIENSTKOMPOSITION DURCH MEHRSTUFIGE SERVER-ANFRAGE

- Server benötigt zur Beantwortung einer Anfrage Dienste anderer Server
  → Server tritt als Client für weiteren Server auf
- Bereitstellung komplexerer Dienste durch Zusammensetzung anderer Dienste
  → Dienststrukturierung durch Schachtelungsprinzip
  (entspricht Modularisierung bei imperativer Programmierung)
- Nachteile
  - Erhöhter Kommunikationsaufwand
  - Synchronisierungsaufwand bei (gängiger) gleichzeitiger Subdienst-Ausführung



Übersicht

BASISKONZEPTE

3.1 RESSOURCEN UND DIENSTE

- 3.1.1 Prinzipien und Phasen für Verteilung
- 3.1.2 Ressource
- 3.1.3 Dienst
- 3.1.4 Anbieter und Konsument

3.2 CLIENT UND SERVER

- 3.2.1 Fundamentalverteilung: Client-Server
- 3.2.2 Netzwerkebene

3.3 INTERAKTIONSMODELLE

3.4 (A)SYNCHRONITÄT

- 3.4.1 Anfrage und Antwort
- 3.4.2 Multiple Anfragen

EIGENSCHAFTEN

3.4 KOHÄRENZ UND TRANSPARENZ

- 3.4.1 Definition
- 3.4.2 Transparenz von Verteilungseigenschaften
- 3.4.3 Bedeutung und Realisierbarkeit

3.5 SKALIERBARKEIT

- 3.5.1 Definition und Typen
- 3.5.2 Typ: Größe
- 3.5.3 Typ: Geographie
- 3.5.4 Typ: Administration

3.6 NEBENLÄUFIGKEITS-GRAD

- 3.6.1 Bedeutung und Konsequenz
- 3.6.2 Ausschlussverfahren
- 3.6.3 Grad und Auswirkung

3.7 KOMMUNIKATIONS-GRAD

- 3.7.1 Schalenmodell
- 3.7.2 Sockets
- 3.7.3 Nachrichten
- 3.7.4 Entfernter Prozeduraufruf
- 3.7.5 Entfernter Methodenaufruf
- 3.7.6 Runtime, Dienste, Komponenten

ARCHITEKTUREN

3.8 SOFTWARE- UND SYSTEM-ARCHITEKTUR

- 3.8.1 Software-Architektur

Übersicht

K A P I T E L 3

Vom zentralen Fall übertragbare Basiskonzepte, daraus ableitbare grundlegende Eigenschaften, erste teilweise auch zentral gültige Strukturen und Architekturelemente

Übersicht

K A P I T E L 3

Vom zentralen Fall übertragbare Basiskonzepte, daraus ableitbare grundlegende Eigenschaften, erste teilweise auch zentral gültige Strukturen und Architekturelemente