Objektorientierte Datenbanken

Vorlesung 3 vom 22.04.2004

Dr. Sebastian Iwanowski
FH Wedel

Inhalt heute:

Das Transaktionskonzept der ODMG

Gewährleistung von Integritätsbedingungen durch objektorientierte Programmierung

Die Java-Anbindung der ODMG

Die Realisierung des ODMG-Standards in FastObjects

Das Transaktionskonzept der ODMG

Warum brauchen wir ein Transaktionskonzept ?

Bsp.: Bankkontoführung

Prozess 1: Umbuchung eines Betrages von Konto A nach Konto B

geplant:

Umbuchung

read (A, a1)
a1 := a1 - 300
write (A, a1)
read (B, b1)
b1 := b1 + 300
write (B, b1)

tatsächlicher Verlauf:

Umbuchung

read (A, a1)
a1 := a1 - 300
write (A, a1)
read (B, b1)

Störung

Wo sind die 300 € geblieben?

Warum brauchen wir ein Transaktionskonzept ?

Bsp.: Bankkontoführung

Prozess 1: Umbuchung eines Betrages von Konto A nach Konto B

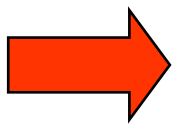
Prozess 2: Zinsgutschrift für Konto A

Umbuchung

```
read (A, a1)
a1 := a1 - 300
write (A, a1)
read (B, b1)
b1 := b1 + 300
write (B, b1)
```

Zinsgutschrift

```
read (A, a2)
a2 := a2 * 1.03
write (A, a2)
```



Möglicher verzahnter Ablauf:

Umbuchung	Zinsgutschrift
read (A, a1)	
a1 := a1 - 300	
	read (A, a2)
	a2 := a2 * 1.03
	write (A, a2)
write (A, a1)	
read (B, b1)	
b1 := b1 + 300	
write (B, b1)	

Wo ist die Zinsgutschrift geblieben?

Transaktionskonzept der ODMG

Was ist eine Transaktion ?

Eine Transaktion ist eine Folge von Operationen, die entweder alle vollständig oder alle überhaupt nicht durchgeführt werden sollen.

Eine Transaktion muss das ACID-Prinzip erfüllen:

Atomicity	unteilbar
Consistency	hinterlässt immer konsistenten Datenbestand
Isolation	beeinflusst keine andere Transaktion
Durability	Wirkung ist permanent

Transaktionskonzept der ODMG

Interface `Transaction`:

begin()	Anfang einer Transaktion
commit()	Ende einer Transaktion (erfolgreich)
abort()	Ende einer Transaktion (nicht erfolgreich)

➔ *Alle zwischen Anfang und Ende einer Transaktion angetroffenen Operationen gehören zur selben Transaktion.*

➔ *Alle Operationen, die sich auf die Datenbank auswirken sollen, müssen innerhalb einer Transaktion ausgeführt werden.*

➔ *Eine Transaktion kann nur ausgeführt werden, wenn eine Datenbank geöffnet worden ist.*

Datenbanken werden durch Objekte des Interfaces `Database` repräsentiert

Gewährleistung von Integritätsbedingungen durch objektorientierte Programmierung

Integritätsbedingungen

Was ist eine Integritätsbedingung ?

Eine Integritätsbedingung ist eine beliebige logische Bedingung, die zwischen den betrachteten Daten gelten muss

statische Integritätsbedingung

Zulässige Daten hängen von einer absoluten Bedingung ab.

dynamische Integritätsbedingung

Zulässige Daten hängen vom Zustand der zuletzt angenommenen Datenbelegung ab. *„weiche“ Bedingung*

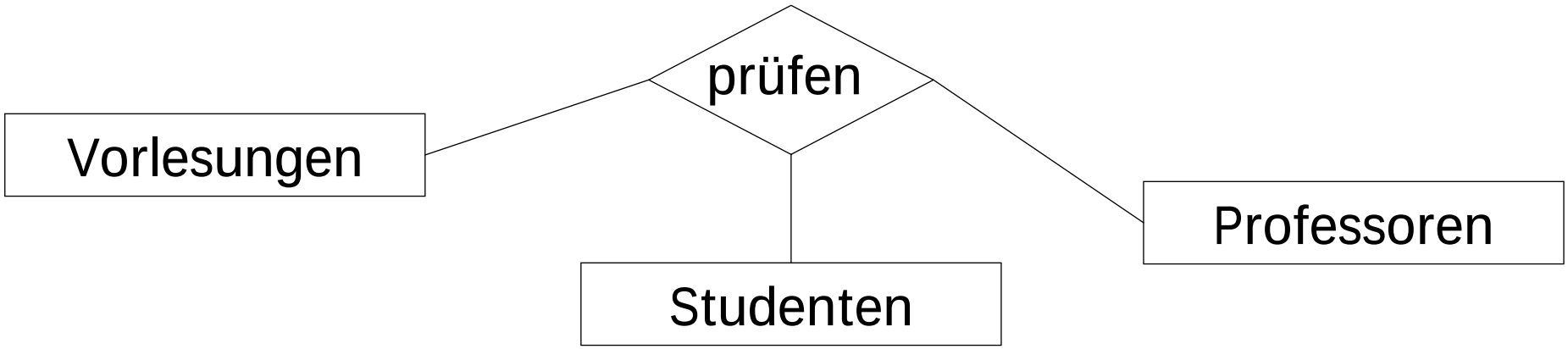
Zulässige Daten hängen von der Entwicklung der in der Vergangenheit angenommenen Zustände ab. *„scharfe“ Bedingung*

In OOP können Integritätsbedingungen immer gewährleistet werden durch:

a) Typvereinbarungen

oder b) Implementierung entsprechender Manipulationsmethoden

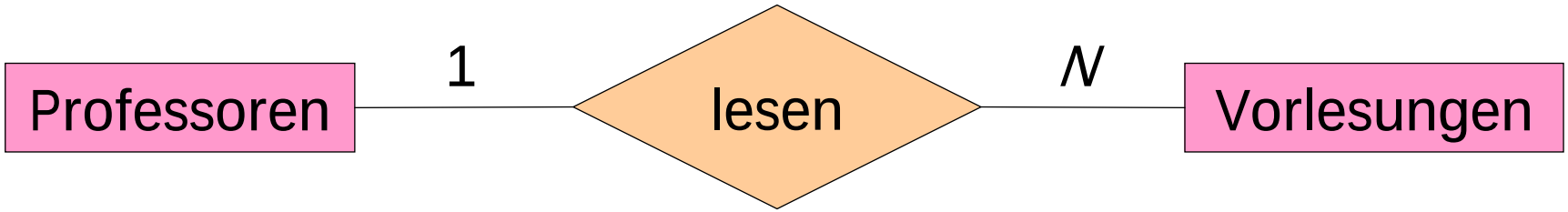
1. Beispiel für eine statische Integritätsbedingung:



Typvereinbarungen legen fest, ob Studenten von Professoren oder Assistenten geprüft werden:

```
class Prüfungen {
    attribute Datum Prüfdatum; // ist Klasse, aber dennoch Attribut
    attribute float Note;
    relationship Professoren Prüfer inverse  Professoren::hatGeprüft;
    relationship Studenten Prüfling inverse  Studenten::wurdeGeprüft;
    relationship Vorlesungen Inhalt inverse  Vorlesungen::wurdeAbgeprüft;
};
```

2. Beispiel für eine statische Integritätsbedingung:



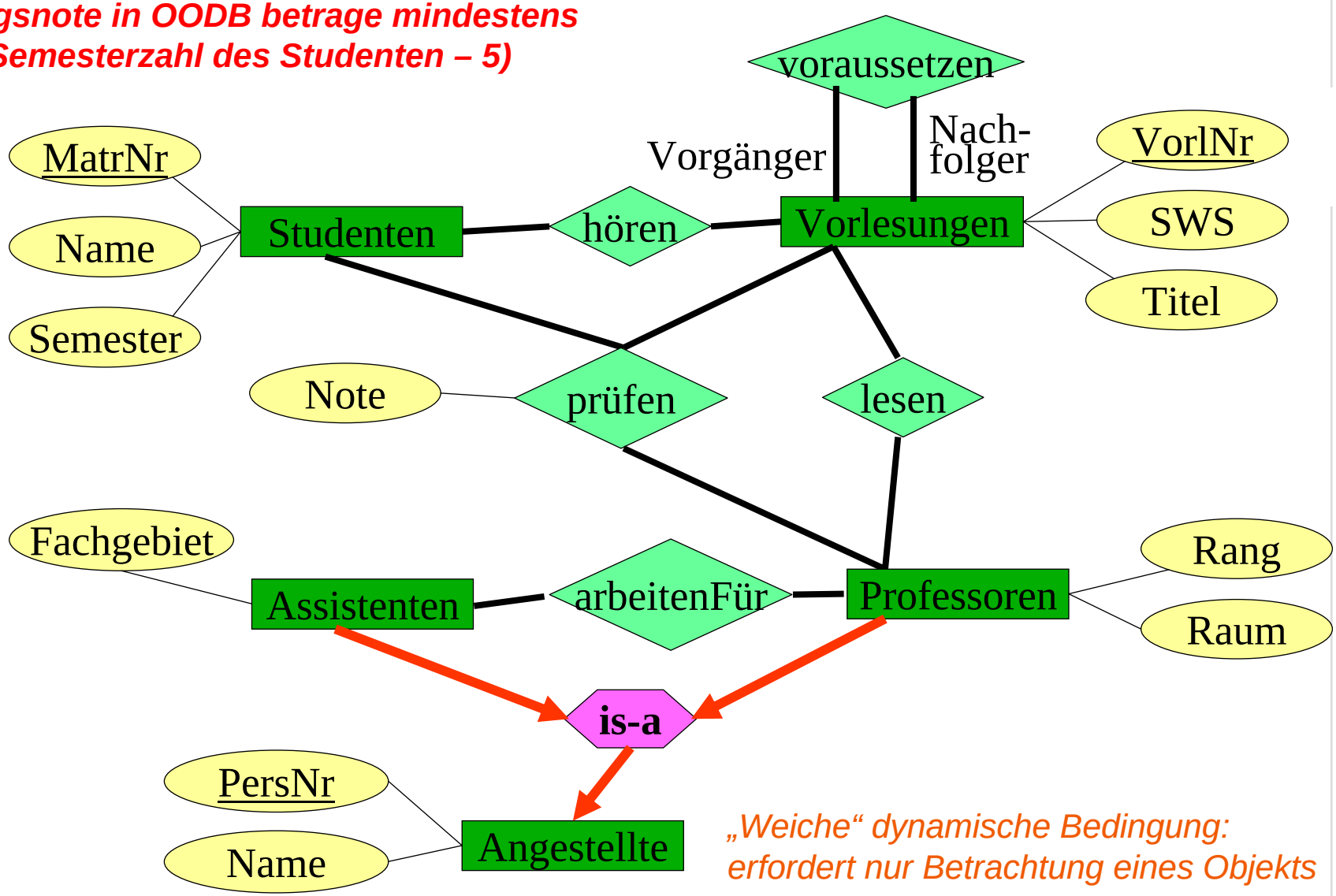

```
class Professoren {
    ...
    relationship set(Vorlesungen) liest inverse Vorlesungen::gelesenVon;
    boolean attachVorlesung (Vorlesung vorlesung)
    {
        if ((Vorlesungen.size() >= N)
                && !(Vorlesungen.contains (vorlesung)))
                return false;
        else
        {
                Vorlesungen.insert (vorlesung);
                vorlesung.wirdGelesenVon (this);
                return true;
        }
    }
    ...
};
```

Der Set-Typ sorgt dafür, dass ein Professor mehrere Vorlesungen halten kann.

Die Zugriffsmethode sorgt dafür, dass ein Professor nicht mehr als N Vorlesungen hält:

Beispiel für eine dynamische Integritätsbedingung:

Die Prüfungsnote in OODB betrage mindestens den Wert (Semesterzahl des Studenten – 5)



„Weiche“ dynamische Bedingung: erfordert nur Betrachtung eines Objekts

Weitere Beispiele für dynamische Integritätsbedingungen:

- Für die Mitarbeiter gilt, dass das Durchschnittsgehalt der Qualitätssicherer unter dem der Arbeitsplaner liegen muss

  „Weiche“ dynamische Bedingung:
  erfordert aber die Betrachtung vieler Objekte

- Der Durchschnittsverkaufspreis eines Produktes bezogen auf die letzten zwölf Monate darf nicht mehr als fünf Prozent vom Durchschnittspreis der letzten beiden Jahre abweichen.

  „Scharfe“ dynamische Bedingung:
  erfordert die Abspeicherung vergangener Zustände

Die Java-Anbindung der ODMG

Transaktionen und Datenbanken

Es gibt die Java-Interfaces Transaction und Database:

Interface Transaction

```
public void begin();

public void commit();

public void abort();

public void checkpoint();

public boolean isOpen();

public void join();

public void leave();

public void lock()
        throws lockNotGrantedException;

public boolean tryLock();
```

Interface Database

```
public void open(String name, int accessMode)
        throws ODMGException;

public void close();

public void bind (Object object, String name)
        throws ObjectNameNotUniqueException;

public Object lookup(String name)
        throws ObjectNameNotFoundException;

public void unbind(String name)
        throws ObjectNameNotFoundException;

public void makePersistent(Object object);

public void deletePersistent(Object object);
```

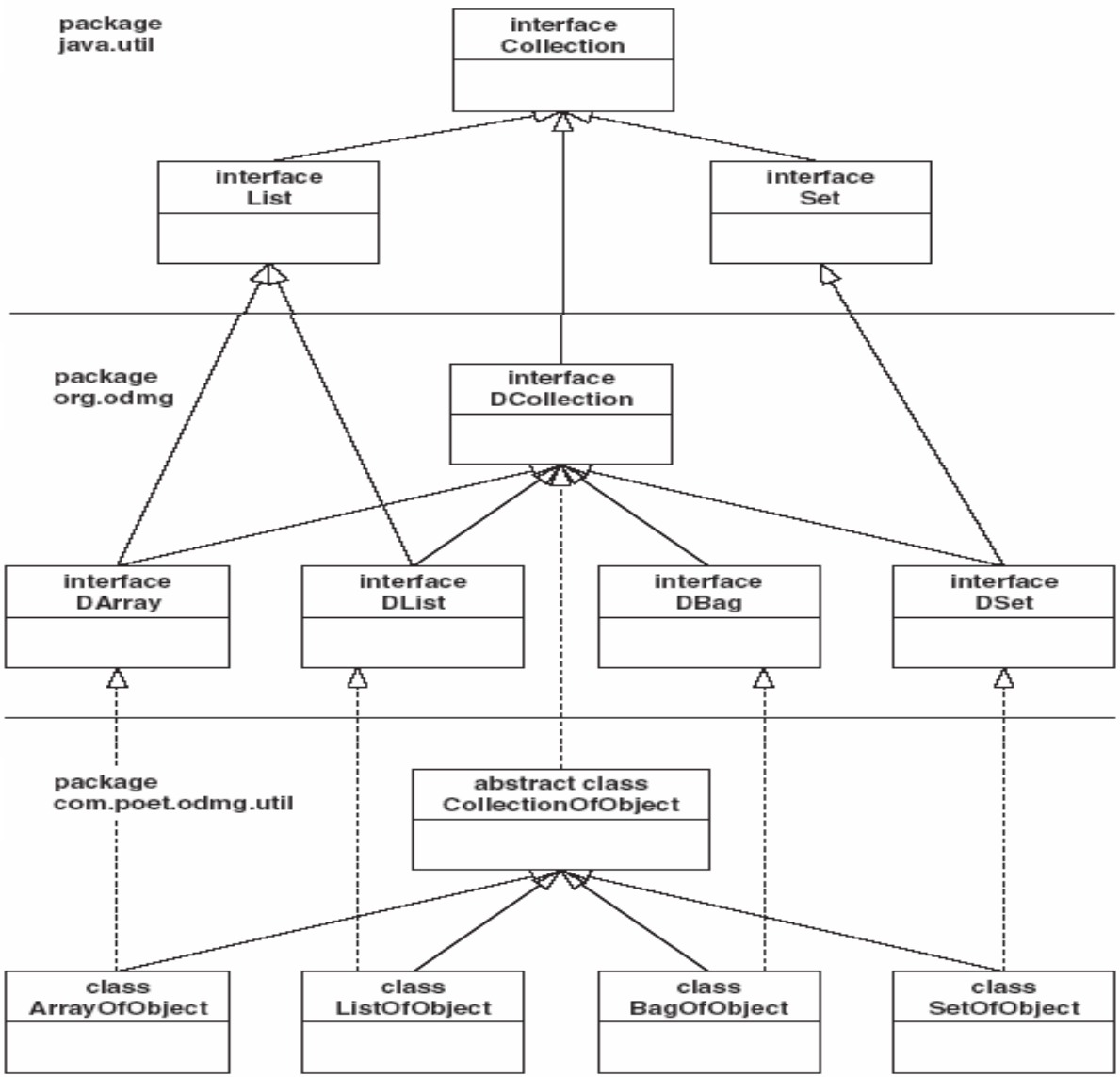
➔ ***In FastObjects: Es gibt Klassen gleichen Namens***

Collections und Sets

Java:

ODMG:

FastObjects:



Collections und Sets

Warum gibt es Subinterfaces von Collections und Sets ?

➔ Übungsaufgabe: Antwort in der Vorlesung

Wie erreicht man die schärferen ODMG-Spezifikationen bzgl. Collections und Sets in Java ?

➔ durch Property Files

☹ wird in FastObjects nicht berücksichtigt ☹

Extensionen und Schlüssel

In ODMG für Java nicht festgelegt

☺ Extensionen in FastObjects implementiert ☺

Class Extent

public Extent (Database db, String objectName);

public void next ();

public boolean hasNext ();

viele weitere Methoden . . .

Die Realisierung des ODMG-Standards in FastObjects

ODMG-Standard in FastObjects

- **Klassendefinitionen in Java (also kein ODL / OIF)**
- **Persistenzfähige Klassen in Datei ptj.opt festgelegt**
- **Die persistenzfähigen Klassen müssen gesondert übersetzt werden**
- **Mehrere Datenbanken werden in einem Schema zusammengefasst**
- **Die Datenbanknamen sind grundsätzlich URL-Namen**

Beim nächsten Mal:
Die ODMG-Abfragesprache OQL

und jetzt gleich:
Anwendung der OODB-Software FastObjects
life
in den Übungen