

Übungsblatt 1

zur Vorlesung

Verteilte Systeme/Ubiquitous Computing

Wintersemester 2007/2008

Achtung: Bitte beachten Sie auch die Web-Site der Vorlesung:
<http://www.mobile.ifi.lmu.de/Vorlesungen/ws0708/vs/>

Wiederholen: Kapitel 11 "Verteilte Systeme" aus dem Skript zu Informatik III

Aufgabe 1: (H) Grundlagen Verteilter Systeme

- Welche spezifischen Eigenschaften besitzen Verteilte Systeme?
- In welche Klassen lassen sich Verteilte Systeme einteilen?
- Welche Vor- und Nachteile haben Verteilte Systeme gegenüber zentralen Systemen?

Antwort:

a. **Eigenschaften:**

- Entferntheit:** Die Komponenten eines Verteilten Systems sind immer auch räumlich voneinander getrennt.
- Asynchronität:** Kommunikations- und Verarbeitungsprozesse werden nicht durch eine globale Systemuhr gesteuert. Änderungen und Prozesse werden demzufolge nicht notwendigerweise synchronisiert.
- Unabhängigkeit der Komponenten:** Die Komponenten sind unabhängig voneinander. Der Ausfall einer Komponente beeinträchtigt andere Elemente des Verteilten Systems nur insofern, als diese auf die ausgefallene Komponente zugreifen wollen. Es entsteht nur ein „Partieller Systemausfall“.
- Autonomie:** Es gibt keine zentrale Einheit, die sämtliche Management- und Steuerungsfunktionen übernimmt. Somit besitzen die einzelnen Komponenten ein gewisses Maß an Autonomie.
- Nebenläufigkeit:** Die Komponenten eines Verteilten Systems können parallel arbeiten.
- Föderative Namensverwaltung:** Ein verteiltes System kann durch den Zusammenschluss von bereits existierenden Systemen entstehen. Innerhalb dieses Zusammenschlusses muss eine eindeutige Interpretation der Namen über die Grenzen eines administrativen oder technologischen Bereichs hinweg gewährleistet sein.
- Mobilität:** Quellen von Informationen, Verarbeitungseinheiten und Nutzer können physikalisch mobil sein.

- (viii) **Migration:** Um die Leistungsfähigkeit eines Verteilten Systems zu erhöhen, können Programme und Daten zwischen verschiedenen Orten bewegt werden; dieses Konzept wird als Migration bezeichnet.
- (ix) **Dynamische Rekonfiguration:** Ein Verteiltes System muss in der Lage sein, dynamische Umstrukturierungen vorzunehmen, also bspw. zur Laufzeit neue Bindungen hinzuzufügen.
- (x) **Evolution:** Ein Verteiltes System unterliegt während seiner Lebenszeit zahlreichen Änderungen, ohne dabei i.d.R. völlig abgeschaltet zu werden.
- (xi) **Lokale Zustandsbetrachtung:** Da globale Systemzustände nur schwer zu realisieren bzw. zu ermitteln sind, muss eine Zustandsbetrachtung jeweils lokal erfolgen.
- (xii) **Heterogenität:** Verteilte Systeme bestehen meist aus Hardware-Komponenten unterschiedlicher Hersteller.

b. **Klassifikation von Systemen:**

Hardware	Software	
	lose gekoppelt	fest gekoppelt
fest gekoppelt		Multiprozessorbetriebssystem
lose gekoppelt	Netzbetriebssystem	Verteiltes Betriebssystem

c. **Vor-, Nachteile:**

Verteilte Systeme zeichnen sich durch folgende Vorteile aus:

- (i) **Skalierbarkeit:** Verteilte Systeme ermöglichen die stetige Anpassung der Größe eines Systems. „Größe“ meint hier die *Anzahl an Benutzer bzw. Zugriffen oder Menge an Daten*.
- (ii) **Erweiterbarkeit bzw. Integrierbarkeit:** Existierende Systeme können von neu hinzugekommenen Systemkomponenten genutzt werden, ohne dass ein System gleicher Funktionalität neu entwickelt werden muss. Zentral ist hier die *Ausweitung der Funktionalität*.
- (iii) **Fehlertoleranz bzw. Ausfallsicherheit:** Die einzelnen Bestandteile eines Verteilten Systems sind weitestgehend autonom. Im Falle eines Fehlers oder sogar Ausfalls einer Systemkomponente können die übrigen Einheiten im Idealfall unbeeinflusst weiterarbeiten und ggf. den Störfall überbrücken.
- (iv) **Kosteneffizienz:** Die Flexibilität und Anpassbarkeit von Verteilten Systemen führen gegenüber zentralen Systemen zu einem leichteren und damit günstigeren Management der IT-Infrastruktur.
- (v) **Teilweise Dezentralität des Managements:** Der Eigentümer einer Ressource hat die Möglichkeit, das Management dieser Komponente selbst zu übernehmen. Dies hat neben eventuell niedrigeren mittleren Managementkosten weitere Vorteile wie Schnelligkeit, mögliche optimale Anpassung an die Bedürfnisse des Einzelnen und automatische Skalierung der mit dem Management beauftragten Personenzahl an den Managementbedarf (hoher Servicegrad bei hoher Kosteneffizienz).

Verteilte Systeme besitzen jedoch auch Nachteile:

- (i) **Komplexere Software:** Die Realisierung eines Verteilten Systems erfordert komplexere Softwarelösungen als die eines zentralen Systems. Dies äußert sich nicht nur in der Komplexität der einzelnen Softwarekomponenten und ihres Zusammenwirkens, sondern auch in der hohen Komponentenzahl, die die betriebswirtschaftlichen Funktionen durch den Erwerb und die Verwaltung zahlreicher Lizenzen belastet.
- (ii) **Eventuell inkonsistente Datenbestände und Zustände:** In Verteilten Systemen kann es vorkommen, dass derselbe Sachverhalt mehrmals gespeichert vorliegt und sich die Datensätze widersprechen. Weiterhin können Zustände, wie z.B. die Zeit, inkonsistent sein.
- (iii) **Schwierige Fehlerdetektion:** Die Lokalisierung von Fehlern kann sich in Verteilten Systemen als schwieriges Unterfangen herausstellen. Hinzukommende Netzwerkkomponenten können völlig neuartige Fehler verursachen.
- (iv) **Datenschutzprobleme:** Datenbestände in Verteilten Systemen ermöglichen generell einfacher den Zugriff auf sensible Daten als dies bei separater Datenhaltung der Fall ist.

- (v) **Schwierigere Überwachbarkeit:** Die teilweise Dezentralität des Managements kann bei unzureichend geschulten Benutzern zu Managementfehlern führen. Die Qualität des IT-Managements ist somit in Verteilten Systemen schwieriger zu gewährleisten als in zentralen.

Aufgabe 2: (T) Mobile Endgeräte

- a. Aktuell kommen die folgenden Betriebssysteme bzw. Plattformen in mobilen Endgeräten zum Einsatz: Brew, Symbian OS, Windows Mobile, Palm OS, diverse Linux Derivate, J2ME und .NET
Nennen Sie deren Hauptcharakteristika und vergleichen Sie diese miteinander.
- b. Welche Trends bezüglich Hardwarekomponenten und Anwendungen lassen sich gegenwärtig erkennen?

Antwort:

- a. **Überblick Betriebssysteme/Plattformen:**
siehe Folien der Übung
- b. **Trends:**
- (i) **Integration von Hardwaremodulen in einer Komponente:** Mobile Endgeräte mit integrierter Digitalkamera, MP3-Player oder GPS-Empfänger
Beispiel: SXG75 Siemens-Benq
 - (ii) **Steigende Rechen- und Speicherkapazitäten:** Beispiel: N91 von Nokia
 - (iii) **Dienstorientierung:** Beispiel: Patent von Google auf sprachgesteuerte Suche für mobile Endgeräte
 - (iv) **Low-Budget-Endgeräte:** Beispiel: Mobiltelefone für Entwicklungsländer oder Prepaid-Wegwerfhandys

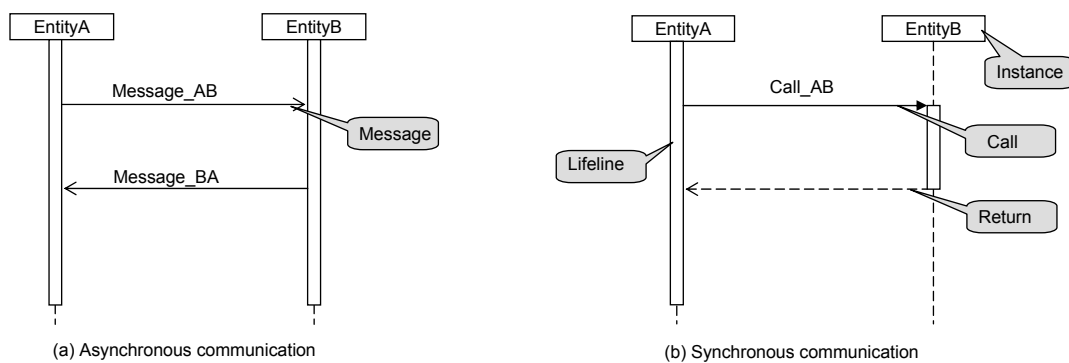
Aufgabe 3: (H) Client/Server-Modell

Das Client/Server-Modell (C/S-Modell) ist eines der Grundmodelle der Verteilten Systeme. Es basiert auf einem einfachen, verbindungslosen Anfrage-Antwort-Protokoll. In der Info III Vorlesung wurden bezüglich der Adressierung, Blockierung, Pufferung und Zuverlässigkeit unterschiedliche Entwurfsmöglichkeiten dargestellt.

- Erläutern Sie die Funktionsweise der verschiedenen Varianten der Adressfindung.
- Erklären Sie den Unterschied zwischen blockierenden (synchronen) und nichtblockierenden (asynchronen) Primitiven mit Hilfe von Sequenz-Diagrammen anhand des folgenden Szenarios:

Ein Benutzer erfragt bei einem Zeit-Server die aktuelle Zeit und wendet sich anschliessend an einen Wetterdienst-Server, um Informationen über das derzeitige Wetter zu erhalten. Wie sieht der jeweilige Ablauf aus, wenn synchrone oder asynchrone Kommunikations-Primitive für die Anfrage benutzt werden?

Bitte verwenden Sie die unten angegebene Notation für die Primitive.



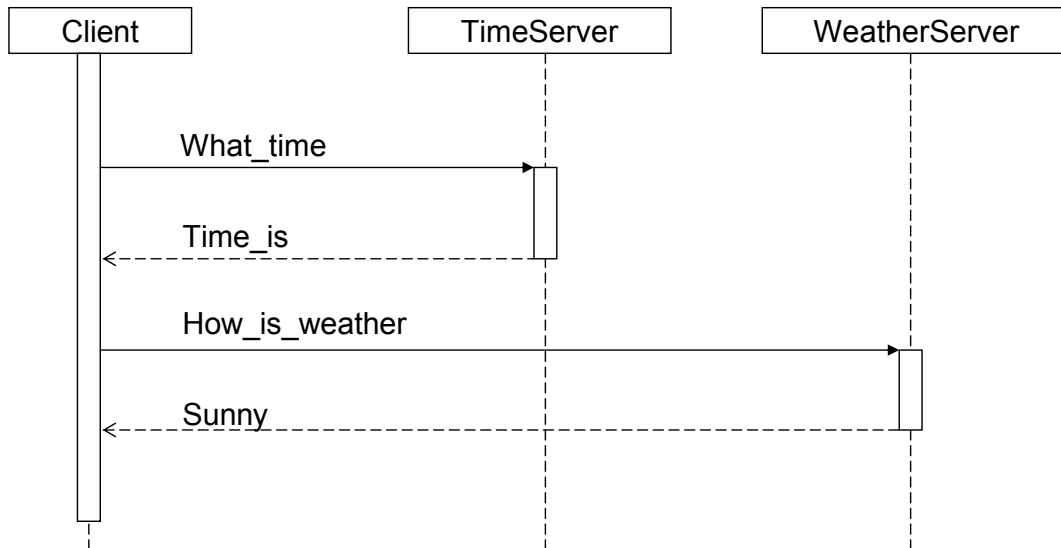
- Ein Entwickler hat die Möglichkeit, Kommunikationsprimitive wahlweise blockierend oder nichtblockierend und gepuffert bzw. nichtgepuffert zu gestalten. Daraus ergeben sich die folgenden vier Möglichkeiten:

	gepuffert	ungepuffert
blockierend	1	3
nichtblockierend	2	4

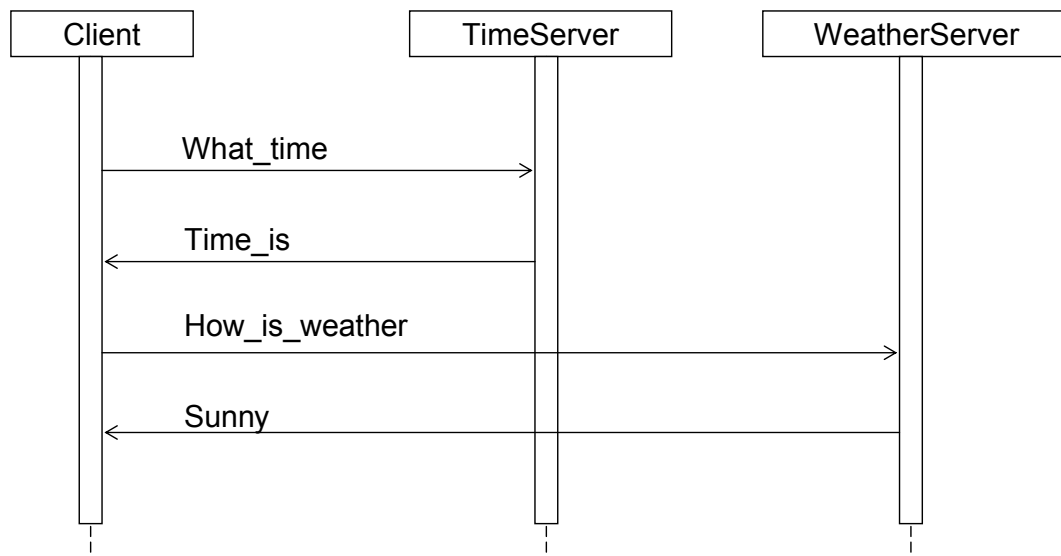
Welche Kombinationsmöglichkeiten würden Sie implementierungstechnisch nicht empfehlen? Bitte begründen Sie Ihre Antwort.

Antwort:

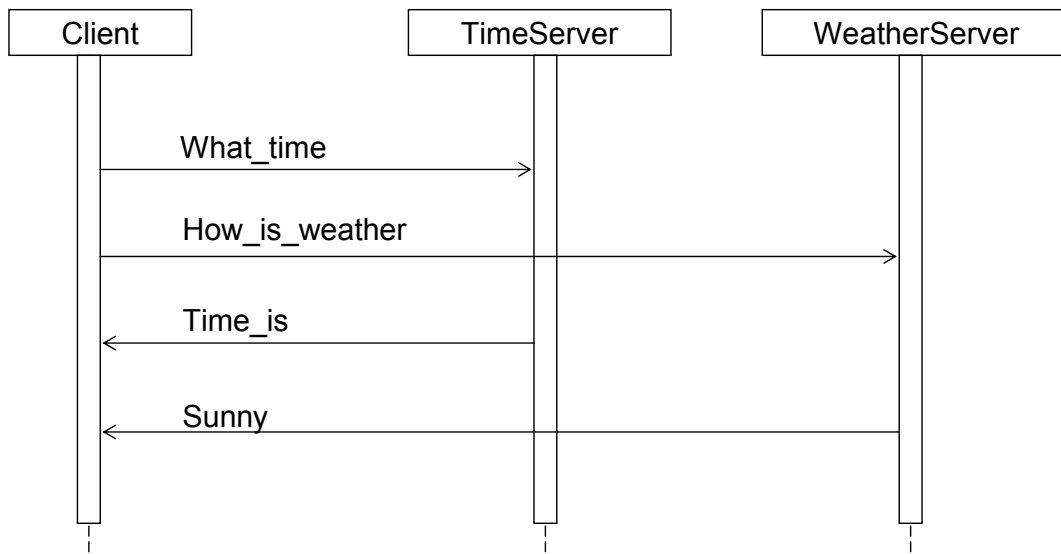
- Adressfindung:
 - machine.process-Adressierung: minimale Netzlast, aber KEINE Ortstransparenz
 - Lokalisierungspaket: orts-transparent, aber hohe Netzlast durch Broadcast
 - Name-Server (Trader): orts-transparent und akzeptable Netzlast
- Sequenz-Diagramm für synchrone Kommunikation:



Sequenz-Diagramm für asynchrone Kommunikation, Variante 1:



Sequenz-Diagramm für asynchrone Kommunikation, Variante 2:



- c. Design-Ziel muss sein, dass die Kommunikation zuverlässig ist (also keine Nachricht verloren geht) und möglichst wenig overhead (Speicherplatz, aber auch Zeitverlust) entsteht. Nicht zu empfehlen sind daher die Möglichkeiten 1 (overhead) und 4 (Nachrichten können verloren gehen).