

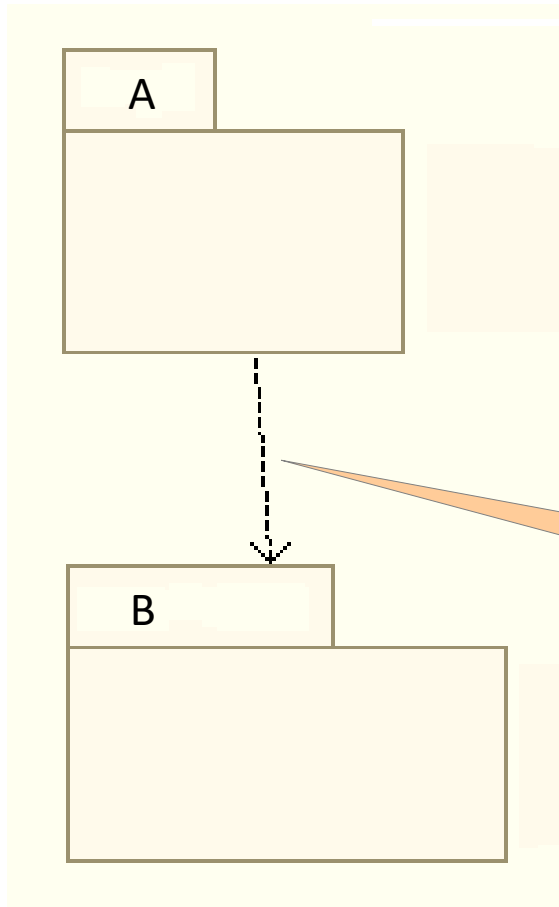


# Architektur und Entwurfsmuster

- ein erster Einblick -

Prof. Dr. Annika Wagner

# Abhängigkeit (aus Java) im Paketdiagramm



Paket A hängt von  
Paket B ab, weil  
mindestens eine Klasse  
in A mindestens eine  
Klasse in B importiert

Bitte nicht mit `<<import>>`, `<<access>>` etc.  
annotieren. Es sei denn Sie wissen, genau  
was Sie tun...

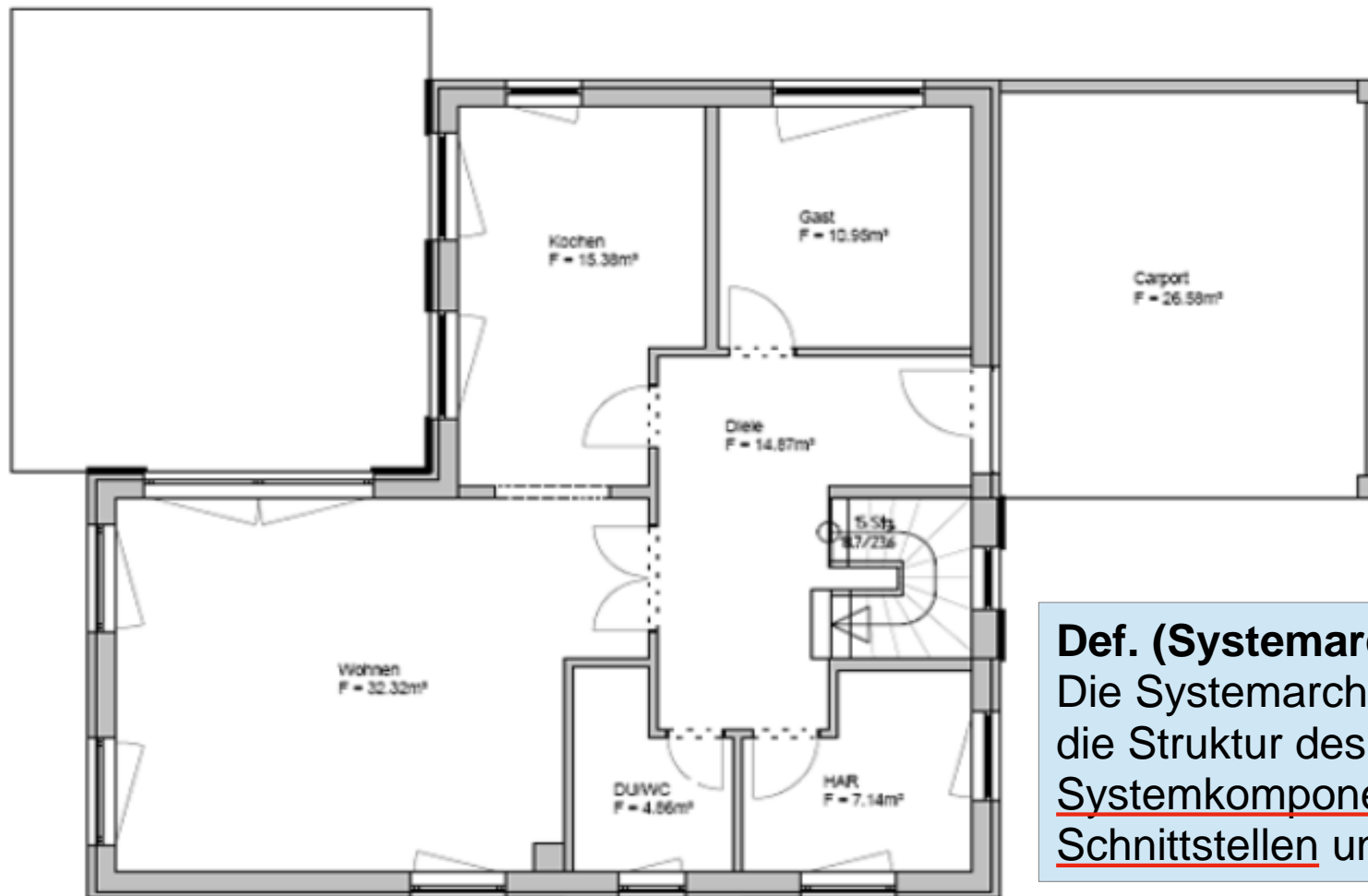


Gruppenarbeit:

Dokumentieren Sie für jede der 4 vorgesehenen  
Lösungen aus der zweiten Programmieraufgabe  
die Abhängigkeiten durch ein Paketdiagramm.

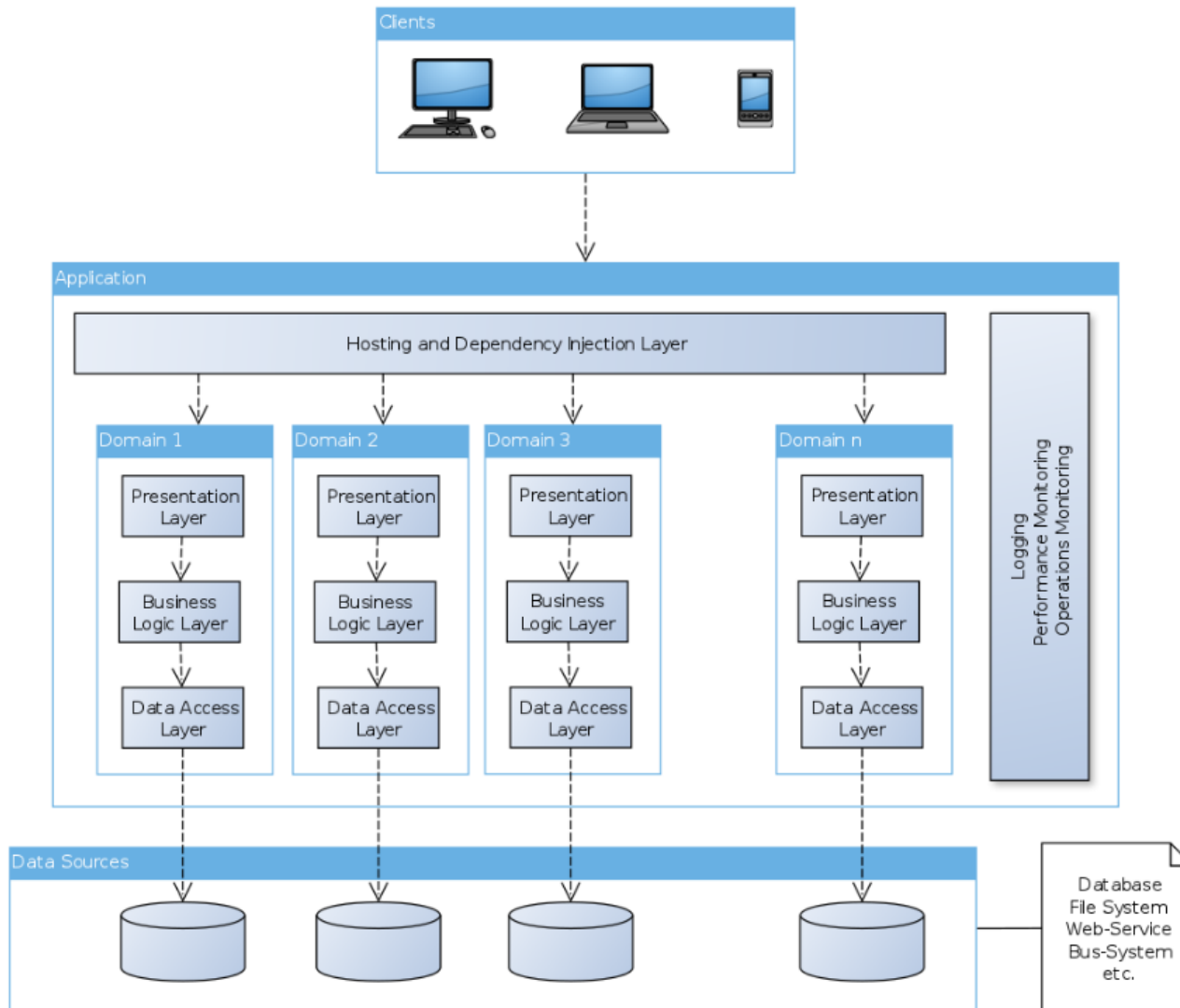
Vergleichen Sie die Ergebnisse!

# Analogie zw. Grundriss und Systemarchitektur



**Def. (Systemarchitektur)**  
Die Systemarchitektur beschreibt die Struktur des Systems durch Systemkomponenten und ihre Schnittstellen untereinander.

# Schichtenarchitektur in Anwendungssystemen



## Entwurfsziel:

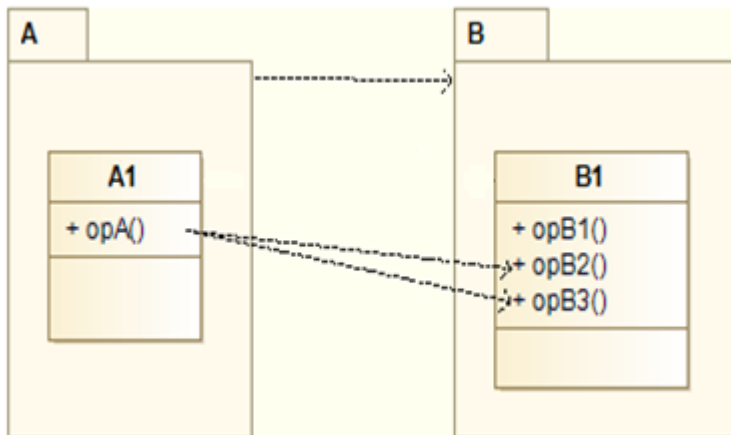
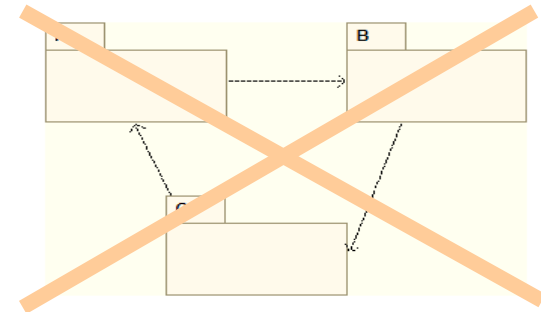
Auswirkungen von Änderungen oder Fehlern auf andere Teile des Systems möglichst gering halten, um Integration und Test zu vereinfachen

## Entwurfsprinzip:

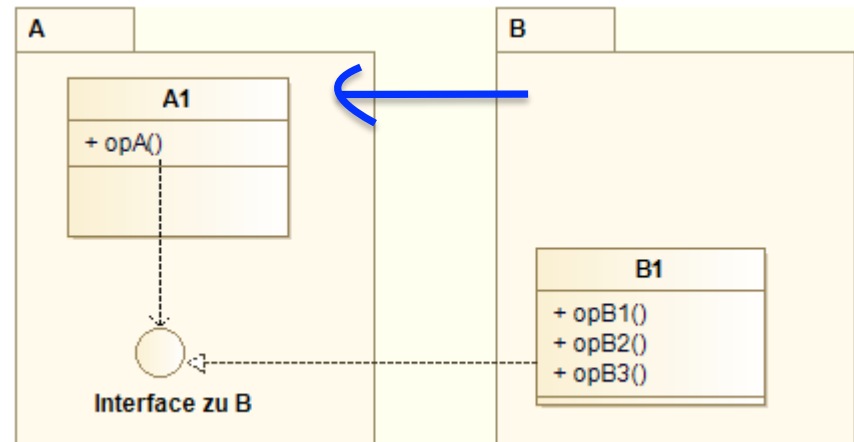
Keine zirkulären Abhängigkeiten

## Entwurfsmittel:

Umkehrung der Abhängigkeiten



durch die Wand



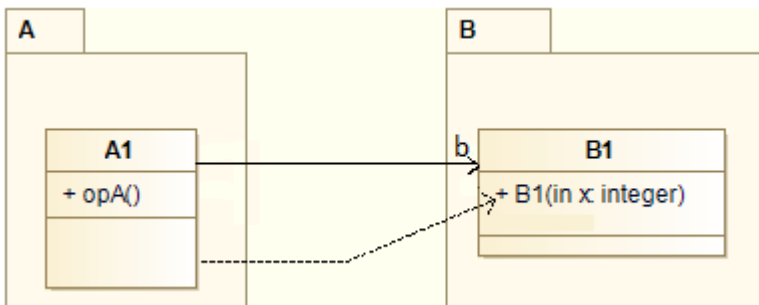
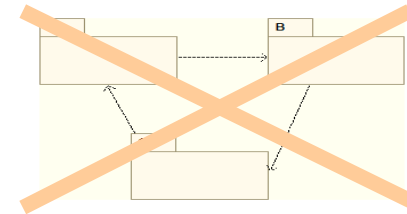
durch die Tür

## Entwurfsziel:

Auswirkungen von Änderungen oder Fehlern auf andere Teile des Systems möglichst gering halten, um Integration und Test zu vereinfachen

## Entwurfsprinzip:

Keine zirkulären Abhängigkeiten



```
public class A1
{
    private B1 b;

    public void opA()
    {
        b = new B1(x);
    }
}
```

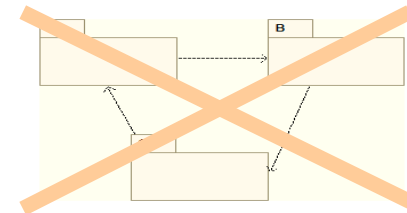
Konstruktoraufruf  
nicht in Interface  
auslagerbar!

## Entwurfsziel:

Auswirkungen von Änderungen oder Fehlern auf andere Teile des Systems möglichst gering halten, um Integration und Test zu vereinfachen

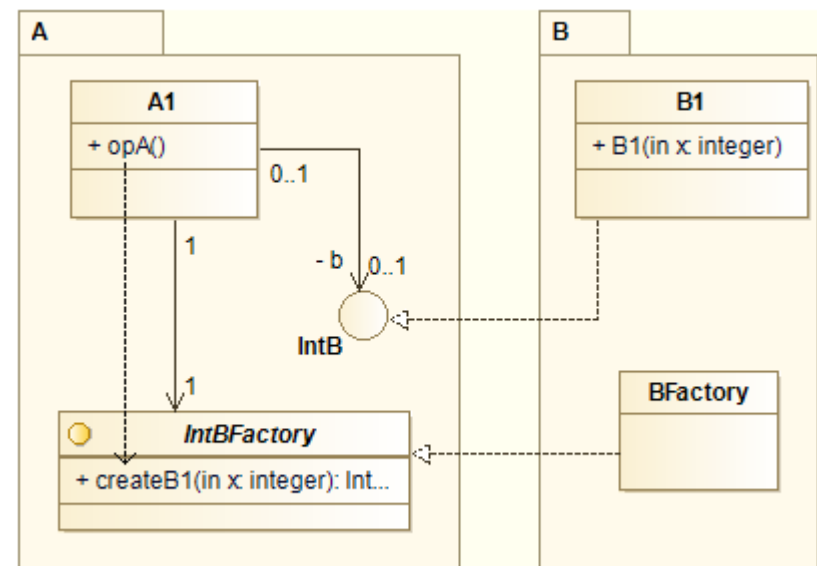
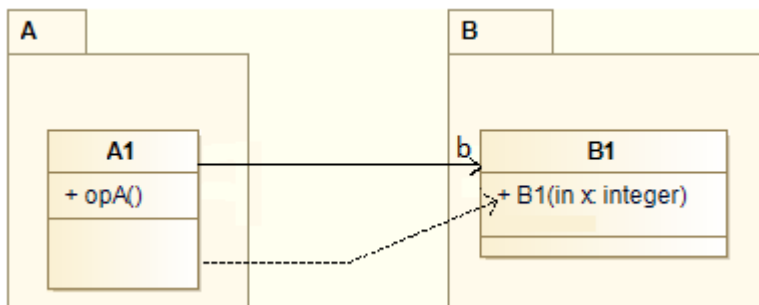
## Entwurfsprinzip:

Keine zirkulären Abhängigkeiten



## Entwurfsmittel:

Umkehrung der Abhängigkeiten mit dem „Fabrik“-Muster





# Anwendung bei Nutzung Hibernate Framework



```
import org.hibernate.*;  
  
public class ManageEmployee {  
    private static SessionFactory factory;  
    public static void main(String[] args) {  
  
        try {  
            factory = new Configuration().configure().buildSessionFactory();  
        } catch (Throwable ex) {  
              
        }  
  
        ManageEmployee ME = new ManageEmployee();  
        Integer empID1 = ME.addEmployee("Zara", "Ali", 1000);  
    }  
  
    /* Method to CREATE an employee in the database */  
    public Integer addEmployee(String fname, String lname, int salary){  
        Session session = factory.openSession();  
        Transaction tx = null;  
        Integer employeeID = null;  
  
        try {  
            tx = session.beginTransaction();  
            Employee employee = new Employee(  
                employeeID = (Integer) session.save(employee);  
                tx.commit();  
            }  
        }  
    }
```

Einmalige (aufwändige) JDBC  
Konfiguration des DB Zugriffs...

... entkoppelt von der häufig  
anfallenden Erzeugung von  
Sessions

Entkopplung schwergewichtiger Aufgaben, die selten durchgeführt werden, von leichtgewichtigen Routineaufgaben

## Entwurfsmittel:

## mit dem „Fabrik“ - Muster



# Was ist ein Entwurfsmuster?



## **Def. (Design Pattern od. Entwurfsmuster)**

Design Pattern sind wiederkehrende Gruppen von Konzepten, die im Rahmen des objektorientierten Entwurfs zielgerichtet gemeinsam verwendet werden, um gleichartige Probleme zu lösen.

Zu ihrer Darstellung wird ein einheitliches Template verwendet.

**Name:** <Durch den Namen des Patterns ersetzen>

**Alternative Bezeichnungen:** <Andere für dieses Pattern bekannte Bezeichnungen inkl. Quelle angeben>

**Anwendbarkeit bzw. Intension:** <Sehr kurze, abstrakte Darstellung des vom Pattern gelösten Problems>

**Motivation:** <Ein Beispiel, das die Wirkung des Patterns illustriert.>

**Lösung:** <Eine abstrakte Beschreibung der Lösung – unabhängig von einem Beispiel.>

**Konsequenzen:** <Bekannte Vor- und Nachteile der Lösung.>

Hausaufgabe: Mit Hilfe des Lehrbuches  
und der Übungsergebnisse ausfüllen für Fabrik(methode)

## Weitere (Arten von) Entwurfsmuster(n)

- Erzeugungsmuster

z.B. Singleton Pattern oder wie man sicher stellt, dass nur eine Instanziierung erfolgt

- Strukturmuster

z.B. Composite Pattern oder wie man eine beliebig tiefe hierarchische Struktur umsetzt

- Verhaltensmuster

z.B. State Pattern oder wie man ein Zustandsdiagramm implementiert

Sprengt den Rahmen ... kein Software Engineering