Verteilte Systeme

Prof. Dr. Martin Becke

CaDS - HAW Hamburg

Version 0.9

Inhalt

Architektur in Verteilten Systemen

Allgemein

- Freiheitsgrade der Realisierung
- Am Ende Aufruf ISA der Architekturen
- Notwendigkeit der Kommunikation
- Zerlegung beherrschbar durch Zerlegungsmethoden

Zerlegungsmethoden

Teile-und-Herrsche-Strategie

- Funktional
- Ressourcenorientiert

Zerlegungsmethoden

Funktional

- Kleinere, im besten Fall atomare, Funktionen/Prozeduren
- Funkion sind von einander unabhängig
- Sprachen theoretisch austauschbar (Sprachen Turing-vollständig)
- Kosten : Zeit und Speicher

- Funktionssignatur
  - Name
  - Parameter
  - Rückgabetyp
  - Bedingungen/Einschränkungen
- Funktionssemantik/-Definition
  - Ausführungslogik
  - Fehlerbehandlung
  - Beschreibung durch Kommentaren, Dokumentation oder Code

```
function sum(a: int, b: int) -> int;
```

Listing 1 – Funktionssignatur

```
function sum(a: int, b: int) -> int {
    return a + b;
}
```

Listing 2 – Funktionssemantik

- Kleinere, auf Ressourcen orientierte Sicht
- Einheit auf eine bestimmte Ressource spezialisiert
- Nur Set von Funktionen : Basis CRUD
- Häufig intuitiver als Funktionen

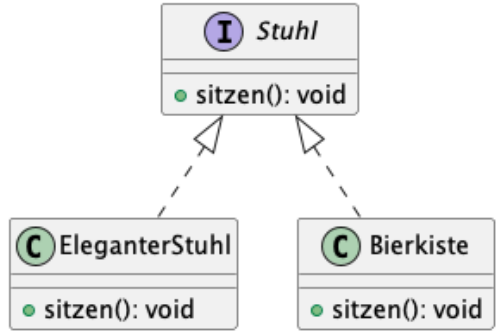

Figure – Funktionale Zerlegung mit Interface

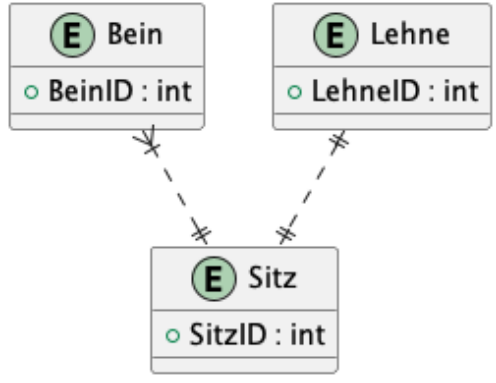

Figure – Resourcen-orientierte Zerlegung mit Relation

```java
public class ChessBoard {
    private Piece[][] board;

    public ChessBoard() {
        this.board = new Piece[8][8];
        // Initialize the board with the starting
            positions of pieces
        // e.g. add new Piece(PieceType.ROOK,
            Color.WHITE) to board[0][0] for the white
            rook in the top-left corner.
    }

    public void setPiece(int row, int col, Piece
        piece) {
        board[row][col] = piece;
    }
}
```

Listing 3 – Schachbrett - Objektorientiert

```
CREATE TABLE ChessBoard (
    id INT PRIMARY KEY,
    row INT,
    col INT,
    pieceType VARCHAR(10),
    pieceColor VARCHAR(5)
);
```

Listing 4 – Schachbrett - Datenbank

Datenseperation

Einführung

- Daten auf mehrere Knoten in einem Netzwerk
- Aufteilung von Daten in kleine Teile
- Verschiedene Methoden

Datenseperation

Methoden

- Horizontale Partitionierung
- Vertikale Partitionierung
- Sharding
- Replikation

Inhalt

Kommunikation

Schnittstellen

- Datenaustausch
- Ressourcenmanagement
- Koordination
- Fehlererkennung und -behebung
- Skalierung

Kommunikation

Art

- Synchrone Kommunikation
- Asynchrone Kommunikation

Kommunikation

Art

- Persistente Kommunikation
- Transiente Kommunikation

Kommunikation

Art

- Signal
- Event
- Nachricht

Kopplung

Idee

- Art und Weise wie interagiert und kommuniziert wird
- Wahl der Kopplungsart wesentlich für Aufbau und Leistung
- Starker Einfluss auf Interoperabilität und Integration

Kopplung

Arten

- Direkte Kopplung
  - Mediator-System
  - Middleware
- Indirekte Kopplung
- Losgekoppelte Kopplung
- Strukturelle Kopplung

States

Einordnung

- stateful
- stateles

Transaktion

Idee

- Eine Sequenz von Operationen
- Wichtig für Konsistenz und Integrität
- ACID-Eigenschaften

Transaktion

Beispiele

- Bankwesen
- E-Commerce
- Verteilte Datenbanken
- Roboter/Urlaub (Diskussion)

Transaktion

Transaktionsmanager

- Verschachtelte Transaktionen
- Größeren Transaktion separat abschliessen
- Koordiniert zurückzurollen

Transaktionsmanager

Aufgaben

- Koordination
- Protokollierung und Wiederherstellung
- Isolierung und Synchronisation
- Commit und Rollback

Transaktion

Message Passing

- Kommunikationsparadigma
- Grundlegendes Konzept
- Nutzt Nachrichten, die Daten oder Anweisungen enthalten

Message Passing

Eigenschaften

- Asynchrone Kommunikation
- Lose Kopplung
- Skalierbarkeit

Message Passing

Beispiele

- Message Queues
- Message Passing Interface (MPI)
- Publish-Subscribe-Systeme
- MOM

Message Passing

Actor-Modell

- Setzt auf das Konzept von Message Passing auf
- Ist ein Konzept für das Design von verteilten Systemen
- Actoren sind grundlegende Recheneinheiten
- Isolation
- Nachrichtenbasiert
- Lokalitätstransparent
- Fehlertolerant
- Beispiel CAF (HAW Hamburg)

Idempotent

Eigenschaft

- Operationen, die wiederholt ausgeführt werden können, ohne dass sich das Ergebnis nach der ersten Anwendung ändert
- Beispiel HTTP Put

DHT

Eigenschaft

- Schlüssel-Wert-Speichersystem
- Bedeutung im Peer-to-Peer-Netzwerk
- Wichtig ist der Schlüsselraum