hochschule für angewandte wissenschaften
FACHBEREICH ELEKTROTECHNIK UND INFORMATIK hamburg
university of applied sciences

Mobile Datenbanksysteme

Mark Thomé
INF – M2 – AW1

01.06.2005

Agenda

- Motivation
- Grundlagen
- Replikation und Synchronisation
- Mobile Transaktionen
- Anfrageverarbeitung
- Mobile Datenbanken im Ferienclub
- Fazit

Agenda

- **Motivation**
- Grundlagen
- Replikation und Synchronisation
- Mobile Transaktionen
- Anfrageverarbeitung
- Mobile Datenbanken im Ferienclub
- Fazit

Motivation

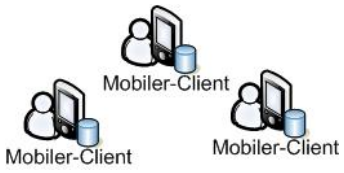
- Komplexität der Anforderungen steigt
- Mobile Clients für Integration von Geschäftsprozessen
- Online/Offline-Szenario
- Speicher auf mobilen Geräten
  - limitierte Ressourcen
  - Replikation und Synchronisation

Agenda

- Motivation
- **Grundlagen**
- Replikation und Synchronisation
- Mobile Transaktionen
- Anfrageverarbeitung
- Mobile Datenbanken im Ferienclub
- Fazit

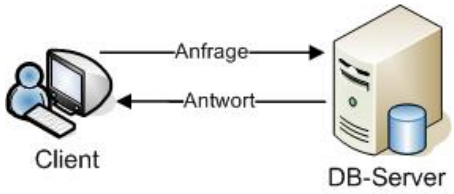
Definitionen

- Mobiles Datenbanksystem[1]
  - Small-Footprint-Datenbanksystem
  - Installiert auf mobilem Client
  - Speichert Daten dauerhaft und in strukturierter Weise
  - Stellt Datenbankfunktionalität bereit
  - Speziell angepasst an die Anforderungen und Bedürfnisse von beschränkten Ressourcen

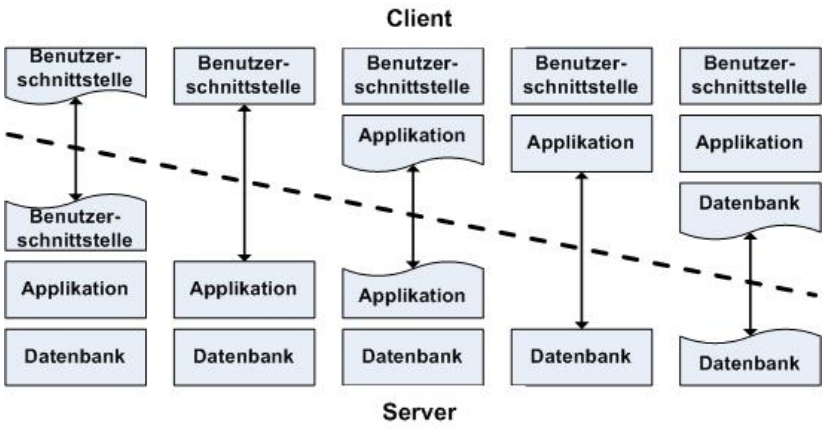


Klassische Client/Server-Architektur

- Client: Initiierender Prozess
  - Stellt Anfragen
  - Erteilt Aufträge
- Server: Reagierender Prozess
  - Bearbeitet Anfragen
  - Erhält Aufträge
- Anfrage-Antwort-Protokoll:



Client/Server-Klassifikation

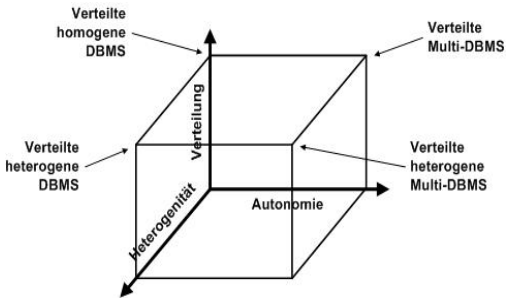


Verteilte Datenbanksysteme

- Idee: Transparente Verteilung von Daten und Datenbankfunktionalität auf verschiedene Knoten
- Jeder Knoten hat eigenes DBS
- Jeder Knoten ist Einstiegspunkt
- Alle Knoten zusammen bilden das verteilte System

Verteilte Datenbanksysteme: Klassifikation

- Klassifizierung durch drei Dimensionen:
  - Heterogenität
  - Verteilung
  - Autonomie

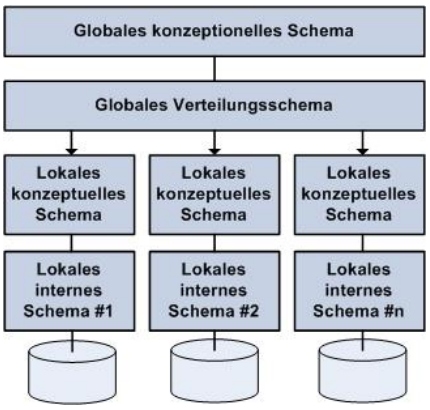


Verteilte Datenbanksysteme: Anforderungen

- Lokale Autonomie
- Unabhängigkeit
- Hohe Verfügbarkeit
- Orttransparenz
- Fragmentierungstransparenz
- Replikationstransparenz
- Verteilte Anfrageverarbeitung
- Verteilte Transaktionsverwaltung
- Hardware-Unabhängigkeit
- Netzwerkunabhängigkeit
- Datenbanksystemunabhängigkeit

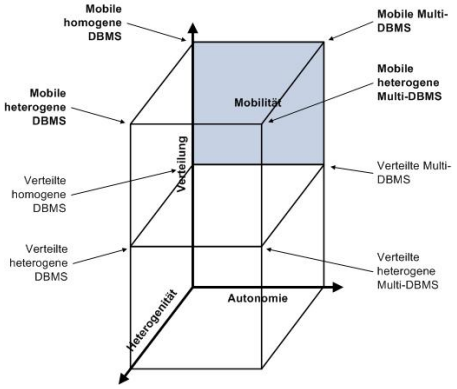
Verteilte Datenbanksysteme: Schemaarchitektur

- Globales konzeptionelles Schema:
  - Beschreibung des logischen Aufbaus der Gesamt-Datenbank
  - Sichert die Verteilungstransparenz
- Globales Verteilungsschema:
  - Festlegung der Datenverteilung
  - Auf logischer Ebene eine prädikative Beschreibung der Verteilung (Partitionierung)
  - Auf physischer Ebene die genaue Festlegung des Speicherortes (Allokation)

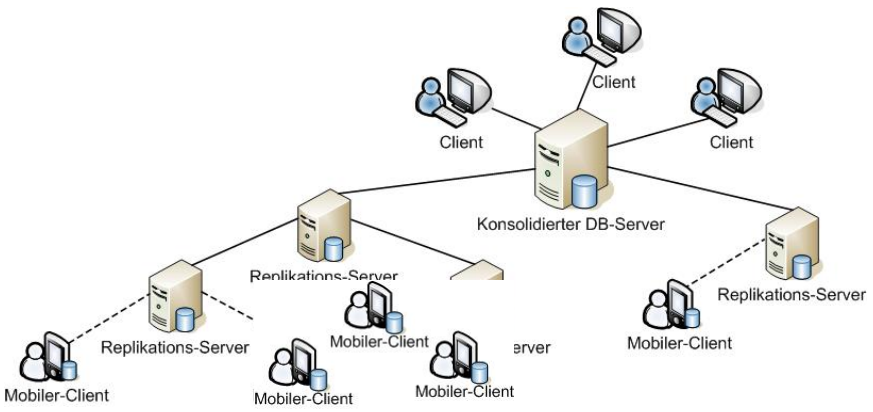


Mobile verteilte Datenbanksysteme

- Mobilität als Erweiterung der Verteilungs-Klassifikation
- Mobile Knoten
- Netzpartitionierung durch häufige Trennung mobiler Knoten von Gesamtsystem



Erweiterte Client/Server-Architektur



Agenda

- Motivation
- Grundlagen
- **Replikation und Synchronisation**
- Mobile Transaktionen
- Anfrageverarbeitung
- Mobile Datenbanken im Ferienclub
- Fazit

Replikation und Synchronisation

- Replikation: Einführung von Datenredundanz
- Synchronisation: Abgleich von redundanten Daten, die geändert wurden
- Replikation und Synchronisation sind für die Umsetzung eines Offline-Szenarios notwendig

Vorteile von Replikation

- Verfügbarkeit
- Kommunikationskosten
- Performanz
- Lastbalancierung
- Datenverlust
- Offline-Szenario

Nachteile von Replikation

- Speicherplatzbedarf
- Konsistenzsicherung
- Komplexität des Systems

Zielkonflikt der Replikation

- Forderung an Replikation sind Verfügbarkeit, Konsistenz und Kosten
- Alle drei Forderungen sind gleichzeitig nicht zu erreichen
- Beispiel: Hohe Verfügbarkeit durch hohe Anzahl von Replikaten erschwert die Konsistenzsicherung

Replikationsstrategien

- Copy-Update-Strategie
- Fehlerbehandlung
- Konsistenzsicherung konkurrierender Zugriffe
  - Erstellung einer Serialisierbarkeitsreihenfolge
  - Nutzung von Sperrverfahren, Zeitstempelverfahren sowie semantischen Synchronisationsverfahren
- Behandlung von Lesetransaktionen
  - Berücksichtigung von Alter und Konsistenz

Replikationsdefinition

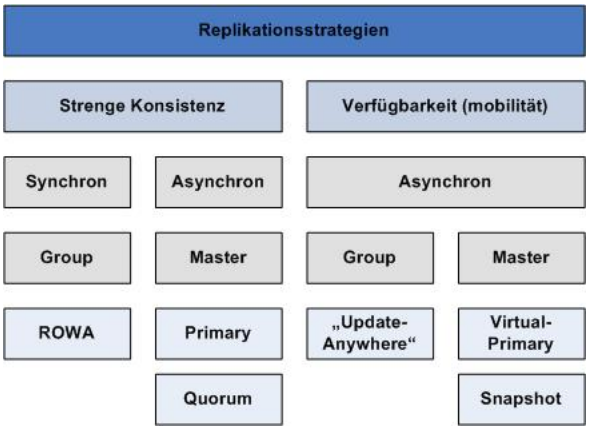
- Replikationsschemadefinition:
  - Definition des Ausschnitts des Datenbankschemas auf der Replikationsquelle, welcher für die Replikation genutzt wird
  - Beschreibt Operationen (z.B. lokal änderbar) und zusätzliche Integritätsbedingungen
  - Dient als Erweiterung für den Synchronisationsprozess
- Replikationsdefinition:
  - Auswahl der zu replizierenden Daten in Abhängigkeit der Replikationsschemadefinition

Synchronisation

- Zielsetzung
  - Bewahrung bzw. Wiederherstellung eines konsistenten Zustands innerhalb der Replikationsumgebung
- Prozess bestehend aus zwei Phasen:
  - Reintegration
  - Rückübertragung
- Bidirektional wenn beide Phasen in einem Arbeitschritt abgehandelt werden
- Unidirektional wenn nur eine Phase in einem Arbeitsschritt abgehandelt wird

Replikations- und Synchronisationsverfahren

- Replikationsstrategien als Grundlage



Konsistenzerhaltene Verfahren

- Pessimistische Verfahren: Erhaltung der Datenkonsistenz durch Sperren
  - ROWA (Synchron)
    - Gleichzeitige Änderung aller Replikate in einer Transaktion
    - Für mobile Umgebungen nicht geeignet
  - Primary Copy-Verfahren (Asynchron)
    - Änderungen auf einer Primärkopie
    - Für mobile Umgebungen nur bedingt geeignet
  - Quorum-Verfahren (Asynchron)
    - Änderungen nur in Abhängigkeit von verfügbaren Replikaten
    - Für mobile Umgebungen nur bedingt geeignet

Verfügbarkeitserhaltende Verfahren

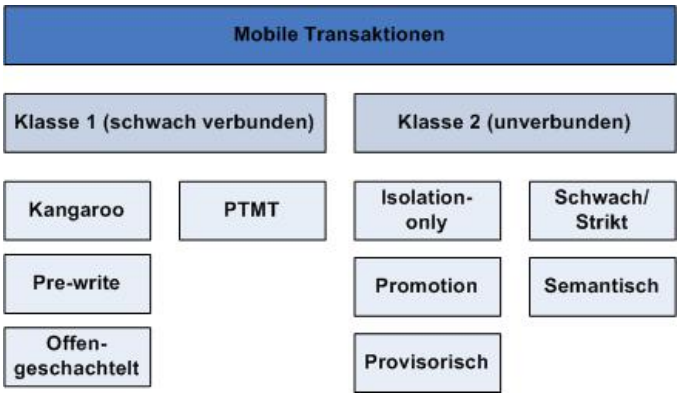
- Virtual Primary-Copy
  - Erweiterung des Primary Copy-Verfahrens
  - Synchronisation zwischen virtueller und realer Primärkopie erforderlich
  - Sehr gut für mobile Umgebungen geeignet
- Snapshot-Verfahren
  - Snapshot als materialisierte Sicht nicht-lokaler Daten
  - Publish & Subscribe-Modell
  - Auf Snapshots kann während Offline-Phase gearbeitet werden
  - Synchrone oder asynchrone Synchronisation
  - Konflikte müssen erkannt und aufgelöst werden
  - Sehr gut für mobile Umgebungen geeignet

Agenda

- Motivation
- Grundlagen
- Replikation und Synchronisation
- **Mobile Transaktionen**
- Anfrageverarbeitung
- Mobile Datenbanken im Ferienclub
- Fazit

Mobile Transaktionen

- Klassisches ACID-Transaktionskonzept als Grundlage



Mobile Transaktionen: Klasse 1

- Kangaroo-Transaktionen
  - Berücksichtigt die Bewegung von mobilen Transaktionen, z.B. Wechsel von Funkzellen
- Preserialisation Transaction Management Technique
  - Ausführung der Transaktion kann temporär unterbrochen werden
- Offen-geschachtelte Transaktionen
  - Teiltransaktionen: Möglichst frühzeitiges Freigeben von belegten Ressourcen
- Prewrite-Transaktionen
  - Zweistufige Transaktionsausführung

Mobile Transaktionen: Klasse 2

- Isolation-Only-Transaktionen
  - Ziel: Möglichst effiziente Transaktionsverarbeitung
  - Nur Isolation-Eigenschaft
- Schwache/Strikte Transaktionen
  - Ziel: Möglichst effiziente Konsistenzsicherung
  - Erstellen von Clustern innerhalb einer Replikationsumgebung
  - Innerhalb eines Clusters strikte Konsistenz
  - Zwischen Clustern schwache Konsistenz
  - Während Offline-Zeitpunkt bildet mobiler Client eigenen Cluster
  - Konfliktauflösung beim Einbinden des mobilen Clusters notwendig

Mobile Transaktionen: Klasse 2 (2)

- Promotion-Transaktionen
  - Ziel: Konfliktvermeidung durch zusätzliche Regeln
  - Compact als Einheit aus zu replizierenden Daten, Datenzugriffsmethoden, Konsistenzbedingungen, Zustandsinformationen und Zusicherungen
  - Ausführung von Transaktionen auf lokalen Compacts
  - Synchronisation von Compacts mit zentralem Datenbestand mit variablen Korrektheitskriterien
- Semantische Transaktionen
  - Ziel: Konfliktvermeidung durch intelligente Fragmentierung der zu replizierenden Daten

Mobile Transaktionen: Klasse 2 (3)

- Provisorische Transaktionen
  - Einfachste Transaktion der Klasse 2
  - Erweiterung des Virtual Primary Copy-Verfahrens
  - Two Tier-Replikation: mobile und stationäre Knoten
  - Grundsätzliche Durchführung von vorläufigen Transaktionen auf lokalem Datenbestand
  - Synchronisation mit globalem Datenbestand durch nachziehen aller vorläufigen lokalen Transaktionen als Basistransaktion auf Master-Knoten
  - Verteilung der Basistransaktion an alle anderen Knoten
  - Konflikterkennung und Konfliktauflösung notwendig
  - Als einziges Klasse 2 Transaktionsmodell in kommerziellen Produkten verfügbar

Agenda

- Motivation
- Grundlagen
- Replikation und Synchronisation
- Mobile Transaktionen
- **Anfrageverarbeitung**
- Mobile Datenbanken im Ferienclub
- Fallbeispiel
- Fazit

Anfrageverarbeitung

- Bekannte Anfrageverarbeitung (Query-Engine) ist nicht auf leistungsschwachen Geräten einsetzbar
- Anfragekomplexität:
  - Anfrageoptimierung nur durch heuristische statt kostenbasierter Optimierungen
  - Gewinn: Einsparung von Rechenleistung
- Änderungsdynamik zugegriffener Daten:
  - Bei statischen Daten bringen vorkompilierte Anfragen und materialisierte Sichten (Snapshots) eine Performanzsteigerung
  - Gewinn: Einsparung von Speichernutzung
- Energieverbrauch und Kommunikationskosten

Agenda

- Motivation
- Grundlagen
- Replikation und Synchronisation
- Mobile Transaktionen
- Anfrageverarbeitung
- **Mobile Datenbanken im Ferienclub**
- Fazit

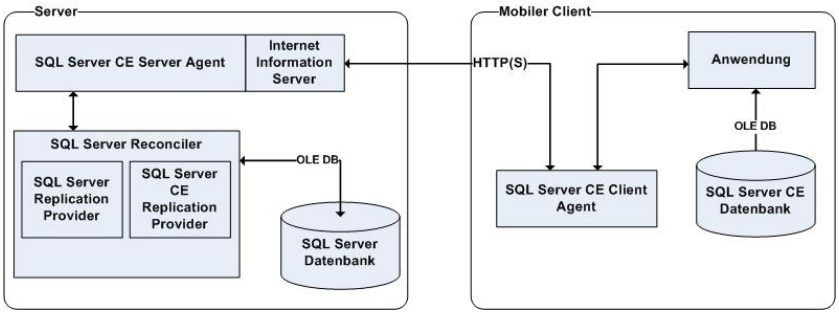
Verfügbare mobile Datenbanksysteme

- Pico-Datenbanksysteme
  - ?
- Eingebettete Datenbanken
  - eXtremeDB, Hypersonic SQL Database, Instant DB, Cloudscape
- Integrierte Datenbanken
  - Sybase SQL Anywhere Studio
  - PointBase Micro
- Erweiterte Client/Server-Architektur
  - IBM DB2 Everyplace
  - Oracle 10g Lite
  - MS SQL Server CE
  - Tamino Mobile Suite

Vorschlag: SQL Server CE

- Remote Data Access
  - Push & Pull-Mechanismus
  - Optimistische Nebenläufigkeitskontrolle
  - Konflikterkennung, aber keine Auflösung
- Merge Replication
  - Basiert auf Publish & Subscribe-Modell
  - Provisorische Transaktionen
  - Replikatsauswahl auf Server
  - Horizontale (Selektion) und vertikale (Projektion) statische und dynamische Filter
  - Tupelbasierte Konflikterkennung, flexible Konfliktauflösung

SQL Server CE Merge-Architektur



SQL Server CE API

- Synchronisationsmethode des Clients:

```
public void Synchronize() {
      SqlCeReplication repl = new SqlCeReplication();

      repl.InternetUrl = "http://awp/sqlerverce/sscesa20.dll";
      repl.Publisher = “SERVER\AWP";
      repl.PublisherDatabase = “AWP";
      repl.PublisherLogin = “name";
      repl.PublisherPassword = “password";
      repl.Publication = “AWP";
      repl.Subscriber = "PPC_AW_P01";
      repl.SubscriberConnectionString = @”Data Source=\AWP.sdf";

      repl.AddSubscription(AddOption.CreateDatabase);

      repl.Synchronize();
}
```

SQL Server CE: Synchronisationskonflikte

- Ursachen
  - Update – Update
  - Insert – Insert
  - Update – Delete
  - Delete – Update
- Erkennung
  - Versionsnummern und GIUDs
  - Konflikte werden protokolliert
- Vermeidung
  - Horizontale Filter

SQL Server CE: Konfliktauflösung

- Default
- Additive
- Averaging
- Earlier Wins
- Later Wins
- Maximum
- Minimum
- Merge Text
- Subscriber Wins
- Custom

SQL Server CE: Konfliktauflösung (2)

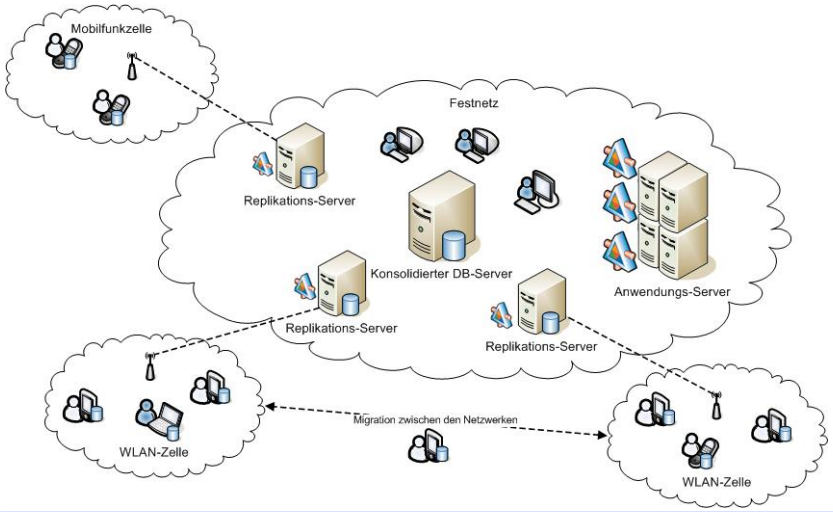
- Custom Stored Procedure:

```
...
// Subscriber
SELECT @select_string =
   'SELECT * FROM OPENQUERY('+QUOTENAME(@subscriber)+','+'"SELECT
   ID,PRICE,ROWGUIDCOL FROM ' +  @sub_qualified_name  +
   " WHERE ROWGUIDCOL = '" + convert(varchar(36),@rowguid) + "'" +
   ' ") '
...
FETCH NEXT FROM @cursor INTO @ID, @SubPRICE, @rowguidvar
...
// Publisher
SELECT ID, @PubPRICE = PRICE,
   @ResolvedPRICE = PRICE + @SubPRICE / 2.0
FROM PRODUCTS
WHERE ROWGUIDCOL = @rowguid
...
```


Bewertung SQL Server CE

- Vorteile:
  - Footprint ca. 800KB
  - Speicherkapazität bis 2GB
  - Komplexe SQL-Abfragen möglich
  - Lokale ACID-Transaktionen
  - Flexible, intelligente Synchronisation inklusive Datenkompression und Konfliktbehandlung
  - Umfassende Sicherheitsmechanismen
- Nachteile:
  - Keine Unterstützung von gespeicherten Prozeduren und Trigger
  - Relativ aufwendige Einrichtung und Administration
  - Microsoft-Technologien erforderlich

Architektur des Ferienclubs



Fazit: Mobile Datenbanken im Ferienclub

- Evaluierung und Bereitstellung einer geeigneten mobilen Datenbankarchitektur
- Integrierung von mobilen Datenbanken in
  - Data Mining
  - Persistence Service
  - Distributed AI
  - Service Oriented Architecture
  - Semantic Web

Agenda

- Motivation
- Grundlagen
- Replikation und Synchronisation
- Mobile Transaktionen
- Anfrageverarbeitung
- Mobile Datenbanken im Ferienclub
- **Fazit**

Fazit

- Nur entfernte Datenhaltung ist nicht ausreichend
- Mobiles Arbeiten und mobile Datenbanken sind nicht voneinander zu trennen
- Bisher viele Konzepte und einige kommerzielle Produkte
- Replikation und Synchronisation in mobilen Datenbanksystemen ein weiterhin spannendes Forschungsgebiet

Vielen dank für die Aufmerksamkeit

Quellen

[1] Bela Mutschler, Günther Specht, Mobile Datenbanksysteme, Springer-Verlag Berlin Heidelberg, 2004

[2] Jim Grey et al., The Dangers of Replication and a Solution, Proceedings of the 1996 ACM SIGMOD international conference on Management of data, 1997

[3] Can Türker, A Survey of Academic and Commercial Approaches to Transaction Support in Mobile Computing Environments, Technical Report #429, ETH Zürich, 2003

[4] Brad Hammond, Merge Replication in Microsoft's SQL Server 7.0, Proceedings of the 1999 ACM SIGMOD international conference on Management of data, 1999

[5] Daniel Barbará, Mobile Computing and Databases – A Survey, IEEE Transactions on Knowledge and Data Enginerring Vol. 11 No. 1, 1999