Disconnected Operations in mobilen Umgebungen

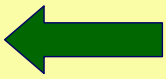
Fachseminar „Verteilte Systeme“

7. Mai 2002 – Oliver Wünsch

Betreuer: Jürgen Bohn
Professor Dr. F. Mattern

Agenda

1. Einführung
   - Was ist „Disconnected Operations?“
2. Lösungsansätze
   1. Mobile Computing
      1. Coda-Dateisystem
      2. Proxy-Mechanismen
   2. Ubiquitous Computing
      1. Message Bus
      2. Tuple Spaces
      3. Event-basierte Infrastruktur
3. Fazit



Disconnected Operations

- Computing heute:
  - Verbund mehrerer Rechner zur Lösung der gegebenen Aufgaben
    - Internet (WWW, Mail, etc.)
    - Daten(bank)zugriffe
    - Zugriff auf Sensoren und Aktuatoren
  - Voraussetzung: Vernetzung, Kommunikation zwischen mehreren Geräten

Disconnected Operations (II)

- Was, wenn Verbindung nicht zur Verfügung steht?
  - Netzausfall, Überlastung
  - Mobiler Rechner unterwegs
  - Mobiles Device mit andauernden Verbindungen und Trennungen.

Disconnected Operations (III)

- „Disconnected Fall" vorsehen in
  - Technischem Design von
    - Betriebssystem
    - Middleware
    - Anwendungen
  - Aber auch: fachlichem Design der (Geschäfts-)Prozesse

Lösungmöglichkeiten

- Abhängig von der Umgebung
  - Anwendungsgebiet
  - Evtl. „Altlasten"
    - Middleware, Anwendungen, etc.
  - Infrastruktur
- Differenzierung nach Rahmenbedingungen zwischen
    - (Distributed Computing)
    - Mobile Computing
    - Ubiquitous Computing

Agenda

1. Einführung
   Was ist „Disconnected Operations?"
2. Lösungsansätze
   1. Mobile Computing
      1. Coda-Dateisystem
      2. Proxy-Mechanismen
   2. Ubiquitous Computing
      1. Message Bus
      2. Tuple Spaces
      3. Event-basierte Infrastruktur
3. Fazit

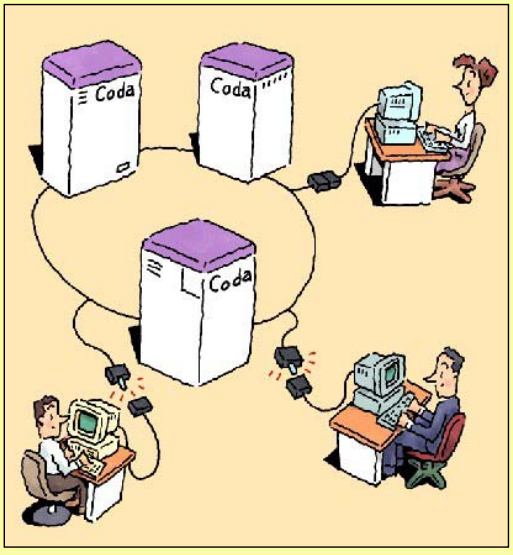
Mobile Computing

- Normalerweise verbunden
- Transparenz nicht notwendig aber wünschenswert
  - Benutzer oder Infrastruktur initiiert Trennungsprozess oder Trennungen vorhersehbar
- Clients sind vollständige PCs (z.B. Laptops)
  - Speicher (RAM, HD), Prozessorleistung
- Asymmetrische Client/Server-Struktur
- „herkömmliche Infrastruktur"
- Protokolltransparenz (TCP/IP, Mobile IP, etc.)

Mobile Computing

- Funktionierende Lösungen/Produkte bereits vorhanden
  - Fortgeschrittener Entwicklungsstand im Bereich Mobile Computing
- Stellvertretend für alle Lösungen hier:
  - Coda-Dateisystem
  - Proxy-Mechansimus

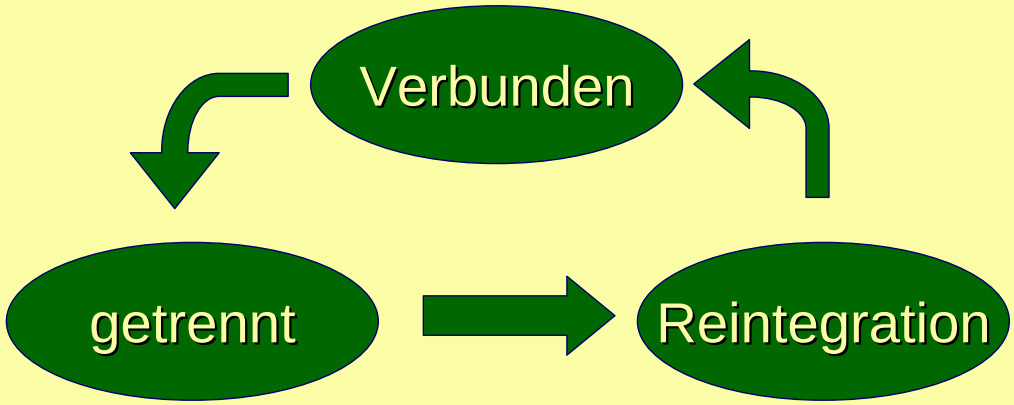
Coda-Dateisystem



- Entwickelt seit 1987
- Hervorgegangen aus dem Andrew File System (AFS)
- Auch aus heutiger Sicht (fast) vollständige Lösung der Probleme bei Disconnected Operations.

Coda - Funktionsweise

Das Verhalten des Dateisystems ist abhängig vom aktuellen Status.



Coda - Funktionsweise

- Wenn verbunden:
  - fortlaufende lokale Speicherung und Aktualisierung aller „relevanten" Objekte auf dem Client (proaktives Caching = „Hoarding")
  - Problem: Welche Objekte sind „relevant"?
  - => Vorherige Definition durch Benutzer und „Last Recently Used-Strategie".
- Wenn getrennt:
  - Arbeiten mit den Daten im Cache
  - Nicht gecachte Objekte stehen nicht zur Verfügung.
- Bei Wiederverbindung:
  - Reintegration

Reintegration

Problem:

Mehrere Benutzer bearbeiten und speichern unabhängig voneinander die gleiche Datei. Getrennte Benutzer erfahren dies nicht.

- Reintegration => Konflikt
- Konfliktlösung
  - Wenn möglich automatisch,
  - sonst Benutzereingriff erforderlich!
- Alternativ: Konfliktverhinderung durch Sperren

Sperren (I)

- Sperre: Recht auf exklusiven Zugrff

Zwei Ansätze:

- Pessimistisch
  - Explizite Rechtevergabe VOR Trennung
    - Shared (andere Clients können nur lesen)
    - Exclusive (andere Clients haben keinen Zugriff)
  - Gangbar bei vom Benutzer gesteuerter Trennung
  - Schwierig
    - ohne Benutzerinteraktion,
    - bei Ad-hoc Trennung,
    - länger andauerden Trennungen (evtl. Leases zum Ablauf einer Sperre).

...

Sperren (II)

...

- Optimistisch
  - Lese- und Schreibzugriffe überall zugelassen.
  - Konfliktlösung bei Wiederverbindung notwendig:
    - Konflikterkennung,
    - vorzugsweise automatische(!) Konfliktlösung.
- Trotzdem: Optimistische Sperren vorziehen!
  - Weniger hinderlich
  - Coda verwendet optimistische Sperren.

Weak connections

- Coda bietet speziellen Modus für schmalbandige Verbindungen:
  - Zwischengespeicherte Daten nutzen, Transfervolumen zu reduzieren (Zeit, Kosten)
    - Proaktives Caching ausgesetzt (Bandbreite schonen)
    - Nicht gecachte Daten werden vom Server geholt.
  - Nur Schreibzugriffe sofort an Server weiterreichen („write-through“)

Probleme

- Caching aller evtl. benötigten Daten
  - Speicherplatzbeschränkungen!
  - Größe des benötigten Datensatzes muss keiner als der Cache sein
- Evtl. hohe Bandbreite notwendig
  - Proaktives Caching
- Konfliktlösungsmechansimen
  - Funktion abhängig vom Dateiformat
  - Coda-Prämisse: Viele kleine Dateien, vernachlässigbar wenige Konflikte.

Proxy-Mechanismus

- Clients nicht direkt mit Server verbunden,
- Proxy vermittelt, versteckt Trennungen vor Server
  - Entlastung des Servers von
    - Andauerndem Verbindungsaufbau und –abbau
    - Disconnection-Logik
- Proxy kennt dabei sowohl Status des Servers als auch des Clients

Proxy-Mechanismus (II)

- Mit Verbindung:
  - Client kommuniziert über Proxy mit Server
  - Proxy lauscht, speichert Status des Client-Speichers
- Ohne Verbindung:
  - Client speichert Nachrichten für Server
  - Proxy speichert Nachrichten für Client
  - Proxy hält Server-Verbindung aufrecht
- Bei Wiederverbindung:
  - Proxy verbindet neue Client-Session mit bestehender Server-Session
  - Proxy und Client leiten zwischengespeicherte Nachrichten weiter
  - Client-Datenspeicher wird dabei aktualisiert

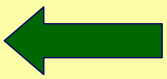
Probleme

- Fall: Client verbindet sich nicht wieder.
  - Einführung von Leases, nach Ablauf Session löschen.
  - Transaktionskonsistenz?
- Nicht jeder Server sendet „Push"-Nachrichten
  - Wie „merkt" der Proxy, dass sich die Daten auf dem Server geändert haben?
  - Polling? Ressourcenintensiv!!
- Client muß Proxies transparent wechseln können
  - Bedingt durch Mobilität oder Infrastruktur
  - Proxies tauschen Datensätze untereinander aus.

Bewertung

- Bisherige Lösungen ausgelegt für Mobile Computing
  - Client/Server orientiert
    - Server und die angebotenen Dienste a priori dem Client bekannt
    - API des Servers bekannt
    - Infrastruktur kann vorausgesetzt werden
  - Basierend auf RPC-artiger Semantik

Agenda

1. Einführung
   Was ist „Disconnected Operations?"
2. Lösungsansätze
   1. Mobile Computing
      1. Coda-Dateisystem
      2. Proxy-Mechanismen
   2. Ubiquitous Computing 
      1. Message Bus
      2. Tuple Spaces
      3. Event-basierte Infrastruktur
3. Fazit

Ubiquitous Computing

- Trennungen sind üblich oder Regel
- Transparenz des Verbindungs- / Trennungsprozesses notwendig
  - Trennungen und Verbindungen unvermittelt
- Clients mit begrenzten Ressourcen
- Beschränktes oder gar kein User Interface
- Ad-hoc Infrastrukur / Ubiquitäre Infrastruktur
  - Bandbreite begrenzt und variierend, evtl. asymmetrisch
  - Alle Kommunikationspartner können das Netzwerk jederzeit betreten oder verlassen.

Ubiquitous Computing

- Netzwerk-Prämissen nicht unbedingt transparent für die Anwendungen auf dem ubiquitären Device
- Peer-To-Peer Kommunikation
  - Keine festgelegte Aufteilung in Clients und Servers
- Spontane Infrastruktur
- Mobilitätstransparenz
- Kommunikationspartner kennen a priori weder Existenz noch angebotene Dienste der Gegenstelle.
- Oft mehr als 2 Kommunikationspartner pro Vorgang
  - Z.B. mehrere Interessenten an den Werten eines Sensors

Remote Procedure Call (RPC)

- Funktionsaufrufe auf fernem System
  - synchron,
  - blockierend,
  - streng typisiert.
  - Kommunikationspartner sind bekannt
  - Genau 2 Partner (Client ruft Funktion auf Server auf.)
  - Vorkehrung zur Umgehung dieser Beschränkung nicht Bestandteil von RPC, sondern aufgesetzt
- Versteckt lediglich eigentliche Netzwerkkommunikation vor der Applikation.
- Geeignet für Ubiquitous Computing?

Ubicomp (II)

- Interessierter Empfänger bei Design oder Auslösung eines Events vielleicht nicht erreichbar oder unbekannt?
- Mehrere Sensoren und Aktuatoren
- Beziehungen (nicht nur Netzkommunikation) ad hoc.
- RPC-Semantik alleine bietet hier keine Lösung

  => ungeeignet
- => Message Passing

Message Passing

- „Publisher" verbreiten Messages mit Kategorisierung (Topics) ohne festes Ziel.
- „Subscriber" empfangen Messages der Topics, die sie interessieren.
- Beziehung zwischen Sender und Empfänger dynamisch festgelegt.
- Strukturmöglichkeiten:
  - Broadcasting der Events über Message Bus
  - Publish-Subscribe-Mechanismus für Events
  - Tuple Spaces

Disconnected Message Passing

- Problem: Nachrichten erreichen nur verbundene Teilnehmer
- => Speicherung der Nachricht durch einen Server oder den Sender selbst
  - Wie lange?
    - Verfalldatum
  - Übermittlung
    - Periodisches Wiederholen einer gültigen Nachricht durch den Sender
    - Aktiver Abruf aller Nachrichten eines Themas durch die Interessenten („Wer hat etwas mit dem Topic x für mich?")

Message Bus

- Existierende Lösungen:
  - CORBA Event Service
  - Java Message Queue (J2EE)
- Schwächen
  - Wieder strenge Typisierung
  - Kein QoS-Support (z.B. Multimedia-Anwendungen)
  - Kein Mobility-Support
  - Kein eingebauter Support für disconnected, d.h. verzögerte Kommunikation ohne Messaging Server.
  - Bedingte Subscriptions fehlen („Alarm, wenn die Tür nach 22 Uhr geöffnet wird!“)

Tuple Spaces

- Tuple Spaces sind Bereiche, in denen Nachrichten abgelegt werden.
  - Shared Memory auf einem Device
  - Device stellt Zugriff auf den Speicherbereich als Service zur Verfügung.
- Teilnehmer können
  - Nachrichten kategorisiert ablegen
  - Nachrichten nach Kategorie gefiltert lesen
  - sich für Benachrichtigungen bei neuen Nachrichten bestimmter Kategorien registrieren

Event-basierte Infrastruktur

- Anwendungsgebiet: Virtual Counterparts
  - Für gewisse „reale“ Objekte existieren virtuelle Gegenstücke im Speicher eines Rechners.
  - Austausch von Zustandsänderungen per Events
  - Events werden per Publish-Subscribe verteilt
- Zustandsänderung ohne Verbindung zum virtuellen Gegenstück?
- Spätere Auslieferung der Events bei Verbindung
  - Persistenz
  - Event-Infrastruktur: Event-Services, Manager

Bewertung – Ubiquitous Computing

- RPC-Semantik nur bedingt geeignet.
- Message/Event Passing-Ansätze erfüllen die Anforderungen meist besser.
  - Voraussetzung für Disconnected Operations: Persistenz
  - Message-Broadcasting nicht geeignet (Skalierung)
  - => Messaging Server, Event Services oder Tuple Spaces für Persistenz und Koordination notwendig
- Aber Architekturen wie CORBA für Ubicomp-Devices zu ressourcenhungrig:
  - => Kombination mit den Message-Servern über Proxies

Fazit

- Keine für alle Anwendungsbereiche optimale Lösung zur Garantie von Disconnected Operations.
  - Unterschiedliche Voraussetzungen
  - Unterschiedliche Probleme

  => Unterschiedliche Lösungen
- Unter Berücksichtigung der Voraussetzungen sind angemessene Lösungen vorhanden.
- Jedoch: Kein Ansatz kann eine Netzverbindung vollständig ersetzen.

Literatur (Auswahl)

- J. Kistler, M. Satyanarayanan: Disconnected Operation in the Coda File System. ACM Transactions on Computer Systems, February 1992
- M. Satyanarayanan: Pervasive Computing: Vision and Challenges. IEEE Personal Communications, August 2001
- M. Esler, J. Hightower, T. Anderson, G. Borriello: Next Century Challenges: Data-Centric Networking for Invisible Computing. MOBICOM, Washington, Seattle, August 1999
- M. Valente, R. Bighonha, M. Bigonha, A. Loureiro: Disconnected Operation in a Mobile Computation System. Workshop on Software Engineering and Mobility, Toronto, Canada, May 2001
- M. Fiuczynski, D. Grove: A Programming Methodology for Disconnected Operation. Technical Report, March 1994
- A. Snoeren, H. Balakrishnan, M. Kaashoek: Reconsidering Internet Mobility. 8th Workshop on Hot Topics in Operating Systems, HotOS-VIII, May 2001
- D. Conan, S. Chabridon, G. Bernard: Disconnected Operations in Mobile Environments. To appear at the 2nd International Workshop on Parallel and Distributed Computing Issues in Wireless Networks and Mobile Computing, Fort Lauderdale, Florida, April 2002
- M. Langheinrich, F. Mattern, K. Römer, H. Vogt: First Steps Towards an Event-Based Infrastructure for Smart Things, Distributed System Group, Department of Computer Science, ETH Zurich, October 2000
- Umar Saif, David J. Greaves: Communication Primitives for Ubiquitous Systems or RPC Considered Harmful, Proceedings of 21st ICDCS International Workshop on Smart Appliancews and Wearable Computing, 2001