Resümee (1)

- Organisatorisches zur Vorlesung

- Einordnung der Vorlesung
- Verteilte Systeme: Begriff, Sichtweisen, Eigenschaften...
- Motivation; Gründe für verteilte Systeme
  - Kooperation von geographisch verteilten Einheiten
  - qualitatives Wachstum des Internet
- Middleware für das Internet
- Transparenzeigenschaften (Verbergen von Verteiltheit etc.)
- Verteilte Systeme als “Verbund”
- Historische Entwicklung von Systemen und Konzepten
- Charakteristika und praktische Problembereiche verteilter Systeme

Resümee (2)

- Phänomene und konzeptionelle Probleme
  - Schnappschussproblem (inkonsistente globale Sicht)
  - Phantom-Deadlocks
  - Uhrensynchronisation
  - kausal inkonsistente Beobachtungen
  - Geheimnisaustausch über unsicheren Kanal

- Multiprozessoren (gemeinsamer Speicher)
  - Buskoppelung
  - Schaltnetzkoppelung (Crossbar, Permutationsnetze)
- Cluster-Computer (verteilter Speicher)
  - Bewertungskriterien für Verbindungstopologien
  - Hypercube (rekursives Konstruktionsprinzip)
  - Torus
  - Cube Connected Cycle
  - Zufallstopologien

Resümee (3)

- Nachrichtenkommunikation
  - Message-passing-Systeme / -Bibliotheken
  - Prioritäten von Nachrichten
  - Zuverlässigkeitsgrade
- Fehlermodelle
  - fehlerhaftes Senden / Empfangen
  - Verlust von Nachrichten
  - crash, fail-stop
  - allgemeine (“byzantinische”) Fehler
- Kommunikationsmuster
  - Mitteilung ↔ Auftrag
  - synchron ↔ asynchron
- Synchrone Kommunikation
  - Definition
  - Realisierung
  - virtuelle Gleichzeitigkeit; Gummibandtransformation
  - Blockaden und Deadlocks
- Asynchrone Kommunikation
  - Vor- / Nachteile gegenüber synchroner Kommunikation

Resümee (4a)

- Synchrone Kommunikation mit asynchroner simulieren
  - Warten auf ein explizites Acknowledgement
- Asynchrone Kommunikation mit synchroner simulieren
  - Puffer(prozess!) zur Entkoppelung dazwischenschalten
- Implementierung von Pufferprozessen
  - durch Inversion der Kommunikationsbeziehung
- Puffer beschränkter Kapazität
  - Implementierungsaspekte
- Alternatives Empfangen von Nachrichten
  - “select”-Anweisung: elegantes und mächtiges Konstrukt
  - aber: Semantik genau festlegen
- Verschiedene Kommunikationsmuster
  - no-wait-send; RPC; asyn. RPC; rendezvous
- Datagramm
- Rendezvous-Protokoll

Resümee (4b)

- RPC
  - Implementierung
  - Parameter-Marshalling
  - Stubs
  - Transparenzproblematik

- RPC-Fehlerproblematik
  - Fehlerursachen (verlorene Nachrichten, Crash von Server / Client)
  - Gegenmassnahmen
  - Probleme (Orphans,...)

- RPC-Fehlersemantik / -klassifikation
  - maybe, at-least-once, at-most-once, exactly once

Resümee (5)

- RPC-Fehlersemantik / -klassifikation
  - maybe, at-least-once, at-most-once, exactly once

- RPC
  - Binding
  - Protokolle
  - Effizienz
  - asynchroner RPC

- Socket-Programmierschnittstelle
  - Client-Server-Beispiel in C
  - als Übungsaufgabe: Sockets in Java

Resümee (6a)

- Java als “Internet-Programmiersprache”

- Adressierungsarten
  - 1:1, direct naming
  - m:n, mailbox
  - n:1, port
  - Kanäle

- Empfangen von Nachrichten
  - non-blocking (→ aktives Warten)
  - Zeitüberwachung
  - selektives Empfangen
  - implizites Empfangen

- Sprachaspekte beim verteilten Programmieren
  - kommunizierbare Datentypen?

- Gruppenkommunikation (Broadcast / Multicast)
  - Anwendungen
  - idealisierte Sicht
  - Fehlerproblematik
  - Zuverlässigkeitsgrad (“best effort”, k-zuverlässig)
  - “reliable Broadcast” mit ACK, NACK

Resümee (6b)

- Algorithmus für “reliable Broadcast”

- FIFO-Broadcasts
  - zwei nacheinander ausgeführte Broadcasts ein und desselben Senders erreichen alle Empfänger in dieser Reihenfolge
  - nicht stark genug, um akausale Beobachtungen zu verhindern

- Kausale Broadcasts
  - kausale Abhängigkeit zweier Nachrichten
  - “Causal Order”: Nachrichtenempfang “respektiert” kausale Abhängigkeit von Nachrichten (“kausaltreu”)

Resümee (7)

- Atomare Broadcasts
  - logisch gleichzeitiger Empfang der Einzelnachrichten eines Broadcasts
  - Realisierung über zentralen Sequencer bzw. Token auf einem logischen Ring
- Kausal atomare Broadcasts
  - virtuelle Synchronität
- Multicast
  - Zweck
  - Adressierung von Multicast-Gruppen
- Gruppenüberlappung
  - lokale / globale Gültigkeit von Reihefolgebedingungen etc.
- Multicast: Membership-Problem
  - atomare Änderung der Gruppenzugehörigkeit
  - Tolerieren von Prozessausfällen

Resümee (8)

- Push-Prinzip und Publish & Subscribe
- Ereigniskanäle als “Softwarebus”
- Tupelräume
  - Linda-Modell
  - JavaSpaces
- Logische Zeit
  - Raum-Zeitdiagramme, Ereignisse
  - Zeitstempel von Ereignissen
  - Uhrenbedingung (als Ordnungshomomorphismus)
- Logische Uhren von Lamport
  - Definition
  - Realisierung
  - injektive Abbildung, eindeutige Zeitpunkte
- Wechselseitiger Ausschluss (mit logischer Zeit)
  - replizierte Warteschlangen von Lamport (request, reply, ack)
  - Verfahren von Ricart / Agrawala 1981
  - Korrektheitsargumente? (Exklusivität, Deadlockfreiheit, Fairness,...)

Resümee (9)

- Namensverwaltung
  - Zweck von Namen
  - Namen und Adressen
  - Binden
  - Namenskontexte, hierarchische Namensräume
  - Aufgaben einer Namensverwaltung
  - Namensverwaltung in verteilten Systemen
- Zufall
  - Pseudozufall, “echter, physikalischer” Zufall
  - Symmetrisierungstrick von J. v. Neumann
- Nameserver
  - Replikation und Caching

Resümee (10)

- Internet Domain Name Service (DNS)
  - Namensauflösung im Internet
  - resource records
  - nslookup
- Client-Server-Modell (<==> Peer-to-Peer-Strukturen)
  - Prinzip
  - Client/Server-*Maschinen*
  - Client/Server-*Rollen*
- Zustandsändernde / -invariante Dienste und Server
  - idempotente und wiederholbare Aufträge
  - stateless / statefull
  - Beispiel Webserver (URL rewriting, cookies)

Resümee (11)

- Konkurrente Server
  - dynamische / statische Handler-Prozesse
- X-Window als “klassisches” Client-Server-System
  - aber: events zur asynchronen Rückmeldung Server → Client
- Servergruppen / verteilte Server
  - Strukturen kooperierender Server
  - Server-Auswahl bei einem Lastverbund
  - Replikation von Servern (“Überlebensverbund”)
- Middleware: historischer Kurzüberblick
- Sun-RPC
  - Identifikation entfernter Prozeduren (host, Programm-, Version-, Prozedur-Nummer)
  - Registrieren eines Dienstes auf Serverseite
  - Generieren von Prozedurstubs und Serverskelett aus Schnittstellenspezif.
- Portmapper
  - Zuordnung Port / Programmnummer eines Dienstes
- Schutzaspekte bei Sun-RPC
  - “UNIX flavor”: Automatisches Mitsenden von Benutzerkennung etc.
  - “Secure RPC”: Authentifizierung mit DES
- DCE: Hauptkomponenten; Threads und deren Problematik

Resümee (12)

- CORBA
  - CORBA-Architektur
  - Object Services und Common Facilities
  - neuere Erweiterungen bei CORBA

Resümee (13)

- Jini
  - Motivation: Dienstparadigma, Netzzentrierung,...
  - Java-Bezug
  - Lookup-Service
  - Discovery
  - Join
  - Proxies und smart Proxies
  - Code-Mobilität
  - Leases
  - verteilte Ereignisse
  - Vorteile und Probleme von Jini

- Sicherheit in verteilten Systemen: Anforderungen
- Einmalpasswörter mit Einwegfunktionen
- One-time-Pads mit XOR
- Symmetrische und asymmetrische Kryptosysteme
- Authentifizierung mit asymmetrischen Schlüsseln

Resümee (14)

- Problem der “Replays” und Lösungsansätze
- Schlüsselvergabe durch Key-Server
- Autonome, “geheime” Schlüsselgenerierung
  - Schlüsselaustausch mit Diffie-Hellman-Prinzip
- “Man in the middle”: Erkennungsmöglichkeit
- Authentifizierung mit geheimen Zertifikaten
- Zero-Knowledge-Proofs
  - Beispiel: Isomorphie von Graphen

- Kerberos
  - Protokoll für Ticket-Granting-Ticket- und Service-Ticket-Erwerb
  - Anwendungsbeispiel: Einrichtung sicherer Kanäle
  - Sicherheitsaspekte