Weitere Definitionen 1/2

- Things that people perceive as hard to change.

  [Fowler, M. (2003). "Design - Who needs an architect?". IEEE Software 20 (5)]

Since designing the architecture takes place at the beginning of a software system's lifecycle, the architect should focus on decisions that “have to” be right the first time, since reversing such decisions may be impossible or prohibitively expensive.

Weitere Definitionen 2/2

- [Architecture is] the set of principal design decisions made about the system

[Taylor et al.]

→ Diejenigen Dinge, über die ich Kontrolle habe, weil ich sie explizit gemacht habe, die ich also auch bewusst ändern kann

- A set of architectural design decisions.

[Jansen, A.; Bosch, J. (2005). "Software Architecture as a Set of Architectural Design Decisions", WICSA'05]

- Software architecture should not be considered merely a set of models or structures, but should include the decisions that lead to these particular structures, and the rationale behind them. This insight has led to substantial research into software architecture knowledge management.

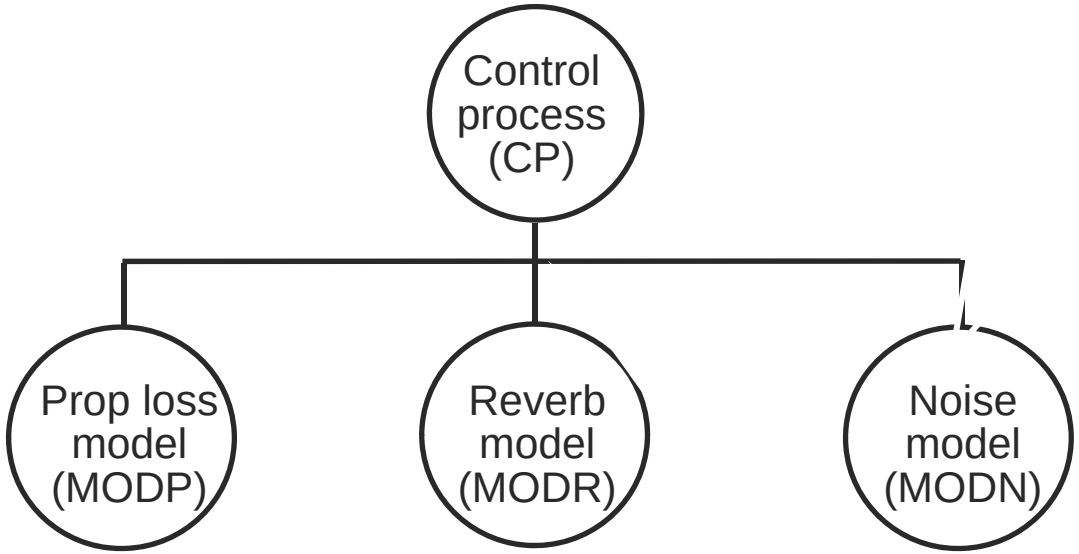
[Ali Babar, Muhammad; Dingsoyr, Torgeir; Lago, Patricia; van Vliet, Hans (2009). Software Architecture Knowledge Management]

Ein typisches Beispiel einer Architekturdarstellung

Begriffsbildung

- Begriff
- Anforderungen
- Definitionen
- Definition: Textur
- Beispiele
- Systemarchitektur
- Konsequenzen
- ...
- ...
- ...
- ...
- ...

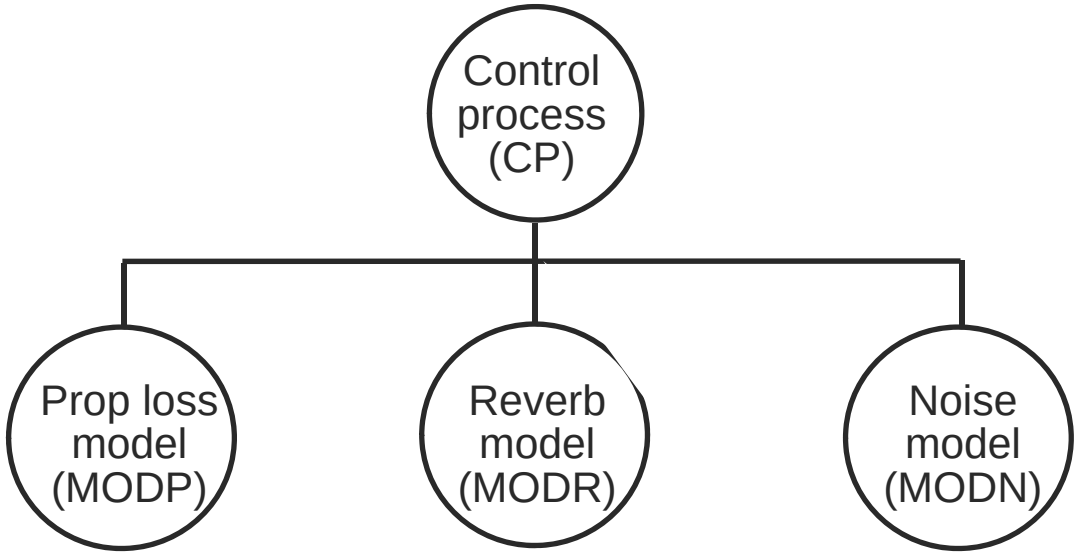
Top-Level-Architektur einer akustischen Unterwasser-Simulation



Was sagt uns dieses Diagramm?

Ein typisches Beispiel einer Architekturdarstellung

Top-Level-Architektur einer akustischen Unterwasser-Simulation

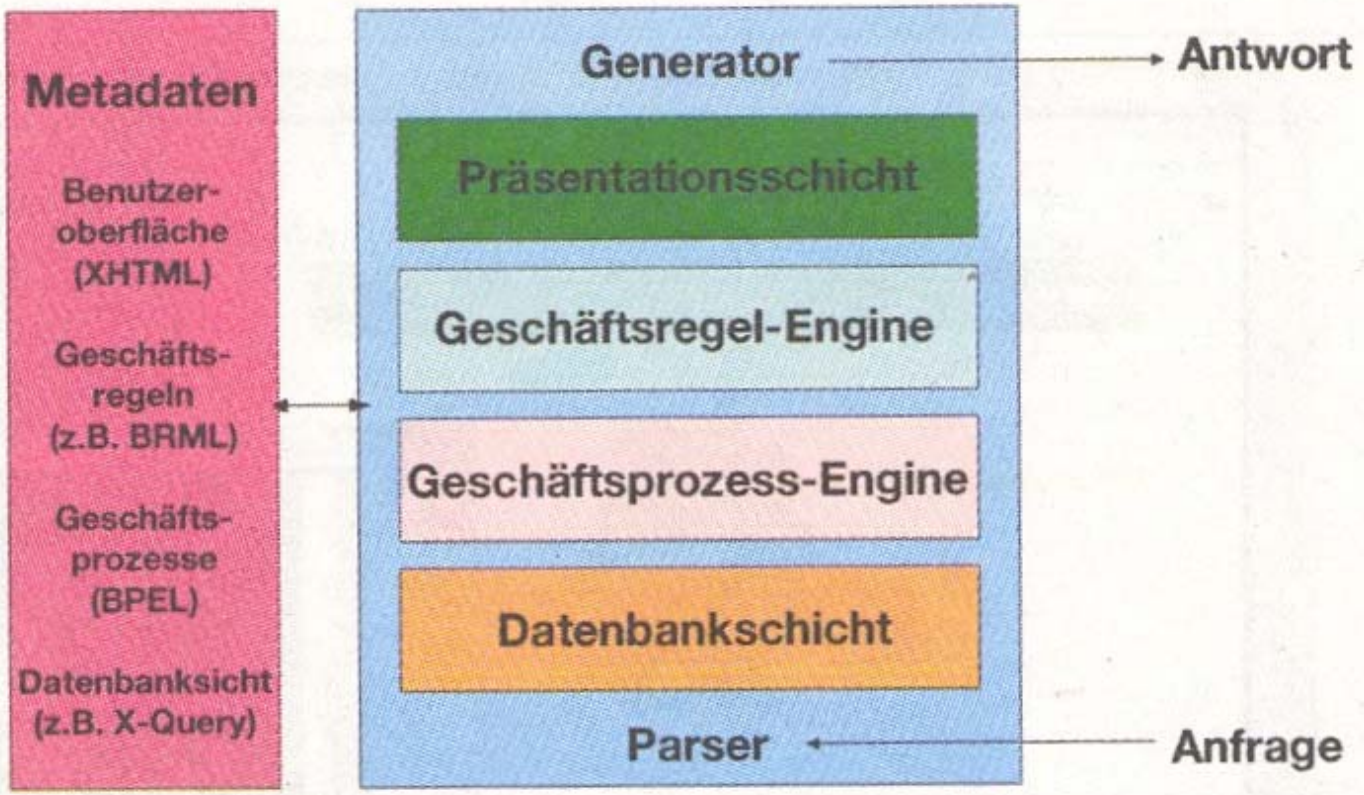


Was sagt uns dieses Diagramm ***nicht***?

Was sagt uns dieses Diagramm (nicht)?

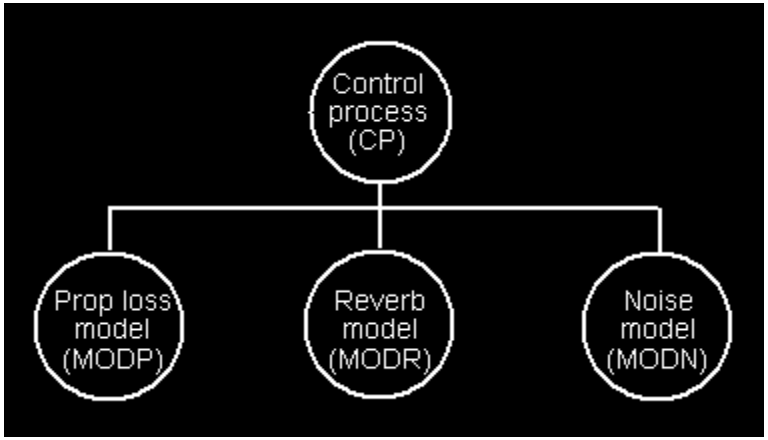
[Computer Zeitung 5/2006]

Schema einer Applikation nach dem Prinzip der weborientierten Architektur (WOA)



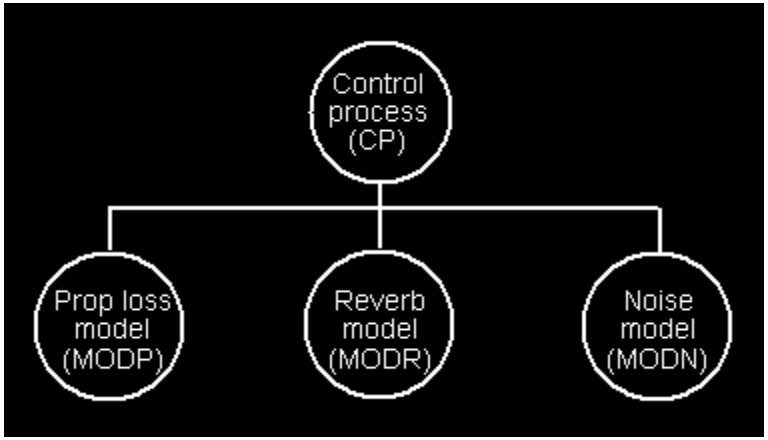
Fragen, die eine Architektur beantworten *sollte* 1/2

- Von welcher Art sind die Komponenten?
  - Tasks? Prozesse?
  - Objekte? Programme? Funktionen?
  - Bibliotheken? Übersetzungseinheiten?
  - Prozessoren?
- Von welcher Art sind die Beziehungen?
  - Aufrufe? Starts? Signale? Benutzt-Beziehungen? Datenfluss?
  - Vererbung?
  - Läuft-zusammen-mit? Schließt-aus?
  - Liegt-auf-dem-gleichen-Prozessor?



Fragen, die eine Architektur beantworten *sollte* 2/2

- Ein System besitzt (viele) unterschiedliche Aspekte
  - Aus Kundensicht: logischer Aufbau
  - Aus Entwicklersicht: Aufbau aus Modulen
  - Aus Administratorsicht: Verteilung von Prozessen auf Rechnern/Prozessoren
  - etc.
- Man sagt, ein System...
  - besitzt unterschiedliche *Strukturen* [Bass, Clements, Kazman]
  - wird dargestellt aus unterschiedlichen *Sichten* [Kruchten]
- Ein Software-System besitzt eine *Mikrostruktur* / *Textur* [Jazayeri, Ran, van der Linden], die beschreibt, wie z. B. folgende Dinge realisiert sind:
  - Kommunikation und Koordination
  - Fehlerentdeckung und –Behandlung



Architectural Texture "Textur": Begriff, Übersetzung

Übersetzungen:

- der Aufbau
- die Beschaffenheit
- das Gefüge
- *das Gewebe*
- *die Oberflächenstruktur*
- die Struktur
- die Textur

[Quelle: www.leo.org]

Definition "Texture"

- [The] *Recurring microstructure* of the software components is at least as important as the system structure.

We call recurring microstructure of components ***texture*** and consider it an important part of software architecture.

[Jazayeri, Ran, van der Linden:
Software Architectures for Product Families,
Addison Wesley]

Motivation "Texture" 1/2

- Ein Software-System wird nicht nur von seiner "Grobstruktur" geprägt, sondern auch von seiner "Mikrostruktur"
- Beispiele: Wie sind folgende Dinge realisiert:
  - Kommunikation und Koordination
  - Fehlerentdeckung und –Behandlung
- Förderlich für Analysierbarkeit, Änderbarkeit, Erweiterbarkeit, ...: *Einheitliche* Realisierung dieser Aspekte/Muster (nicht nur: *Pattern*!)
  - innerhalb von Komponenten und
  - Komponenten-übergreifend

Motivation "Texture" 2/2

- Diverse "Dinge" müssen mehrfach implementiert/benutzt werden
  - Kommunikation und Koordination
  - Ablaufkontrolle und Logging
  - Zustandsüberwachung
  - Sicherheitsaspekte
  - Persistenz
  - Fehlerentdeckung und –Behandlung
- Über ihre Realisierung sollte *nicht* jedes Mal *lokal* entschieden werden:

→ Man muss sich *bewusst werden*, dass es sich überhaupt um wiederholt zu treffende Entscheidungen handelt!

- Programmiersprachen und Tools bieten keine Unterstützung für eine einheitliche Realisierung;
  Ausnahmen = ?

Konsequenz

- Es muss strategische, systemweite Vorgaben (policies) geben, wie solche Dinge (einheitlich) zu realisieren sind (in Design, Implementierung und Qualitätssicherung)

→ Die Architektur (–Dokumentation) ist der (einzige) passende Ort für solche Vorgaben:

  – Standards
  – Richtlinien
  – Stile/Pattern
  – Infrastruktur-Benutzung
  – etc.

Konsequenz

- Es muss strategische, systemweite Vorgaben (policies) geben, wie solche Dinge (einheitlich) zu realisieren sind (in Design, Implementierung und Qualitätssicherung)

→ Die Architektur (–Dokumentation) ist der (einzige) passende Ort für solche Vorgaben:

  – Standards
  – Richtlinien
  – Stile/Pattern
  – Infrastruktur-Benutzung
  – etc.

Mögliche Abgrenzung Architektur <-> Design

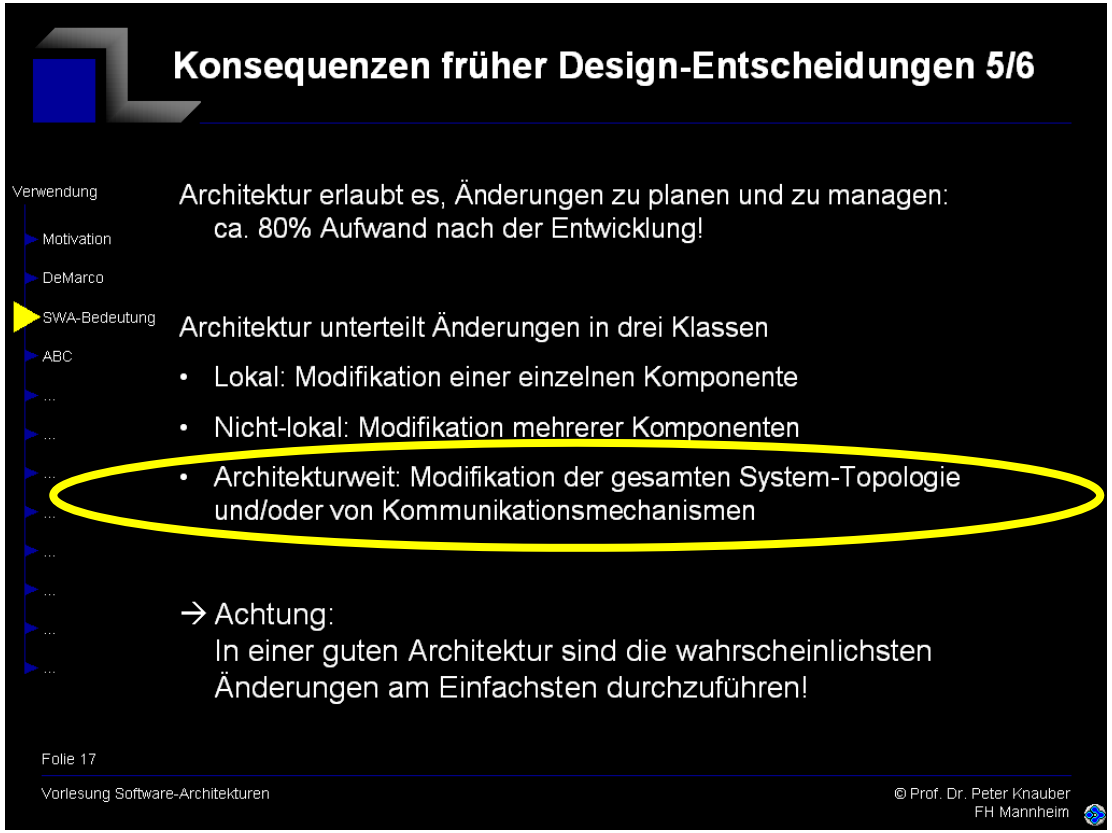
- Die Architektur macht *Vorgaben* fürs Design,
  damit sind alle Design-Vorgaben *per definitionem* Architektur
  - Beispiel: Wenn z.B. ein Observer-Mechanismus verwendet werden **muss**, dann ist das Teil der Architektur, sonst Teil des Designs
  - Beispiel: Fenster-Anordnung übereinander im Wohnhaus

- We call things architecturally significant of they are…
  - Costly to change
  - Risky
  - New

[Grady Booch]

Architektur-Relevanz

Einmal entschieden und realisiert, ist es sehr schwer, Architektur-Entscheidungen rückgängig zu machen
→ Systemweite Auswirkungen (s. Bedeutung von SWA)



Carnegie Mellon University, Pittsburgh

Weitere Definitionen

Begriffsbildung

- Begriff
- Anforderungen
- Definitionen
- Definition: Textur
- Beispiele
- Systemarchitektur
- Konsequenzen
- ...
- ...
- ...
- ...
- ...

Software Engineering Institute | Carnegie Mellon

A-Z Index Search

HOME | OUR WORK | OUR SOLUTIONS | PRODUCTS & SERVICES | LIBRARY | NEWS | CAREERS | ABOUT US | BLOG

Software Architecture

Overview | Getting Started | Research | Tools & Methods | Consulting | Case Studies | Our People

Community Software Architecture Definitions

This page lists definitions that have been contributed by visitors to this website. Contact us to submit your definition of software architecture.

Background

SEI work in architecture-centric engineering exploits the relationship between a system's architecture and its quality attributes, and leverages architecture as the key means for ensuring that systems will support their business and mission goals.

Learn More

- Explore our architecture-related training opportunities.
- Learn about our research in architecture-centric engineering.
- Network with other software, systems, and enterprise architects.

Definitions

Stephan Zehrer (Senior System/SW Engineer, Privat, Markdorf, Germany): In relation to ISO 42010, the term "architecture" is defined here: http://www.iso-architecture.org/ieee-1471/defining-architecture.html Fundamental concepts or properties of a software in its environment embodied in its elements, relationships, and in the principles of its design and evolution. I see software as a subclass of system.

Edgar Gomez Reyes (Desarrollador de Proyectos, Aspec System, Veracruz, Mexico): Es el equipamiento logico y la arquitectura de los servicios que se desarrollan para ofrecer una vida mas fácil y cómoda para la humanidad. Los desarrolladores "Trabajamos en lo difícil para que usted lo haga fácil".

Arvind Kiewelekar (Associate Professor, DBATU, Maharashtra, India): Software architecture as a discipline aims at creating techniques to describe existing software systems, prescribing how to build a new software system and evaluate the configuration of software systems.

Nelu Suciu (Solution Architect, ISDC, Cluj, Romania): The core element of any software system is data. Along with data any software system provides the following capabilities: collection, processing, storage, presentation, and distribution of data. Software architecture has a two-fold purpose: first it has to

Glossary
- Bibliographic Software Architecture Definitions
- Classic Software Architecture Definitions
- Community Software Architecture Definitions
- Modern Software Architecture Definitions
- Published Software Architecture Definitions
- What is your definition of software architecture?

Training
Publications
Educators Workshops
Events

Related Links

News

SEI Makes SMART Resources Freely Available

SATURN Conference Announces Speakers, Opens Registration

See more related news »

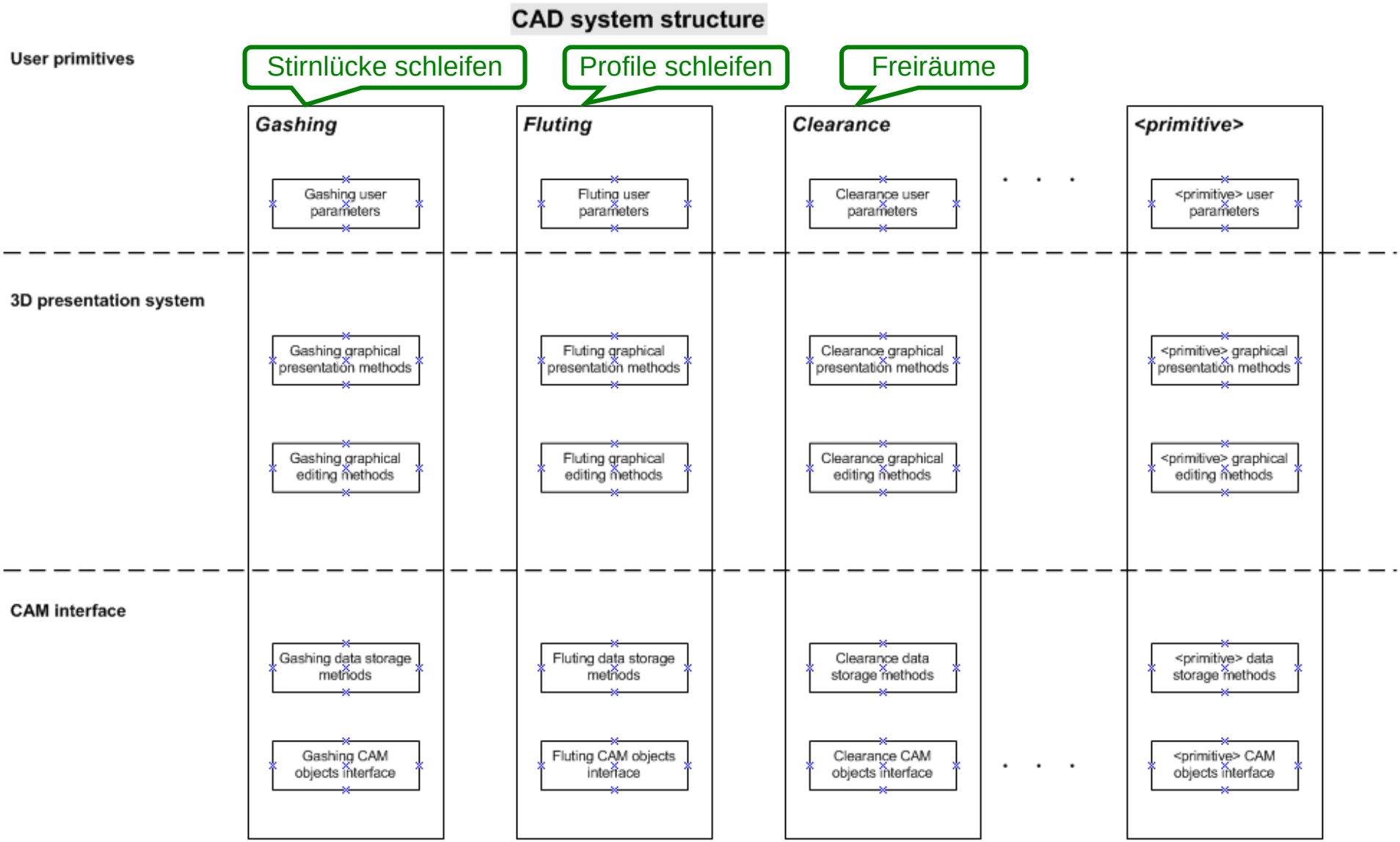
Training

Documenting Software Architectures - eLearning

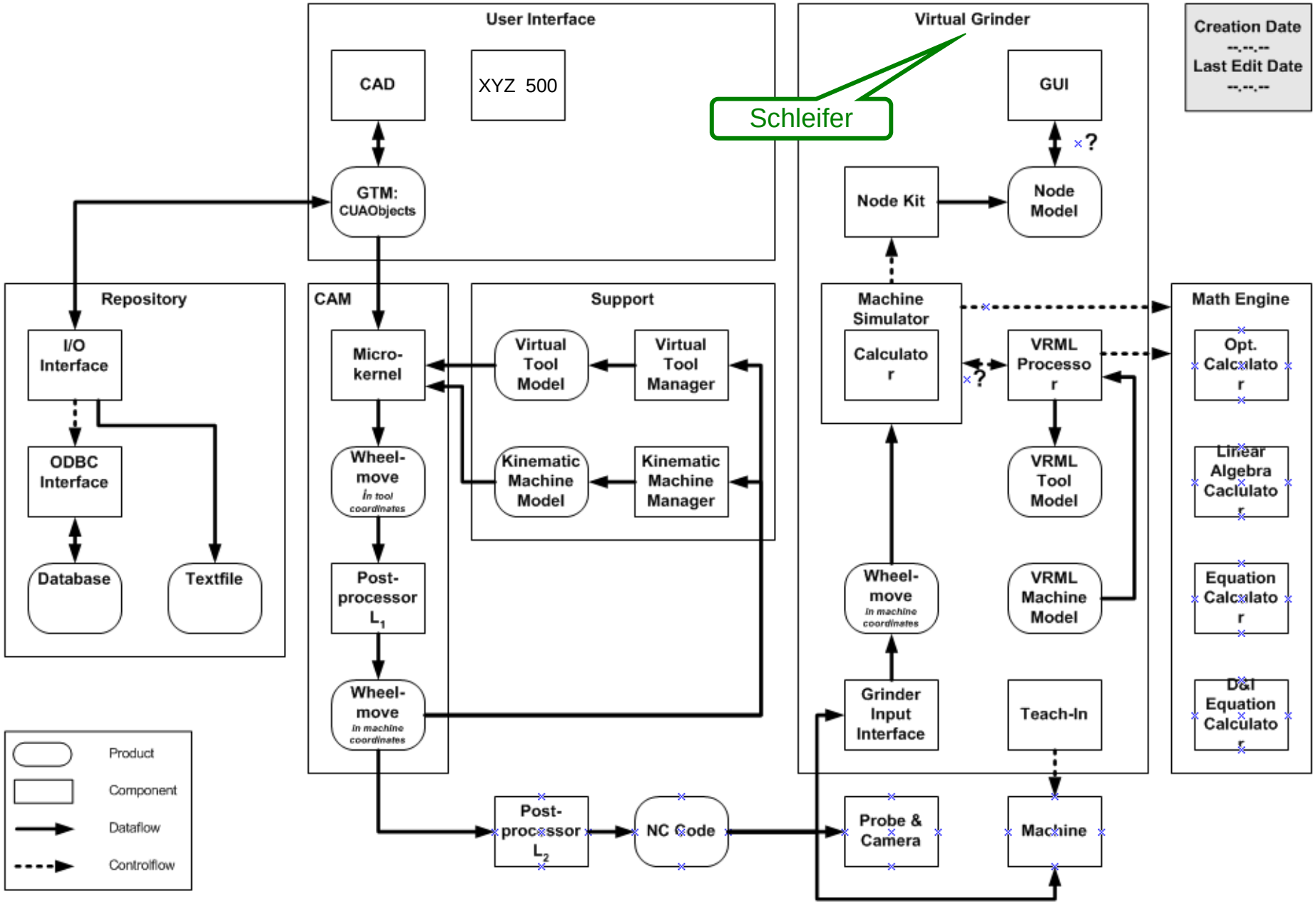
Software Architecture: Principles & Practices - eLearning

See more related courses »

Beispiel für eine schlechte Darstellung (nicht: ... für ein schlechtes System!)

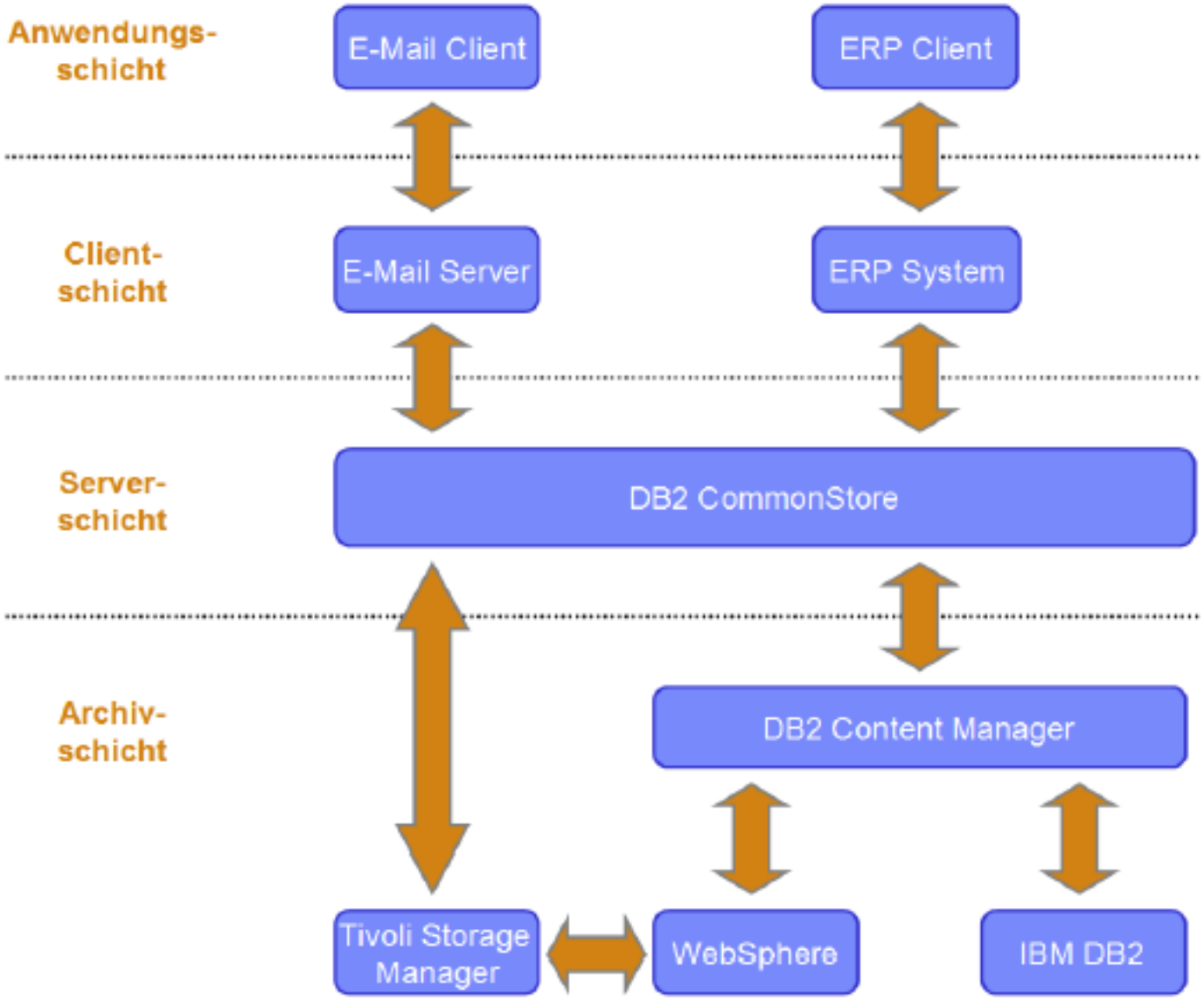


Beispiel für eine bessere Darstellung: Data-Flow-Sicht

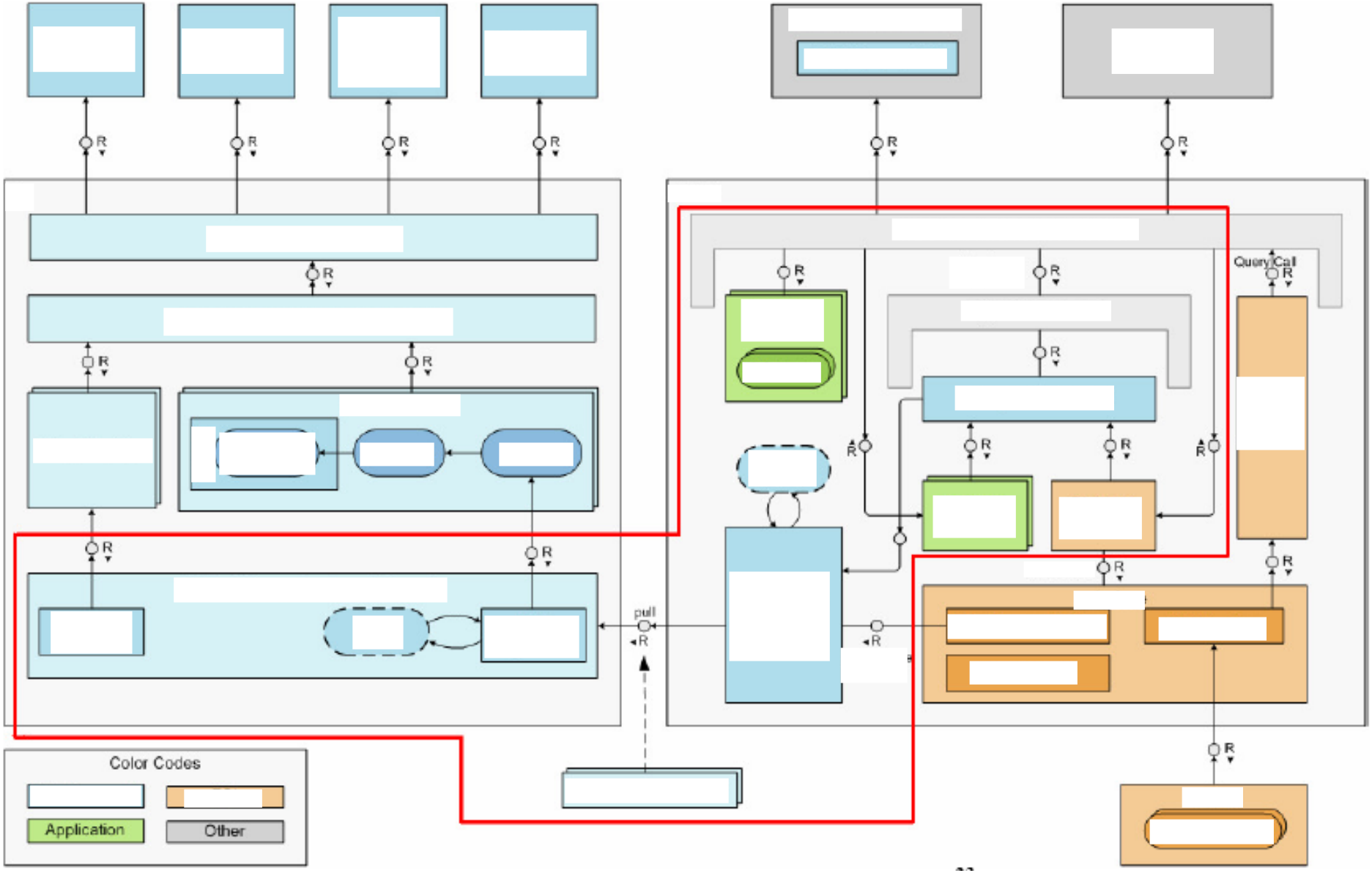


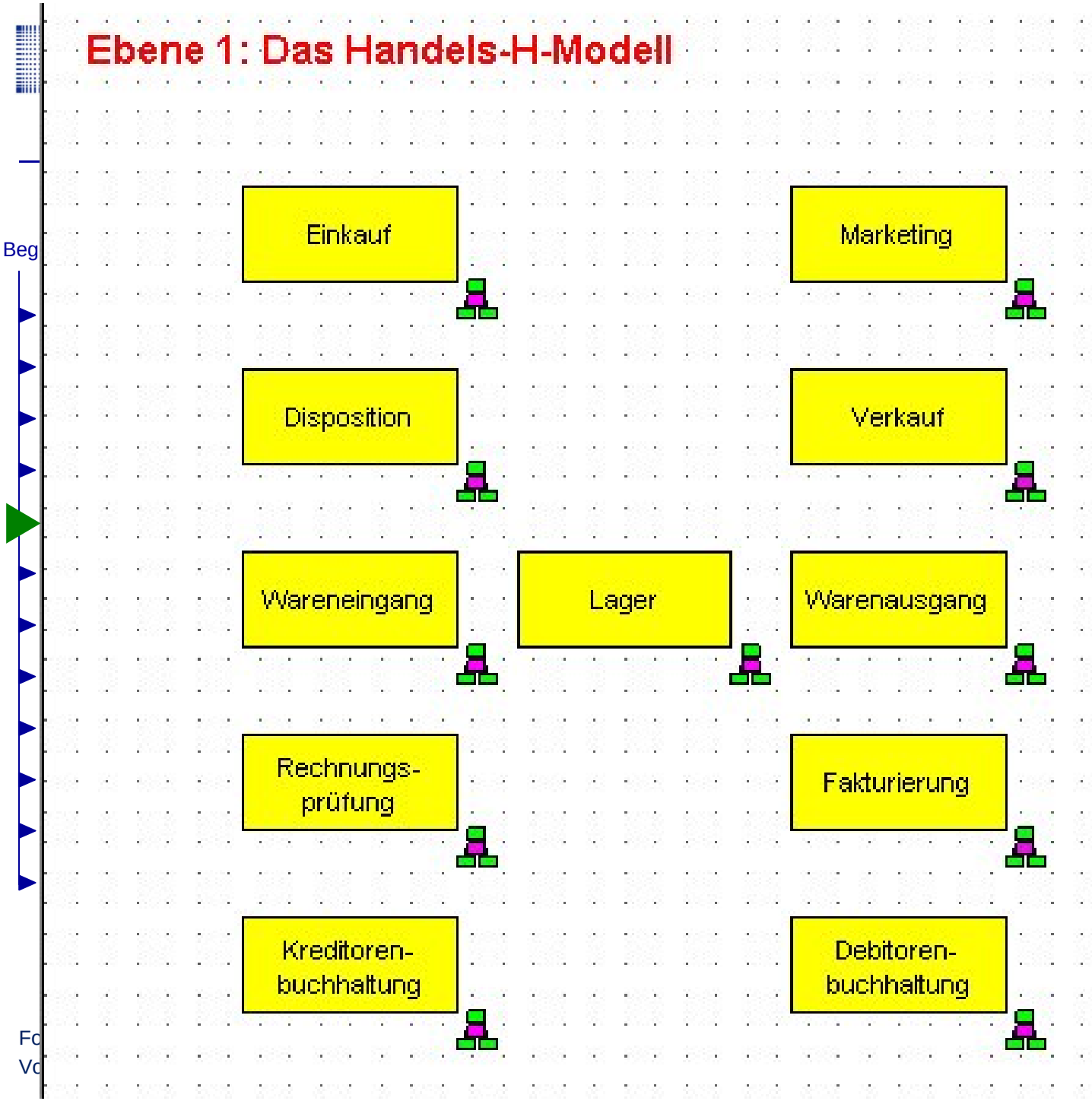
DB Common Store – Architektur

[IBM Corporation: IBM Content Management. http://www-304.ibm.com/jct03001c/services/learning/ites.wss/us/en?pageType=page&c=a0006792. Version: Juli 2005]



Darstellung einer anonymen Architektur...





(Extrem-) Beispiel für eine geschickte *Marketing-* Darstellung

Gute oder schlechte Darstellung? Warum?

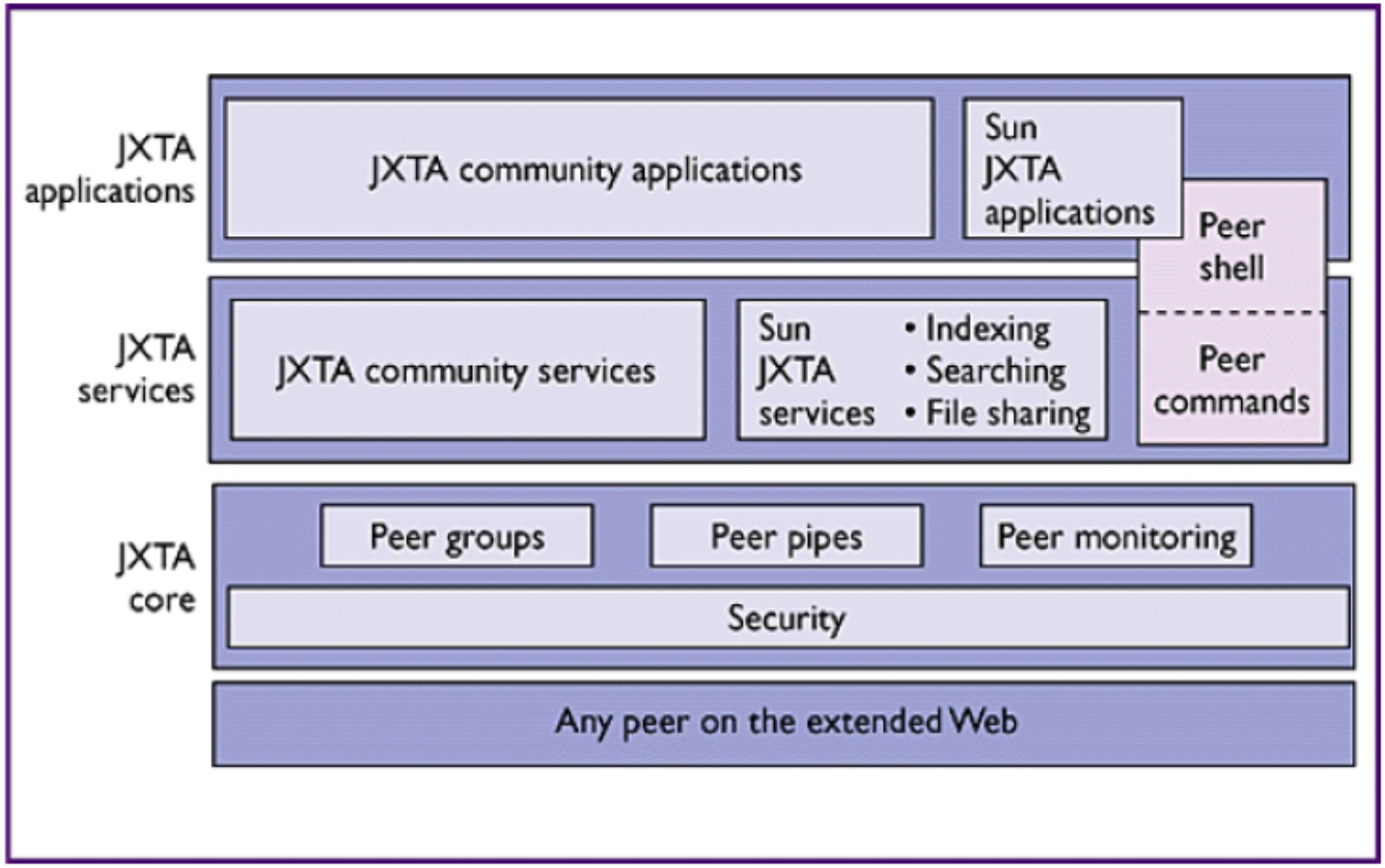


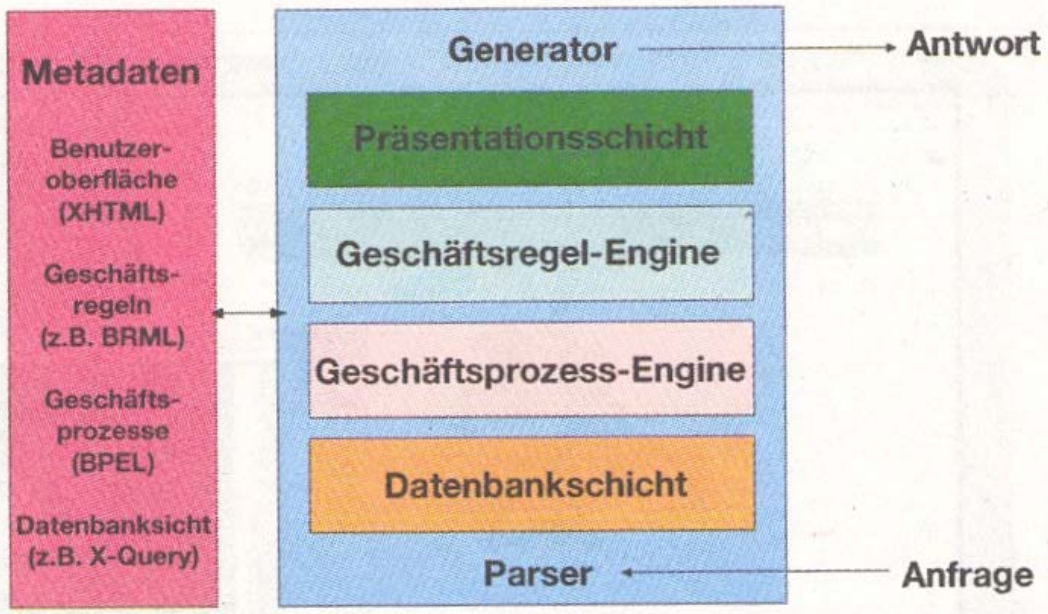
Abb. 2.9: Die Architektur von JXTA (Quelle: [24])

CZ Nr. 22 / 29. Mai '06

9

Webmodell stößt auf Skepsis

Schema einer Applikation nach dem Prinzip der weborientierten Architektur (WOA)

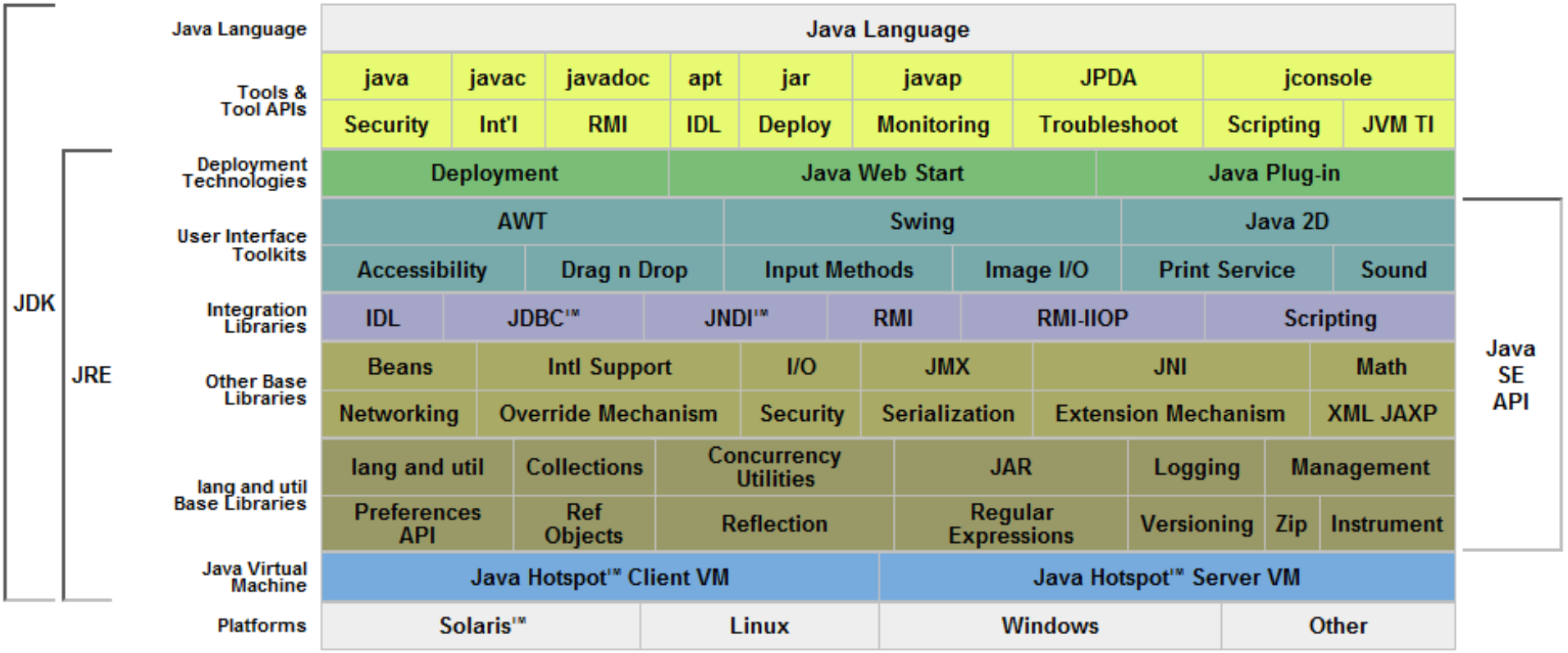


Die so genannte weborientierte Architektur (WOA) soll helfen, die **Schwächen serviceorientierter Ansätze** (SOA) zu beseitigen – so hofft zumindest Gartner. Nach Ansicht der Analysten Nick Gall und David

Was ist an dieser Architektur (-Sicht) nicht OK?

Java™ SE 6 Platform at a Glance

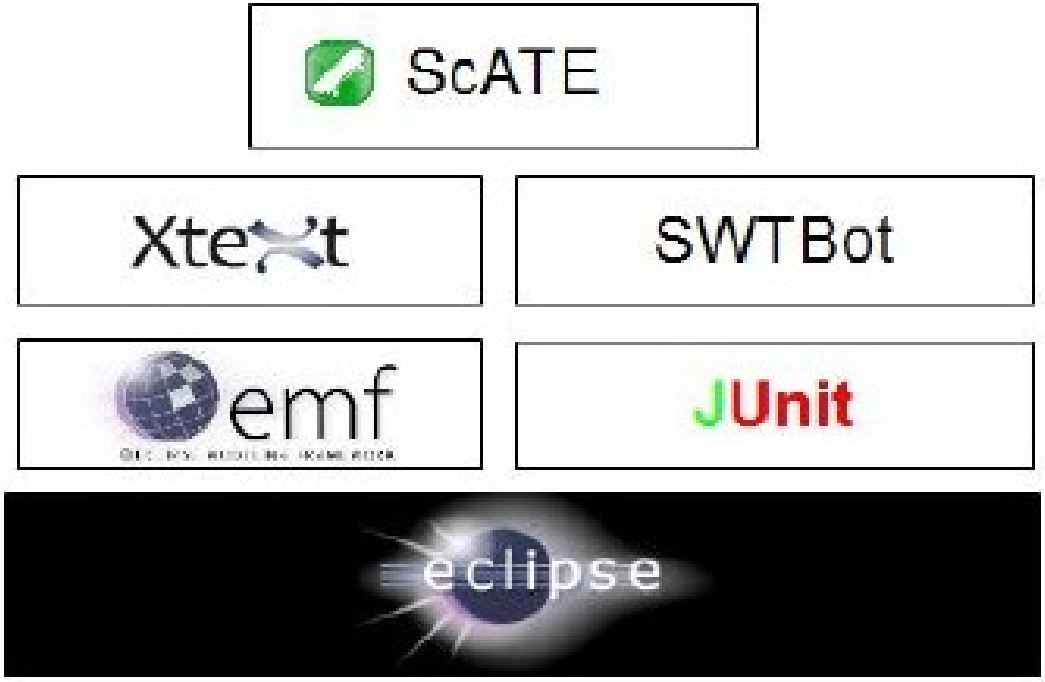
Begriffsbildung



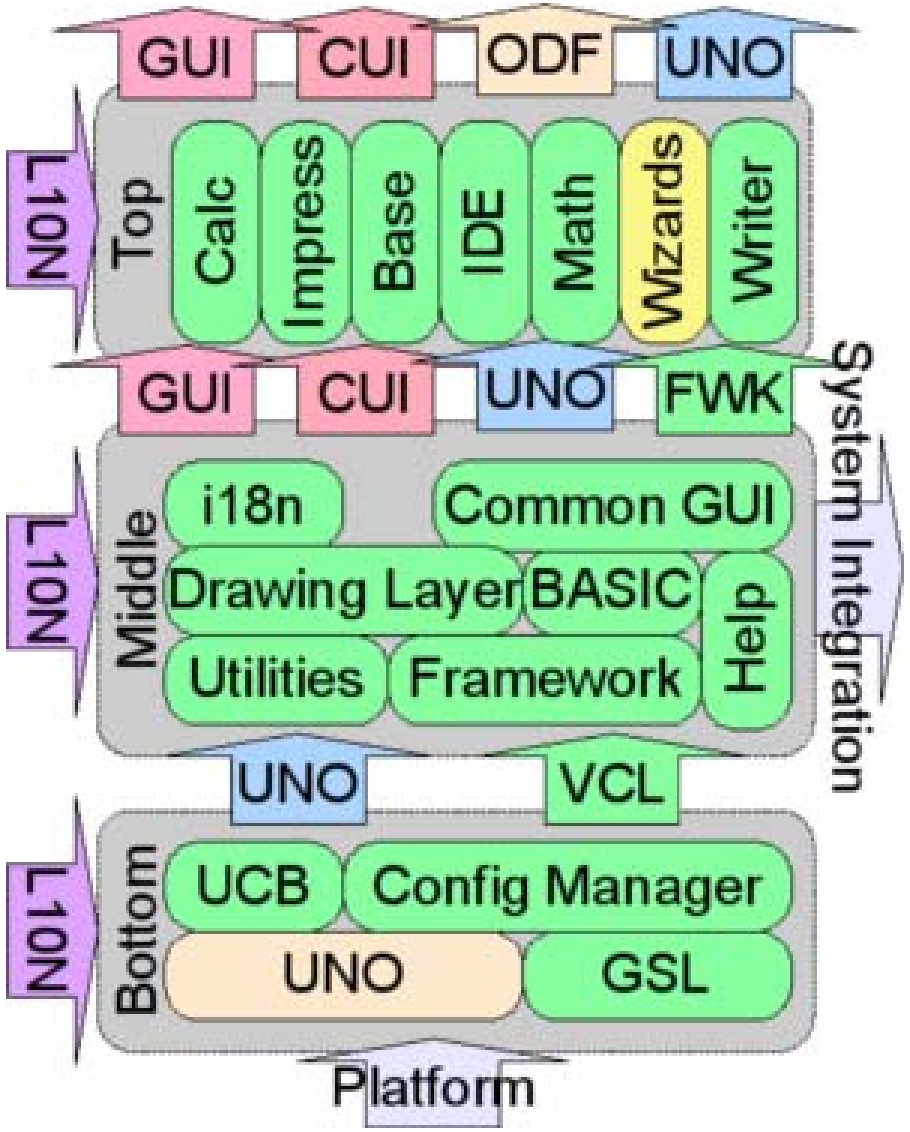
Gute oder schlechte Darstellung? Warum?

Begriffsbildung

- Begriff
- Anforderungen
- Definitionen
- Definition: Textur
- Beispiele
- Systemarchitektur
- Konsequenzen
- ...
- ...
- ...
- ...
- ...



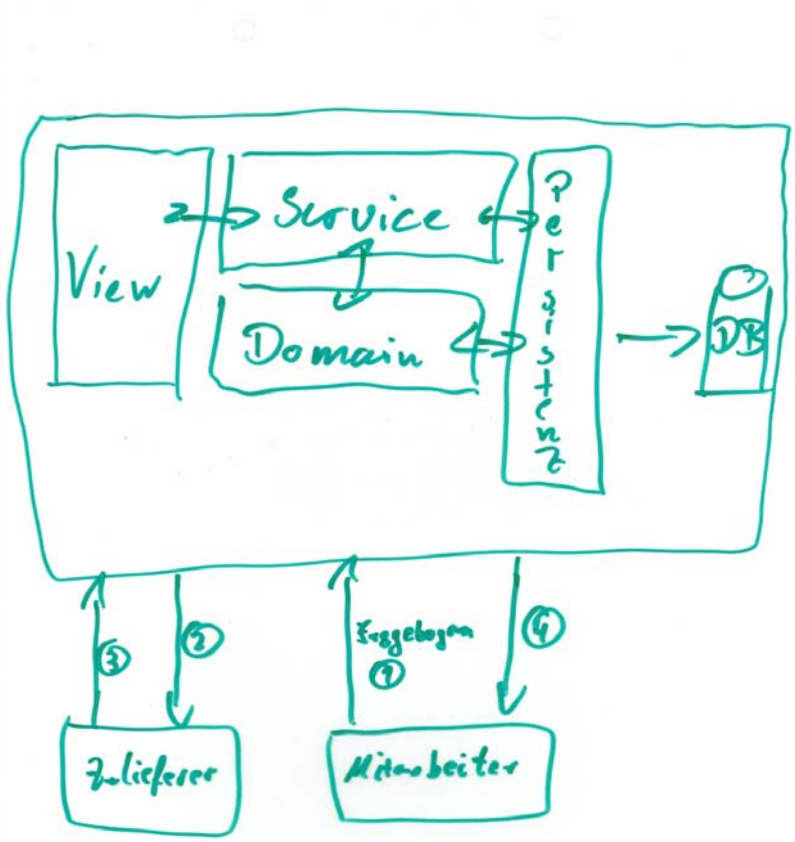
Gute oder schlechte Darstellung? Warum?



Ihre Beiträge

Begriffsbildung

- Begriff
- Anforderungen
- Definitionen
- Definition: Textur
- Beispiele
- Systemarchitektur
- Konsequenzen
- ...
- ...
- ...
- ...
- ...



Software-Architektur vs. Systemarchitektur

- Arbeiten Sie mit Ihrer Nebenfrau/Ihrem Nebenmann zusammen
- Finden Sie die Unterschiede der Begriffe "Software-Architektur" und "Systemarchitektur";
  was ist das eine, was ist das andere?
- Halten Sie Ihr Diskussionsergebnis schriftlich in Stichworten fest

Zeit: 3 Minuten

Definitionen [Siedersleben]

- Eine **Systemarchitektur** besteht aus
  - TI-Architektur und
  - Software-Architektur
- Eine **Software-Architektur** *für Informationssysteme* besteht aus
  - A-Architektur und
  - T-Architektur
- **TI-Architektur** → Technische Infrastruktur:
  Geräte, Systemsoftware, Programmiersprachen
- **A-Architektur** → Anwendungsarchitektur:
  die eigentliche Architektur der Anwendung *ohne* Bezug zur Technik
- **T-Architektur** → Technikarchitektur:
  Ablaufumgebung für die A-Architektur, z.B. Fehlerbehandlung, GUI-Umsetzung, Persistenzschicht, Prozessorganisation etc.

Konsequenzen aus der Definition (von BCK) 1/4

- Architektur ist eine Abstraktion eines Systems
  - Sie definiert Komponenten und wie diese miteinander interagieren
  - Sie enthält keine Komponenten-interne Information; private Details gehören *nicht* auf Architekturniveau

- Systeme haben mehrere Strukturen / Views / Sichten
  - *Keine einzelne* Sicht beschreibt die Struktur eines Systems ausreichend
  - Die Sichten, die für eine Architektur relevant sind, sind *nicht für jedes System gleich*;

  → genau das und nur das, was für das Verständnis oder die Analyse eines Systems oder die Kommunikation/Diskussion darüber benötigt wird, wird dargestellt

***Die* Definition des Software Engineering Institute (SEI)**

The software architecture of a program or computing system is

- the structure or structures of the system, which comprise software components
- the externally visible properties of those components, and
- the relationships among them.

[*Software Architecture in Practice*, Len Bass, Paul Clements, and Rick Kazman]

Konsequenzen aus der Definition 2/4

- Eigenschaften von Komponenten ("properties of … components ") sind gegenseitige Annahmen / Verträge, z.B.
  - Angebotene Dienste
  - Geforderte Dienste
  - Performanz-Charakteristika
  - Fehlerbehandlung (und –kommunikation)
  - Nutzung gemeinsamer Ressourcen
- Box-and-Line-Diagramme reichen oft nicht als Architekturdarstellung
  - Sie spezifizieren Komponenten nicht genau, sondern
  - Sie vermitteln nur einen intuitiven Eindruck davon, was eine Komponente tut (z.B. "Datenbank")
  - Sie informieren nicht, warum die Architektur so und nicht anders aussieht

***Die* Definition des Software Engineering Institute (SEI)**

The software architecture of a program or computing system is

- the structure or structures of the system, which comprise software components,
- the externally visible properties of those components, and
- the relationships among them.

[*Software Architecture in Practice*, Len Bass, Paul Clements, and Rick Kazman]

Konsequenzen aus der Definition 3/4

Beziehungen ("relationships") sind mehr als *Konnektoren* ("connectors"):

- Konnektoren gibt es nur zur Laufzeit
  - schickt-Daten-an
  - ruft auf
  - stößt-an
- Beziehungen existieren (zusätzlich) auch außerhalb der Laufzeit
  - ist-ein-Teil-von
  - erbt-von
  - ist-Instanz-von

Die **Definition des Software Engineering Institute (SEI)**

The software architecture of a program or computing system is

- the structure or structures of the system, which comprise software components,
- the externally visible properties of those components, and
- the relationships among them.

[*Software Architecture in Practice*, Len Bass, Paul Clements, and Rick Kazman]

Konsequenzen aus der Definition 4/4

- Jedes System *besitzt* eine Architektur
  - *If you don't develop an architecture, you will get one anyway – and you might not like what you get!*
  - Jedes System besteht aus Komponenten mit Beziehungen untereinander
  - Das einfachste System besteht aus einer Komponente mit Beziehungen zu sich selbst
- Aber:
  Es macht einen Unterschied, ob die Architektur eines Systems bekannt ist oder nicht;
  Stichpunkte in diesem Zusammenhang:
  - Einhalten von Vorgaben
  - Vergleich Soll-Architektur vs. Ist-Architektur; Architektur-*Recovery*
    (deutsch: Wiederbeschaffung?)
  - Begründung für die Wahl / Ausprägung der Architektur eines Systems

Die **Definition des Software Engineering Institute (SEI)**

The software architecture of a program or computing system is

- the structure or structures of the system, which comprise software components,
- the externally visible properties of those components, and
- the relationships among them.

[*Software Architecture in Practice*, Len Bass, Paul Clements, and Rick Kazman]