

Entwurf

Rainer Schmidberger

Rainer.Schmidberger@informatik.uni-stuttgart.de

Zweck des Entwurfs

□ Aus Sicht der funktionalen Anforderungen ist der Entwurf eines Systems beliebig wählbar

⇒ Überspitztes Beispiel: Wenn eine Klasse mit einer Methode, die 10.000 Zeilen lang ist, die geforderte Funktion erfüllt, ist das aus Sicht der funktionalen Anforderungen in Ordnung.

aber:

- ⇒ Lesbarkeit
- ⇒ Wiederverwendungsmöglichkeit
- ⇒ Projektorganisation
- ⇒ ..

sind deutlich erschwert. Diese (i.d.R nichtfunktionalen) Anforderungen beeinflussen den Entwurf.

Was leistet der Entwurf?

- ❑ Legt die prinzipielle Lösungsstruktur fest
- ❑ Gliedert in überschaubare Einheiten
- ❑ Integriert bereits bestehender Module
- ❑ Liefert wiederverwendbarer Module

Definitionen

- ❑ Entwurf ist eine **Aktivität**, bei der eine technische Lösungsstruktur für ein System entwickelt wird
- ❑ **Entwurfsdokument**: Resultat des Entwurfs. Kann als Text, UML und formlose Skizzen beschrieben werden
- ❑ Zentrale Merkmale des Entwurfs werden auch als **Architektur** bezeichnet.

- ❑ Modul = Komponente = Subsystem
- ❑ Eine Modul definiert sich durch seine festgelegte Zuständigkeit

Was steht im Entwurfsdokument

- Verweis auf Standards, z.B. Referenzarchitekturen wie
 - ⇒ J2EE
 - ⇒ Model-View-Controller Prinzipien
 - ⇒ usw.
- Beschreibung des grundlegenden Aufbaus des Systems
 - ⇒ Paketaufteilung
 - ⇒ Komponenten mit Schnittstellen
 - ⇒ Art der Kommunikation zwischen den Komponenten
 - ⇒ Grundlegende Datenstrukturen (z.B. Datenbankmodelle, xml-Schemata)
- Kurzum: Alles Prinzipielle der technischen Lösung
- Was steht nicht im Entwurfsdokument: Projektplanung, Aufwand, Implementierungsdetails

Datenflüsse beim Entwurf

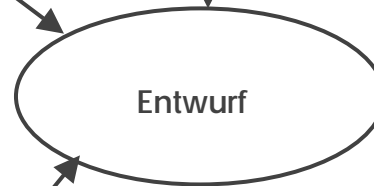
Spezifikation

- Funktionale Anforderungen
 - Architekturrelevante Use Cases
 - Erkennbare domänenspezifische Komponenten

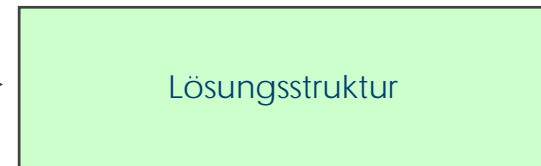
- Nicht-Funktionale Anforderungen
 - Vorweggenommene Entwurfsentscheidungen (z.B. schnittstellen)
 - Nicht Use-case spezifische Anforderungen wie z.B. Wartbarkeit, Wiederverwendbarkeit, Laufzeit, Robustheit, Verfügbarkeit, Speicherbedarf, usw

Standards

- Best Practices, Erfahrungen aus vorhergehenden Projekten
- Design Patterns
- Domänenspezifische Standardarchitektur
- Unternehmensstandards

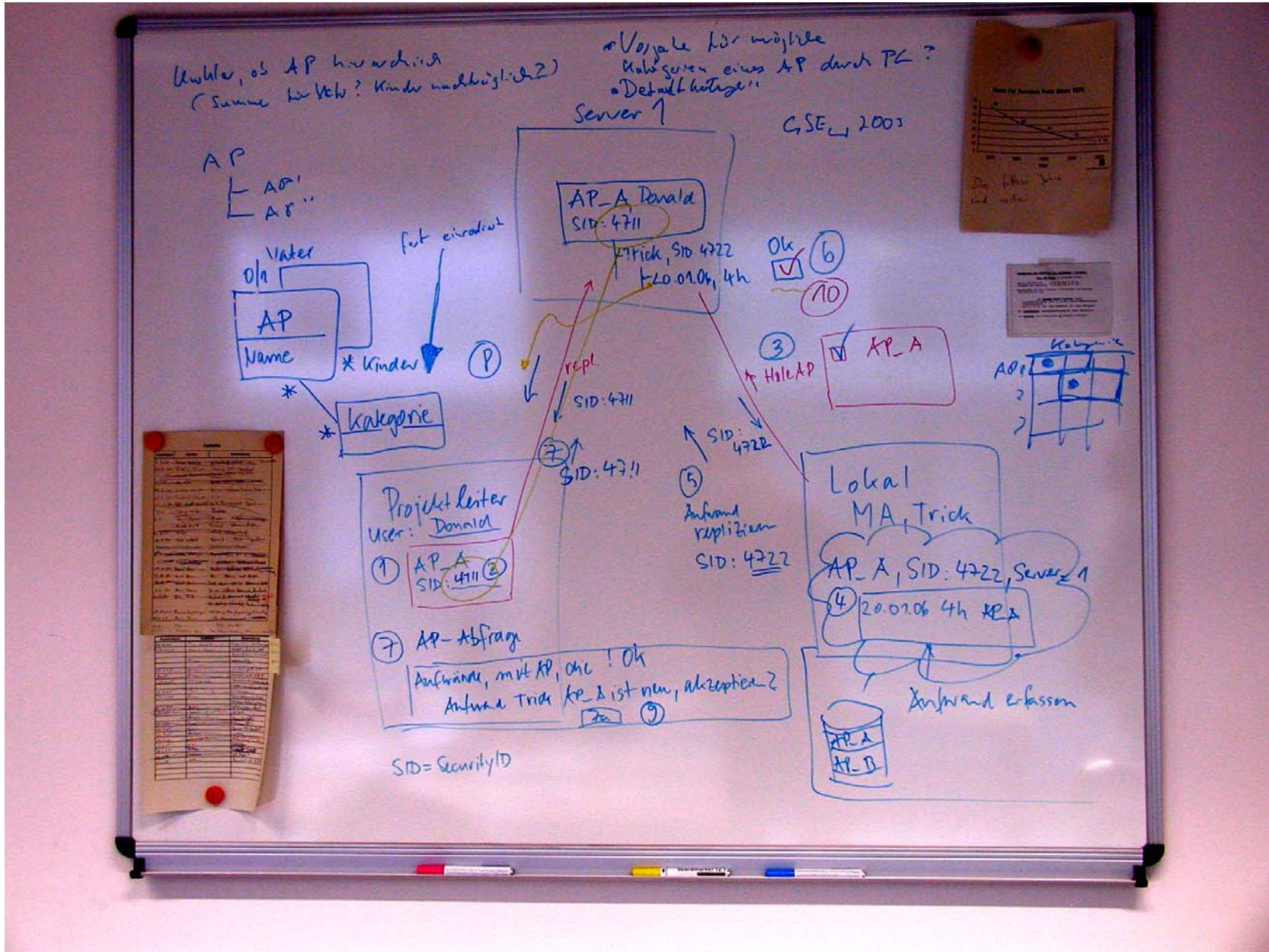


Entwurfsdokument

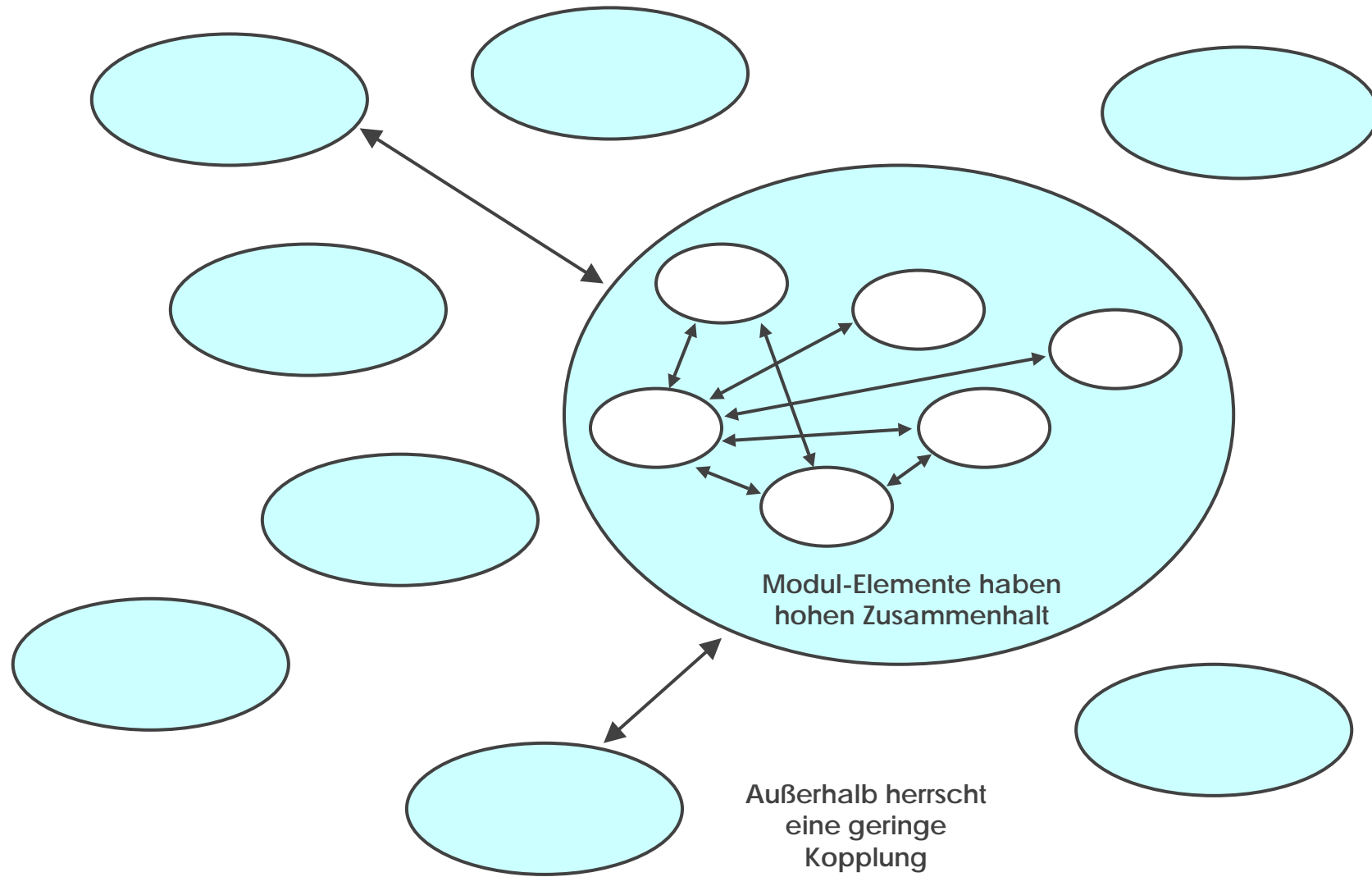


Strategische Entwurfsentscheidungen

- Einsatz von Bibliotheken
- Einsatz von Infrastruktur (z.B. Datenbanken oder Middleware, ...)
- Integration von Altsystemen (Legacy Systems)
- Distributionsanforderungen



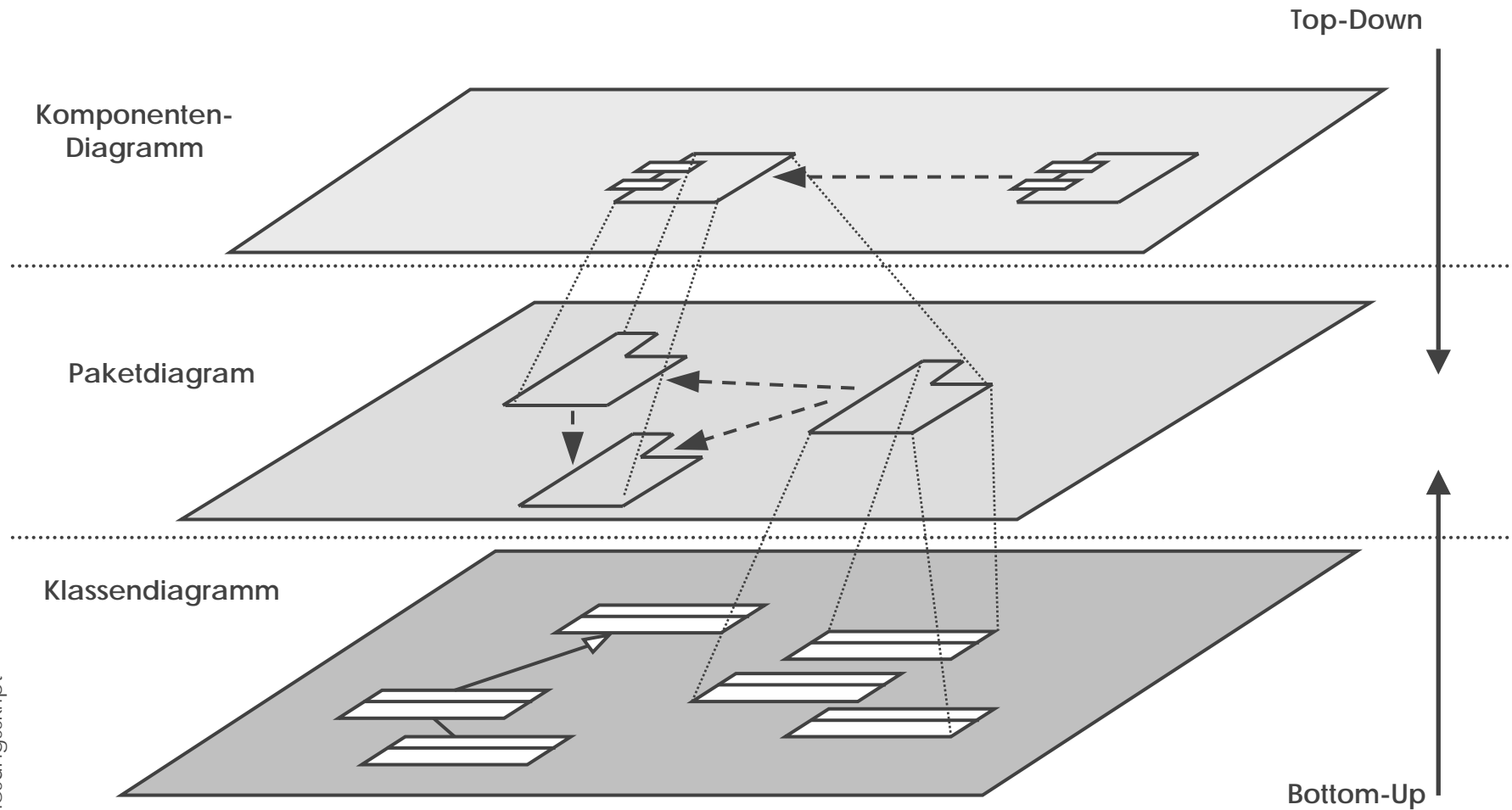
Kopplung und Zusammenhalt



Entwurfsbeschreibung in der UML (1)

08.03.2007 © 2004-2006, Rainer Schmidberger, ISTE

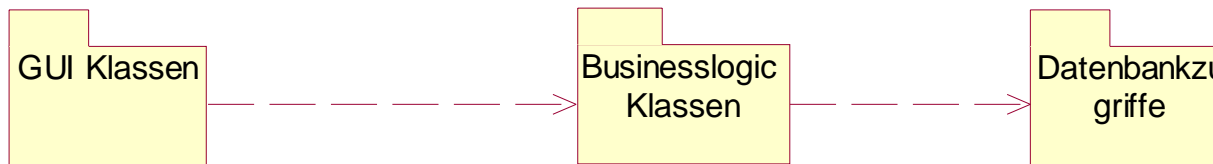
Programmentwicklung,
Vorlesungsskript



Entwurfsbeschreibung in der UML (2)

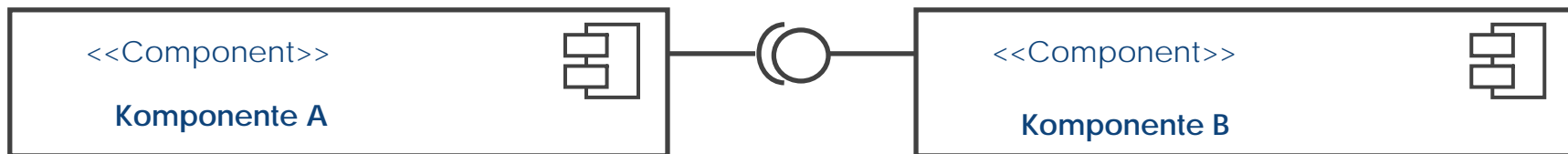
□ Pakete

- ⇒ Gruppierung von Klassen (und weiteren Paketen)
- ⇒ Liefern keine gemeinsame Funktion, sondern sind mehr „Werkzeugkasten“. Beispiel: java.util



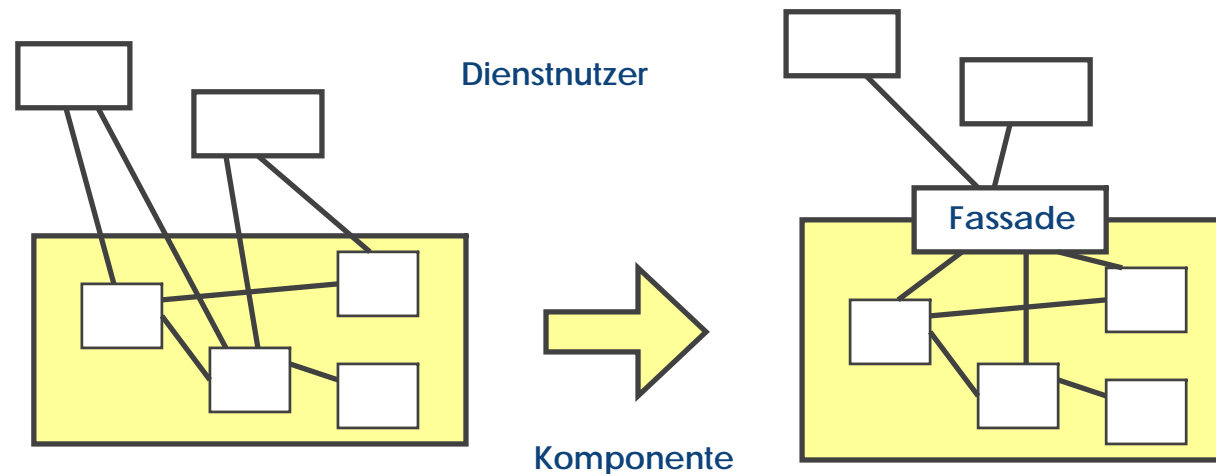
□ Komponenten

- ⇒ Werden durch die außen verfügbare Funktion definiert
- ⇒ Nicht „Werkzeugkasten“, sondern „Gerät“
- ⇒ Nutzen die Klassen verschiedener Pakete



Komponenten

- ❑ Werden durch (ein oder mehrere) Klassen implementiert
- ❑ Schnittstellen der Komponenten sind Klassen oder Interfaces
- ❑ Möglichst wenige der in einer Komponente enthaltenen Klassen sollten außerhalb der Komponente sichtbar sein.
- ❑ Öffentlichen Schnittstellen durch eine „Fassade“ von der internen Implementierung entkoppeln.



Vorgehen beim Entwurf

1. Anforderungen z.B. in Form von Use Cases nach Wichtigkeit („Business value“) ordnen. Wichtiges (aus Kundensicht) zuerst!
2. Standard-Entwurf prüfen
3. Falls kein Standard-Entwurf gefunden wird:
 - ⇒ Zuständigkeiten auf Module aufteilen
 - ⇒ Kopplung soll klein sein: am besten sind autarke Module
 - ⇒ Zusammenhalt soll groß sein: die Bestandteile eines Moduls sollen zusammengehörende Dinge tun
4. Angemessenheit des Entwurfs prüfen (keep it simple!)
5. Durchgängigkeit des Entwurfs prüfen
6. Realisierbarkeit z.B. durch einen Prototypen prüfen
7. Wiederholen, bis alle Anforderungen umgesetzt sind
8. Unwichtige Anforderungen, die sich nicht in den Entwurf integrieren lassen, werden in Rücksprache mit dem Kunden verworfen

Abschließende Tipps

- ❑ Natürliche Vorgehensweise: Top-Down, Dekomposition
- ❑ Es existieren bereits Klassen, die integriert werden sollen: Bottom-Up
- ❑ Conway's Law: „Any organization which designs a system (...) will inevitably produce a design whose structure is a copy of the organization's communication structure.“, Melvin E. Conway, 1968
- ❑ Keep it simple!! Angemessenheit!
- ❑ Externe Systeme möglichst „Einpacken“
- ❑ Fassaden nutzen
- ❑ Zum Entwurfsdokument: Alles, was man einem Außenstehenden zum prinzipiellen Aufbau eines Systems erklären würde, steht im Entwurfsdokument.