Vorlesung
Component Ware und Web-Services

1. Einführung

Dr. Hans-Gert Gräbe, F. Schumacher
Wintersemester 2003/2004

Vorlesungen am AIS-Lehrstuhl im WS 03/04

- Software-Technik
  - 3. Semester Grundstudium
  - 2 SWS + Übung
  - freitags, 11:15–12:45, HS 11
- eBusiness II
  - Kernfach: Praktische oder Angewandte Informatik
  - donnerstags, 13:15–14:45, HS 21
- Anwendungen mit XML
  - Kernfach: Praktische oder Angewandte Informatik
  - donnerstags, 15:15–16:45, HS 4

Vorlesungen am AIS-Lehrstuhl im WS 03/04

- Component Ware und Web-Services
  - Kernfach: Praktische oder Angewandte Informatik
  - montags, 17:15–18:45, HS 22
- Algorithmen für Zahlen und Primzahlen
  - Kernfach: Theoretische oder Angewandte Informatik
  - dienstags, 15:15–16:45, SG 3-11
- Algebraische Komplexitätstheorie
  - Schwerpunkt: Theoretische oder Angewandte Informatik
  - donnerstags 13:15–14:45, HS 4
- Konstruktive Invariantentheorie
  - Schwerpunkt: Theoretische oder Angewandte Informatik
  - montags 09:15–10:45, HS 1

Seminare am AIS-Lehrstuhl im WS 03/04

- Forschungsseminar: Anwendungsspezifische Informationssysteme
  - Für Diplomanden und Promovenden
- Internet Dienstleistungen
  - Praktische, Angewandte oder Versicherungsinformatik
  - mittwochs, 13:15–14:45, HG R 1-25
- Prozess-orientierte Content-Nutzung
  - Praktische, Angewandte oder Versicherungsinformatik
  - dienstags, 15:15–16:45, SG 3-07/HG R 1-25
- Integration- und Service- Engineering
  - Praktische, Angewandte oder Versicherungsinformatik
  - mittwochs, 15:15–16:45, HG R 1-25

Praktika und Diplomarbeiten am AIS-Lehrstuhl

- Praktika
  - Bewerbungsbogen unter http://ais.informatik.uni-leipzig.de/
  - Ansprechpartner:
    - Fr. Pfretzschner <pfretzschner@informatik.uni-leipzig.de>
    - Sprechzeiten: nach Vereinbarung
- Diplomarbeiten
  - Bewerbungsbogen unter http://ais.informatik.uni-leipzig.de/
  - Ansprechpartner:
    - Hr. Thränert <thraenert@informatik.uni-leipzig.de>
    - Sprechzeiten: nach Vereinbarung

Internet Anwendungen unter z/OS (OS/390)

Dr. rer. nat. Paul Herrmannn, Prof. Dr.-Ing. Wilhelm G. Spruth

Freitags, 11:15-12:45 Uhr, im Hörsaal 22

erstmalig am Freitag, dem 24. Oktober 2003.

Alle größeren Unternehmen investieren derzeitig in neuartige Client/Server, Multimedia und e-Business Anwendungen. Diese Anwendungen kommunizieren über das Internet mit den existierenden zentralen Unternehmensservern.

Hierbei setzen 90 von 100 Großunternehmen das z/OS (früher OS/390) Betriebssystem auf ihrem zentralen Server ein.

Im Jahr 2000 war die Anzahl der OS/390 CICS Transaktionen größer als die weltweite Zahl der Zugriffe auf das World Wide Web.

Die Vorlesung "Internet Anwendungen unter z/OS" erläutert praxisnahe Systemstrukturen, die derzeitig für neuartige und leistungsfähige Internet-Anwendungen entwickelt werden.

Die Vorlesung wird durch praktische Übungen ergänzt. Hierfür steht uns ein eigener OS/390 Rechner `jedi.informatik.uni-leipzig.de` zur Verfügung.

Die Vorlesung kann als Wahlfach innerhalb der praktischen Informatik mit 2 SWS anerkannt werden. Die Übungen können mit 4 SWS anerkannt werden.

Organisatorisches

- Termine, montags 17:15–18:45 Uhr, HS 22
- Abschluss: Klausur
- Ansprechpartner
  - Dr. Gräbe <graebe@informatik.uni-leipzig.de>
  - Hr. Schumacher <schumacher@informatik.uni-leipzig.de>
  - Sprechzeiten: nach Vereinbarung per Mail
- Informationen
  - http://ais.informatik.uni-leipzig.de/

Inhalt dieser Vorlesung

1. Zielsetzung der komponentenbasierten Software-Entwicklung
   - Componentware
   - Der Komponentenbegriff
   - Granularität von Komponenten

2. Designprinzipien der komponentenbasierten Software-Entwicklung
   - Design for Component
   - Design from Component
   - Design to Component

3. Komponententechnologien
   - Benutzerschnittstelle
   - Geschäftslogik
   - Datenschicht
   - Komponentenplattformen
   - Frameworks

Warum Komponenten?

- Ausgangssituation
  - Monolithische Bauweise der Programme
  - Sich wiederholender Code wird häufig neu geschrieben
- Steigende Anforderungen
  - Qualität und Quantität
  - Zunehmende Vernetzung
- Probleme
  - Realisierungsdauer/Time to Market
  - Kosten
  - Stabilität
- Lösung
  - Softwarebausteine
  - Zusammensetzen der Produkte aus Komponenten

Definition

Definition nach Szyperski:

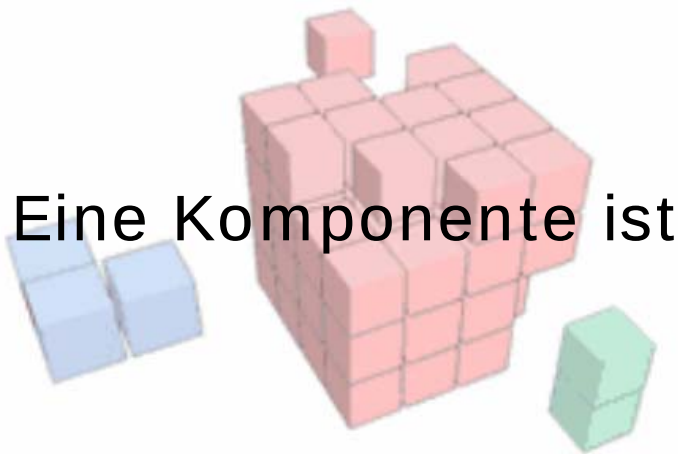
A software component is a **unit of composition** with contractually specified **interfaces** and **explicit context dependencies** only. A software component can be **deployed independently** and is subject to composition by third party.

Eigenschaften

- 4 Haupteigenschaften von Komponenten:

... eine funktional und technisch abgeschlossene Einheit

... unabhängig als Einheit entwickel- und verteilbar



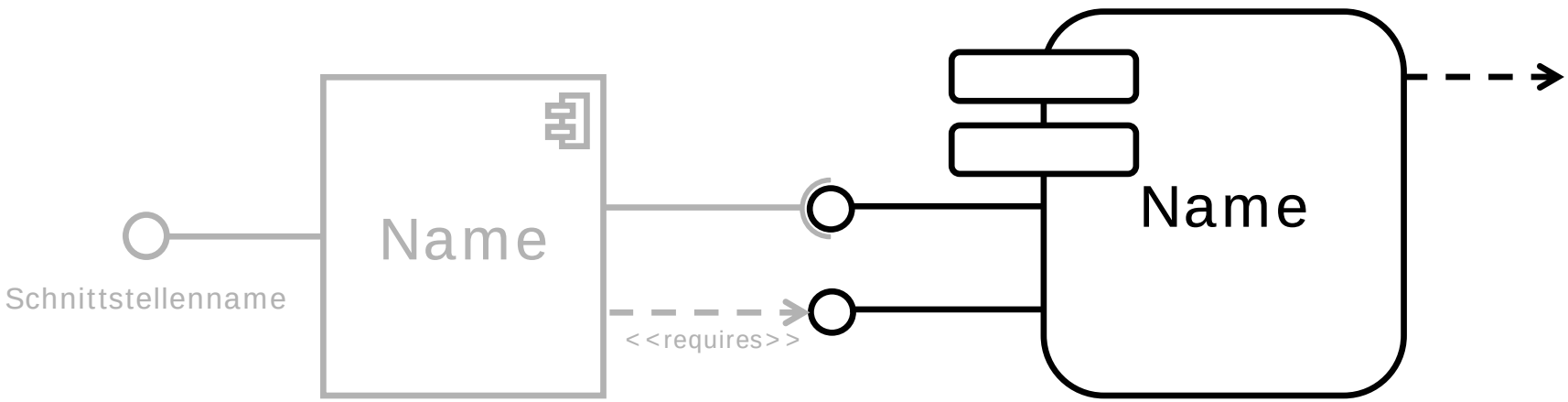
Eine Komponente ist...

... wiederverwendbar

... nur durch genau spezifizierte Schnittstellen ansprechbar

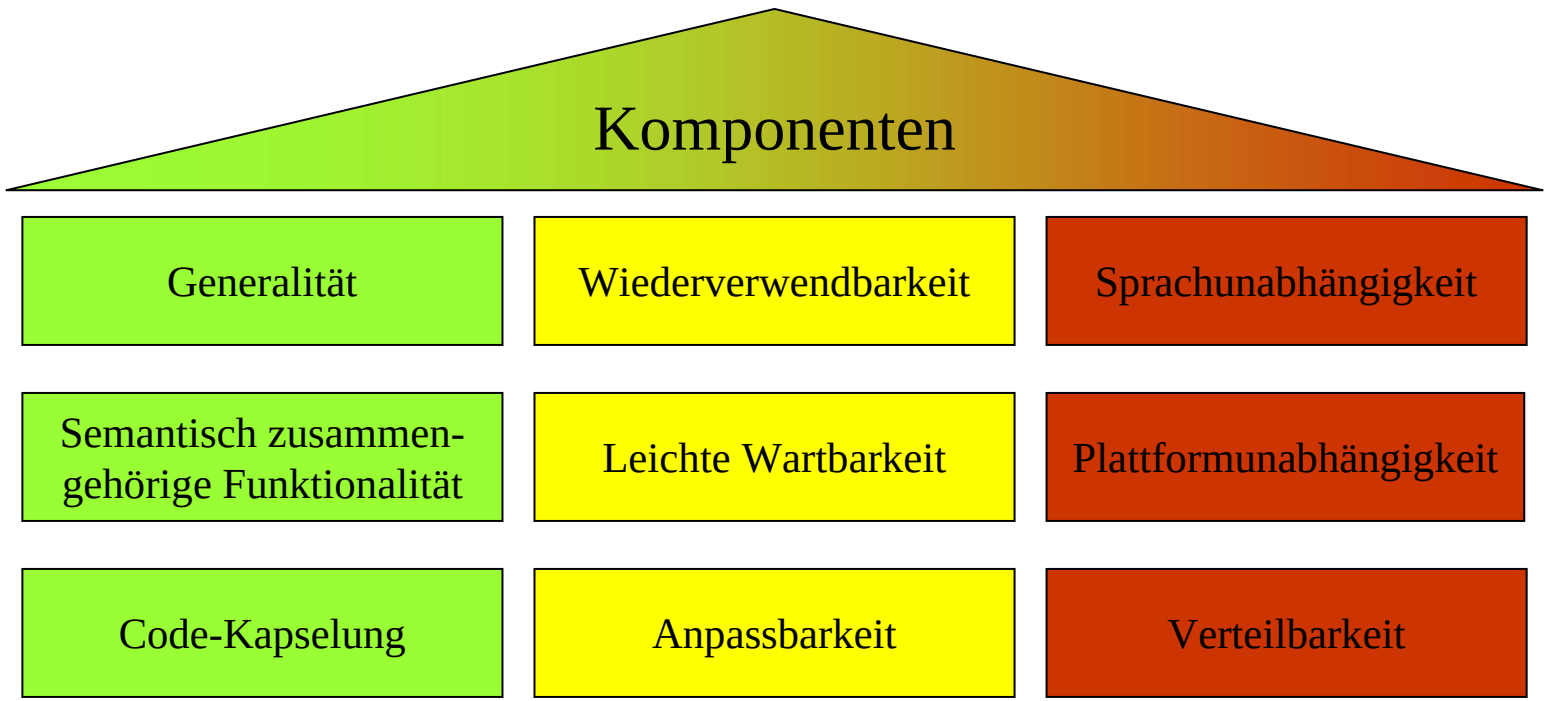
Aufbau einer Komponente

- Schnittstellen
  - Operationen / Funktionalität
  - Attribute / Signatur
  - Veröffentlichte Schnittstellen
  - Konsumierte Schnittstellen



Vorteile

- Kombiniert Top-Down und Bottom-Up Development
- Reduziert die Kosten und Entwicklungszeit



Begriffsdefinition

- Komponenten
  - **Gekapselte**, generalisierte **Softwareobjekte**, die einen **Dienst** zur Verfügung stellen und aus denen größere Komponenten oder Systeme gebaut werden können
- Kapselung
  - wohlspezifizierte Diensteschnittstelle, Kontextunabhängigkeit
- Generalisierung
  - Parametrisierung, Erweiterbarkeit, Nutzbarkeit in unterschiedlichen Anwendungen
- Dienst
  - Zusammenhängende Sammlung von in Beziehung stehender Funktionalität. Nachfrage nach Dienst (service) von Klienten (clients)
- Systemfähigkeit
  - Kaskadierbarkeit, Katalogisierbarkeit, Aufbau nach vorgegebenen Architekturprinzipien

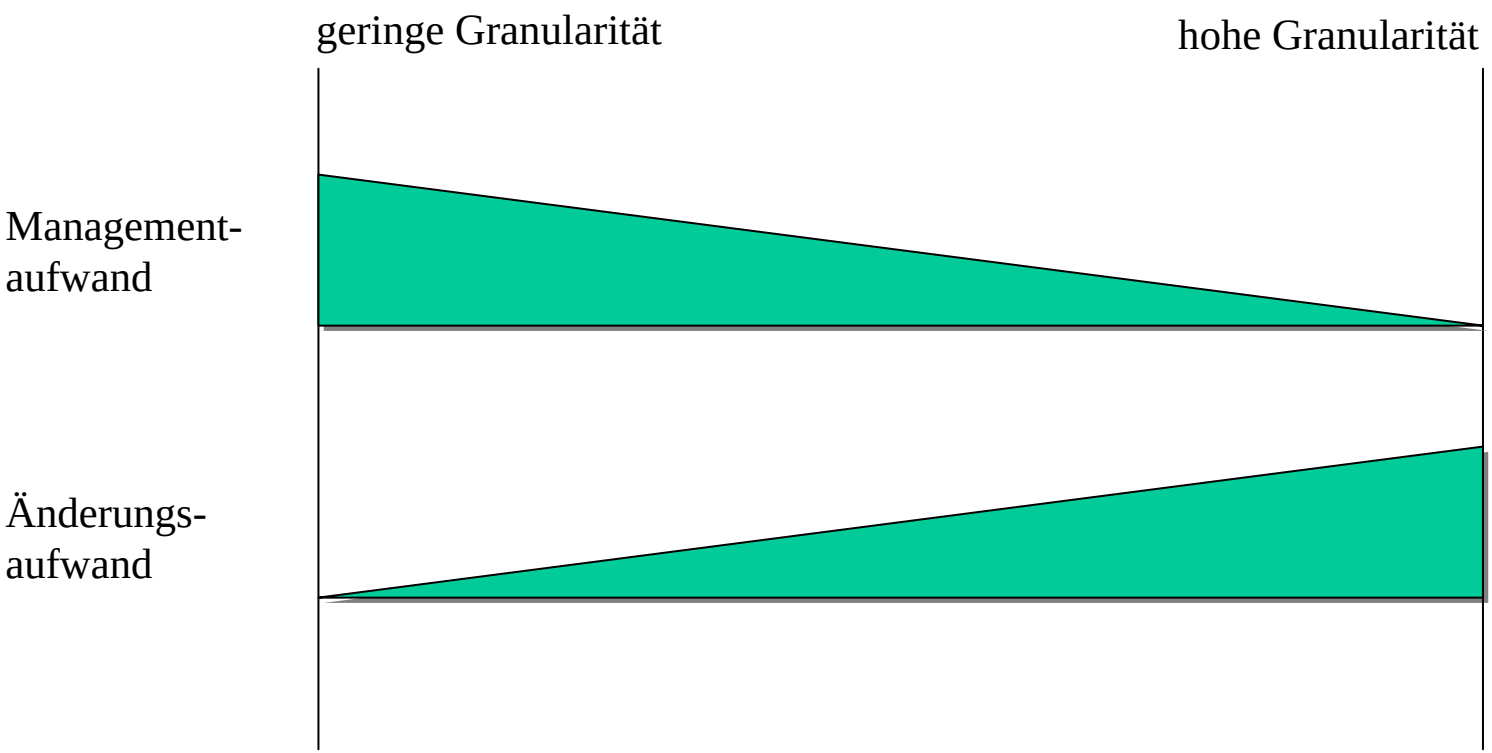
Begriffsdefinition

- Komponenten-Plattform
  - Hilfsmittel und Technologien, die zur Erstellung und zum Betrieb bzw. zur Anpassung flexibler und erweiterbarer Anwendungen auf der Basis von Komponenten erforderlich sind

- Komponenten-Engineering
  - Modelle, Methoden und Werkzeuge, die zur Analyse, Entwicklung und Design von auf Komponentensoftware beruhenden betrieblichen Anwendungssystemen dienen
  - Design to / from / for component

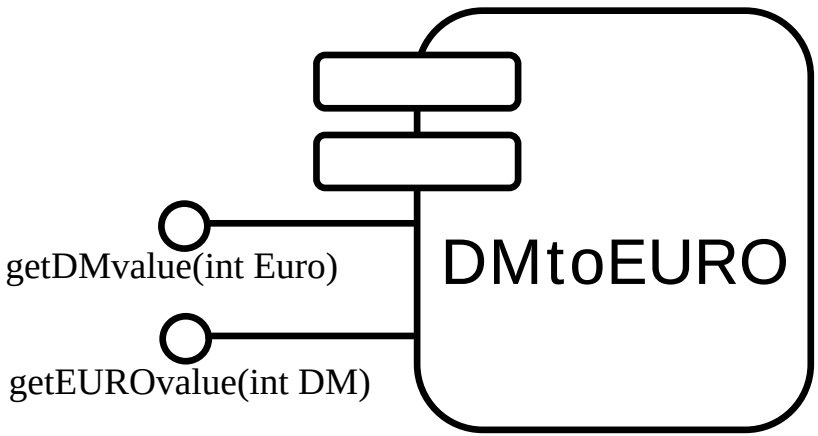
Granularität

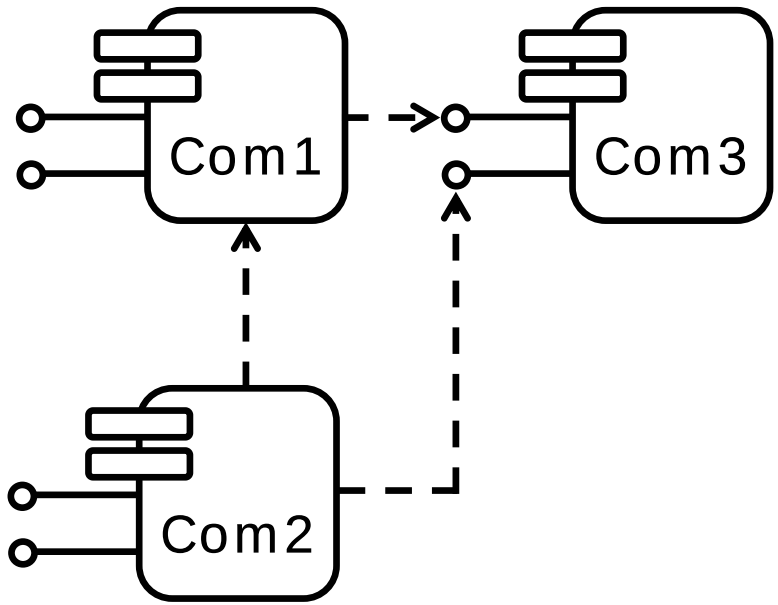
- „Maß“ für Komplexität (Funktionalität, Schnittstellen) der eingesetzten bzw. entwickelten Komponenten
- wichtig für Qualitätsbeurteilung einer Komponente (Erfolg)



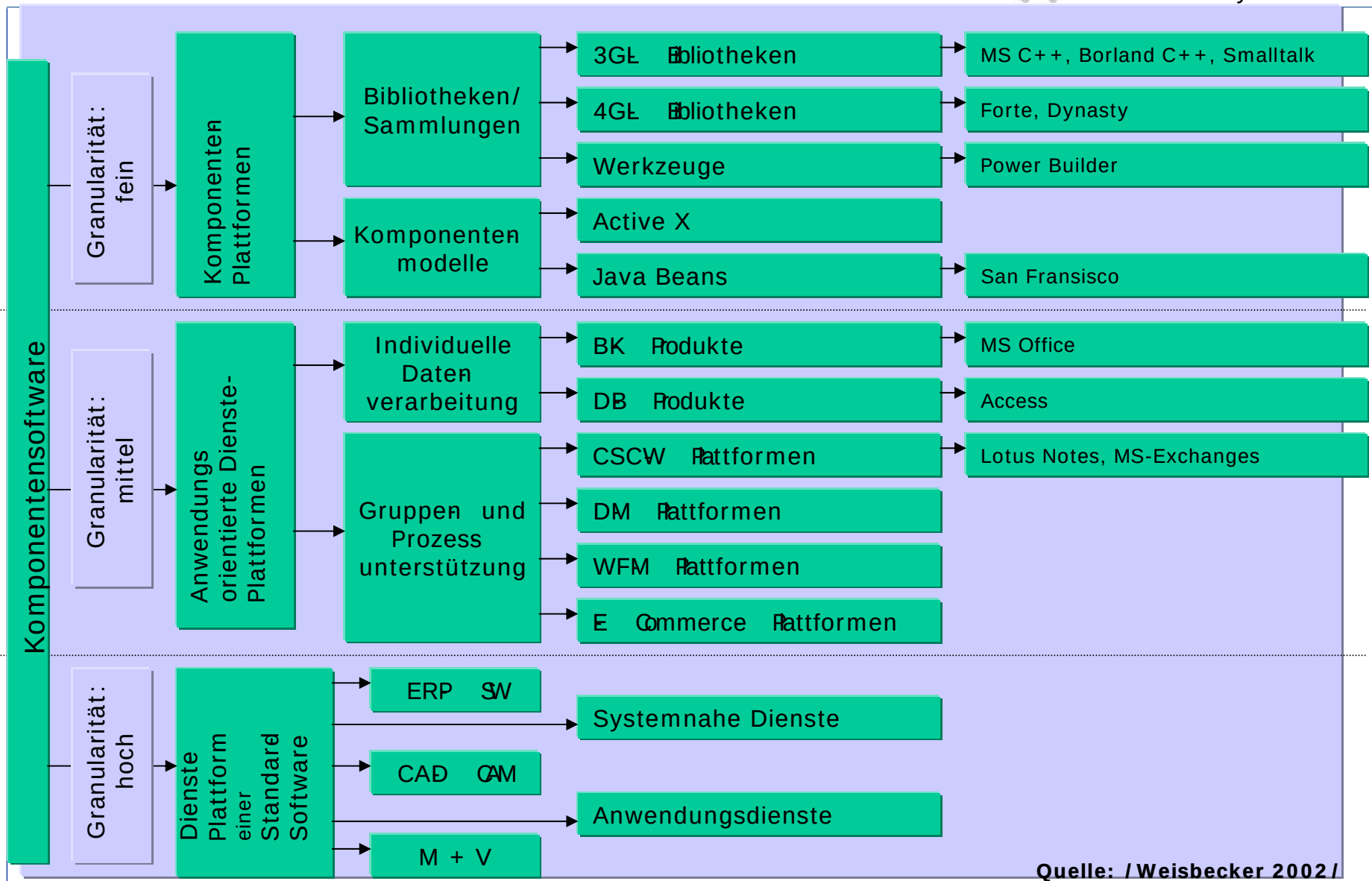
Granularität

- Geringe Granularität
  - Wenige Schnittstellen
  - Simple Funktionalität
- Hohe Granularität
  - Viele Schnittstellen
  - Komplexe Funktionalität





Granulatität



Quelle: /Weisbecker 2002/

Designprinzipien

- Design **for** Component
  - Initiale Entwicklung atomarer Komponenten zum Ziele der Bereitstellung spezifischer, gekapselter Dienst, welche später in neue Anwendungen schnell und einfach integriert werden können.

- Design **from** Component
  - Inkrementelle Entwicklung von komplexeren Komponenten und Anwendungssystemen unter Nutzung vorhandener und noch zu erstellender Bausteine, sowie der Dienst der Componentware-Plattform.

- Design **to** Component
  - Methoden zur Transformation konventionell erstellter Anwendungssysteme in eine flexible komponentenbasierte Umgebung.

Design for Component

- Fokus: Komponente als Endprodukt
- Anbieter / Produzenten-Sicht
- Ausrichtung an:
  - Standards
  - angestrebte Zielumgebung
- Ergebnis:
  - Bereitstellung von Komponenten
  - Erfüllung einer spezifischen Aufgabe

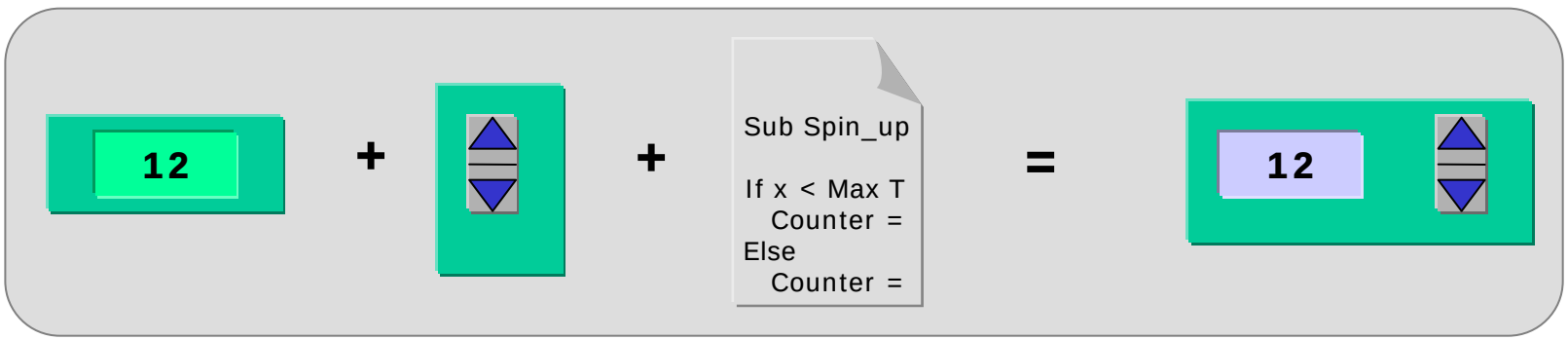


Abb: Zusammenbau einer Komponente aus Teilkomponenten

Design from Component

- Fokus: Zusammenbau komplexer Anwendungssysteme
- Käufer / Anwender-Sicht
- Aggregation von Komponenten auf höheren Niveau
- Grundlagen:
  - Spezialisierte Komponenten
  - "Komponenten-Kleber"
  - Componentware-Plattform-Dienste
- Ergebnis:
  - Aufgabenspezifische Anwendungskomposition

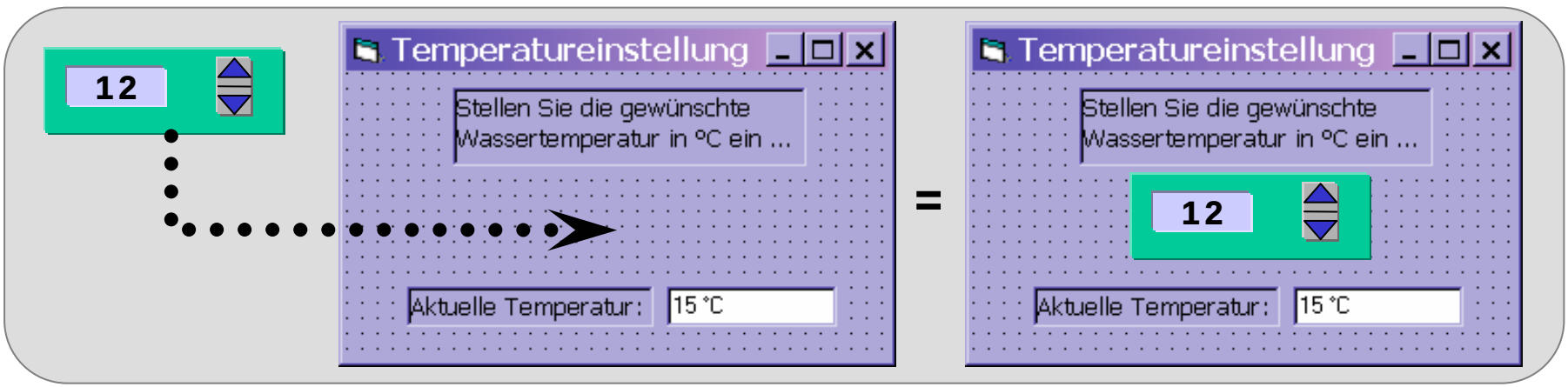


Abb: Verwenden einer Komponente

Design to Component

- Migration zu komponentenbasierten Lösungen
- Anwendungs-Reengineering
  - "Zerschlagen" von Altanwendungen
  - Monolithische Altanwendung wird zu einer flexiblen neuen komponentenbasierten Anwendung umgebaut
- Verwendung vorhandener Komponenten
- Ersetzung alter Komponenten durch neue Komponenten

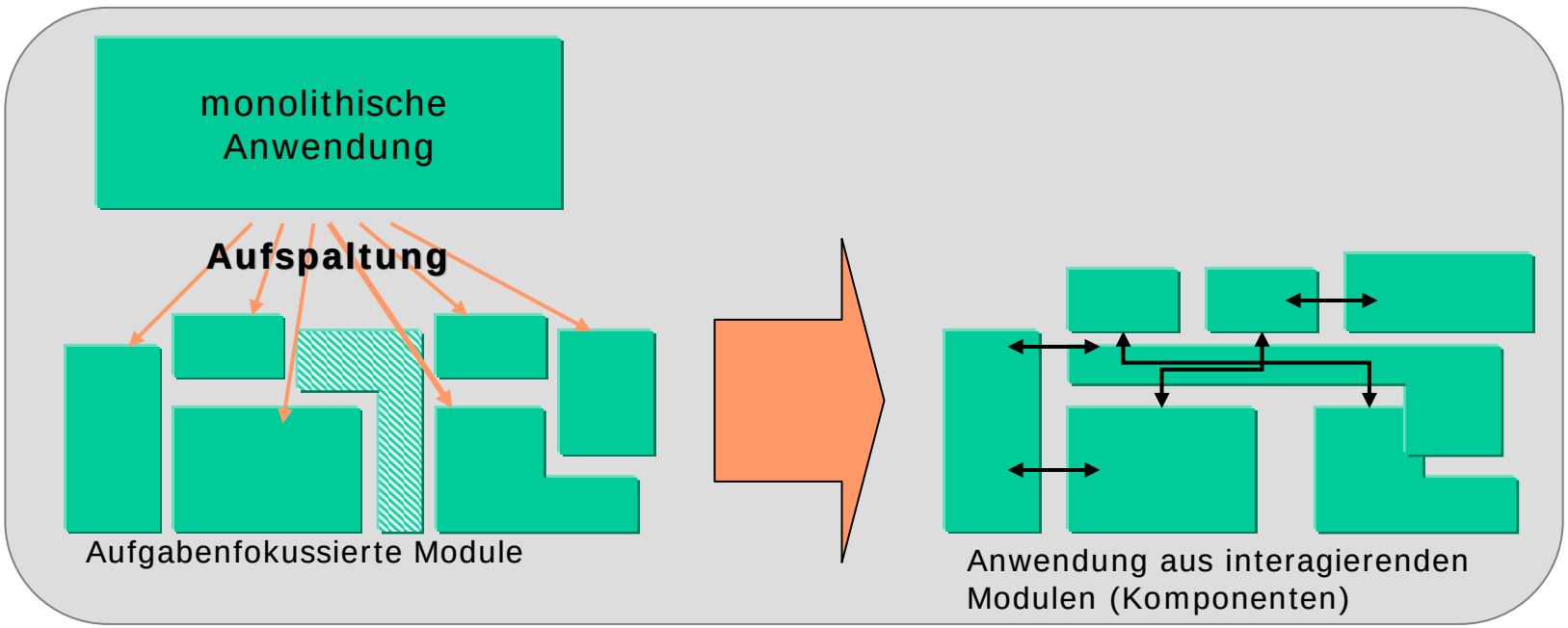
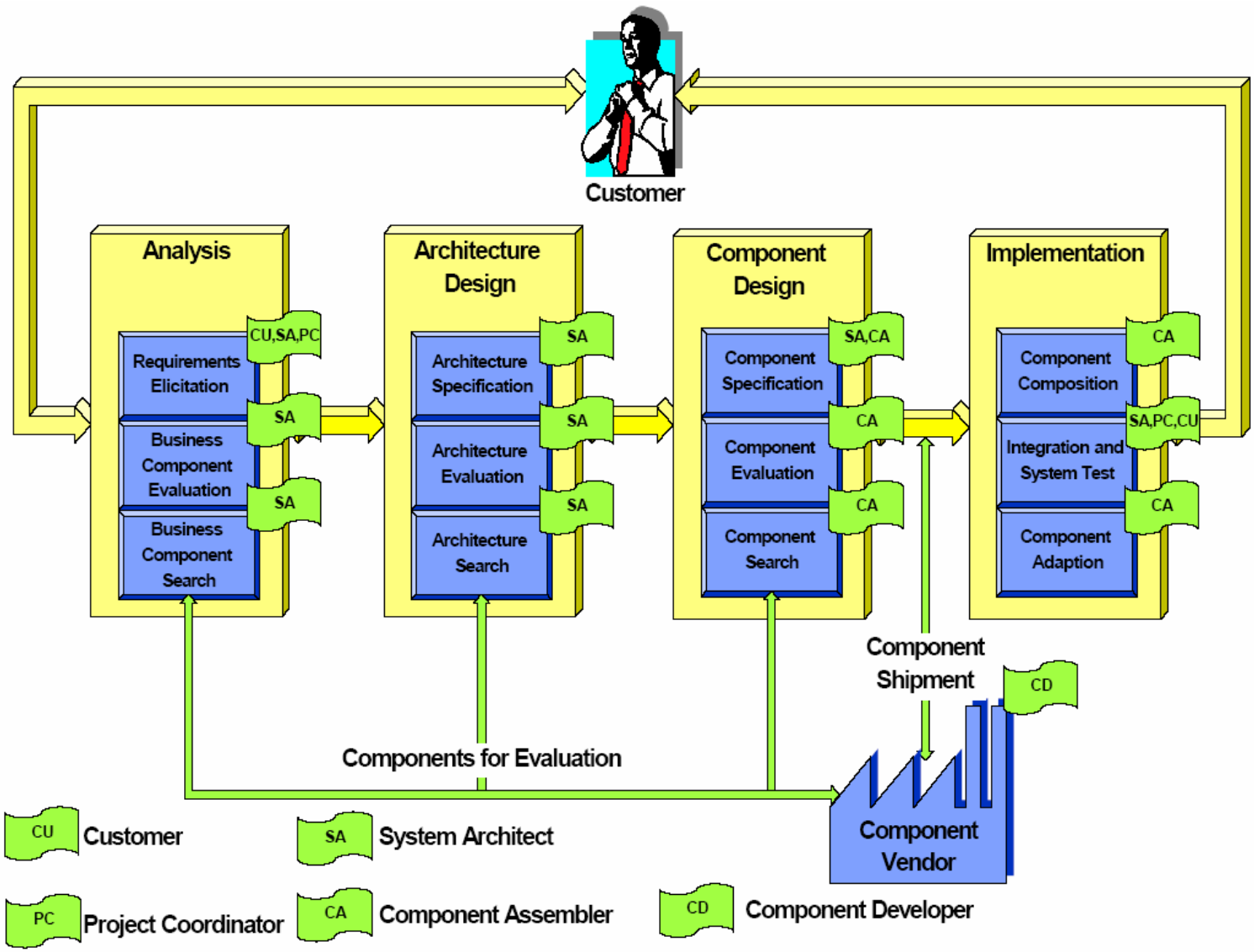


Abb: von einer monolithischen zu einer modularen Anwendung

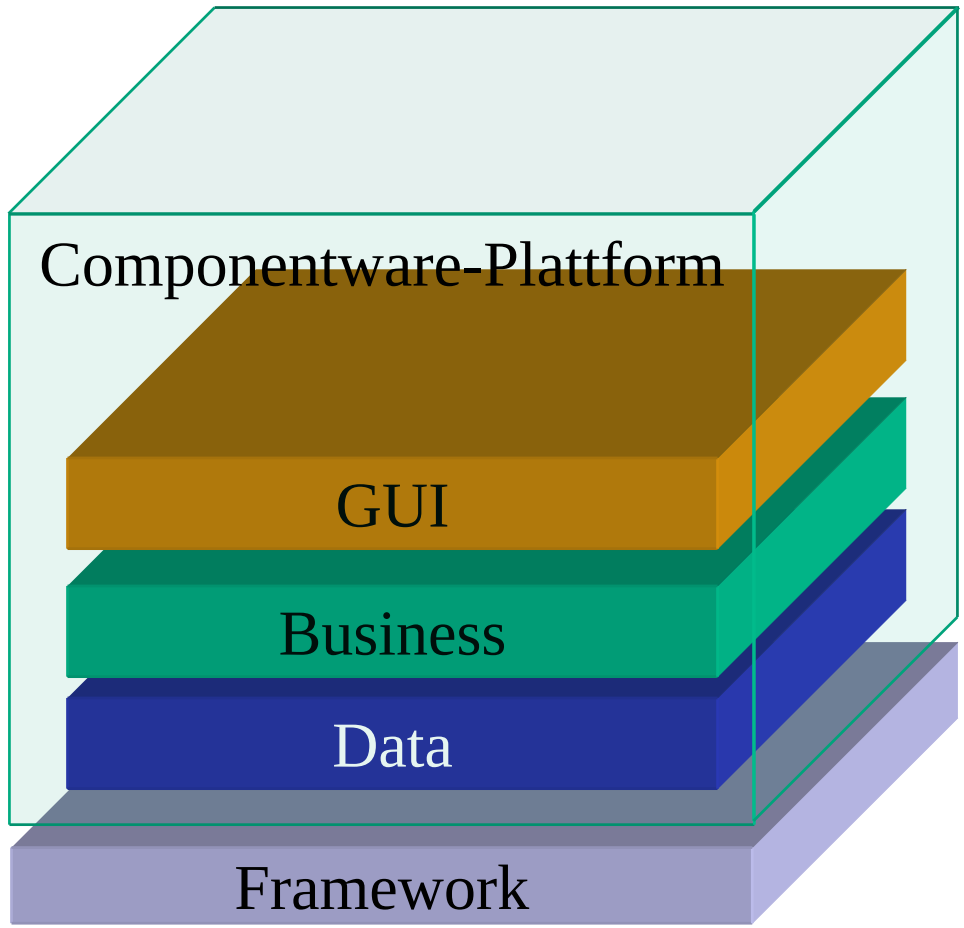
Entwicklungszyklus



Quelle: /Sihling 1998/

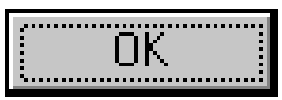
Komponententechnologien

- Anwendungen bestehen aus mehreren Schichten
- Jede Schicht besteht aus Komponenten
- Die Komponenten können aus unterschiedlichen Technologien erstellt werden
- Damit Vorteile unterschiedlicher Technologien in einem Produkt vereinbar

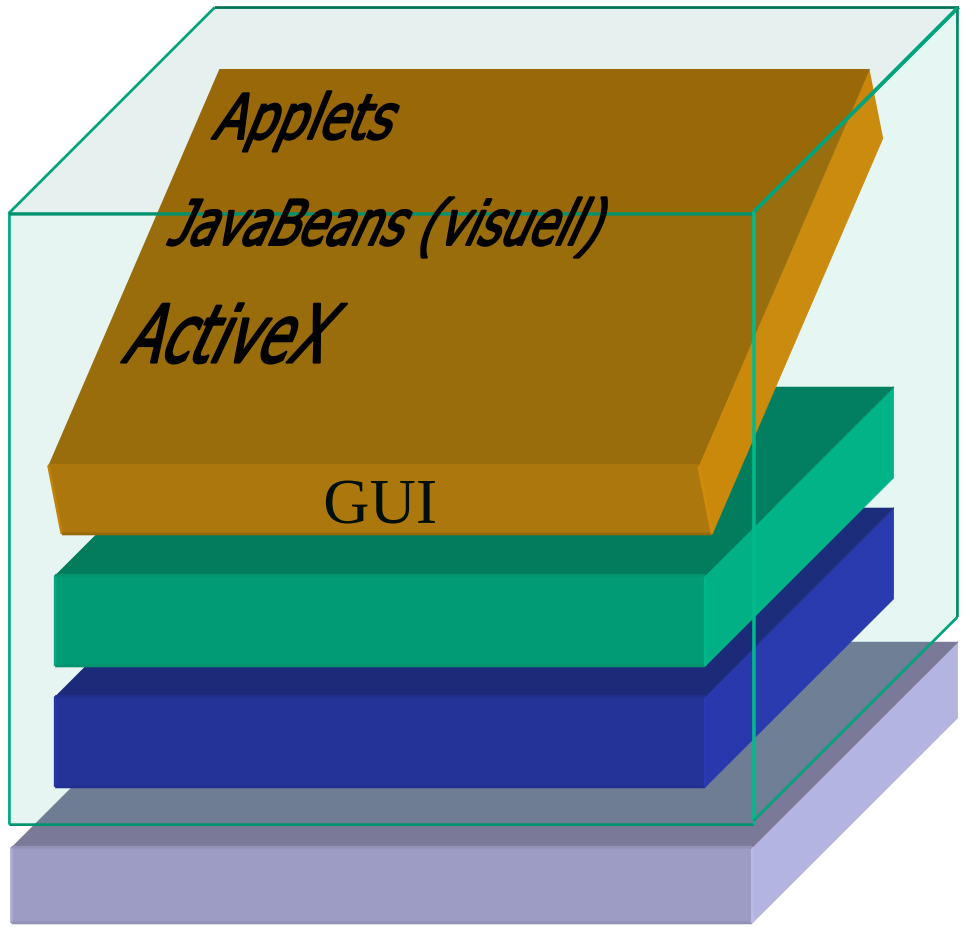


Komponententechnologien

- **Graphische Benutzerschnittstelle**
- Ausgabe von Informationen
- Entgegennehmen von Benutzereingaben
- Interaktion mit dem Benutzer
- Standards bei der Präsentation

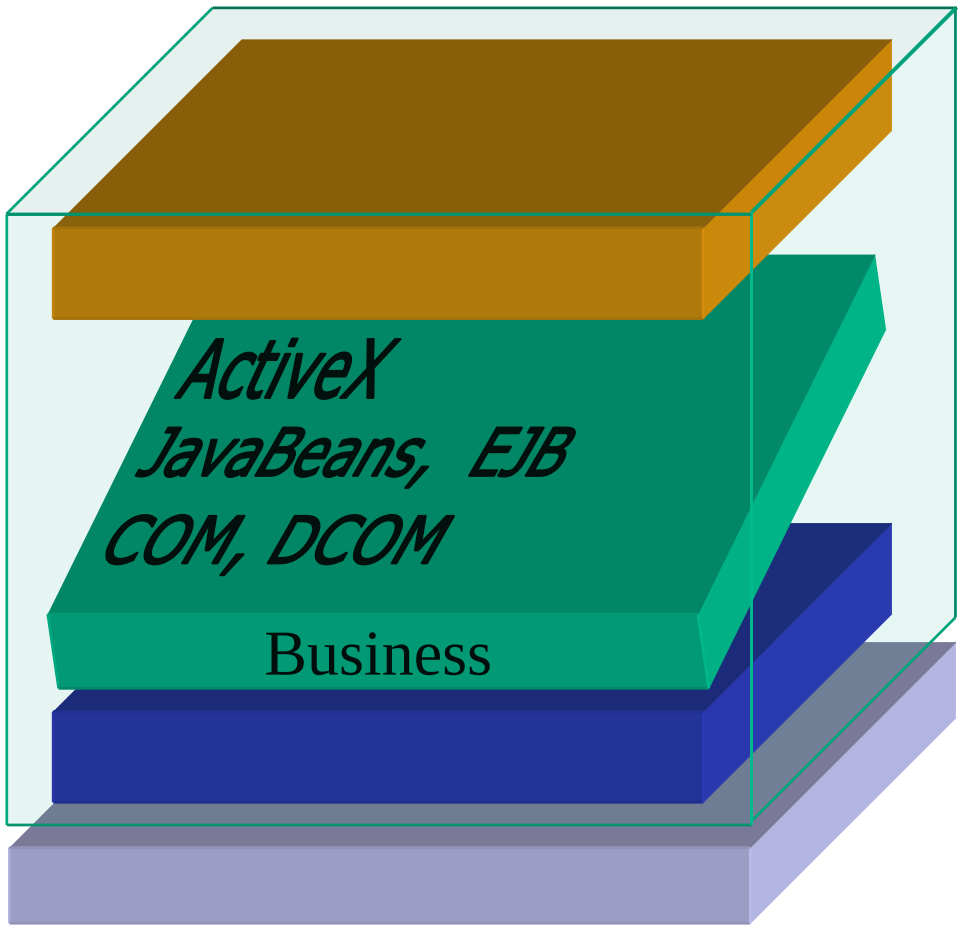


- Softwareergonomie



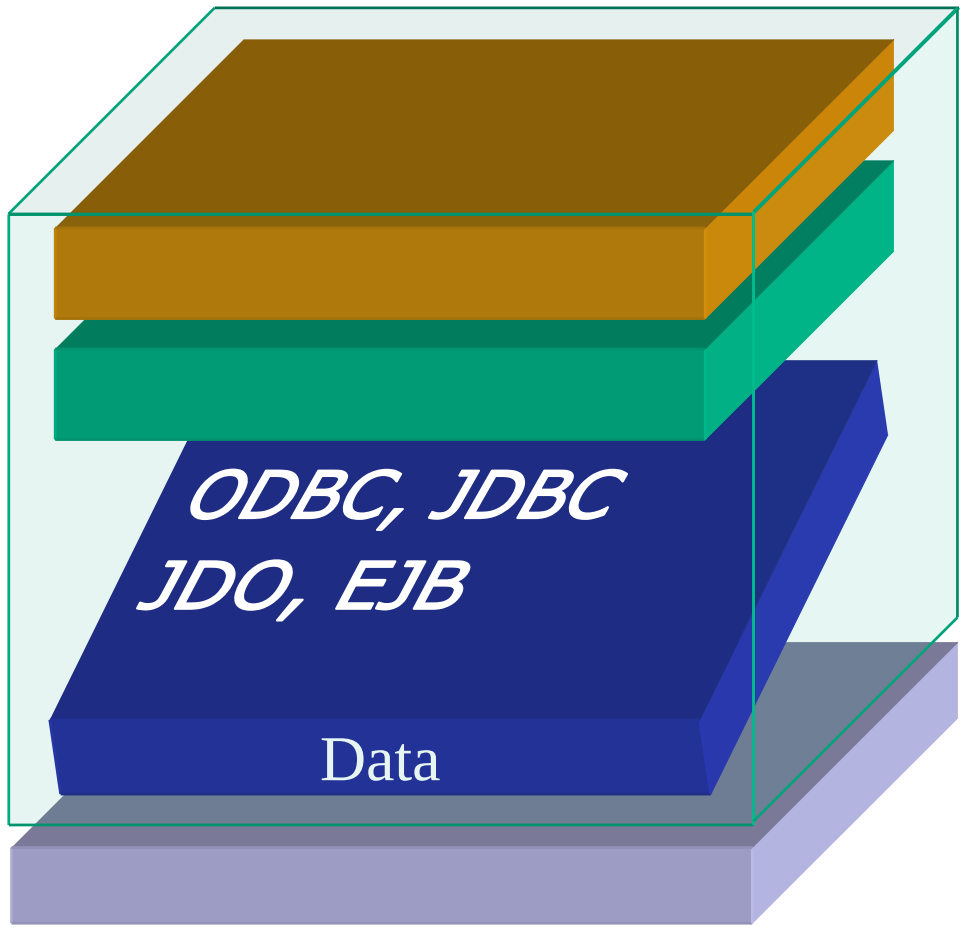
Komponententechnologien

- **Geschäftslogik**
- Implementierung von Funktionalität und Geschäftsregeln
- Komponenten zur Durchführung von Standardaufgaben (Drucken, Dateizugriff ...)
- Aufgabenspezifische, spezialisierte Komponenten

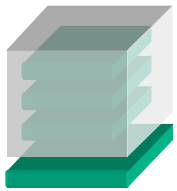


Komponententechnologien

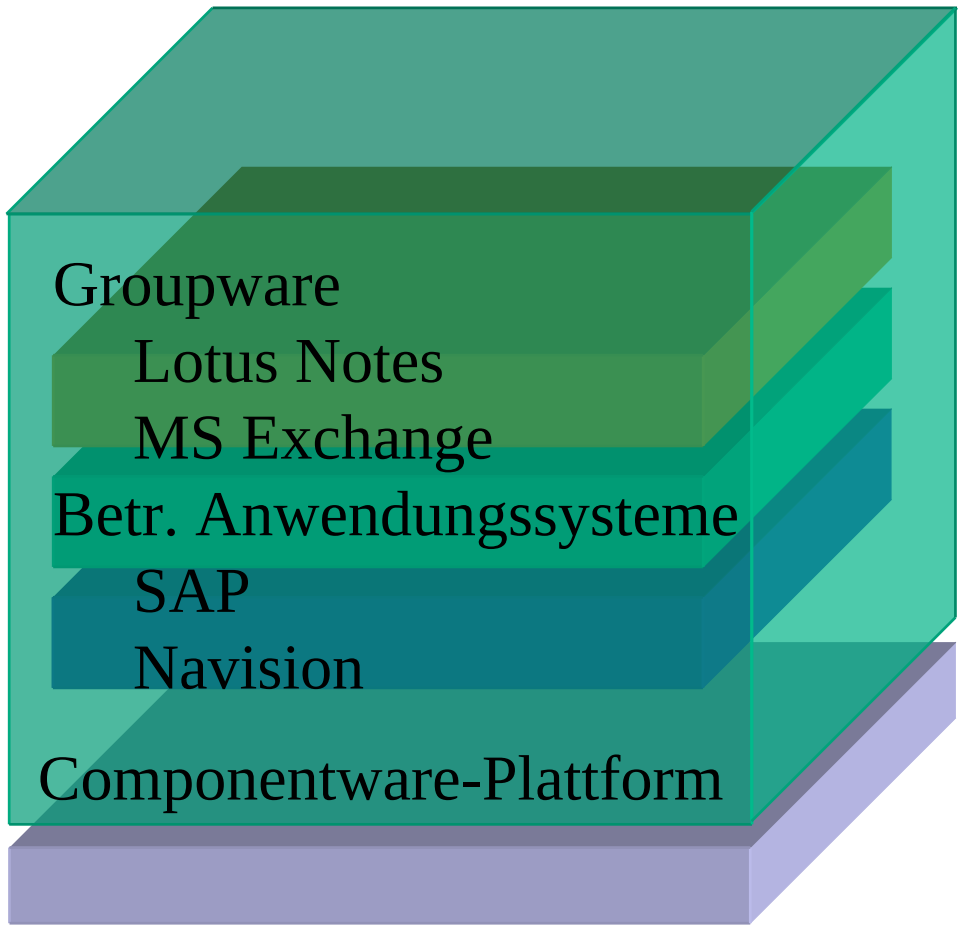
- **Datenschicht**
- Persistenzebene
- Kapseln des Zugriffs auf Datenquellen (relationale Datenbank, XML, ...)
- Daten als Objekte (Komponenten)



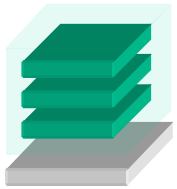
Komponententechnologien



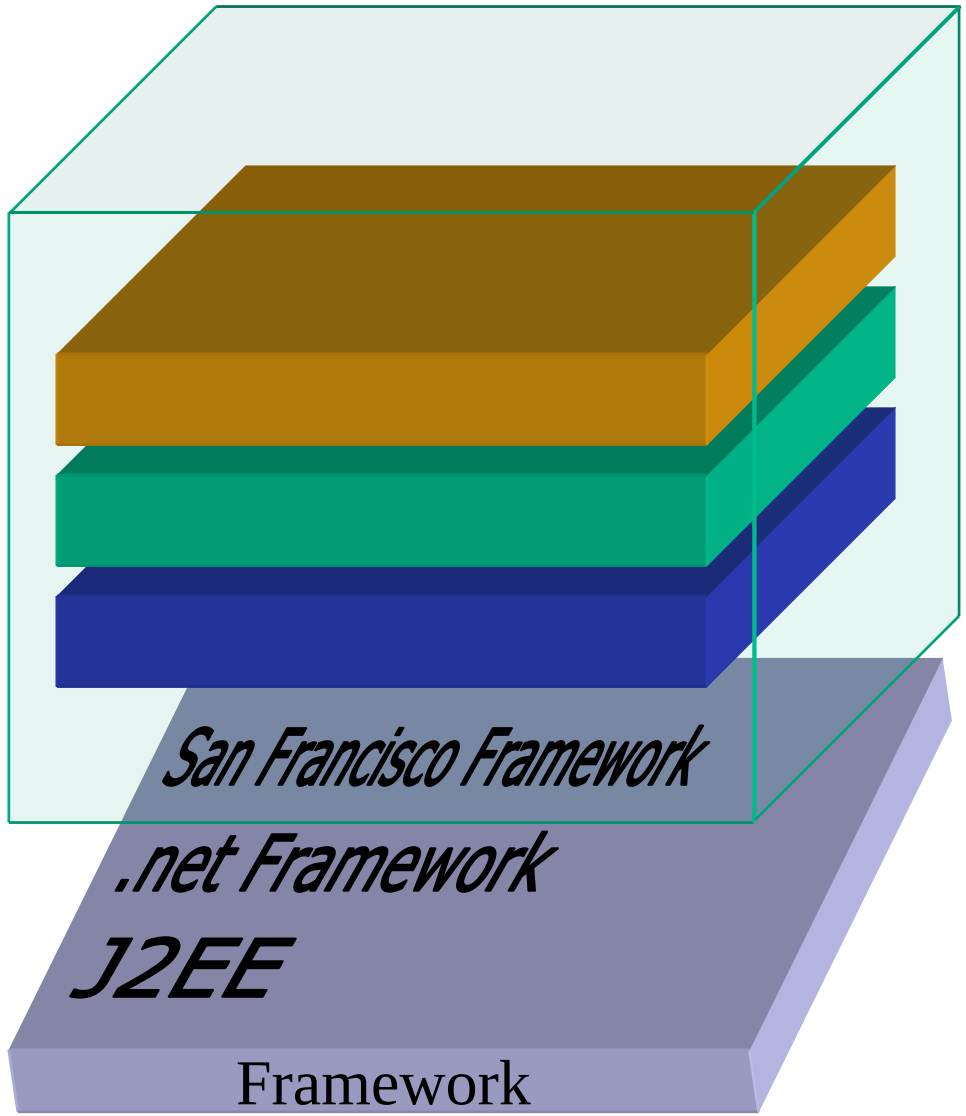
- Verteilte Anwendungen
- Modular erweiterbare Anwendungen
- Geschäftslogik / Datenschicht auf Server
- GUI auf Clientsystemen
- **Groupware**
  - Unterstützung von Teamarbeit
- **Betriebliche Anwendungssysteme**
  - Unterstützung wertschöpfender Geschäftsprozesse in Unternehmen



Komponententechnologien



- Plattform zur Erstellung ganzheitlicher Software
- Make or Buy
- Wiederverwendung
- Komponentenmärkte
- Kommunikation von Anwendungen untereinander



Zusammenfassung

- ***Komponententechnologie***:
  Mehrstufiger modularer Softwareaufbau aus vorgefertigten Komponenten, in denen Funktionalität gekapselt ist

- ***Anforderungen:***
  - formal fundiertes **Komponentenkonzept** als Basis
  - **Beschreibungstechniken** für derartige Komponenten
  - Entwicklung eines **Prozessmodells** zur Entwicklung, Verwaltung und Zusammensetzung von Komponenten
    - Unterstützung der Zuweisung verschiedener Rollen
  - **Werkzeuge**, welche die Beschreibung und das Prozessmodell unterstützen
    - zur Systemgenerierung selbst
    - zur Dokumentation
    - zur Verifikation und Sicherung wichtiger und kritischer Systemeigenschaften

Quellenverzeichnis

Quellen

- Balzert H., Lehrbuch Grundlagen der Informatik, Spektrum Akademischer Verlag, Heidelberg, 1999
- Weisbecker A., Software-Management für komponentenbasierte Software-Entwicklung, Jost Jetter Verlag, Heimsheim 2002
- Sihling M., Componentware - The Big Picture, http://www4.informatik.tu-muenchen.de/~sihling/publications/1998/CBSE98.pdf