Verteilte Systeme

1. Paradigmenwechsel für Softwarelösungen durch Verteilung

1.2 Allgemeine Anforderungen und Techniken verteilter Systeme

Sebastian Iwanowski
FH Wedel

Verteilte Systeme: Vorlesungsinhalte

1. Paradigmenwechsel für Softwarelösungen durch Verteilung
    - 1.1 Einführung durch Beispiele aus der Praxis
    - 1.2 Allgemeine Anforderungen und Techniken verteilter Systeme
    - 1.3. Pheromonbasierte Verkehrssteuerung als Beispiel für einen verteilten Ansatz
2. Die Client-Server-Beziehung und daraus resultierende Techniken
    - 2.1 Grundlagen der Client-Server-Beziehung
    - 2.2 Nebenläufigkeitstechniken in Java
    - 2.3 Entfernte Aufrufe
    - 2.4 Objektmigration
3. Dienstevermittlung
    - 3.1 Aufgaben
    - 3.2. Serviceorientierte Architektur (SOA)
    - 3.3. Web Services als Mittel zur Realisierung einer SOA

Verteilte Systeme: Literatur

Allgemeine Lehrbücher:

Ulrike Hammerschall:
Verteilte Systeme und Anwendungen, Architekturkonzepte, Standards und Middleware-Technologien,
Pearson Studium 2005, ISBN 3-8273-7096-5

George Coulouris / Jean Dollimore / Tim Kindberg:
Distributed Systems, Concepts and Design,
Addison-Wesley 2001, ISBN 0201-61918-0
Deutsche Übersetzung auch erhältlich:
Pearson Studium 2002, ISBN 3-8273-7022-1

Andrew Tanenbaum / Marten van Steen:
Verteilte Systeme, Grundlagen und Paradigmen,
Pearson Studium 2003, ISBN 3-8273-7057-4

Michael Wooldridge: *An Introduction to MultiAgent Systems*,
Wiley 2002, ISBN 0-471-49691-X (nur für Agententechnologie)

Verteilte Systeme: Literatur

Bücher für die Realisierung verteilter Systeme mit Java:

Marko Boger:
Java in verteilten Systemen, Nebenläufigkeit, Verteilung, Persistenz,
dpunkt-Verlag 1999, ISBN 3-932588-32-0

Thomas Stark: *J2EE, Einstieg für Anspruchsvolle*,
Pearson Studium 2005, ISBN 3-8273-2184-0

Manfred Hein / Henner Zeller:
Java Web Services, Entwicklung plattformübergreifender Dienste mit J2EE, XML und SOAP,
Addison-Wesley 2005, ISBN 3-8273-2231-6

David Chappell / Tyler Jewell: *Java Web Services*,
O'Really 2002, ISBN 0-596-00269-6

Verteilte Systeme: Literatur

Lehrbücher für SOA und Web Services:

Wolfgang Dostal / Mario Jeckle / Ingo Melzer / Barbara Zengler:
Service-orientierte Architekturen mit Web-Services, Konzepte - Standards - Praxis,
Spektrum 2005, ISBN 3-8274-1457-1

Gustavo Alonso / Fabio Casati / Harumi Kuno / Vijaj Machiraju:
Web Services, Concepts, Architectures, and Applications
Springer 2004, ISBN 3-540-44008-9

Thomas Erl: *Service-Oriented Architecture, A Field Guide to Integrating XML and Web Services,*
Prentice Hall 2004, ISBN 0-13-142898-5

Thomas Erl: *Service-Oriented Architecture, Concepts, Technology, and Design*,
Prentice Hall 2005, ISBN 0-13-185858-0

Definition: Verteiltes System

Andrew Tanenbaum / Maarten van Steen:

> *Ein verteiltes System ist eine Menge unabhängiger Computer, die dem Benutzer wie ein einzelnes System erscheint.*

Günther Bengel (FH Mannheim):

> *Ein verteiltes System ist ein System, in dem eine Reihe einzelner Funktionseinheiten, die miteinander über ein Transportsystem verbunden sind, in Zusammenarbeit Anwendungen bewältigen.*

Erweiterung (lw):

> *Die Funktionseinheiten verarbeiten softwaretechnisch Daten und die korrekte Funktionalität des Transportsystems ist nicht gewährleistet.*

Allgemeine Anforderungen an verteilte Systeme

Benutzerspezifische Anbindung an das System

Offenheit

Transparenz

Skalierbarkeit

} **wird im Folgenden näher erläutert**

Transparenzforderungen nach ISO (1995)

Transparenztyp	**Beschreibung**
Zugriff (access)	Verberge, wie auf einzelne Ressource zugegriffen werden muss
Ort (location)	Verberge, wo einzelne Ressourcen liegen (von wo gefragt wird)
Persistenz	Verberge, ob sich Ressource im Hauptspeicher oder auf der Festplatte befindet
Migration	Verberge, dass Ressourcen verschoben werden können
Relokation	Verberge, dass Ressourcen verschoben werden können, *während sie benutzt werden*
Replikation	Verberge, wievielfach eine Ressource vorhanden ist
Konkurrenz	Verberge, wie viele Nutzer gleichzeitig auf die Ressource zugreifen
Ausfall (failure)	Verberge, welche Ressourcen nicht zur Verfügung stehen

Weitere Transparenzforderungen

Transparenztyp	Beschreibung
Parallelität	Verberge, wie viele Prozesse gleichzeitig laufen
Leistung	Verberge, wie groß die Kapazitäten der einzelnen Rechner sind
Skalierung	Verberge, wie viele Teilnehmer und Funktionen das gesamte System verkraftet

Grundsatz:

Es ist nicht immer sinnvoll, dass alle Transparenzforderungen erfüllt sind.

Es sollte aber immer klar sein, welche Transparenzforderungen erfüllt sind und welche nicht.

Skalierung

Kriterien:

1. *Physische Kapazität des Gesamtsystems*
2. *Geographische Ausdehnung*
3. *Anzahl der unabhängigen Systemteile*

Skalierung

Probleme mit zentralen Konzepten:

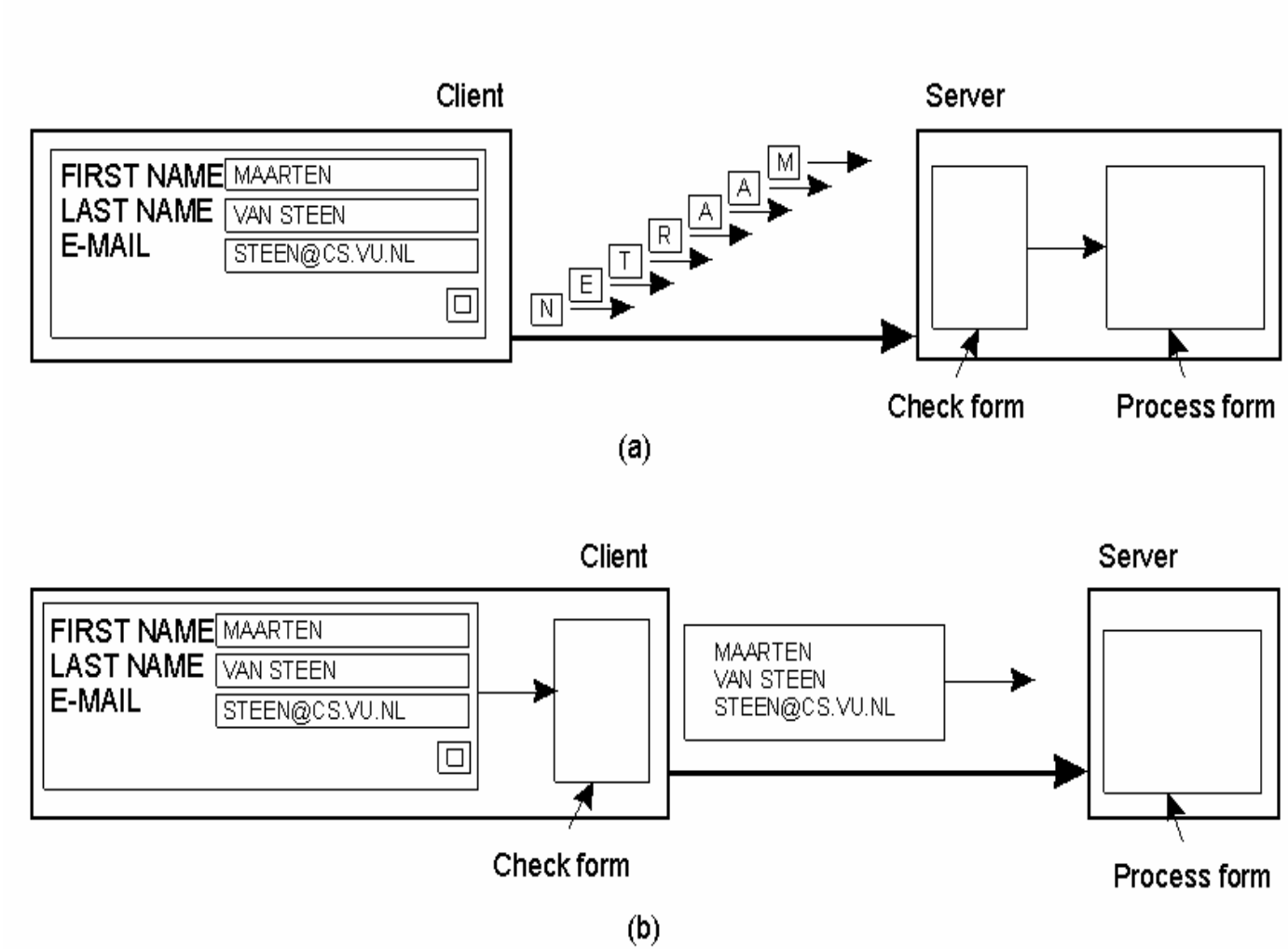
Zentrales Konzept	Beispiel
Zentraler Anbieter	Ein einzelner Server für alle
Zentrale Datenhaltung	Ein einzelnes on-line-Telephonbuch
Zentraler Algorithmus	Routingalgorithmus auf vollständigen Systemdaten

Skalierung

Lösungen mit verteilten Ansätzen:

Ansatz	**Beispiel**
Halte Software so lokal wie möglich	Formularausfüllung
Hierarchisch aufgebaute Namensverwaltung	Internet-DNS (Domain Naming System)
Verteilte Algorithmen	Pheromonbasierter Ansatz

Beispiel: Formularausfüllung



aus Tanenbaum / van Steen: S. 31

Beispiel: Mobile verteilte Datenhaltung