Vorlesung Implementierung von Datenbanksystemen

1. Einführung

Prof. Dr. Klaus Meyer-Wegener

Wintersemester 2019/20

Lehrstuhl für Informatik 6 (Datenmanagement)
Friedrich-Alexander-Universität Erlangen-Nürnberg

Datenabstraktion

- **Auch: Datenkapselung, Datenunabhängigkeit (der Anwendung), Information Hiding, …**

- **Speichern und Wiedergewinnen von Daten**
- **ohne Kenntnis der Details der Speicherung**
  - Keine Adressen, sondern Namen
  - Keine Bytefolgen, sondern Typen (einer Programmiersprache)
- **Persistente (dauerhafte) Speicherung**
  - Daten bleiben über Programmende oder Rechnerabschaltung hinaus erhalten
- **Wiedergewinnen =**
  Auffinden (Suchen, Lokalisieren) und
  Aushändigen (Übergeben – im richtigen Format!)

Generische Software

- **Altes Prinzip der Informatik bzw. der Programmierung:**
  - Was man in verschiedenen Anwendungen immer wieder tun muss
    - Ausgabe auf Drucker oder Bildschirm,
      Kommunikation über ein Netz,
      persistente Speicherung von Daten usw.
  - verallgemeinern (Abstraktion)
  - und in Unterprogramme auslagern.
- **Generische (anwendungsübergreifende) Lösungen**
  - **Konfigurierbar** für die Zwecke einer bestimmten Anwendung
  - durch Parameter oder **Schema**
- **Operationen, Schnittstelle**
  - Was braucht man?
  - Vollständigkeit …

1.1 Schichtenbildung

- **Strukturierung komplexer Software-Systeme**
  - Z.B. ISO/OSI für Kommunikationsprotokolle
- **Entwurfsmuster**
- **Schicht**
  - realisiert einen bestimmten **Dienst** (abstrakt)
    den sie an der **Schnittstelle** (konkret) "nach oben"
    der darüber liegenden Schicht
    zur Verfügung stellt
  - nimmt dafür Dienste darunter liegender Schichten in Anspruch
  - verbirgt die darunter liegenden Schichten vollständig
    (Datenkapselung, siehe oben)
  - muss daher *alle* erforderlichen Funktionen anbieten

Schichtenbildung (2)

- **Beobachtung**
  - Entlang der Schichten wird nach oben hin "abstrahiert" (Veredelungshierarchie).
    - Weniger lästige Details, mächtigere Funktionen
  - "Eine Hauptaufgabe der Informatik ist die systematische Abstraktion." (Hartmut Wedekind)

- **Problem: Anzahl *n* der Schichten in einem Software-System**
  - *n* = 1: monolithisches System

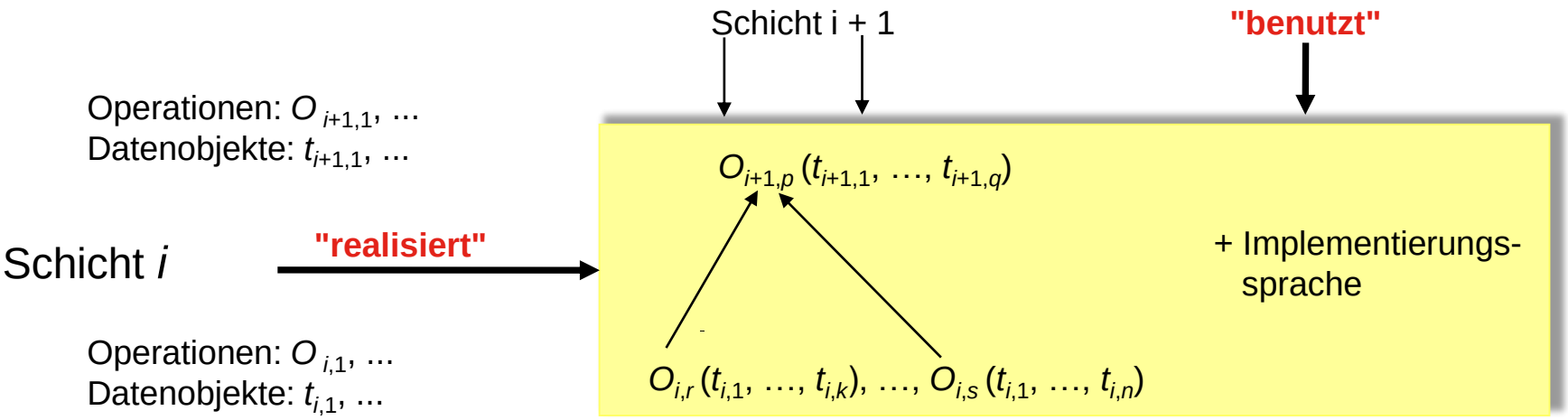
    → keine Schichtenbildung
  - *n* sehr groß:

    → hoher Koordinierungsaufwand, Leistungseinbußen (Schnittstellenüberquerung)
  - Faustregel: *n* typischerweise zwischen 3 und 10

Aufbauprinzip einer Schicht

- **Genauere Begriffsbestimmung "Schicht", "Schnittstelle"**
  - $\{ O_{i,j} \}$: Menge der **Operationen** an der Schnittstelle der Schicht *i*
  - $\{ t_{i,j} \}$: Menge der **Datenobjekte** (Adressierungseinheiten) der Schnittstelle *i*

- **Aufbauprinzip:**



Vorteile der Schichtenbildung

- **Konsequenzen der Nutzung hierarchischer Strukturen:**
  - Höhere Ebenen (Systemkomponenten) werden einfacher, weil sie Dienste der tieferen Ebenen (Systemkomponenten) benutzen können.
  - Änderungen auf höheren Ebenen haben keinen Einfluss auf tiefere Ebenen.
  - Höhere Ebenen können abgetrennt werden, tiefere Ebenen bleiben trotzdem funktionsfähig (wiederverwendbar).
  - Tiefere Ebenen können getestet werden, bevor die höheren Ebenen lauffähig sind.
- **Jede Hierarchieebene kann als abstrakte oder virtuelle Maschine aufgefasst werden.**
  - Programme der Schicht *i* benutzen die Programme der Schicht *i*–1 als abstrakte Maschine.
  - Abstrakte Maschine der Schicht *i* dient wiederum als Basismaschine für die Implementierung der abstrakten Maschine der Schicht *i*+1.

Vorteile der Schichtenbildung (2)

- **Eine abstrakte Maschine entsteht aus der Basismaschine durch Abstraktion.**
  - Einige Eigenschaften der Basismaschine werden verborgen.
  - Zusätzliche Fähigkeiten werden durch Implementierung höherer Operationen für die abstrakte Maschine bereitgestellt.

DBVS als Beispiel für Schichtenarchitektur

- **Motivation**
  - Bisher bekannt:
    - Ein DBMS ist ein "normales" Anwendungsprogramm, das auf der Grundlage eines Betriebssystems abläuft.
  - Aber:
    - Ein DBMS ist ein großes und komplexes System, das strukturiert aufgebaut sein muss.
- **Folgerung**
  - DBMS ist *das* Beispiel eines schichtweisen Systementwurfs.
- **Fokus**
  - Die richtigen Schnittstellen (Abstraktionen) finden!
  - Mehrere alternative Techniken zur Realisierung eines Dienstes in einer Schicht
  - Zusammenspiel mehrerer Schichten

1.2 Datenbanktechnologie

Konzepte, Methoden, Werkzeuge und Systeme für die

- **dauerhafte**, Lebensdauer Daten > Dauer Erzeugungsprozess
- **zuverlässige**, Integrität, Konsistenz, Sicherheit vor Verlust
- **unabhängige** wechselseitige Änderungsimmunität AP ↔ DBS

Verwaltung und

- **komfortable**, "höhere" abstrakte Schnittstelle
- **flexible** Ad-hoc-Zugriffsmöglichkeit

Benutzung von

- **großen**, Datenvolumen >> Hauptspeicher
- **integrierten**, kontrollierte Redundanz von/für mehrere Anwendungen
- **mehrfach benutzbaren** gleichzeitiger Zugriff

Datenbanken

Einordnung wichtiger DBS-Konzepte

- Datenintegration
- anwendungsorientierte Datenbeschreibung
- Datenunabhängigkeit

} Datenmodelle, Schemata, Sichten

- Konsistenzkontrolle
- Mehrbenutzerbetrieb
- Datenintegrität
- Wiederanlauf

} Transaktionen

- Zugriffskontrolle
- Programmankopplung

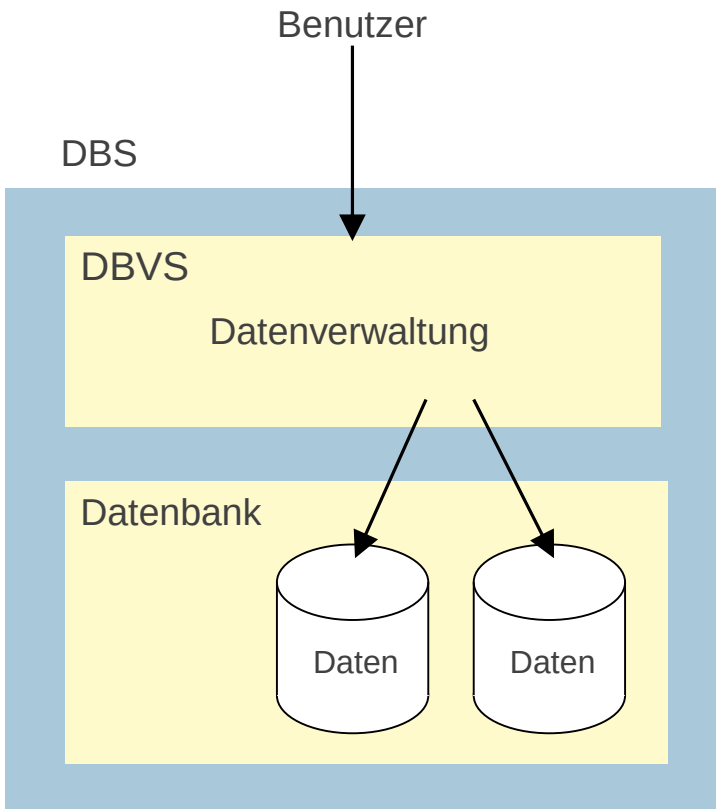
} Datenschutz, Programmiersprachen-einbettung

- Ad-hoc-Anfragen
- Zugriffspfade
- Speicherungsstrukturen
- Verteilung

} Leistungsaspekte, Verfügbarkeit

Datenbanksystem (DBS)

- **Begriff**
  - System zur Beschreibung, Speicherung und Wiedergewinnung von umfangreichen Datenmengen, die von mehreren Anwendungs-programmen benutzt werden
- **Komponenten**
  - **Datenbank (DB)**, in der die Daten abgelegt werden
  - **Datenbank-Verwaltungssystem (DBVS)**:
    Software, die die Daten den vorgegebenen Beschreibungen entsprechend abspeichert, auffindet oder weitere Operationen mit den Daten durchführt
    - (Quelle: Informatik-Duden)



Wann ein Datenbanksystem?

- **Großes Datenvolumen**
  - Auch, aber nicht das Entscheidende

- **Daten sollen wiederverwendbar sein.**
  - Speicherung offen für neue Anwendungen
- **Von mehreren Anwendungen gleichzeitig nutzbar, bei hoher Aktualität der Daten**

(Ziel)

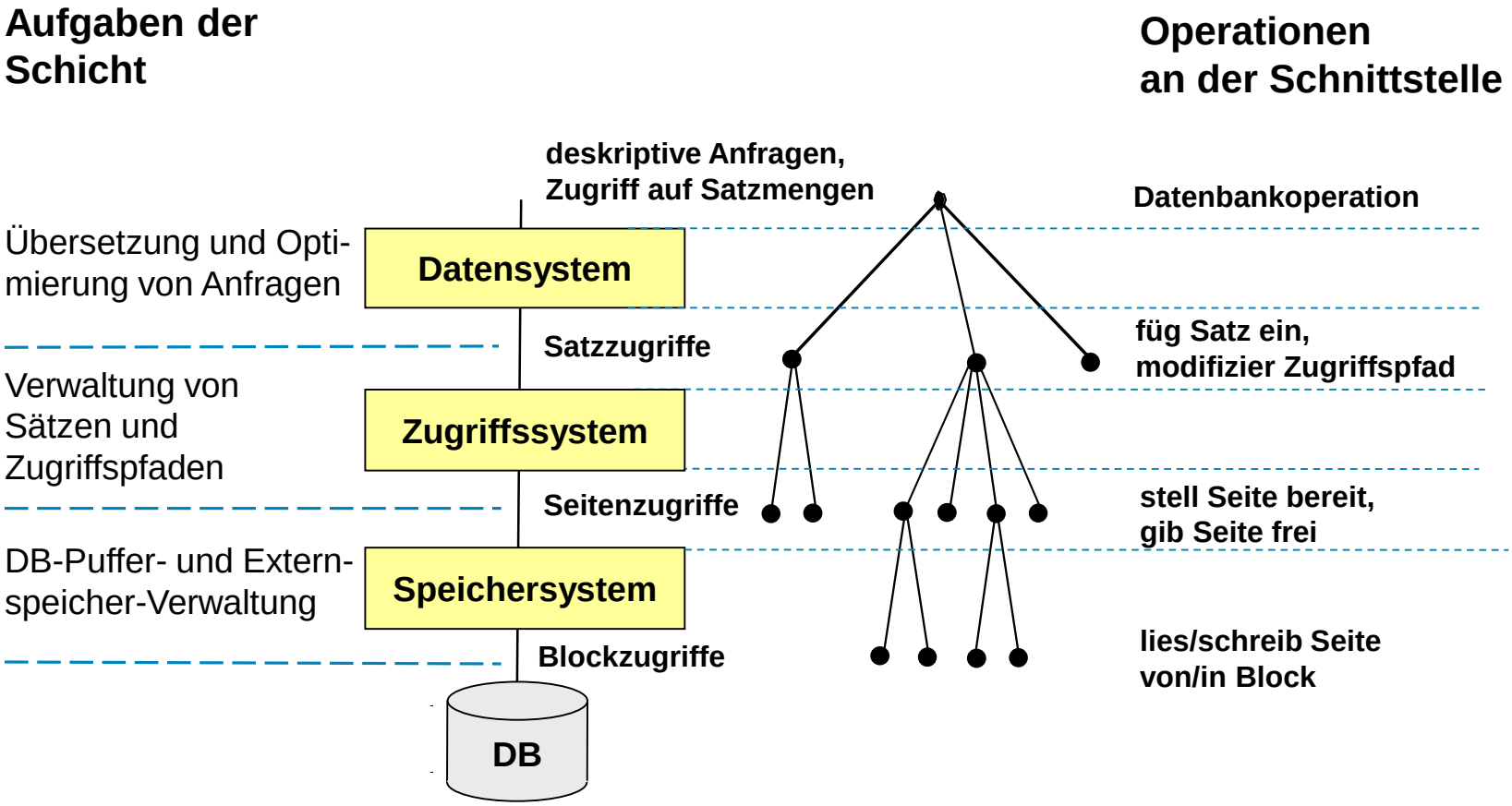
- **Wohlstrukturiert**
- **Redundanzfrei**
- **Flexibel abfragbar (recherchierbar)**
- **Ausfallsicher**

(Konsequenzen)

- **Ach ja, und leistungsfähig, schnell, …**

(Randbedingung)

Schichtenmodell eines DBVS (vereinfacht)



1.3 Zusammenfassung

- **"Schichtologie"**
  - Wesentliches Hilfsmittel zur Konstruktion großer Software-Systeme
  - Eine **Schicht *i*** realisiert für **übergeordnete Schicht *i*+1** einen Dienst unter Inanspruchnahme **darunter liegender Schicht *i*–1**.
- **Schichtenmodell eines Datenbanksystems**
  - Zentrales Hilfsmittel
    - zur Strukturierung
      und auch Visualisierung
      der vielfältigen Aufgaben eines Datenbanksystems und
    - zum generischen Systementwurf
      - Anwendungsspezifika (Schema, Anfragen)
        werden erst zur Laufzeit dem System bekanntgegeben