Objektorientierte Datenbanken

Vorlesung 2 vom 15.04.2004

Dr. Sebastian Iwanowski
FH Wedel

Offene Frage vom letzten Mal:

Wie können wir objektorientiert modellieren und dennoch eine Datenbank benutzen ?

Technische Ziele für die Objektmodellierung

Standardisierung des Objektmodells

OMG, ODMG

Persistenzkonzept

Dauerhaftigkeit von Daten

Transaktionskonzept

Verknüpfung von Operationen zu einer Einheit

Unterstützung von Integritätsbedingungen

Semantischer Zusammenhang zwischen den Daten

Inhalt heute:

Gründe für die Entstehung der ODMG

- Warum ist Standardisierung notwendig ?
- Smalltalk versus C++
- Java als Kompromiss

Das Objektmodell der ODMG

- Objektdefinitionen
- Persistenzkonzepte

Gründe für die Entstehung der ODMG

Warum ist Standardisierung notwendig ?

Antwort:

Wenn Datenbanken Objekte aus verschiedenen Programmen zum Zweck der Manipulation und des Austauschs halten sollen, dann müssen diese Programme dasselbe Verständnis von Objekten und ihren Eigenschaften haben.

Datenbanken haben einen de-facto-Standard:

Relationales Modell, SQL als DML

Objektorientierte Sprachen unterscheiden sich in:

Vererbungskonzept

Typkonzept

Kapselungskonzept

Speicherverwaltung und Speicherzugriff

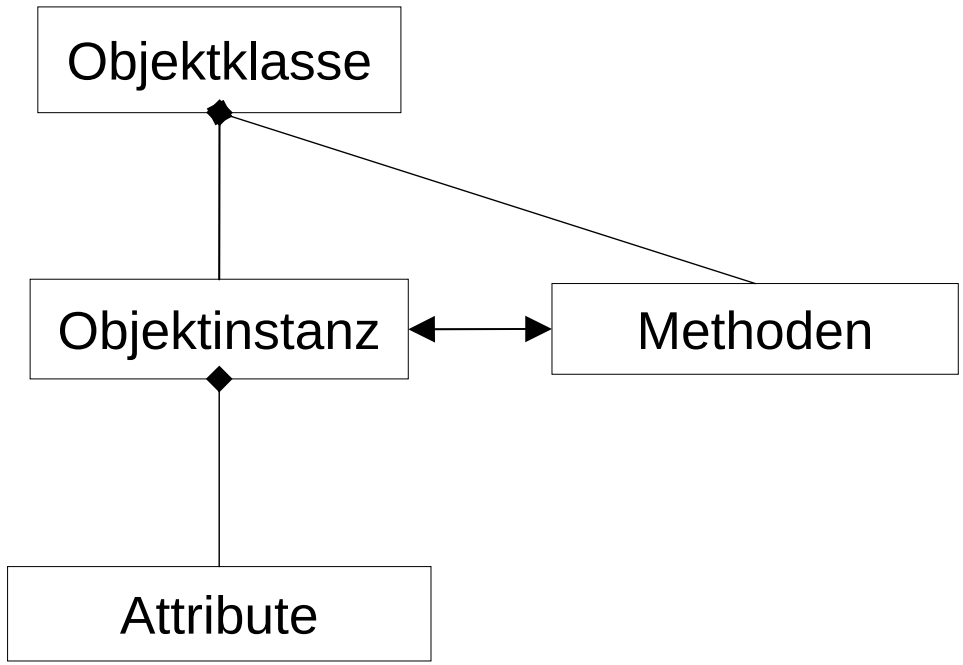
Gemeinsamkeiten aller OOP

Objekthierarchie und -aufbau:

Deklaration der Attribute
Deklaration und Festlegung der Methoden

Festlegung des Zustands:
definiert durch Werte der Attribute
Festlegung des Methodenparameters self

Festlegung der Werte



Klassen können spezialisiert und generalisiert werden (Vererbung)

Unterschiede am Beispiel Smalltalk und C++

Konzept	Smalltalk	C++
Vererbungskonzept	Unterklassen erben alle Methoden, keine Attribute Methoden werden überlagert, wenn die Namen gleich sind keine Mehrfachvererbung	Unterklassen erben speziell dafür frei gegebene Attribute und Methoden Methoden werden überlagert, wenn Namen, Parameteranzahl und Parametertypen gleich sind Mehrfachvererbung
Typkonzept	keine Datentypen, alle Daten sind Objekte	alle Datentypen, alle Objekte sind intern `structs`

Unterschiede am Beispiel Smalltalk und C++

Konzept	Smalltalk	C++
Kapselungskonzept	Alle Methoden sind öffentlich, alle Attribute privat	Methoden und Attribute können öffentlich, privat oder dazwischen sein und müssen als solche explizit deklariert werden
Speicherverwaltung und Speicherzugriff	Speicher wird automatisch eingerichtet und bereinigt Call by reference für alle Objekte	Speicher wird nur bei einfachen Datentypen automatisch eingerichtet und bereinigt, bei anderen Datentypen und bei Objekten nur per Befehl Call by value für alle Daten und Objekte ➔ Pointerkonzept notwendig

Geschichte der ODMG

gegründet 1991 als unabhängige Untergruppe der OMG

Mitglieder: 90 % des damaligen OODB-Marktes

Leitung: Rick Cattell (Sun, später SunSoft, später JavaSoft)

Ziele:

- Vereinheitlichung der Smalltalkwelt und C++-Welt durch Objektstandard
- Anbindung an die Sprachen Smalltalk und C++ (ab 1997 auch Java)
- Persistenzkonzepte
- Transaktionskonzepte
- Abfragekonzepte

Das tat ab 1995 auch Java !

2001 aufgelöst

parallel entwickelter Standard: JDO (nur für Java)

Das ODMG-Objektmodell

Das ODMG-Objektmodell

Objekte Unterscheidung: atomar oder strukturiert

Literale Werte (Unterscheidung: atomar oder strukturiert)

Eigenschaften Unterscheidung: Attribute oder Beziehungen

Verhalten Methoden (Unterscheidung: Schnittstelle oder Implementierung)

Typen Oberbegriff für Objekte mit gemeinsamen Eigenschaften und Verhalten
(Schnittstelle **mit** Implementierungen)

Interface Schnittstelle **ohne** Implementierungen

Klasse Implementierung eines Typs

Extension Menge aller Instanzen eines Typs bzw. einer Klasse und der Subtypen (Unterklassen)
Beachte: Eine Instanz gehört eindeutig zu einem Typ (Klasse) !

Schlüssel zum eindeutigen Identifizieren von Objekten

Das ODMG-Objektmodell

Konzept	ODMG
Vererbungskonzept	Unterklassen erben alle Methoden, Attribute und Beziehungen Überlagerung von Methoden bei Vererbung ist verboten Mehrfachvererbung nur bei Interfaces, nicht bei Klassen
Kapselungskonzept	Kapselung wird nicht beachtet: Alles ist öffentlich !
Speicherverwaltung und Speicherzugriff	ist Sache der Programmiersprache

Das ODMG-Objektmodell

Vordefinierte Typen für die Objekte:

keine vorgeschriebenen atomaren Typen

(wird der Implementierung überlassen)

Collection Types: Set, Bag, List, Array, Dictionary

(Typen der Elemente müssen gleich sein und explizit spezifiziert werden)

Structured Types: Date, Interval, Time, TimeStamp

Vordefinierte Typen für die Literale:

atomare Typen: sehr viele (alle von Java und noch mehr)

Collection Types: set, bag, list, array, dictionary

Structured Types: date, interval, time, timestamp

Unterscheidung zwischen Objekt und Literal nur für nicht objektorientierten Teil der C++ - Implementierung

Das ODMG-Objektmodell

Objektspezifikationssprachen:

Object Definition Language (ODL)

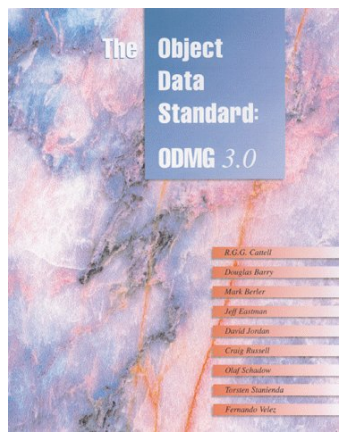
(zum Definieren von Objekten)

Object Interchange Format (OIF)

(zum Speichern und Laden von Objekten mit ihren gegenwärtigen Zuständen)

Ausführliche Syntax und Beispiele in Kap. 3 von:

Richard G.G. Cattell / Douglas K. Barry (Edts.): *The Object Data Standard: ODMG 3.0*



Persistenzkonzepte der ODMG

Persistenzkonzepte der ODMG

Was ist Persistenz ?

Persistenz ist die Fähigkeit von Daten, beliebig lange Lebensdauern anzunehmen unabhängig von dem Programm, in dem sie erzeugt wurden.

Nicht persistente Daten heißen transient:
Transiente Daten leben nur so lange wie das Programm (der Programmteil), in dem sie erzeugt wurden.

Die Funktionsweise von Programmen darf sich nicht ändern, wenn das Programm mit persistenten statt mit transienten Daten arbeitet.

(Persistenz ist Programm-orthogonal)

Persistenz muss prinzipiell mit jedem Typ funktionieren

(Persistenz ist Typ-orthogonal)

Persistenzkonzepte der ODMG

klassenabhängige Persistenz

Jede Klasse kann persistent gemacht werden.

Alle Instanzen einer persistenten Klasse sind persistent.

ODMG: **objektabhängige Persistenz**

Jedes Objekt einer persistenzfähigen Klasse kann persistent gemacht werden.

Die Persistenz eines Objekts erfolgt durch explizite Kennzeichnung.

Smalltalk **1) Automatische Persistenzfähigkeit**

C++ **2) Persistenzfähigkeit durch Vererbung**

Java **3) Persistenzfähigkeit durch explizite Kennzeichnung**

Persistenzkonzepte der ODMG

Objekte mit Beziehungen zu anderen Objekten:

C++

1) Explizite Persistenz

Persistenz eines Objekts erfolgt nur durch explizite Kennzeichnung.

Smalltalk
Java

2) Transitive Persistenz

Auch Persistenz durch Erreichbarkeit genannt:

Jedes Objekt, das durch ein persistentes Objekt erreicht werden kann,
ist ebenfalls persistent, sofern es einer persistenzfähigen Klasse angehört
(anderenfalls ist es transient)

Beim nächsten Mal:

Das ODMG-Transaktionskonzept
Das ODMG-Abfragekonzept
Integritätsbedingungen (eventuell)
Beginn der praktischen Übungen