

AI-B.41: Verteilte Systeme

- Lecture Notes [SL] -

C. Schmidt | SG AI | FB 4 | HTW Berlin

Stand: WiSe 18/19

Urheberin: Prof. Dr. Christin Schmidt

Verwertungsrechte: keine außerhalb des Moduls

Ablauf heute

- Organisatorisches [~20 min.]
- Einführung [~70 min.]

AI-B.41: Verteilte Systeme

- Lecture Notes [SL] -

I. Organisatorisches

C. Schmidt | SG AI | FB 4 | HTW Berlin

Stand: WiSe 18/19

Urheberin: Prof. Dr. Christin Schmidt

Verwertungsrechte: keine außerhalb des Moduls

Kontakt

Prof. Dr. Christin Schmidt (christin.schmidt@htw-berlin.de)

SG Angewandte Informatik

Campus Wilhelminenhof, Gebäude C, Raum 612

Schwerpunkte:

- Verteilte Systeme
- Data Science / Machine Learning
- Gesellschaftliche / Ethische Aspekte der Informatik
- (IT-)Projektmanagement

Sprechstunde:

- In der Vorlesungszeit (vgl. Website)
- Nach vorheriger Anmeldung per E-Mail

Kontakt

Frank Dornheim (dornhef@htw-berlin.de)

Fachbereichsleiter im ITDZ

Masterabschluss an der HTW im Studiengang BUI 2010

2 Rechenzentren mit etwa 2500m² Fläche

Arbeitsbereich:

RZ Facility,

Storage und SAN,

Backup

Automatisierungsinfrastruktur (IT-Fabric oder Cloud)

Etikette

- Lieber vollen Herzens abwesend als halbherzig anwesend
- Handy aus/lautlos
- Notebook zuklappen
(außer bei vorlesungsbezogenen Anwendungen)
- Keine Bild- oder Tonaufnahmen



Alle einverstanden?

Quellen/Literatur

- Literaturangaben finden sich immer auf der letzten Folie:
 - » Zur Vertiefung des Stoffes
 - » Als Referenz für Inhalte auf den Folien
 - » Zur Bearbeitung der Übungsaufgaben
 - » Zur Vorbereitung auf Klausuren
- Modulbegleitende Materialien / Foliensätze
 - » Werden ggf. in Moodle zur Verfügung gestellt
 - » Keine Bringschuld:
 - Dozentinnen / Dozenten sind nicht dazu verpflichtet, Ihnen Unterlagen anzufertigen / zur Verfügung zu stellen.
 - Freiwilliges Angebot Ihrer Dozentinnen / Dozenten, um Sie beim Lernprozess zu unterstützen.
 - Über Form, Ausgestaltung und zeitliche Zurverfügungstellung entscheidet Dozent_in.
 - » Nicht weitergeben.
 - » Keine Verwertungsrechte, d.h. keine weitere Nutzung außerhalb der Hochschule (z.B. Studi-Dropbox, Slideshare, ...)!
- Ihr Engagement / Ihre Mitarbeit zur Erarbeitung der Inhalte sollte sich keinesfalls auf das „Durchlesen“ der Folien beschränken

Daraus folgt:

- Überprüfen Sie Ihre Erwartungshaltung(en)
- SL / Ü: Vortrag wird ggf. durch Materialien ergänzt
 - » Aus Erfahrung können Materialien manchmal umfangreich und die Schriftgröße vortragsbegleitend nicht gut lesbar sein (dies ist beabsichtigt!)
 - » Materialien dienen Ihrer Vor- und Nachbereitung als freiwilliges Unterstützungsangebot Ihrer Dozentin / Ihres Dozenten (es ist NICHT das Ziel, die Zuhörerschaft mittels der Materialien visuell zu unterhalten / ihr zu schmeicheln)
- Sie sollten selbst Ihren inhaltlichen Fortschritt dokumentieren / mitschreiben und das Gesagte kognitiv reflektieren!
- Übungen haben (oftmals) keine Standardlösungen (one-fits-all)
- Lehrveranstaltungen sind ein ANGEBOT zur Vertiefung im Eigenstudium und eigenverantwortlicher Vor- und Nachbereitung! (Zeiten hierzu sind in ECTS-Punkten

Quellen und Lesetipps zur Vertiefung (insbesondere für Studienanfängerinnen und -anfänger)

Verhaltenskodex (an der HTW Berlin gilt ein (nicht nur) an Hochschulen / Universitäten üblicher Verhaltenskodex)

- Adam, K.; Rachfall, T. (2016) Verhaltenskodex im Studiengang Wirtschaftsingenieurwesen; HTW Berlin; online: <http://wiw-bachelor.htw-berlin.de/studium/verhaltenskodex> [letzter Zugriff: 2 X 2017]
- Greß, M. (2014) Anrede Sprechstunde - Sieben Knigge-Regeln für die Uni; Mitteldeutsche Zeitung; online: <http://www.mz-web.de/karriere/anrede-sprechstunde-sieben-knigge-regeln-fuer-die-uni,20651404,28029902.html> [letzter Zugriff: 2 X 2017]
- Scharbert, K. (1995) Von der Schule ins Studium – Vorlesungsknigge; FH München; online: http://www.stiftung-swk.de/bridge_course_math/d_knigge.html; [letzter Zugriff: 2 X 2017]

„E-Mail - Netiquette“:

- Manschwetus, U. (2015) Wie man Professoren eine E-Mail schreibt; online: <http://wissenschafts-thurm.de/wie-man-professoren-eine-e-mail-schreibt/> [letzter Zugriff: 2 X 2017]
- Leddy, M. (2005) How to e-mail a professor; online: <http://mleddy.blogspot.de/2005/01/how-to-e-mail-professor.html> [letzter Zugriff: 2 X 2017]

Mitschriften in Vorlesungen:

- May, C. (2014) A Learning Secret: Don't Take Notes with a Laptop; Scientific American; online: <https://www.scientificamerican.com/article/a-learning-secret-don-t-take-notes-with-a-laptop/> [letzter Zugriff: 2 X 17]

Sonstiges:

- Litzcke, S. M.; Linssen, R. (2007) Studieren lernen. Arbeits- und Lerntechniken, Prüfungen und Studienarbeiten. Schriftenreihe der FH Bund (50); online: http://psydok.psycharchives.de/jspui/bitstream/20.500.11780/434/1/Studieren_lernen.pdf [letzter Zugriff: 2 X 2017]

Tipps

- Erwartungshaltung anpassen und Eigeninitiative ergreifen, d.h.
 - » über die in den Veranstaltungen / Übungen bereit gestellten Beispiele hinaus sollten Beispiele / Code selbständig recherchiert, bewältigt und bewertet werden
 - » eigenes Schließen von Verständnislücken u.a. auf Basis der Quellen, die der Lehrveranstaltung zu Grunde gelegt werden

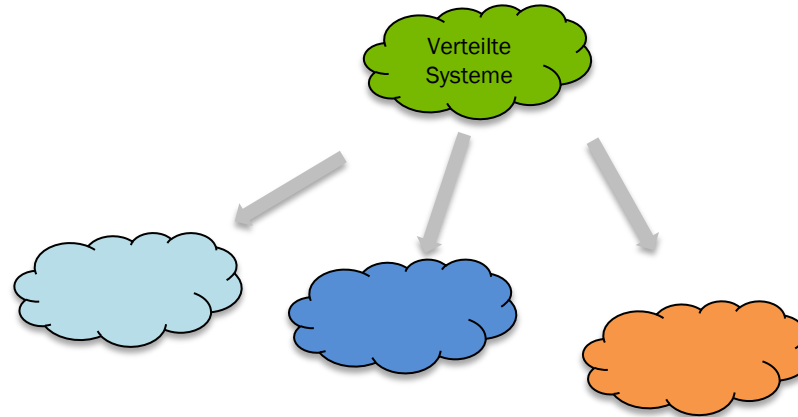
?

Was erwarten Sie von der Lehrveranstaltung “Verteilte Systeme“?

Was wollen Sie lernen?

Welche Themen interessieren Sie besonders?

Wie wollen Sie lernen?



Timeboxing [~5 min.]

- ✓ Setzen Sie sich mit Ihrem Nachbarn/Ihrer Nachbarin zusammen und beantworten Sie sich gegenseitig die obigen Fragen.
- ✓ Machen Sie sich Notizen, damit Sie danach erläutern können, welche Erwartungen Ihr Nachbar hat.

Tipps:

- Legen Sie zuerst fest, wer mit dem Fragen beginnt. Wechseln Sie nach 2 Minuten.
- Hinterfragen Sie die Antworten Ihres Nachbarn, z.B.: "Warum?", "Was speziell?" usw.

Ich frage Sie am ggf. Ende nach den Erwartungen Ihres Nachbarn bzw. Ihrer Nachbarin.

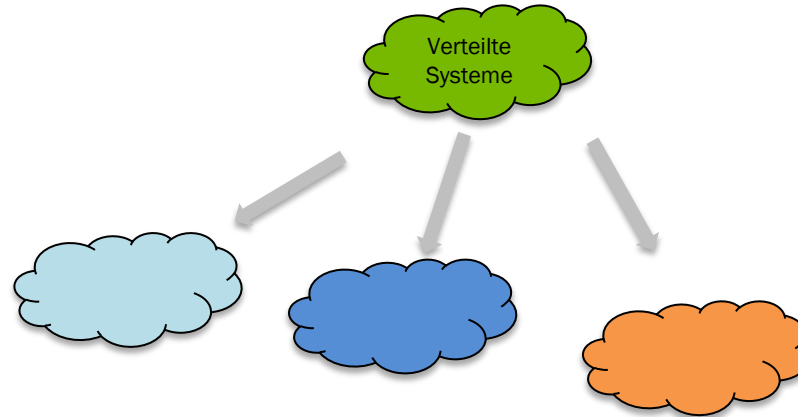
?

Was erwarten Sie von der Lehrveranstaltung “Verteilte Systeme“?

Was wollen Sie lernen?

Welche Themen interessieren Sie besonders?

Wie wollen Sie lernen?



Ihre Ergebnisse:

Was?

Welche besonderen Themen?

Wie?

Lernziele des Moduls

Seminaristischer Lehrvortrag -> Praktisch-gestalterische Kompetenz:
Analyse und Design verteilter Systeme (Fähigkeit zur Analyse, Bewertung
und zum Vergleich verschiedener Technologien zur Erstellung verteilter
Anwendungen)

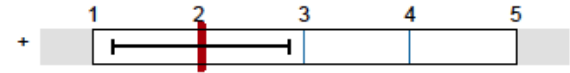
Übung -> Technologische Kompetenz: Fähigkeit zum Entwurf / zur
Entwicklung einfacher verteilter Anwendungen

- ✓ Das Modul ist ein Querschnittsmodul, u.a. mit Bezug zu vielen anderen Modulen im AI-Curriculum (kein Programmier- und / oder Produktschulungskurs!)
- ✓ SL & Ü sind ein Angebot (eigene Vertiefung im Rahmen Ihrer Vor- und Nachbereitung gefordert)

Evaluationsergebnisse (WiSe 14/15)

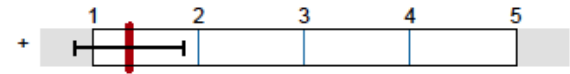
Globalwerte

3.1. Die Lehrveranstaltung hat (Skalenbreite: 5)



mw=2
s=0,8

5.1. Der Dozent / Die Dozentin (Skalenbreite: 5)



mw=1,3
s=0,5

6. Stärken / Schwächen

6.1. Was gefällt Ihnen besonders gut an diesem Lehrangebot? (Antwort wird eingescannt, bitte in Blockschrift ausfüllen und Rand beachten!)

- Aktuelle und relevante Themen
- Sehr entspannt, kompetent und Konzentration aufs Wesentliche. Reagiert blitzschnell auf Mails
- Sehr nett
- Sinnvolle, gut strukturierte Lehrveranstaltung mit guter alternative Prüfungsleistung
- offene Lernatmosphäre

6.2. Was kann besser werden? (Antwort wird eingescannt, bitte in Blockschrift ausfüllen und Rand beachten!)

- - Moodle
- online Evaluation!
- Bestimmte Materialien bzw. Folien reichen
- Bitte online Evaluation
- Die Prüfungsleistungen ins Semester zu verlegen, ist an sich eine gute Idee. Allerdings führte es zu erheblichen Stress, wenn mehrere Kurse so geregelt werden.
- Fragebogen in Papierform bei der Evaluation
- Moodle könnte besser werden!
Nicht gegen Frau Schmidt gerichtet.
- bitte elektronische Evaluation!!
- gut

Roadmap: Verteilte Systeme / SoSe 18

(o.G.)

Zug	1					2				
LV (Dozent_In)	SL (LB F. Dornheim, M.Sc.)			Ü (LB C. Wolf, B.Sc.)		SL (Prof. Dr. C. Schmidt)			Ü (LB T. Dumke, B.Sc.)	
Zeit:	Freitags: 08:00 - 09:30 Uhr			Mittwochs: 15:45 - 17:15 Uhr (1. Zug, 1. Gruppe) Mittwochs: 17:30 - 19:00 Uhr (1. Zug, 2. Gruppe)		Donnerstags: 12:15 - 13:45 Uhr			Donnerstags: 14:00 - 15:30 Uhr (2. Zug, 1. Gruppe) Donnerstags: 14:00 - 15:30 Uhr (2. Zug, 2. Gruppe)	
Ort / Raum:	Raum WH C 446			Raum WH C 624		Raum WH C 446			Raum WH C 625	
KW	Datum	Teil (SL)		Datum	Teil (Ü)	Datum	Teil (SL)	Beschreibung	Datum	Teil (Ü)
40	05.10.18	1	Organisatorisches, Einführung	03.10.18	keine LV: Orientierungstag	04.10.18	keine LV		04.10.18	keine LV
41	12.10.18	2	Systemmodelle & Architekturstile	10.10.18	1	11.10.18	1	Organisatorisches, Einführung	11.10.18	1
42	19.10.18	3	Systemarchitekturen I: C/S	17.10.18	2	18.10.18	2	Systemmodelle & Architekturstile	18.10.18	2
43	26.10.18	4	Systemarchitekturen II: P2P	24.10.18	3	25.10.18	3	Systemarchitekturen I: C/S	25.10.18	3
44	02.11.18	5	Cloud Computing	31.10.18	4	01.11.18	4	Systemarchitekturen II: P2P	01.11.18	4
45	09.11.18	6	Netzwerke	07.11.18	5	08.11.18	5	Cloud Computing	08.11.18	5
46	16.11.18	7	Interprozesskommunikation I: Socket-Primitives	14.11.18	6	15.11.18	6	Netzwerke	15.11.18	6
47	23.11.18	8	Interprozesskommunikation II: Middleware (RPC/RMI)	21.11.18	7	22.11.18	7	Interprozesskommunikation I: Socket-Primitives	22.11.18	7
48	30.11.18	9	Koordination & Synchronisation I: Concurrency / Thread & Lock-Primitives	28.11.18	8	29.11.18	8	Interprozesskommunikation II: Middleware (RPC/RMI)	29.11.18	8
49	07.12.18	10	Koordination & Synchronisation II: Time	05.12.18	9	06.12.18	9	Koordination & Synchronisation I: Concurrency / Thread & Lock-Primitives	06.12.18	9
50	14.12.18	11	Koordination & Synchronisation III: Gruppenkommunikation	12.12.18	10	13.12.18	10	Koordination & Synchronisation II: Time	13.12.18	10
51	21.12.18	12	Ausgewählte Kapitel: z.B. Globale Zustände, Sicherheitsaspekte	19.12.18	11	20.12.18	11	Koordination & Synchronisation III: Gruppenkommunikation	20.12.18	11
52	28.12.18		keine LV: Weihnachtszeit	26.12.18	keine LV: Weihnachtszeit	27.12.18	keine LV: Weihnachtszeit		27.12.18	keine LV: Weihnachtszeit
1	04.01.19		keine LV	02.01.19	keine LV	03.01.19	keine LV		03.01.19	keine LV
2	11.01.19		ggf. Exkursion	09.01.19	Prüfungsvorbereitung: Erstellung Klausurantworten / Spickzettel	10.01.19	Gastvortrag: Cloud Computing in der Projektpraxis		10.01.19	Prüfungsvorbereitung: Erstellung Klausurantworten / Spickzettel
3	18.01.19		Prüfungsvorbereitung (PV)	16.01.19	Prüfungsvorbereitung: Erstellung Klausurantworten / Spickzettel	17.01.19	Prüfungsvorbereitung		17.01.19	Prüfungsvorbereitung: Erstellung Klausurantworten / Spickzettel
4	25.01.19		1. PRZ	23.01.19	1. PRZ	24.01.19	1. PRZ		24.01.19	1. PRZ



Prüfungsleistung

- 1. & 2. Zug: Klausur (Dauer: 90 min.) im 1. und 2. PRZ
- Voraussetzung zur Teilnahme an der Klausur:
 - » 60% sinnvolle Bearbeitung der Übungsaufgaben
 - » Mehr dazu in den Übungen

AI-B.41: Verteilte Systeme

- Lecture Notes [SL] -

I. Einführung

C. Schmidt | SG AI | FB 4 | HTW Berlin

Stand: WiSe 18/19

Urheberin: Prof. Dr. Christin Schmidt

Verwertungsrechte: keine außerhalb des Moduls

Lernziele

Nach dieser Lehrveranstaltung kennen Studierende idealerweise:

- (Arbeits-)Definitionen wichtiger Begriffe und deren Abgrenzung im Themenfeld verteilter Systeme
 - Merkmale und Eigenschaften verteilter Systeme
 - Bedeutung und Kategorien von Transparenz
 - Vor- und Nachteile verteilter Systeme
 - Erste Ansätze zur Bewertung von verteilten Systemen / verteilten Architekturen
 - Grundlegende Aspekte hinsichtlich Skalierbarkeit und Interaktionskomplexität in verteilten Systemen
 - Ausgewählte Beispiele verteilter Systeme
- ✓ Studierende sollen grundlegende Begriffe und Aspekte im Bereich „Verteilte Systeme“ kennen lernen.

„Verteiltes System“: Begriffseingrenzung [I]



„A distributed system is one in which the failure of a computer you didn't even know existed can render your own computer unusable.“

Lamport [1987]; online:
<http://research.microsoft.com/en-us/um/people/lamport/pubs/distributed-system.txt> [last accessed: 1 XI 17]

Freie Übersetzung:

„Sie wissen, dass Sie eines haben, wenn der Absturz eines Computers, von dem Sie nie zuvor gehört haben, verhindert, dass Sie Ihre Arbeit erledigen können.“

Lamport [1987; in: Tanenbaum & van Steen 2008: 23]

?

Welche Software-Anwendungen haben Sie auf welchen Endgeräten in den letzten beiden Wochen am häufigsten genutzt?



Timeboxing [~5 min.]

- ✓ Welche von diesen Anwendungen benötigen andere, nicht auf Ihrem Endgerät selbst verfügbare Ressourcen (z.B.: Web-, Fileserver, Dienste)?
- ✓ Beschreiben Sie das Zusammenspiel Ihrer ausgewählten Anwendung mit anderen Ressourcen.

Verteilungsaspekte

vgl. Anthony [2016:10]

- Welche Aspekte eines Systems sind verteilt?
 - » Dateien
 - » Daten
 - » Betriebssystem
 - » Sicherheitsmechanismen
 - » Workload
 - » Supplementäre Prozesse (zur Aufrechterhaltung des Verteilten Systems)

- Wie sind diese Aspekte verteilt?

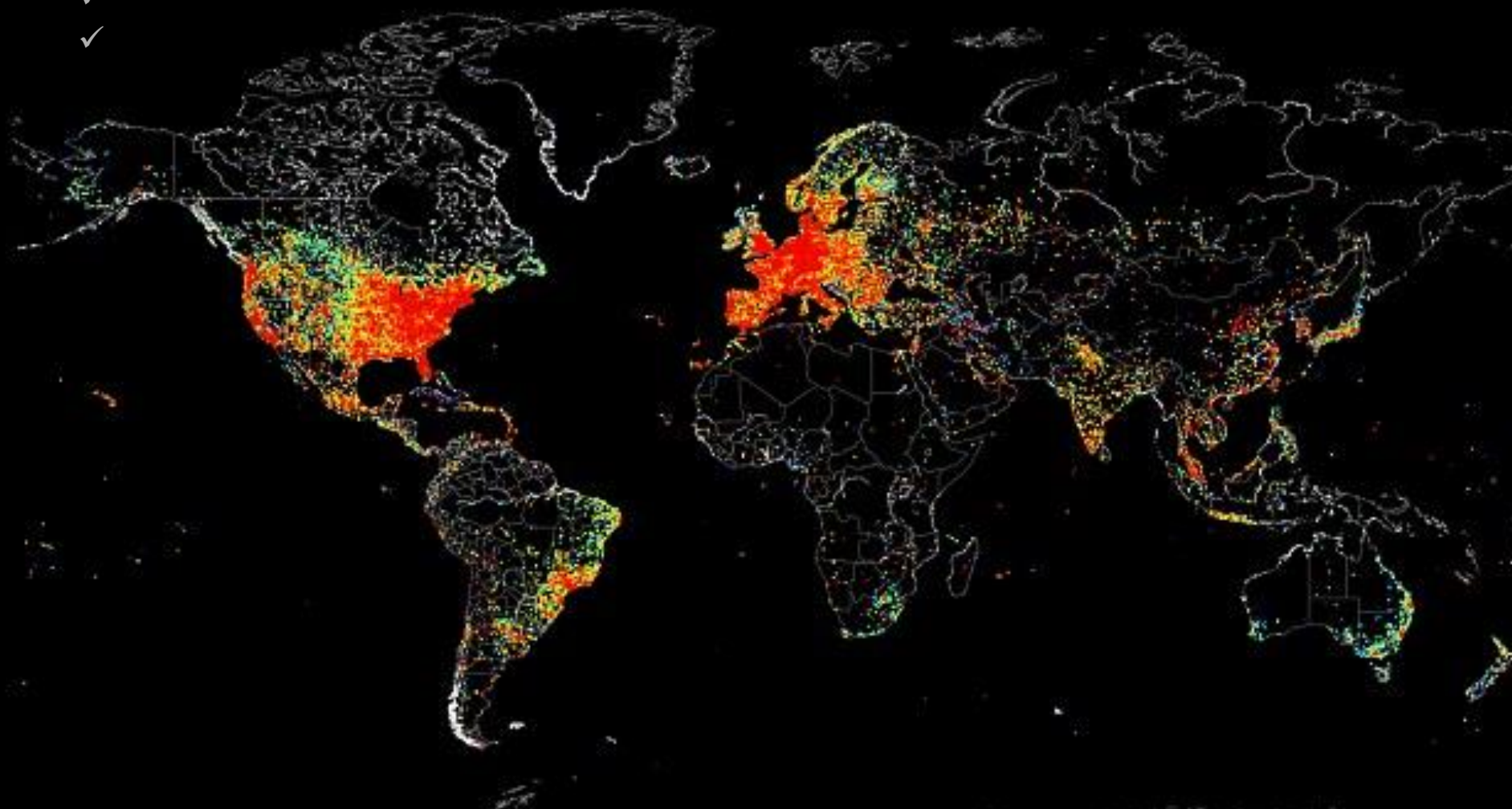
Beispiele verteilter Systeme [I]

Internet / World Wide Web

SHODAN

✓

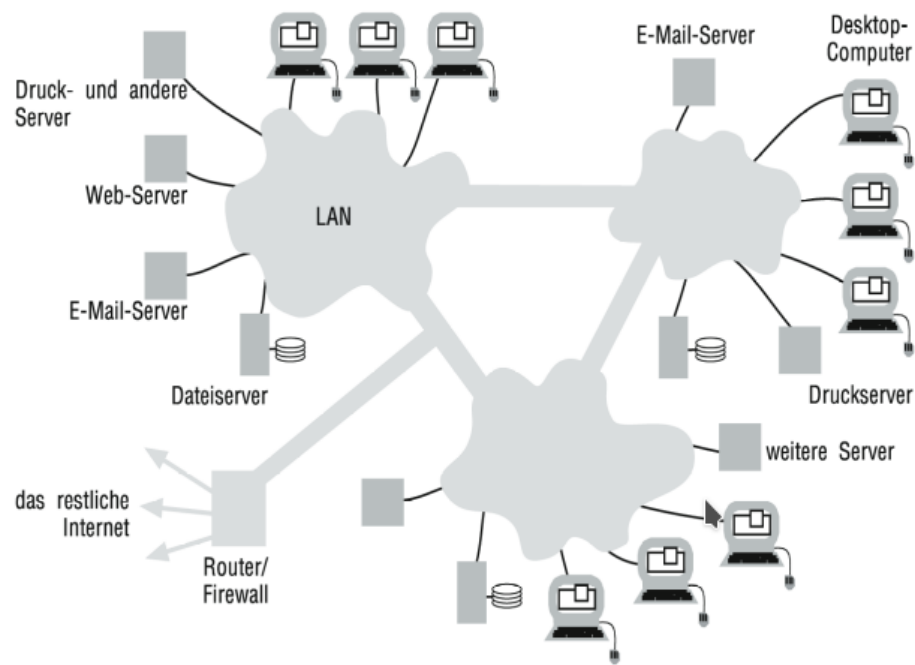
✓



Beispiele verteilter Systeme [II]

Intranet

- Teil des Internets
- Separate Verwaltung mit Abgrenzung zum Internet
 - » Router = Bindeglied zwischen Internet und Intranet
 - » Firewall = Schutzmechanismus für das Intranet



Quelle: Coulouris et al. [2002: 21]

Beispiele verteilter Systeme [III]

Viele weitere als „Cloud Services“, z.B.:

- Filesysteme: Dropbox
- Social Media: Twitter, Facebook
- Entertainment: Streaming-Provider (z.B.: Netflix)
- eCommerce-Anwendungen: Amazon
- Performance: Amazon EC2
- Unternehmensanwendungen (Enterprise Application Integration)
- Multimedia-Systeme
- Sensornetzwerke (z.B.: im Gesundheitswesen, Industrie 4.0)
- E-Mail-Applikationen

Was macht diese Systeme zu einem verteilten System? Schauen wir uns nun an, was man darunter versteht ...

?



Was verstehen Sie unter einem
„verteilten System“ ?

Timeboxing [5 min.]

- ✓ Erklären Sie mit eigenen Worten, was sich hinter dem Begriff verbergen könnte.
- ✓ Versuchen Sie, ein oder zwei praktische Beispiele zu beschreiben.

„Verteiltes System“: Begriffseingrenzung [II]



Definitionen aus der Fachliteratur (Auszug):

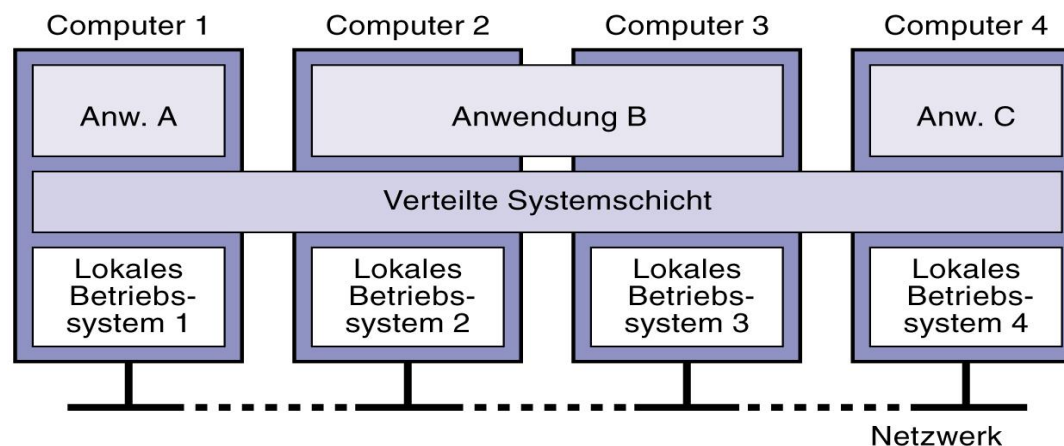
Definition	Quelle(n)
„...als System, in dem sich Hardware oder Software-Komponenten auf vernetzten Computern befinden und nur über den Austausch von Nachrichten kommunizieren und ihre Aktionen koordinieren .“	Coulouris et al. [2002:18]
„Ein verteiltes System ist eine Ansammlung unabhängiger Computer, die den Benutzern wie ein einzelnes kohärentes System erscheinen.“	Tanenbaum & van Steen [2008:19]
„Ein verteiltes System (distributed system) besteht aus unabhängigen über ein Rechnernetz kommunizierenden Rechnern, wobei keine zentrale Systemsteuerung existiert und der Verteilungsaspekt für die Benutzer des Systems möglichst transparent ist.“	Schneider & Werner [2001:533]
„Ein Verteiltes System setzt sich aus mehreren Einzelkomponenten auf unterschiedlichen Rechnern zusammen, die in der Regel nicht über gemeinsamen Speicher verfügen und somit mittels Nachrichtenaustausch kommunizieren, um in Kooperation eine gemeinsame Zielsetzung – etwa die Realisierung eines Geschäftsablaufs – zu erreichen .“	Schill & Springer [2012:4]
„Ein lose gekoppeltes System (auch verteiltes System genannt) besteht aus mehreren gekoppelten Prozessoren ohne gemeinsamen Speicher (Hauptspeicher).“	Oechsle [2011:12]
„Distributed Systems are therefore inherently nondeterministic: running a system twice from the same initial configuration may yield different results .“	Fokkink, W. [2013: 1]
„Reduced to the simplest terms, a distributed computing system is a set of computer programs , executing on one or more computers, and coordinating actions by exchanging messages .[...] Most distributed systems operate over computer networks , but one can also build a distributed computing system in which information flows between the components by means other than message passing .“	Birman, K. P. [2012: 50]
„A distributed computing system is one where the resources used by the applications are spread across numerous computers which are connected by a network. [...] A distributed application is one in which the program logic is spread across two ore more software components which may execute on different computers within a distributed system. The components have to communicate and coordinate their actions in order to carry out the computing task of the application.“	Anthony [2016:9]

„Verteiltes System“: Merkmale

- Vorhandensein von Kommunikation (meist über Nachrichtenaustausch (engl.: „message passing“), d.h. verschiedene Prozesse kommunizieren miteinander)
- Gleichzeitiges / zusammen hängendes Ablaufen von Vorgängen:
 - » auf mehreren Rechnern (impliziert in den meisten Fällen die Existenz eines Netzwerks), oder
 - » Auf einem Rechner (verschiedene Cores)
 - » Nebenläufigkeit (parallel / pseudoparallel) von Prozessen
- Kein gemeinsamer Hauptspeicher
- Kooperation durch Koordination
 - » Keine globale Uhr
 - » keine zentrale Steuerung
- Verteilungsaspekte „transparent“ für den Nutzer, d.h. er hat das Gefühl, mit einem einzigen System zu interagieren
- Möglichkeit des Ausfalls von Komponenten eines Verteilten Systems

„Verteiltes System“: Der Begriff „Middleware“

- Häufig werden Verteilte Systeme mittels einer Softwareschicht angeordnet
- Jede (verteilte) Anwendung hat eine (gleiche) Schnittstelle zu dieser Softwareschicht als verteilte Systemschicht (Middleware)
 - » Anwendungen können über die Middleware mit anderen Anwendungen kommunizieren
 - » Anwendungen können auf mehrere Knoten verteilt werden
- Verteilung ist (idealerweise) „transparent“



Quelle: Tanenbaum & van Steen [2008:20]

Ziele

vgl. Tanenbaum & van Steen [2008:20ff.]

1. Ein verteiltes System sollte Ressourcen leicht zugreifbar machen.
1. Es sollte die Tatsache vernünftig **verbergen**, dass Ressourcen über ein Netzwerk verteilt sind.
1. Es sollte offen sein.
1. Es sollte **skalierbar** sein.

„Transparenz“: Begriffseingrenzung

Eine Definition:

„Ein verteiltes System, das in der Lage ist, sich Benutzern und Anwendungen so darzustellen, als sei es nur ein einziges Computersystem, wird als transparent bezeichnet.“

Tanenbaum & van Steen [2008:21]

- ✓ „Transparent“ = durchsichtig, im Sinne einer nicht sichtbaren Struktur
- ✓ Ziel/Herausforderung bei der Entwicklung von verteilten Systemen:
 - ✓ Verschleierung/Verbergung einer Verteilung und der Existenz verschiedener, heterogener Komponenten
 - ✓ Schaffung der „Illusion einer Einzelanwendung“ [Anthony 2016:12]

Transparenz: Arten

Transparenz	Beschreibung
Zugriff	Verbirgt Unterschiede in der Datendarstellung und die Art und Weise, wie auf eine Ressource zugegriffen wird (Zugriff auf lokale und entfernte Ressourcen erfolgt in gleicher Weise und unter Verwendung identischer Operationen)
Ort	Verbirgt, wo sich eine Ressource befindet (Zugriff auf Ressource ist ohne Kenntnis ihres Orts/ihrer Position möglich)
Nebenläufig-keit	Verbirgt, dass eine Ressource von mehreren konkurrierenden Benutzern gleichzeitig genutzt werden kann (erlaubt Prozessen die gleichzeitige Nutzung von Ressourcen ohne gegenseitige Störung und Konsistenzverletzungen)
Replikation	Verbirgt, dass eine Ressource repliziert ist (Verwendung mehrerer Instanzen von Ressourcen ohne Kenntnis des Prozesses/Benutzers darüber)
Fehler	Verbirgt den Ausfall und die Wiederherstellung einer Ressource (Adäquate Fehlerverarbeitung)
Mobilität/Relokation	Verbirgt, dass eine Ressource an einen anderen Ort ohne Beeinträchtigung von Nutzern/Prozessen innerhalb eines Systems (auch während der Nutzung) verschoben (migriert) werden kann
Leistung	Verbirgt die Möglichkeit der Neukonfiguration zur Leistungsverbesserung
Skalierung	Verbirgt die Möglichkeit der Erweiterung des Systems ohne Änderung der Gesamtstruktur / der Anwendungen des Systems (vgl. „Skalierbarkeit“: Begriffseingrenzung)

Quelle: in Anlehnung an Tanenbaum & van Steen [2008:22]; Coulouris et al. [2002:42f.]; Anthony [2016:371ff.]

Transparenz: Arten

Transparenz	Beschreibung
Zugriff	Verbirgt Unterschiede in der Datendarstellung und die Art und Weise, wie auf eine Ressource
<div>?</div> <div>Was verstehen Sie unter einer „Ressource“?</div>	
	werden kann (erlaubt Prozessen die gleichzeitige Nutzung von Ressourcen ohne gegenseitige Störung und Konsistenzverletzungen)
Replikation	Verbirgt, dass eine Ressource repliziert ist (Verwendung mehrerer Instanzen von Ressourcen ohne Kenntnis des Prozesses/ Benutzers darüber)
Fehler	Verbirgt den Ausfall und die Wiederherstellung einer Ressource (Adäquate Fehlerverarbeitung)
Mobilität/Relokation	Verbirgt, dass eine Ressource an einen anderen Ort ohne Beeinträchtigung von Nutzern/Prozessen innerhalb eines Systems (auch während der Nutzung) verschoben (migriert) werden kann
Leistung	Verbirgt die Möglichkeit der Neukonfiguration zur Leistungsverbesserung
Skalierung	Verbirgt die Möglichkeit der Erweiterung des Systems ohne Änderung der Gesamtstruktur / der Anwendungen des Systems

„Ressource“: Begriffseingrenzung

„Ressourcen sind abstrakte Konzepte, die nicht notwendigerweise eine Speicherung in irgendeiner Form beinhalten müssen.“

Tanenbaum & van Steen [2008:21]

- ✓ Abstraktion von Dingen, die in einem Computersystem oder in einem verteilten System sinnvoll gemeinsam genutzt werden können
- ✓ Beispiele:
 - » Hardware (z.B.: Drucker, CPU, Festplattenkapazität)
 - » Software (z.B.: Dateien, Software-Objekte/Applikationen, Datenobjekte)
- ✓ Dienste (Services) verwalten eine Menge verwandter Ressourcen (z.B.: Dateidienst, Druckdienst)

„Verteilte Systeme“: Vor- und Nachteile



- Gemeinsame Ressourcennutzung
- Hohe Leistung durch Parallelisierung von Prozessen
- Lastausgleich
- Verfügbarkeit
- Fehlertoleranz
- Ausfalltoleranz
- Skalierbarkeit
- Erweiterbarkeit
- Herstellerunabhängigkeit
- Kostenreduktion



- Erhöhte Komplexität beim Design- und Implementierungsprozess
 - » Heterogenität
 - » Koordination
 - » Synchronisation / Timing
 - » Verifikation
 - » Sicherheit
 - » ...

Falsche Grundannahmen für Entwickler_innen



Tipps / Quellen zur Vertiefung:

Rotem-Gal-Oz, A. [o.J.], Wilson [2015], Tanenbaum
& Van Steen [2008:33f.]

1. Das Netzwerk ist zuverlässig.
The network is reliable.
1. Das Netzwerk ist sicher.
The network is secure.
1. Das Netzwerk ist homogen.
The network is homogeneous.
1. Die Topologie ändert sich nicht.
Topology doesn't change.
1. Die Latenzzeit beträgt null.
Latency is zero.
1. Die Bandbreite ist unbegrenzt.
Bandwidth is infinite.
1. Die Übertragungskosten betragen null.
Transport cost is zero.
1. Es gibt nur einen Administrator.
There is one administrator.

Bewertung von verteilten Systemen / verteilten Architekturen

- ✓ Es besteht eine Vielfalt möglicher Ausprägungen / Kategorien und Parameter

Parameter	Quelle(n)
Grad: <ul style="list-style-type: none">– Heterogenität– Offenheit– Sicherheit– Skalierbarkeit– Fehlerverarbeitung– Nebenläufigkeit– Transparenz	Coulouris et al. [2002:34ff.]
<ul style="list-style-type: none">– Positionierung im SW-Lebenszyklus (Analyse, Design, Entwicklung, Test, Betrieb)– Management und Umfeld– Interaktivität– Teilnehmerzahl– Ressourcenbedarf– Dynamik– Robustheit (gegenüber Fehlern)	Dunkel et al. [2008:237]

- ✓ Inhomogene Schemata & Interpretationsspielraum begründen Herausforderungen:
 - ✓ Für den Entwurf / Implementierung eines Verteilten Systems
 - ✓ Evaluation und (weitere) Verbesserung des Betriebs eines bestehenden Verteilten Systems
- ✓ Im Folgenden: beispielhafte Vertiefung nach Coulouris et al. [2002]

Heterogenität

vgl. Coulouris et al. [2002:34ff.]

- Netzwerke
 - Computerhardware-komponenten
 - Betriebssysteme
 - Programmiersprachen
 - Implementierungen
-
- ✓ Ziel: Überwindung der Heterogenität (Schaffung von Transparenz) z.B. durch Nutzung von anerkannten Standards in der Kommunikation (Protokolle, Middleware: Java RMI, Webservices, Rest-API,...)

Offenheit

vgl. Coulouris et al. [2002:34ff.]; Bengel et al. [2008:31ff.]

- Grad der Erweiterbarkeit eines Systems auf Basis der Ausprägung in den folgenden Dimensionen:
 - » Offenlegung/Dokumentation von nutzbaren Schnittstellen
 - » Nutzung von einheitlichen Kommunikationsmechanismen
 - » Herstellerunabhängigkeit
- Eigenschaft eines VS, nicht in sich abgeschlossen zu sein, sondern:
 - » Erweiterbar zu sein
 - » Skalierbar zu sein
 - » Mit Sub-Systemen interagieren zu können
- Herausforderungen:
 - » Verbreitungs-Monotonie: Informationen / Nachrichten in einem offenen VS können nach Verbreitung nicht mehr zurück genommen werden
 - » Pluralismus: Sub-Systeme enthalten ggf. heterogene, überlappende (evtl. in Konflikt) stehende Information (fehlen einer zentralen Instanz zur Ermittlung der „Wahrheit“)
 - » Unbegrenzter Nichtdeterminismus: asynchrone Subsysteme können kommen und gehen, Verbindungen und Kanäle können sich ändern (es ist nicht vorhersehbar, wann eine Operation innerhalb eines VS abgeschlossen / beendet ist)

Sicherheit

vgl. Coulouris et al. [2002:34ff.]

- Qualitativ: Wert von Informationsressourcen im Hinblick auf:
 - » Vertraulichkeit (Schutz gegen die Offenlegung gegenüber nicht berechtigten Personen)
 - » Integrität (Schutz gegen Veränderung / Beschädigung)
 - » Verfügbarkeit (Schutz gegen Störungen der Methoden zum Ressourcenzugriff)
- Strategien:
 - » Schaffung von Barrieren (Firewalls)
 - » Verbergung des Inhalts einer Nachricht (Verschlüsselungstechniken)
 - » Sicherstellung der Identität eines Benutzers (Authentifizierungsverfahren)
- ✓ Ziel: Angemessene Sicherheit, d.h. VS müssen sicherstellen, dass sensible Informationen in einer Nachricht sicher innerhalb des Netzwerkes übertragen werden
- ✓ Nicht Fokus dieses Moduls, da eigenes Lehrangebot im Curriculum (Bachelor) des Studiengangs Angewandte Informatik

„Skalierbarkeit“: Begriffseingrenzung

Definition	Quelle(n)
„Ein System, dass als skalierbar bezeichnet wird, bleibt auch dann effektiv, wenn die Anzahl der Ressourcen und die Anzahl der Benutzer wesentlich steigt.“	Coulouris et al. [2002:38]
„Die Skalierbarkeit eines Systems lässt sich in mindestens drei unterschiedlichen Dimensionen messen (Neumann, 1994). Erstens kann ein System skalierbar in Hinblick auf seine Größe sein, was bedeutet, dass wir dem System ganz einfach weitere Benutzer und Ressourcen hinzufügen können. Zweitens ist ein System geografisch skalierbar, wenn die Benutzer und die Ressourcen weit auseinanderliegen können. Drittens kann ein System administrativ skalierbar, also auch dann noch einfach zu verwalten sein, wenn es sich über viele unabhängige administrative Organisationen erstreckt.“	Tanenbaum & van Steen [2008:26]
„Skalierbarkeit bezeichnet die Fähigkeit eines Systems, wachsende quantitative Anforderungen durch Hinzufügen von Ressourcen auszugleichen , ohne dass eine Änderung von Systemkomponenten notwendig wird.“	Schill & Springer [2012:6]
„Das Verhalten der Leistung eines parallelen Programmes bei steigender Prozessoranzahl wird durch die Skalierbarkeit (engl. scalability) erfasst. Die Skalierbarkeit eines parallelen Programmes auf einem gegebenen Parallelrechner ist ein Maß für die Eigenschaft, einen Leistungsgewinn proportional zur Anzahl p der verwendeten Prozessoren zu erreichen.“	Rauber & Rüniger[2012:179]
„Ein skalierbares System lässt sich leicht und flexibel ändern in der Anzahl der Benutzer, der Betriebsmittel, der Rechner, der Anwendungen und der Größe der Datenspeicher. Für den Benutzer sind diese Änderungen transparent (Skalierungstransparenz) und der Benutzerbetrieb bleibt von diesen Änderungen unbeeinflusst.“	Bengel et al. [2008:29f.]

Skalierbarkeit: Einflussfaktoren

vgl. Coulouris et al. [2002:34ff.]

- Kosten für physische Ressourcen
- Art der genutzten Algorithmen (zentral vs. dezentral¹, linear vs. hierarchisch²)
- Architektur: Aufteilung von Ressourcen auf verschiedene Komponenten (Caching, Replikation)
- Erschöpfung von Ressourcen
- Qualität der Evaluation/Antizipation der aktuellen/zukünftigen Bedarfssituation der Ressourcen eines VS

¹ Dezentrale Algorithmen sind in verteilten Systemen gemäß Tanenbaum & van Steen [2008:28] im Vergleich zu zentralen Algorithmen besser skalierbar und zu präferieren. Eigenschaften dezentraler Algorithmen umfassen, dass 1.) kein Computer eine vollständige Information über den Systemstatus besitzt, 2.) Computer nur aufgrund lokaler Informationen entscheiden, 3.) der Ausfall eines Computers nicht den Algorithmus schädigt und 4.) nicht angenommen wird, dass es eine globale Uhr gibt (zu Zeitkonzepten und Problemen vgl. Lecture Notes „rdination & Synchronisation II: Time“).

² Coulouris et al. [2002:38] konstatieren, dass Algorithmen besser skalierbar sind, wenn sie hierarchische Strukturen anstelle von linearen Strukturen verwenden (vgl. „Beispiel: Verteilung“)

Skalierung

vgl. Tanenbaum & van Steen [2008:26]; Bengel et al. [2008:29ff.]

Dimensionen:

- Lastskalierbarkeit: Hinzufügen / Einschränken von Ressourcen in Abhängigkeit der Last (groß / gering) und in für den Benutzer transparenter Weise (Migrationstransparenz)
- Geographische Skalierbarkeit: keine Einschränkung des Gebrauchs durch geographische Entfernung zwischen Ressource und Nutzer_in
- Administrative Skalierbarkeit:
 - » Anzahl der Organisationen, die sich ein VS teilen ist nicht beschränkt
 - » Management, Monitoring und Gebrauch sollte einfach und von überall aus möglich sein

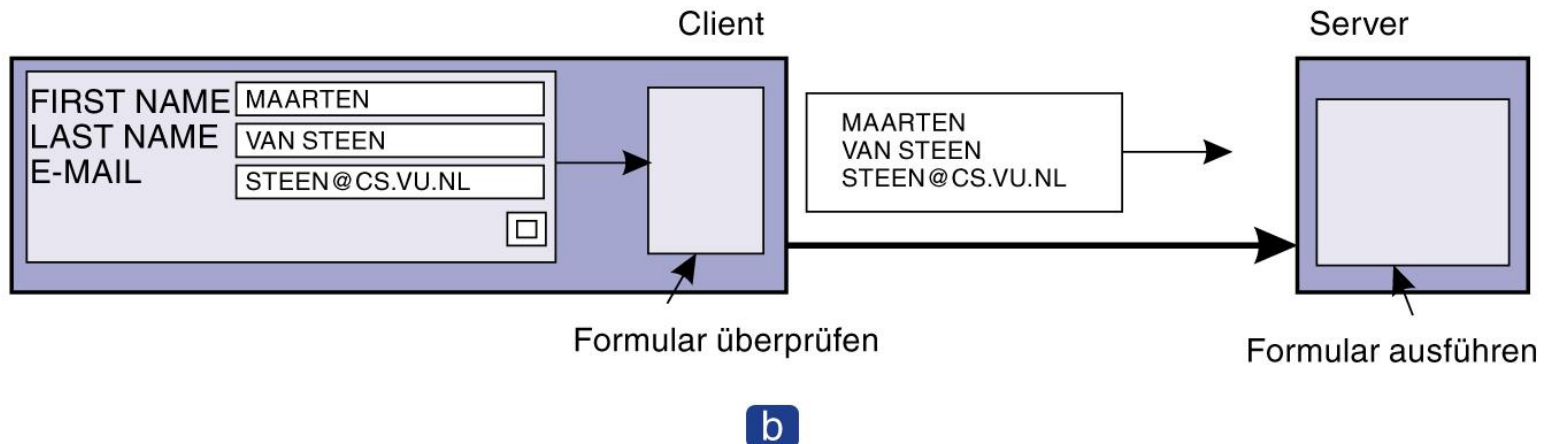
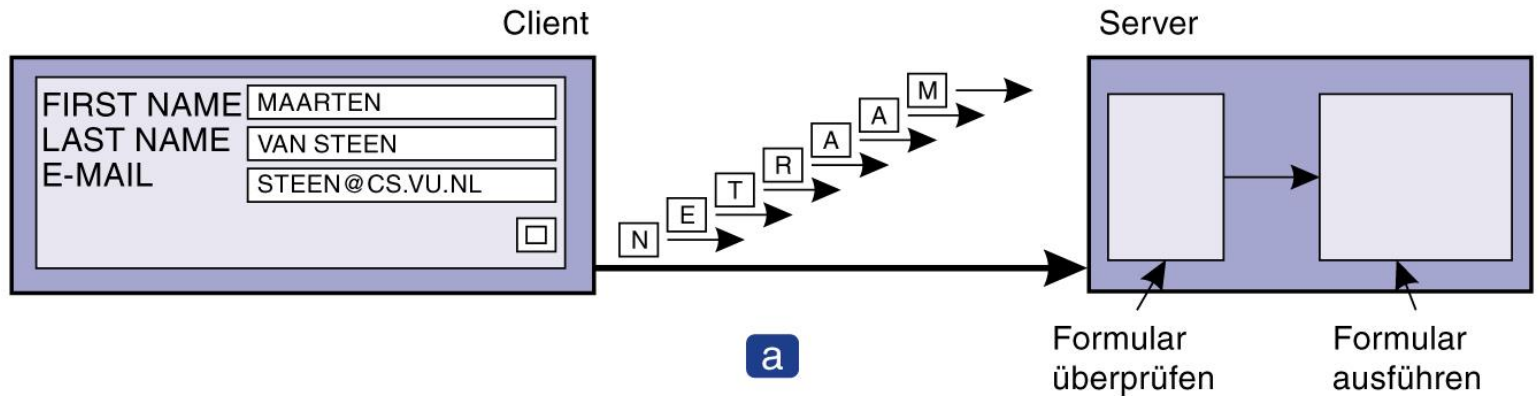
Techniken (Beispiele vgl. ff.)

1. Verbergen der Latenzzeiten der Kommunikation
2. Verteilung
3. Replikation (& Caching)

Skalierungstechniken [I]

Verbergung der Latenzzeiten der Kommunikation

Beispiel: Überprüfung beim Ausfüllen von Datenbank-Formularen (a) auf dem Server oder (b) auf dem Client



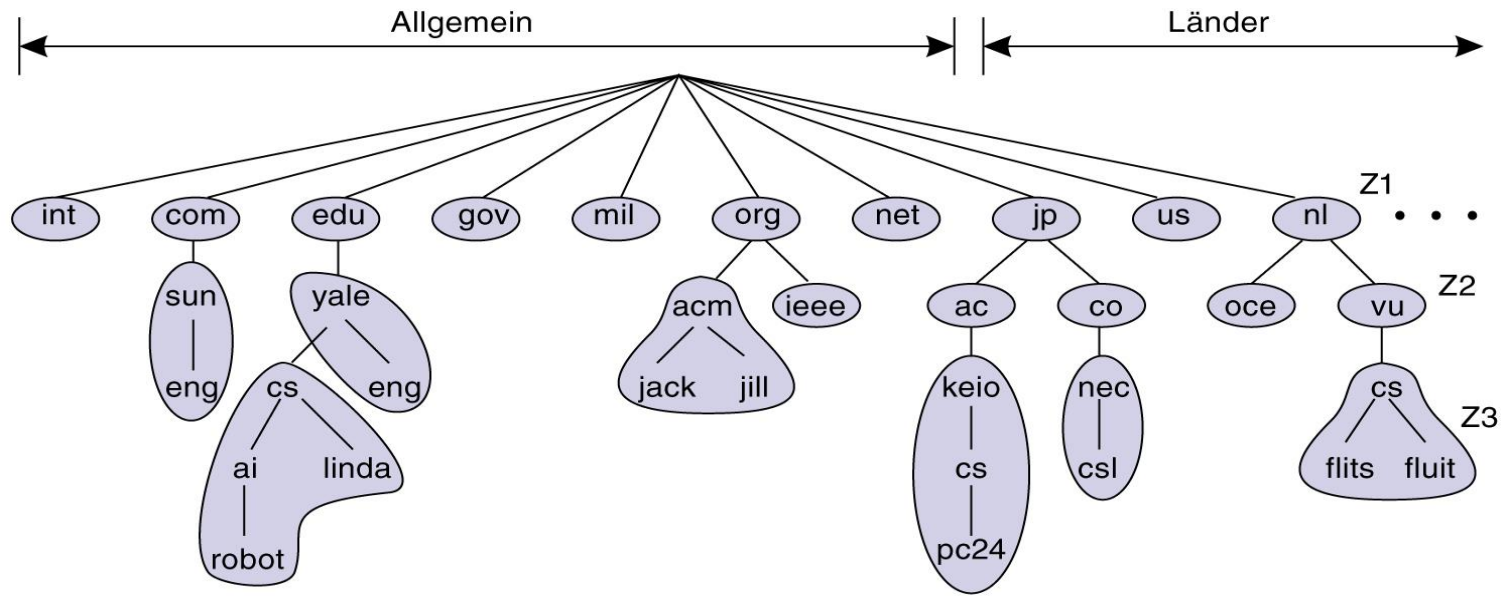
Quelle: Tanenbaum & van Steen [2008:30]

Skalierungstechniken [II]

Verteilung / Unterteilung

Beispiel: Internet-Domain-Name-System (DNS)

- ✓ Namensdienst ist über mehrere Computer verteilt
- ✓ Es wird vermieden, dass lediglich ein einzelner Server alle Anforderungen zur Namensauflösung behandeln muss (Verteilung: ein Server pro Zone im Baum)



Quelle: Tanenbaum & van Steen [2008:31]

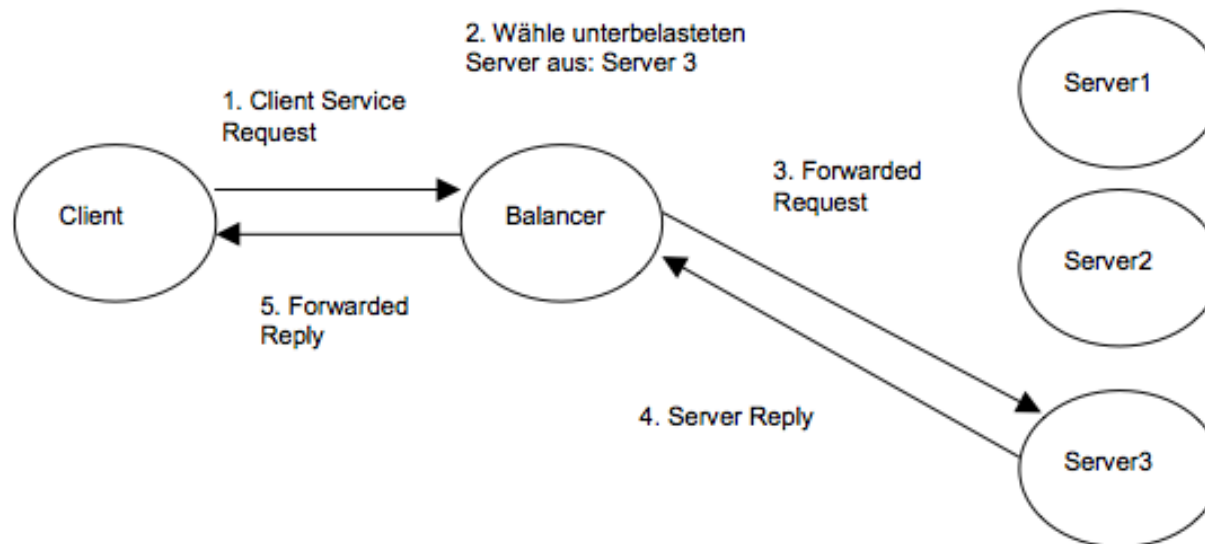
Verteilung & Skalierung: horizontal vs. vertikal

	Verteilung	Skalierung
Vertikal	<ul style="list-style-type: none">- Verteilung unterschiedlicher Funktionalitäten einer mehrstufigen Architektur auf mehrere Rechner- Jede Stufe ist auf einem anderen Rechner implementiert (Multi-Tier-System“)	„Scale up“: Steigern der Leistung durch das Hinzufügen von Ressourcen zu einem Knoten/Rechner des Systems (z.B. Speicherplatz, CPU)
Horizontal	<ul style="list-style-type: none">- Verteilung von Funktionalität in logische Schichten auf einem Rechner- Beispiel: IP-Implementierung auf verschiedenen Schichten	„Scale out“: Horizontale Skalierung bedeutet die Steigerung der Leistung eines Systems durch das Hinzufügen zusätzlicher Rechner/Knoten.

Skalierungstechniken [III]

Replikation

Beispiel: Skalierbares C+S_{Ba}S+-System: Server verteilt als *Balancer* Arbeitslast auf mehrere replizierte Server in niedrigster Stufe (vgl. „Architektur“)

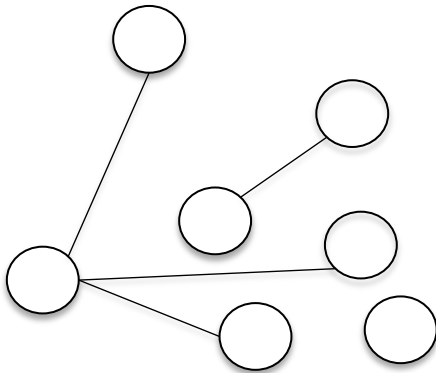


Quelle: Bengel [2004:76]

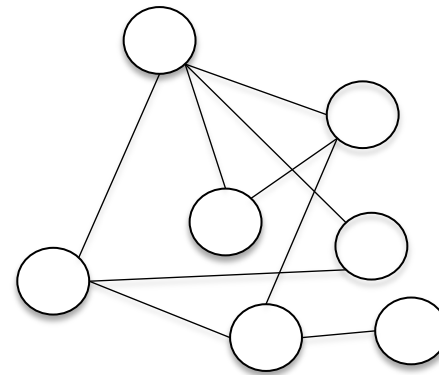
Interaktionskomplexität [I]

vgl. Anthony [2016:401ff.]

- Bezeichnet Anteil / Anzahl der Kommunikationsbeziehungen eines Komponenten eines Verteilten Systems mit anderen
- Beeinflusst Skalierbarkeit



Geringe Interaktionskomplexität
(aufgrund loser Kopplung)



Hohe Interaktionskomplexität
(aufgrund enger Kopplung)

Interaktionskomplexität [II]

vgl. Anthony [2016:401ff.]

- Bezeichnet Anteil / Anzahl der Kommunikationsbeziehungen eines Komponenten eines Verteilten Systems mit anderen
- Beeinflusst Skalierbarkeit

Anteil der Kommunikationen zwischen den Komponenten (N = alle Komponenten)	Interaktionskomplexität	Interpretation
1	$O(N)$	Jede Komponente kommuniziert mit einer anderen Komponente. Dieses Szenario zeichnet sich durch hohe Skalierbarkeit aus, da die Interaktionskomplexität linear mit der Systemgröße ansteigt.
2	$O(2N)$	Jede Komponente kommuniziert mit zwei anderen Komponenten. Die Interaktionskomplexität steigt linear mit der Systemgröße an.
$N/2$	$O(N^2/2)$	Jede Komponente kommuniziert mit annähernd der Hälfte des Systems. Die Interaktionskomplexität steigt exponentiell mit der Systemgröße an (beeinflusst Skalierbarkeit!)
$N-1$	$O(N^2-N)$ Syn.: $O(N(N-1))$	Jede Komponente kommuniziert mit annähernd allen Komponenten des Systems. Die Interaktionskomplexität steigt exponentiell mit der Systemgröße an (beeinflusst Skalierbarkeit!)

Fehlerverarbeitung

vgl. Coulouris et al. [2002:34ff.]

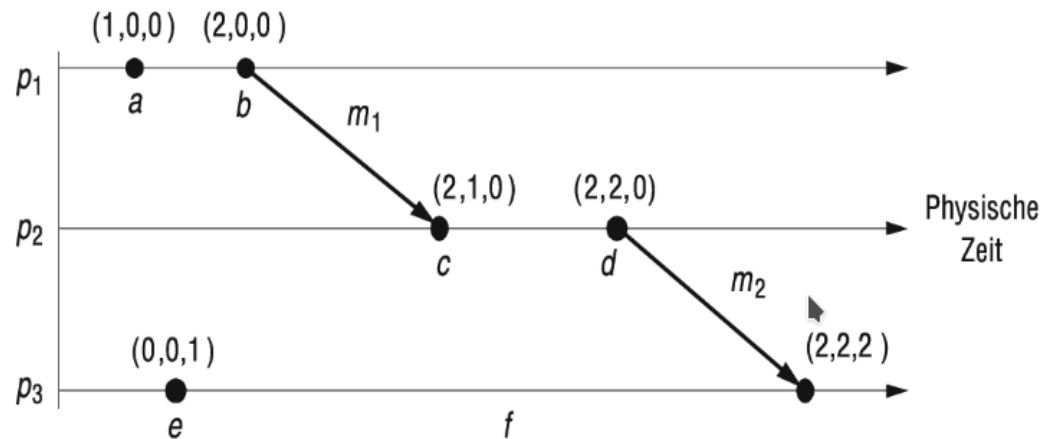
Ausgangspunkt: Fehler treten in Verteilten Systemen partiell auf (Konsequenz: falsche Ergebnisse, Verarbeitungsabbruch)

- Ziel bei Fehlerverarbeitung: Sicherung der Funktionsfähigkeit des Verteilten Systems
- Techniken:
 - » Fehler erkennen (z.B. durch Verwendung von Prüfsummen/Hashfunktionen)
 - » Fehler maskieren
 - Wiederholung der Übertragung einer Nachricht
 - Toleranz durch Redundanz der Komponenten (Routingwege, Server, Replikation von Datenobjekten)
 - » Fehler tolerieren (Komponent/Nutzer_in wird über Fehler lediglich informiert)
- Herausforderungen:
 - » Fehlererkennung
 - » Fehlerkategorisierung (+ Ableitung von Maßnahmen) Wiederherstellung nach Fehlern zur Erhaltung eines konsistenten Datenbestands

Nebenläufigkeit

vgl. Coulouris et al. [2002:34ff.]

- Anforderungen zur (pseudo-)parallelen Nutzung von Ressourcen/nebenläufigen Ausführung von Prozessen:
 - » Kausale unabhängig der Prozesse
 - » Möglichkeit zur Realisierung gleichzeitiger Ausführung
- Herausforderung: Beachtung der Konkurrenz um die Nutzung der Betriebsmittel der Komponente, die die Ressource zur Verfügung stellt bzw. den Prozess verarbeitet



Quelle: Coulouris et al. [2002:465]

Danke. Lernziele erreicht?

Nach dieser Lehrveranstaltung kennen Studierende idealerweise:

- (Arbeits-)Definitionen wichtiger Begriffe und deren Abgrenzung im Themenfeld verteilter Systeme
 - Merkmale und Eigenschaften verteilter Systeme
 - Bedeutung und Kategorien von Transparenz
 - Vor- und Nachteile verteilter Systeme
 - Erste Ansätze zur Bewertung von verteilten Systemen / verteilten Architekturen
 - Grundlegende Aspekte hinsichtlich Skalierbarkeit und Interaktionskomplexität in verteilten Systemen
 - Ausgewählte Beispiele verteilter Systeme
- ✓ Studierende sollen grundlegende Begriffe und Aspekte im Bereich „Verteilte Systeme“ kennen lernen.

Quellen [I]

Aleksy, M.; Korthaus, A.; Schader, M. (2005) *Implementing Distributed Systems with JAVA and CORBA*; Berlin et al.: Springer.

Anthony, R. (2016) *Systems Programming - Designing and Developing Distributed Applications*; Amsterdam et al.: Morgan-Kaufman / Elsevier.

Bengel, G.; Baun, C.; Kunze, M.; Stucky, K.-U. (2008) *Masterkurs Parallele und Verteilte Systeme*; Wiesbaden: Vieweg & Teubner.

Birman, K. P. (2012) *Guide to reliable Distributed Systems – Building High-Assurance Applications and Cloud-Hosted Services*. London et al.: Springer.

Bollmann, T.; Zeppenfeld, K. (2010) *Mobile Computing*, Herdecke: W3L

Coulouris, G.; Dollimore, J.; Kindberg, T. (2002) *Verteilte Systeme - Konzepte und Design*; 3., überarbeitete Auflage; München: Pearson Studium.

Dunkel, J; Eberhart, A.; Fischer, S.; Kleiner, C. ; Koschel, A. (2008) *Systemarchitekturen für Verteilte Anwendungen*; München: Hanser.

Fokkink, W. (2013) *Distributed Algorithms: an intuitive approach*, Cambridge, MA (USA): MIT Press.

Neumann, B. (1994) *Scale in Distributed Systems*; in: Casavant, T. & Singhal, M. (Hrsg.) *Readings in Distributed Computing Systems*, pp. 463-489. Los Alamitos: IEEE Computer Society Press.

Oechsle, R. (2011) *Parallele und verteilte Anwendungen in JAVA*; 3., erweiterte Auflage; München: Carl Hanser.

Quellen [II]

Schill, A.; Springer, T. (2012) *Verteilte Systeme*; 2. Auflage; Berlin, Heidelberg: Springer Vieweg.

Rauber, T.; Rünger, G. (2012) *Parallele Programmierung*; 3. Auflage; Berlin, Heidelberg: Springer (examen.press).

Rotem-Gal-Oz, A. [o.J.] Fallacies of Distributed Computing Explained ;
online: <http://www.rgoarchitects.com/Files/fallacies.pdf> [letzter Zugriff: 1 XI 17]

Stein, E. (2004) *Taschenbuch Rechnernetze und Internet*; München/Wien: Hanser.

Tanenbaum, A.; van Steen, M. (2008) *Verteilte Systeme – Prinzipien und Paradigmen*; 2., überarbeitete Auflage; München: Pearson Studium.

Varela, C. A. (2013) *Programming Distributed Computing Systems*; Cambridge / London: The MIT Press.

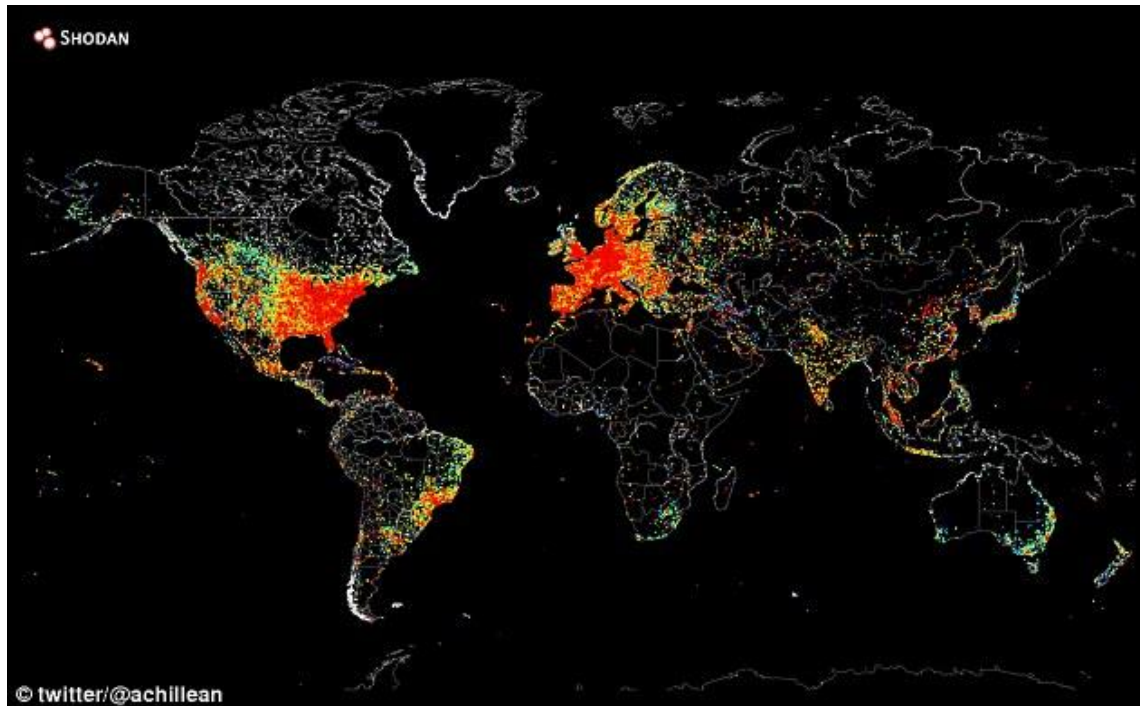
Wilson, G. [2015] Eight Fallacies of Distributed Computing (Tech Talk); online: <https://blog.fogcreek.com/eight-fallacies-of-distributed-computing-tech-talk/> [letzter Zugriff: 1 XI 17]

Appendix

Vertiefung / Ergänzung: Beispiele verteilter Systeme

Internet / World Wide Web [I]

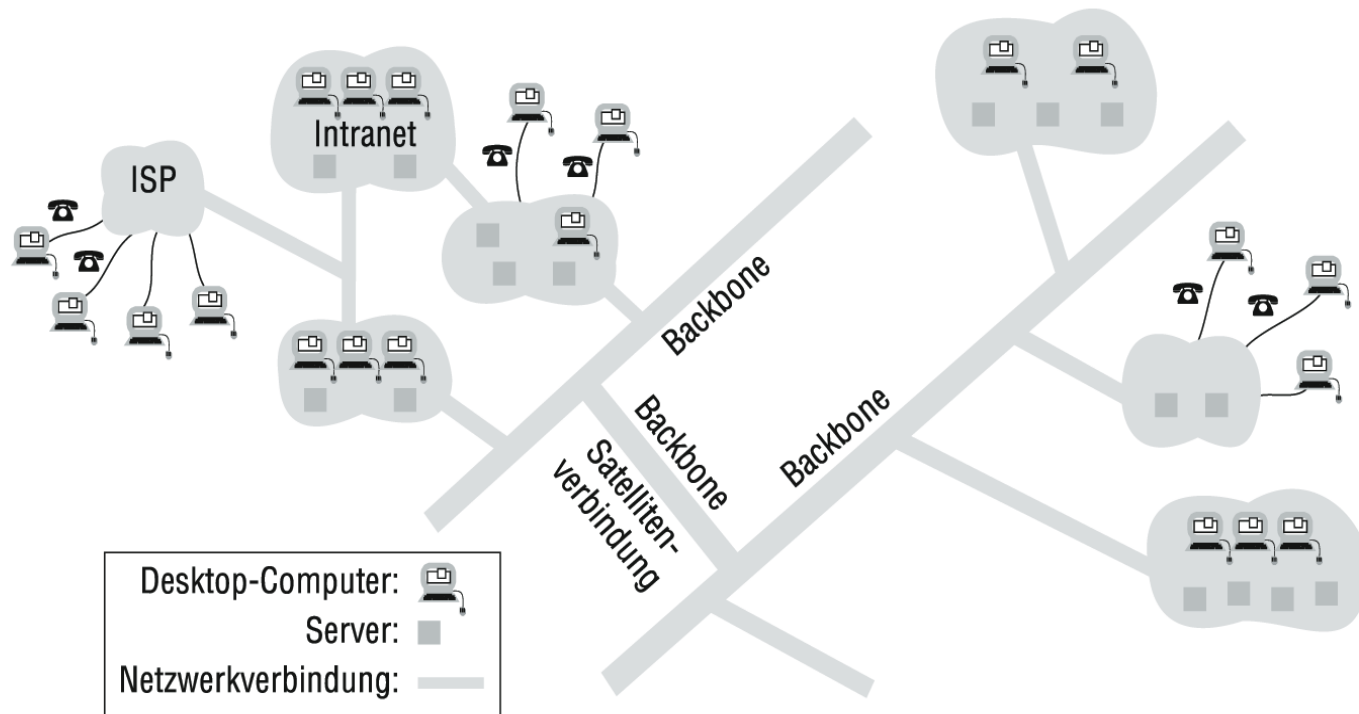
- ✓ Zusammenschluss von Computernetzwerken unterschiedlichster Art
- ✓ Clients (z.B.: Browser) nutzen Dienste von Servern (z.B.: Darstellung einer Website)
- ✓ Riesiges VS



Quelle: http://i.dailymail.co.uk/i/pix/2014/08/29/1409308429430_wps_9_BwJL_bRCcAEt4Hz_png_pinge.jpg ; [letzter Zugriff: 3 XII 17]

Vertiefung / Ergänzung: Beispiele verteilter Systeme

Internet / World Wide Web [II]



Quelle: Coulouris et al. [2002: 19]

Backbone = Netzwerkverbindung mit hoher Übertragungskapazität, z.B. Satellitenverbindungen, Glasfaserkabel

Vertiefung / Ergänzung: Beispiele verteilter Systeme

Internet / World Wide Web [III]

WWW:

- Offenes System
- Dienst des Internet
- Technologie-Standardkomponenten:
 - » HTML (HyperText Markup Language): Sprache zur Darstellung von Seiteninhalt und -layout in einem Browser
 - » URL (Uniform Resource Locator): Bezeichner, der Ressourcen als Teil des Webs identifiziert.
 - » Systemarchitektur: C/S
 - Standardregeln zur Kommunikation (HyperText Transfer Protocol)
 - Browser fungieren als Clients

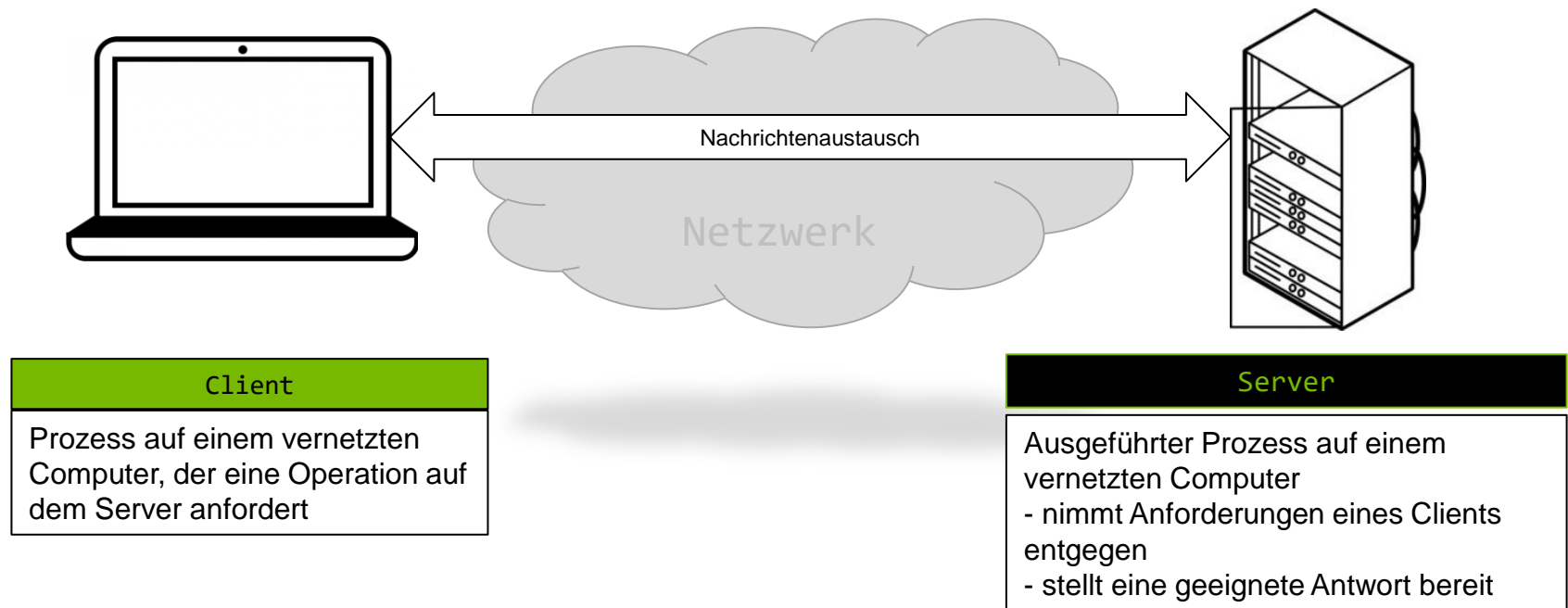
Vertiefung / Ergänzung: Beispiele verteilter Systeme

Internet / World Wide Web [IV]

Gemeinsame Ressourcennutzung im WWW:

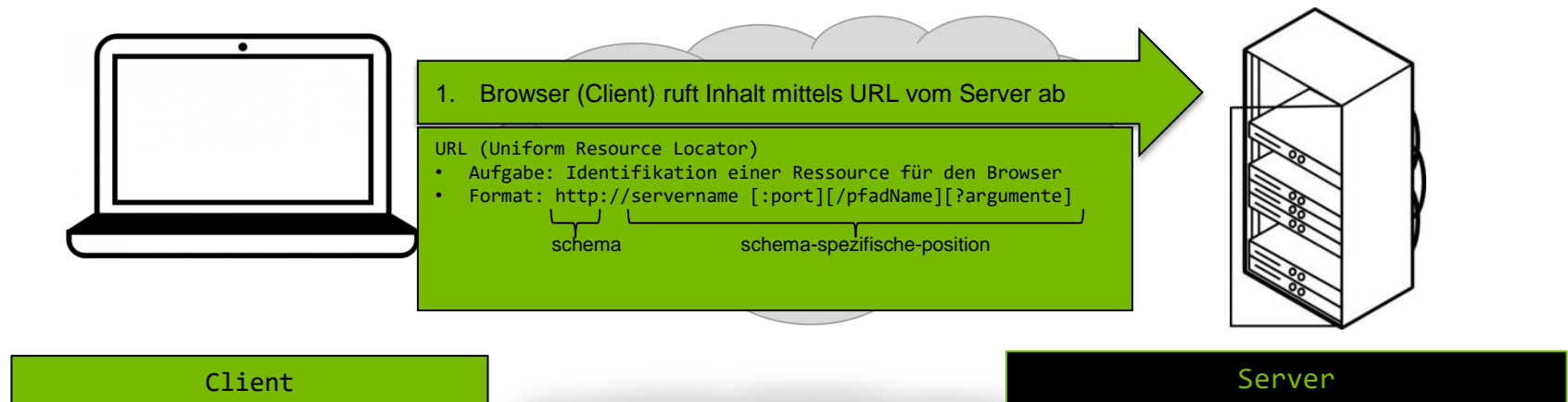
- Ressourcen in einem verteilten System sind (physisch) in Computer gekapselt
- Zugriff auf Ressourcen durch andere Computer kann nur durch Kommunikation erfolgen
- Kommunikation erfolgt innerhalb einer Architektur
- Vorherrschende Architektur des Internet: Client-Server (C/S)
 - » Client und Server sind Prozesse
 - » Ein Prozess kann gleichzeitig Client und Server sein (je nach Rolle innerhalb einer Anforderung)
 - » Clients sind aktiv, Server sind passiv
 - » Clients werden nur für die Dauer der Applikation ausgeführt, deren Bestandteil sie sind
 - » Server werden kontinuierlich ausgeführt

Kommunikation: C/S-Architektur [I]



Kommunikation: C/S-Architektur [II]

Aufruf einer HTML-Ressource



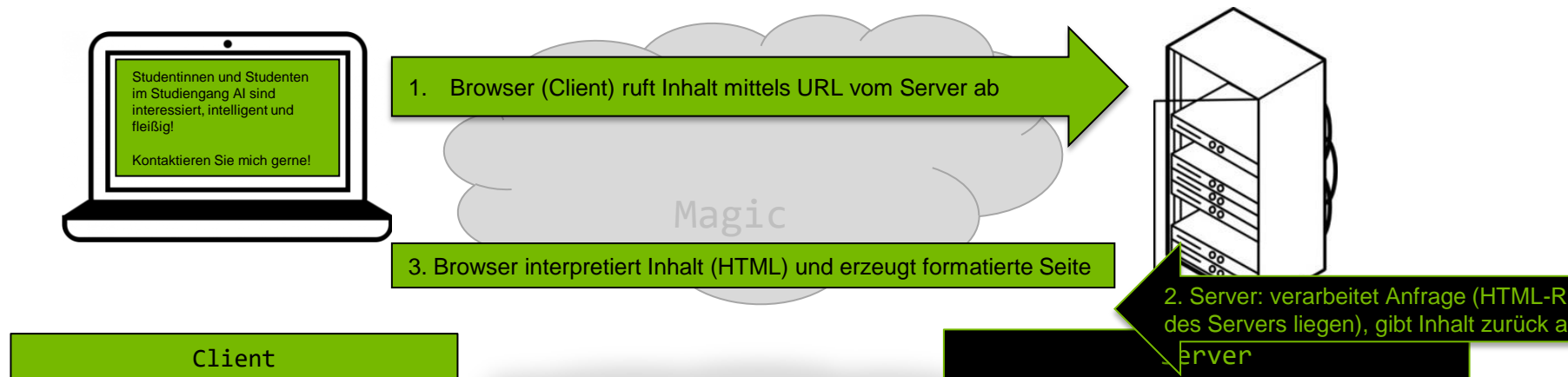
Kommunikation: C/S-Architektur [III]

Aufruf einer HTML-Ressource



Kommunikation: C/S-Architektur [IV]

Aufruf einer HTML-Ressource



Vertiefung / Ergänzung: Beispiele verteilter Systeme

Internet / World Wide Web [V]

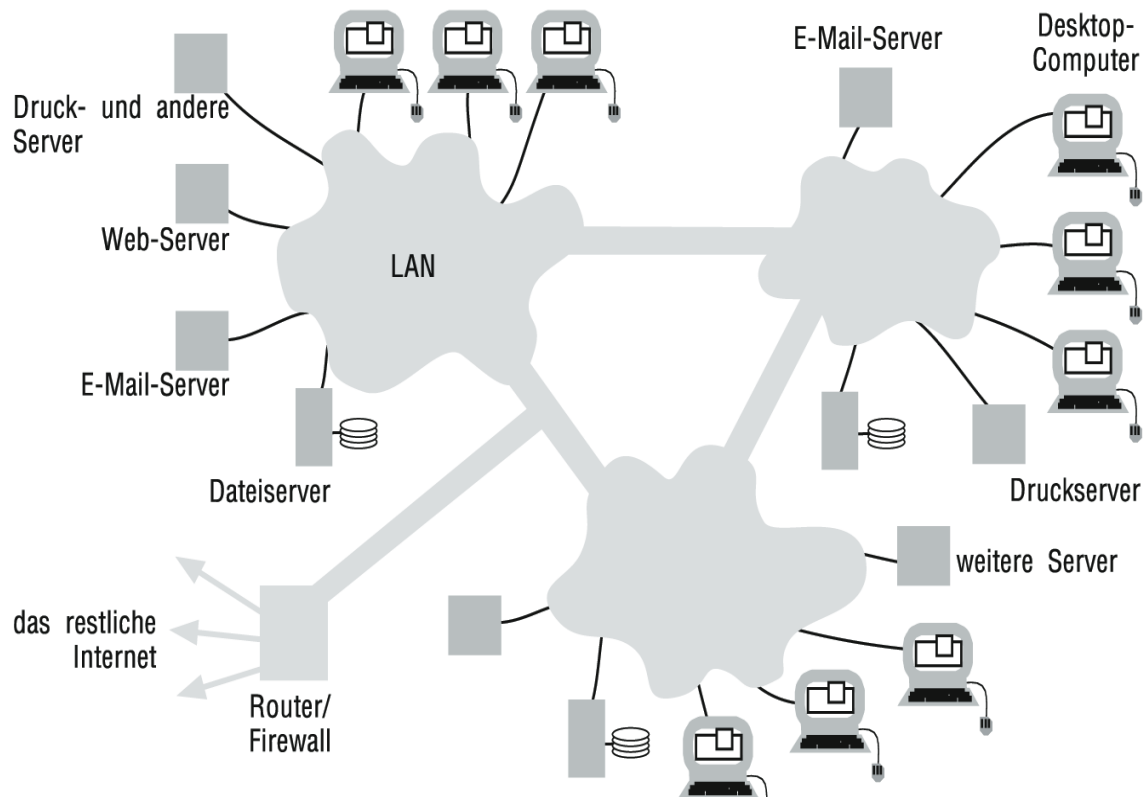
Gemeinsame Ressourcennutzung im WWW:

- Hypertext Transfer Protocol (HTTP)
 - » Schema: definiert den Kommunikationsablauf zwischen Client(s) und Server
 - » Hauptfunktion: Abruf von Ressourcen
 - » Merkmale
 - Prozess der „Anforderung/Antwort-Kommunikation“
 - Inhaltstypen werden per Zeichenketten sowohl vom Browser, als auch vom Server kommuniziert (Multipurpose Internet Mail Extensions / MIME-TYPES)
 - Eine Ressource pro Anforderung
 - Einfacher Zugriff
- Common Gateway Interface (CGI) – Programme:
 - » Auf Webserver ausgeführte Programme, welche Inhalte für Clients erzeugen (z.B. Abfrage eines Kontostands i.A. des Kunden)

Vertiefung / Ergänzung: Beispiele verteilter Systeme

Intranet [I]

- Teil des Internets
- Separate Verwaltung mit Abgrenzung zum Internet
 - » Router = Bindeglied zwischen Internet und Intranet
 - » Firewall = Schutzmechanismus für das Intranet



Quelle: Coulouris et al. [2002: 21]

Vertiefung / Ergänzung: Beispiele verteilter Systeme

Intranet [II]

- Konkretisierung der Dienste, die Benutzern innerhalb eines Intranets die gemeinsame Nutzung von Ressourcen ermöglichen sollen
- Konfiguration der potenziellen Firewall
- Kosten:
 - » Software-Installation
 - » Support

?

Aus welchen Gründen kann es sinnvoll sein, auf eine physische Verbindung zum Internet zu verzichten?
Nennen und beschreiben Sie ein Beispiel!

Vertiefung / Ergänzung: Beispiele verteilter Systeme

Pervasive Computing / Mobile Computing [I]

„Mobile Computing bezeichnet die Gesamtheit von Geräten, Systemen und Anwendungen, die einen mobilen Benutzer mit den auf seinen Standort und seine Situation bezogenen sinnvollen Informationen und Diensten versorgt.“

Bollmann & Zeppenfeld [2010:2]

?

Wie äußert sich die „Mobilität“ hinsichtlich

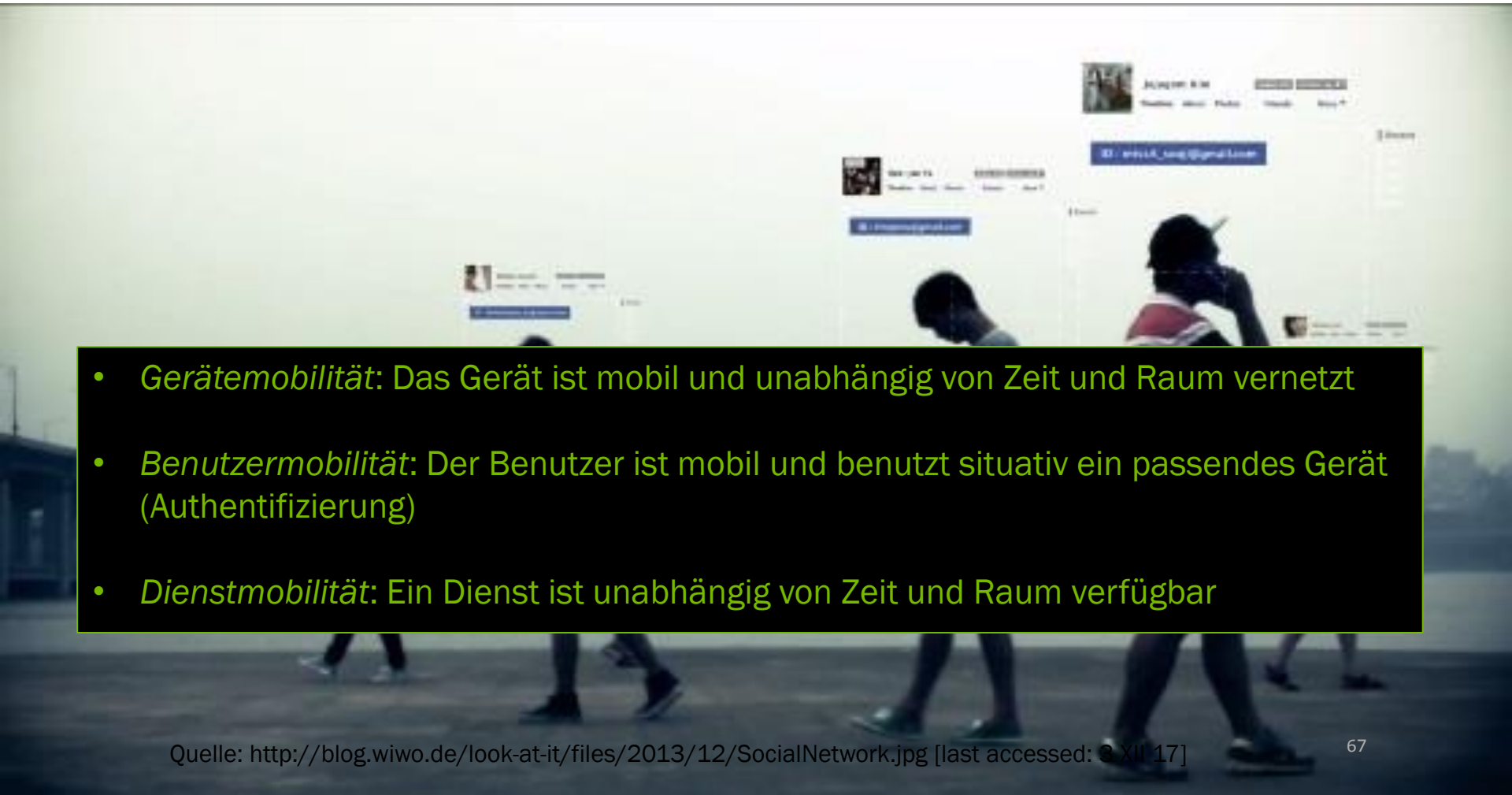
- » des Nutzers ?
- » des Geräts ?
- » der genutzten Ressourcen ?

Vertiefung / Ergänzung: Beispiele verteilter Systeme

Pervasive Computing / Mobile Computing [II]

„Mobile Computing bezeichnet die Gesamtheit von Geräten, Systemen und Anwendungen, die einen mobilen Benutzer mit den auf seinen Standort und seine Situation bezogenen sinnvollen Informationen und Diensten versorgt.“

Bollmann & Zeppenfeld [2010:2]

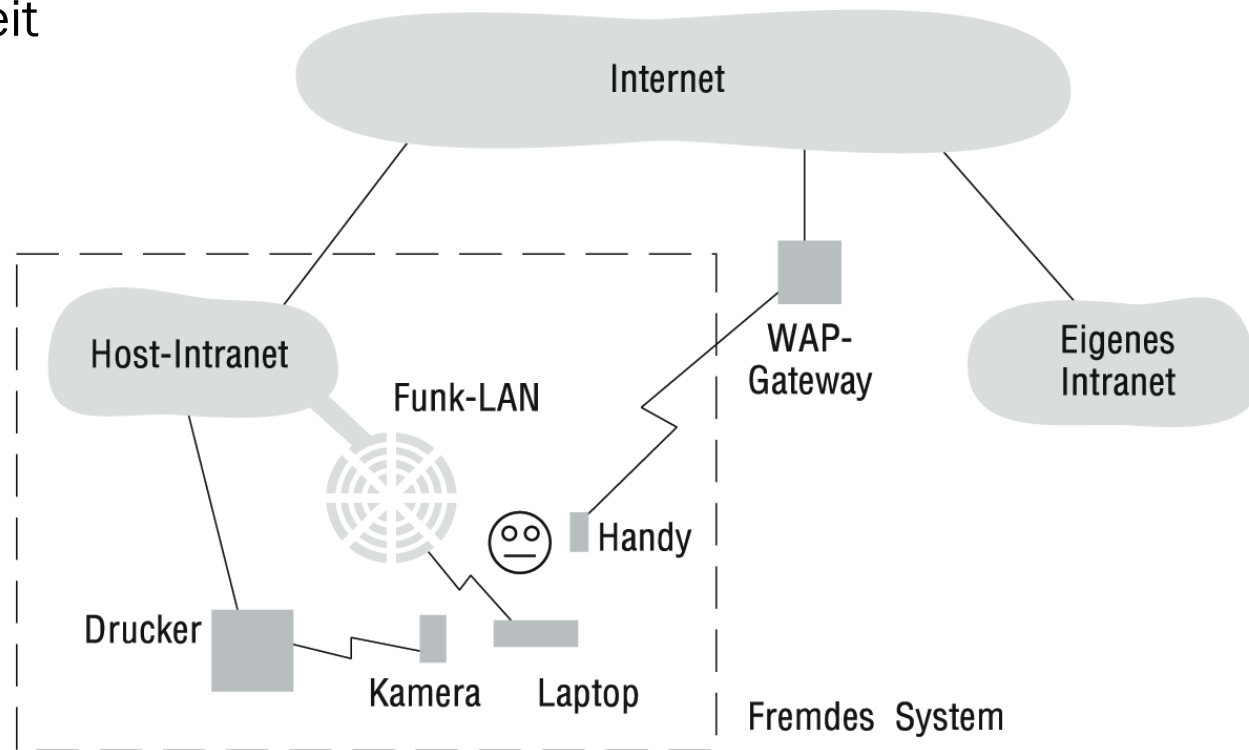
- 
- **Gerätemobilität:** Das Gerät ist mobil und unabhängig von Zeit und Raum vernetzt
 - **Benutzermobilität:** Der Benutzer ist mobil und benutzt situativ ein passendes Gerät (Authentifizierung)
 - **Dienstmobilität:** Ein Dienst ist unabhängig von Zeit und Raum verfügbar

Vertiefung / Ergänzung: Beispiele verteilter Systeme

Pervasive Computing / Mobile Computing [III]

Herausforderungen beim Mobile Computing (Entwurf):

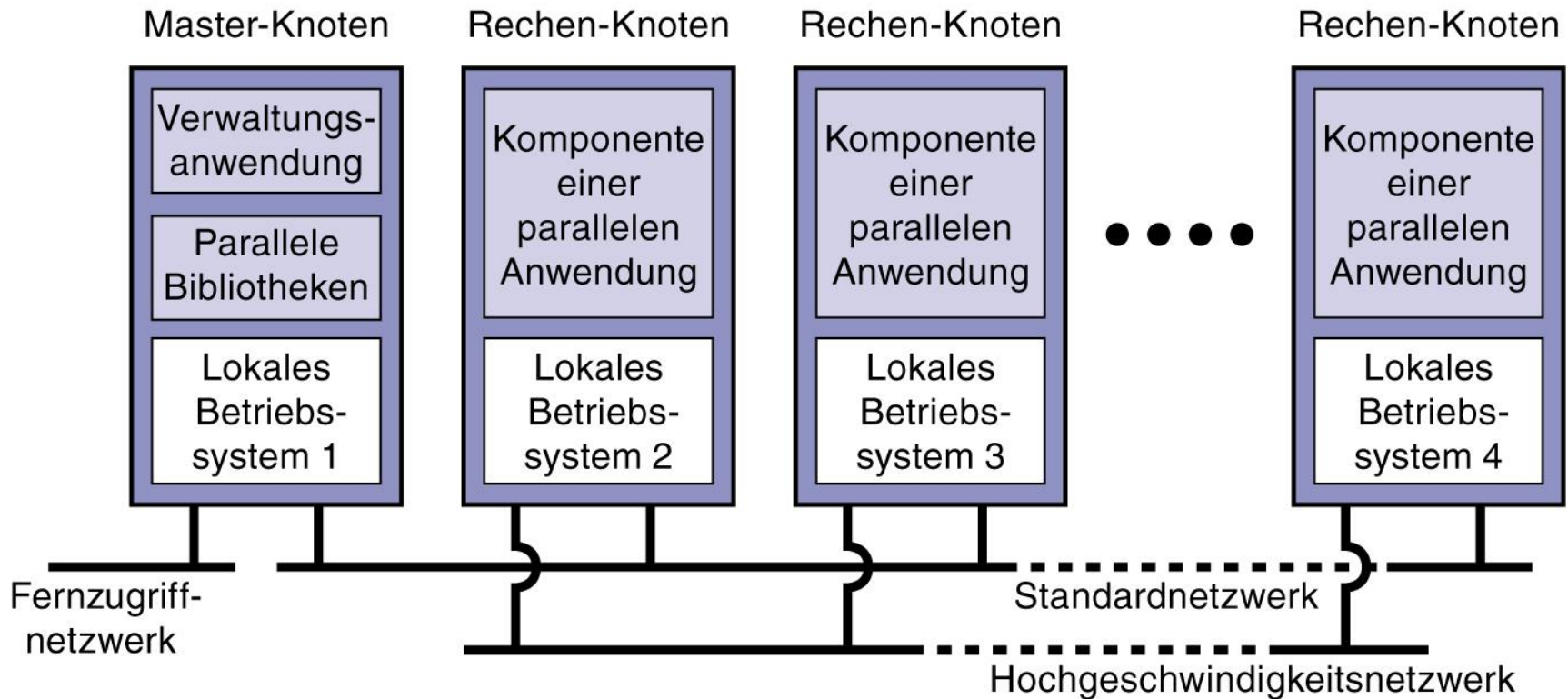
- ✓ Erkennen von Ressourcen
- ✓ Konfigurationsaufwand
- ✓ Sicherheit



Quelle: Coulouris et al. [2002: 23]

Vertiefung / Ergänzung: Beispiele verteilter Systeme

Cluster



Quelle: Tanenbaum & van Steen [2008:35]