NICOLAI M. JOSUTTIS
solutions in time

SYSTEM BAUHAUS

Design objektorientierter Systeme

OOP 2001

Nicolai Josuttis

NICOLAI M. JOSUTTIS
solutions in time

SYSTEM BAUHAUS

Analyse vs. Design

Analyse

ist etwas für

Feiglinge

NICOLAI M. JOSUTTIS
solutions in time

SYSTEM BAUHAUS

Worum geht es?

- Wie designt man objektorientiert?
- Was ist das Besondere an Architekturen im objektorientierten Umfeld?
- Beispiele aus der Praxis
- keine Design-Patterns

These

These:

Objektorientierte Systeme sind nicht objektorientiert

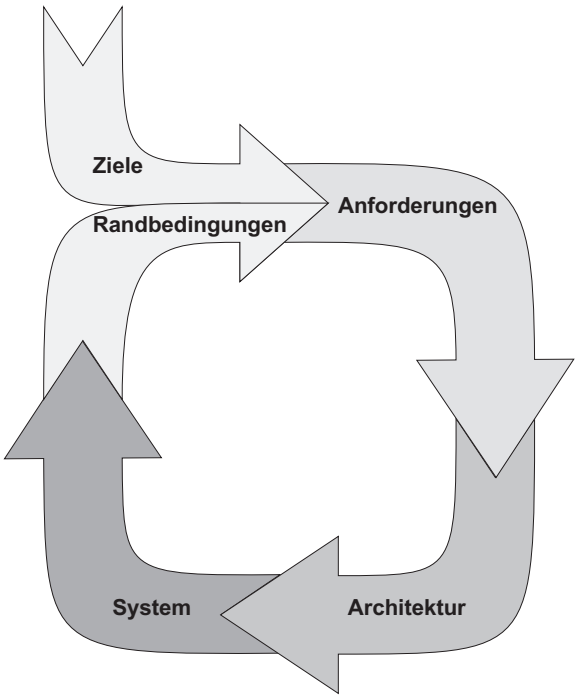
Einschränkung:

- Es geht um „ernsthafte Systeme“
  - keine „Spielzeug-Systeme“
  - keine Schulungsbeispiele
- Das ist nicht die ganze Wahrheit

NICOLAI M. JOSUTTIS
solutions in time

SYSTEM BAUHAUS

Architecture Business Cycle



Kategorien von Anforderungen

System-Randbedingungen

- Der Zweck des Produkts
- Kunden und andere Beteiligte/Betroffene
- Nutzer des Produkts
- Randbedingungen für das Projekt
- Namenskonventionen und Definitionen
- Relevante Fakten
- Annahmen

Projektrandbedingungen

- Offene Punkte
- Fertiglösungen
- Neue Probleme
- Aufgaben
- Inbetriebnahme und Migration
- Risiken
- Kosten
- Benutzerdokumentation

Funktionale Anforderungen

- Die Abgrenzung des Systems
- Anforderungen an Funktionen und Daten des Systems

Nichtfunktionale Anforderungen

- Oberflächenanforderungen
- Benutzbarkeitsanforderungen
- Performance/Durchsatz/Sicherheit
- Operationelle Anforderungen
- Wartungs- und Portierungsanforderungen
- Zugriffsschutzanforderungen
- Politische Anforderungen
- Gesetzliche Anforderungen

”Warteraum” / Sonstiges

NICOLAI M. JOSUTTIS
solutions in time

SYSTEM BAUHAUS

Anforderungen und Objektorientierung

These:

Objektorientierte Systeme sind nicht objektorientiert

denn:

- Die Behandlung von Anforderungen ist weitgehend unabhängig von der Frage, ob objektorientiert designt wird

Doch das ist nicht die ganze Wahrheit, denn:

- Man kann UML-Diagramme als Diskussionsvorlage verwenden
- Man kann die Anforderungen aus objektorientierter Sicht kategorisieren

Hilfsmittel für OOD

- **Architekturen**
  - Die Grobstruktur des Systems bzw. der Software
- **Design Patterns (Entwurfsmuster)**
  - Lösung zu einem Problem in einem Kontext
  - vordefiniertes Zusammenspiel einiger Klassen und Objekte
  - in der Praxis mehrfach erprobt
- **Werkzeuge und Tools**
  - Programmiersprachen
  - Bibliotheken
  - Datenbanken
  - ...

Architektur ist ...

- ... a framework for change

[Tom DeMarco]

- Die Systemarchitektur beschreibt die Gesamtorganisation des Systems in Komponenten oder Teilsysteme. Sie stellt den Kontext bereit, innerhalb dessen in späteren Entwurfsphasen detaillierte Entscheidungen getroffen werden.

[James Rumbaugh]

- Architektur ist die logische und physikalische Struktur eines Systems, geschmiedet aus allen strategischen und taktischen Entwurfsentscheidungen während der Entwicklung

[Grady Booch]

Definitionen von Software-Architektur

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.

[Bass, Clemens, Kazman 1998 in „Software Architecture in Practice“]

The **architecture of an IT system** is the structure or structures of the system, which comprise

- software and hardware components,
- the externally visible properties of those components, and
- the relationships among them.

[Youngs, Redmond-Pyle, Spaas, Kahan 1999 in „A Standard for Architecture Description“]

Definition von Architektur

- **Architektur** ist
  - die **Struktur oder Strukturen eines Systems**
- **Architektur** umfaßt
  - die **Komponenten**
  - die nach außen sichtbaren **Eigenschaften** der Komponenten
  - die **Beziehungen** zwischen den Komponenten

Objektorientierte Architektur-Komponenten

Offen ist:

- Was sind die Komponenten (Elemente) eines objektorientierten Systems?
  - Objekte/Instanzen?
  - Pakete/Packages?
  - Subsysteme?
  - Module?
  - Komponenten (Beans etc.)?
- Was sind die Beziehungen zwischen diesen Komponenten?

NICOLAI M. JOSUTTIS
solutions in time

SYSTEM BAUHAUS

Erfahrungen erster OO-Systeme

- Ein Fehler der frühen 90‘er war, Objekte als Komponenten objektorientierter Systeme zu betrachten
- Erste objektorientierte Groß-Projekte mit erfahrenen Programmieren führten zu massiven Problemen (Mentor Graphics, Borland, Taligent, ...)
- Von den gängigen Entwurfsmethoden unterstützte nur Booch das Konzept der Gruppierung (class categories, subsystems)
- „*Object-oriented computing has failed*“ [Byte Magazine, 1994]

NICOLAI M. JOSUTTIS
solutions in time

SYSTEM BAUHAUS

Don‘t forget

- „Objektorientierung macht es einfach,
  Dinge komplex zu machen“
  [André Weinand 1997 in „Tücken der Objektorientierung“]

- Das menschliche Gehirn mag keine Zyklen
  [frei nach Minsky 1985 in „The Socienty of Mind“]

- „So flexibel wie nötig,
  nicht so flexibel wie möglich“
  [André Weinand 1997 in „Tücken der Objektorientierung“]

NICOLAI M. JOSUTTIS
solutions in time

SYSTEM BAUHAUS

Erfolgreiche Systeme

Eigenschaften erfolgreicher komplexer Systeme:

- Schichten/Layers
- einfache Hierarchien
- entkoppelte Subsysteme
- echte Dekomposition
  - Interaktion innerhalb statt zwischen Subsystemen

[siehe auch: Simon, „The Science of the Artificial“, 1992]

NICOLAI M. JOSUTTIS
solutions in time

SYSTEM BAUHAUS

Erfolgreiche Systeme

Eigenschaften erfolgreicher komplexer Systeme:

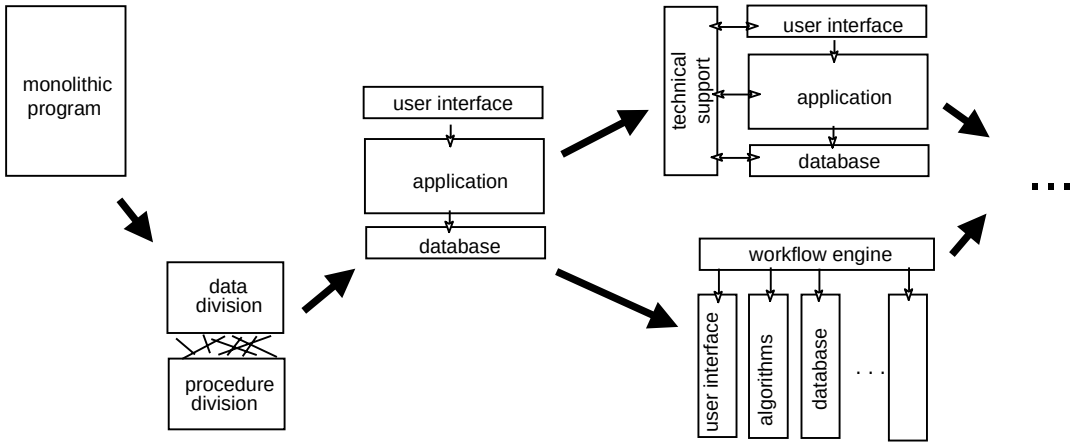
- keine Objekte
- keine Vererbung

auf oberster Ebene

NICOLAI M. JOSUTTIS
solutions in time

SYSTEM BAUHAUS

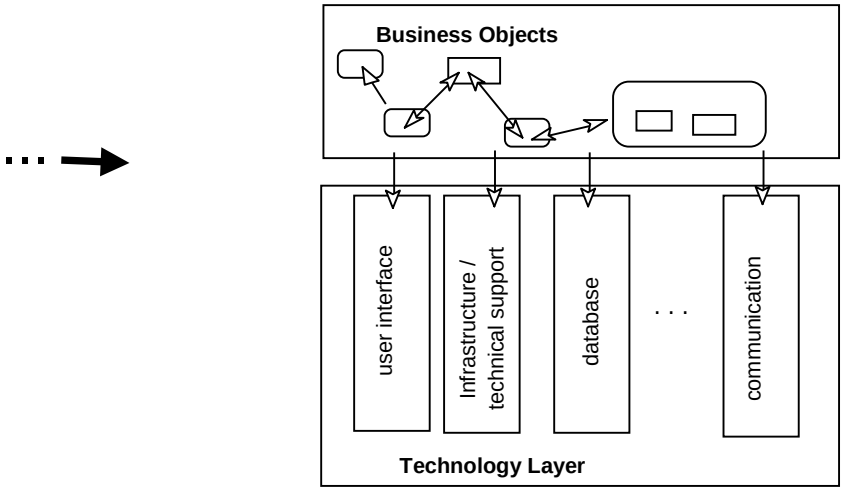
Architekturen im Wandel der Zeiten



[aus Hatley,Hruschka,Pirbhai 2000 „Process for System Architecture and Requirement Engineering“]

Architekturen im Wandel der Zeiten

Business Object Architecture:

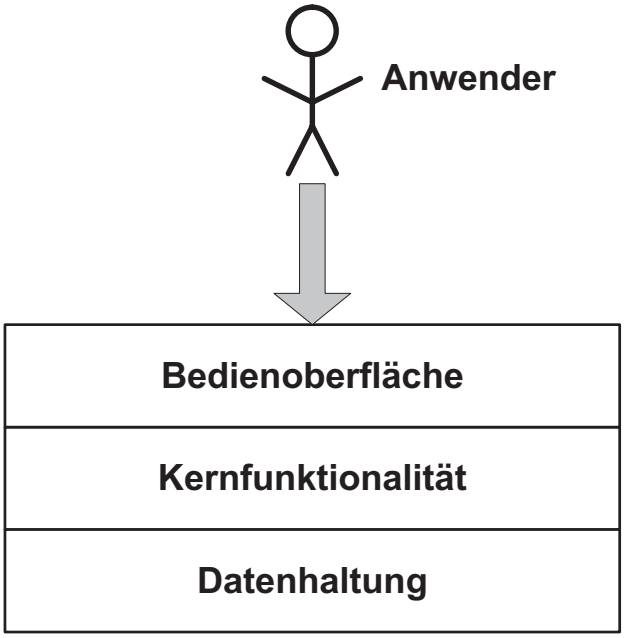


[aus Hatley,Hruschka,Pirbhai 2000 „Process for System Architecture and Requirement Engineering“]

NICOLAI M. JOSUTTIS
solutions in time

SYSTEM BAUHAUS

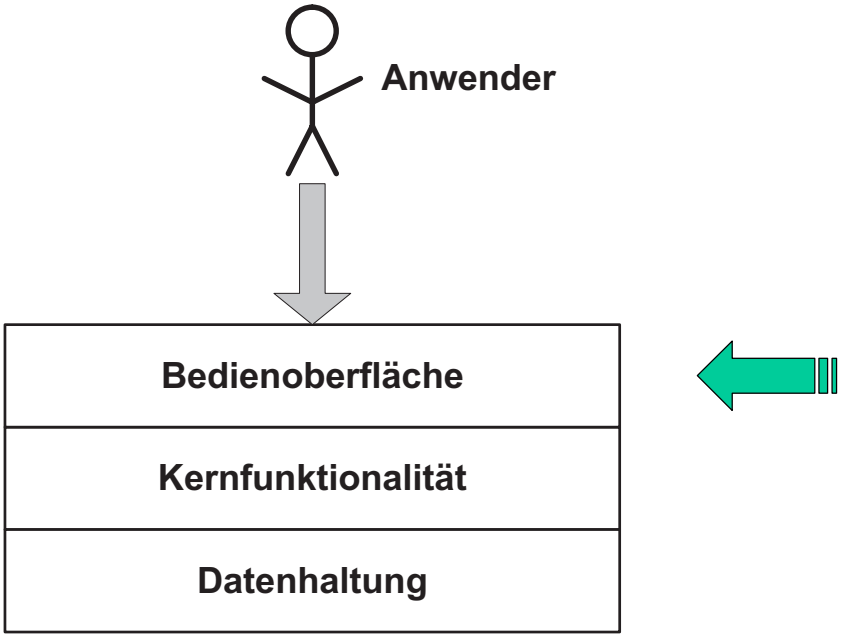
Standard-Schichten von Software-Systemen



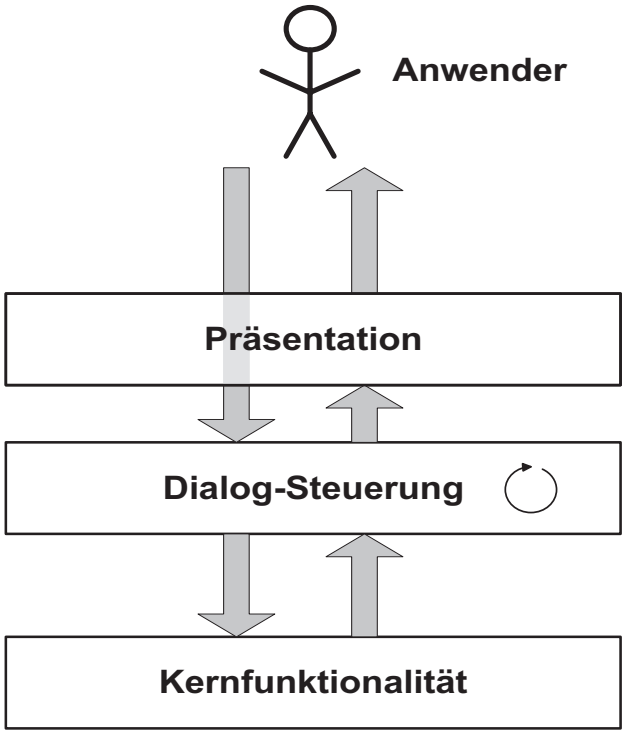
NICOLAI M. JOSUTTIS
solutions in time

SYSTEM BAUHAUS

Standard-Schichten von Software-Systemen



Bedienoberfläche: Seeheim-Modell

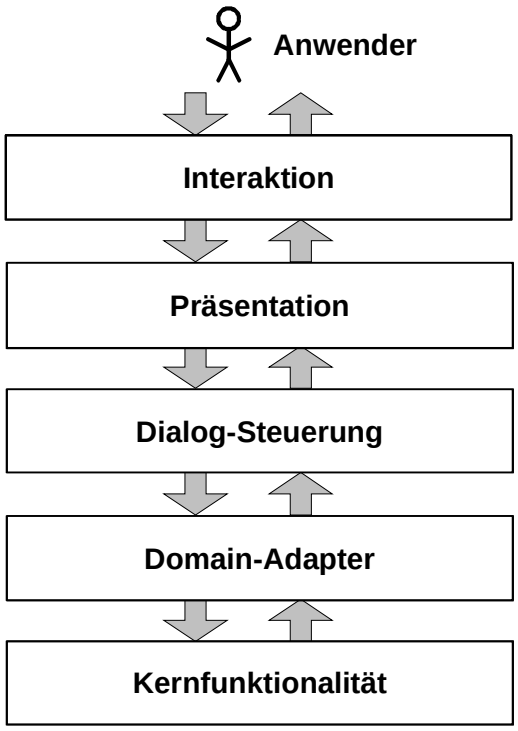


1985

NICOLAI M. JOSUTTIS
solutions in time

SYSTEM BAUHAUS

Bedienoberfläche: Arch-Modell

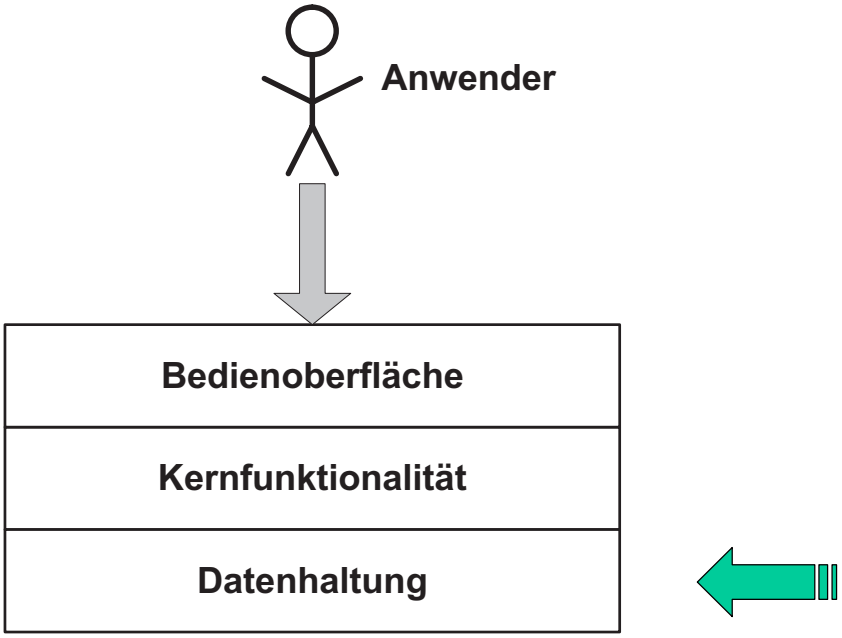


1992

NICOLAI M. JOSUTTIS
solutions in time

SYSTEM BAUHAUS

Standard-Schichten von Software-Systemen



Datenhaltung

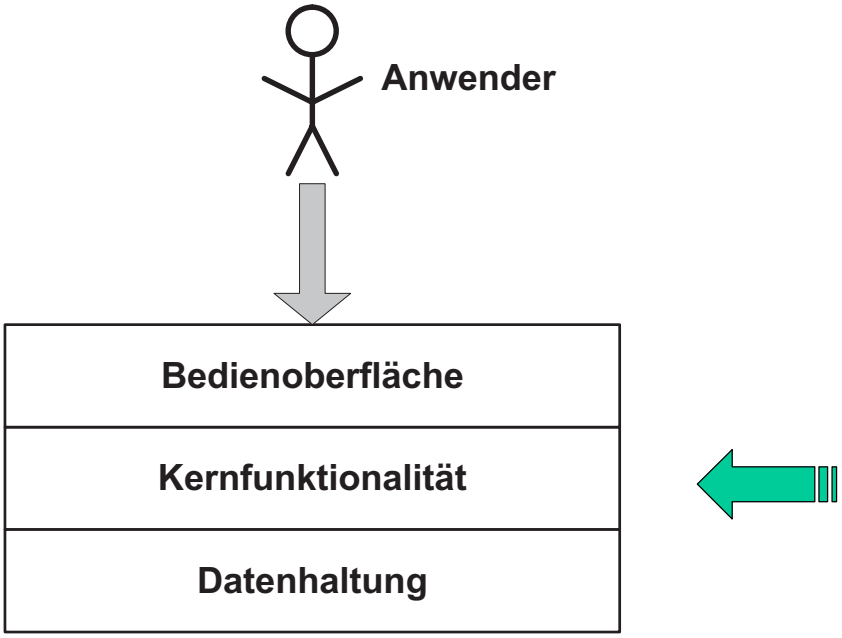
Datenhaltung:

- relationale Datenbank
- objektorientierte Datenbank
- Legacy
  - Host/Mainframe
- Fremdsysteme
  - SAP
  - ...

NICOLAI M. JOSUTTIS
solutions in time

SYSTEM BAUHAUS

Standard-Schichten von Software-Systemen



NICOLAI M. JOSUTTIS
solutions in time

SYSTEM BAUHAUS

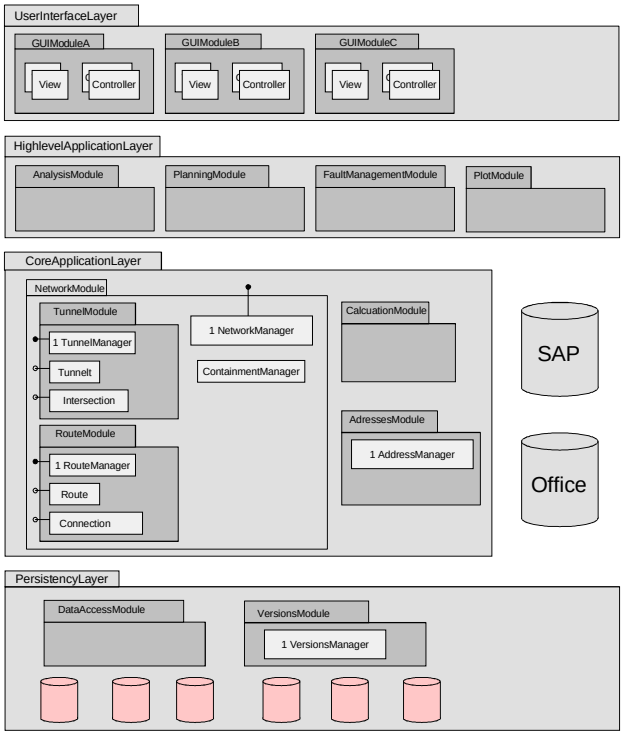
Beispiel für eine 4-Schichten-Architektur

Präsentation

Highlevel-Funktionalität

Basis-Funktionalität

Datenhaltung



Architekturen und Objektorientierung

These:

Objektorientierte Systeme sind nicht objektorientiert

denn:

- System-Komponenten sind unabhängige Subsysteme mit einfachen Daten-Schnittstellen

Doch das ist nicht die ganze Wahrheit, denn:

- Was ist mit verteilten Objekten (z.B. CORBA)?

NICOLAI M. JOSUTTIS
solutions in time

SYSTEM BAUHAUS

OO-System mit Host-Anbindung

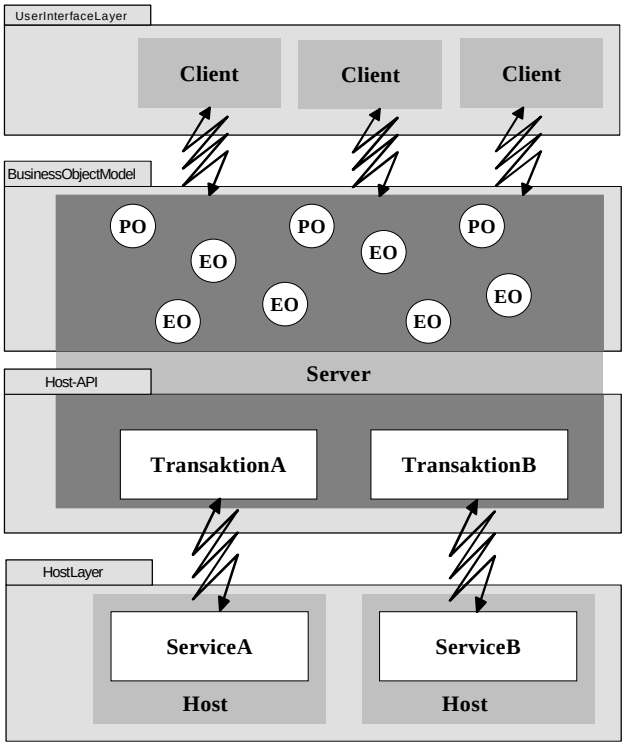
Präsentation

Business Object Model

Datenhaltung

Host-API

Host/Mainframe



NICOLAI M. JOSUTTIS
solutions in time

SYSTEM BAUHAUS

CORBA in realen Systemen

- Es liegt nahe, vom Client aus auf alle BOM-Objekte zuzugreifen
- Das ist problematisch:
  - Keine kleine entkoppelte Schnittstelle zwischen Kernfunktionalität und Präsentation
    - keine unabhängigen Tests
    - Änderungen bleiben nicht lokal
  - Viele Client Stubs (Code zum Zugriff auf Objekte) machen Thin-Clients ziemlich thick
  - Jeder Attributzugriff auf dem Client geht über das Netzwerk
- Auch mit CORBA gelten die Grundregeln zur Strukturierung von Systemen

CORBA-System mit Host-Anbindung

Präsentation

Thin Client

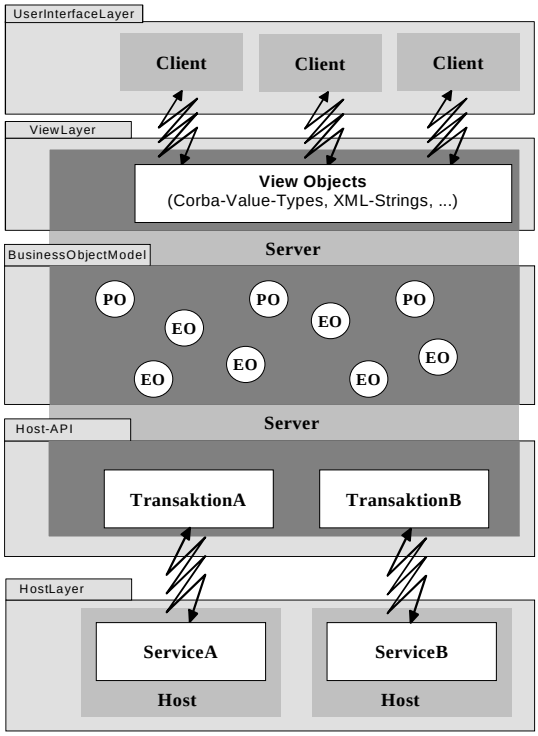
View Layer

Business Object Model

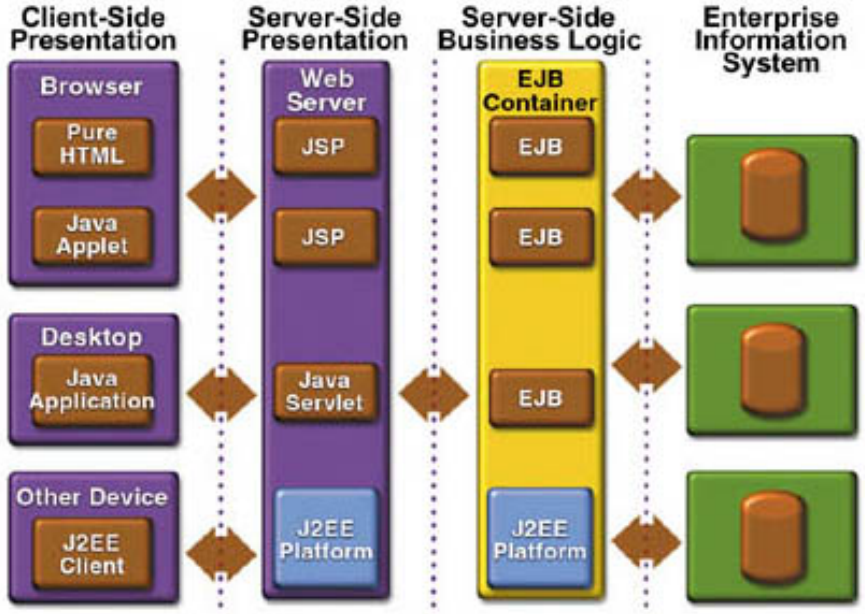
Datenhaltung

Host-API

Host/Mainframe



J2EE Application Model



NICOLAI M. JOSUTTIS
solutions in time

SYSTEM BAUHAUS

(J2E)E-Architektur

Präsentation

Thin Client

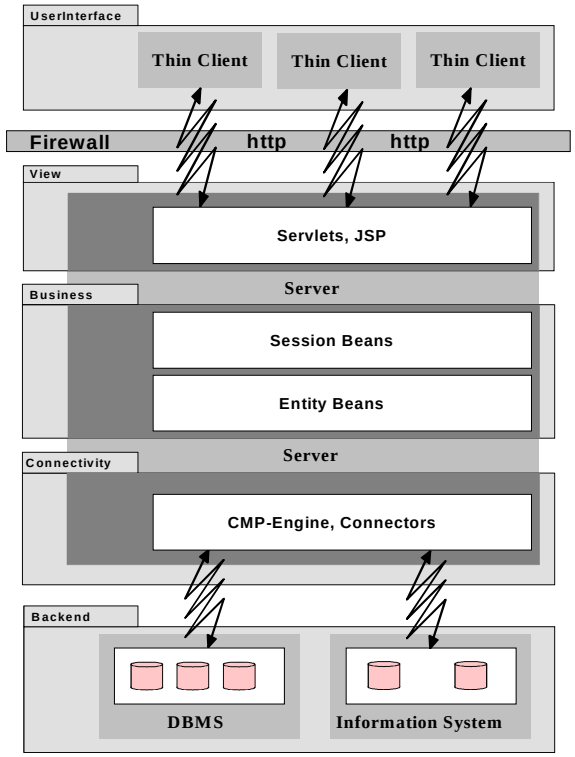
View Layer

Business Object Model

Datenhaltung

Data-API

DBMS, Host, SAP



Anforderungen und Objektorientierung

These:

Objektorientierte Systeme sind nicht objektorientiert

denn:

- System-Komponenten sind unabhängige Subsysteme mit einfachen Daten-Schnittstellen
- Das gilt auch für verteilte Systeme (CORBA, EJB/AS)

Doch das ist nicht die ganze Wahrheit, denn:

- Verteilte Objekte (CORBA, RMI) helfen bei der Kommunikation zwischen Client und Server
- Objekttechnologie hilft bei der Implementierung

Flexibilität bei Erweiterungen

Doch das ist nicht die ganze Wahrheit, denn:

- Jedes nicht-triviale System ist einzigartig

Eine kleine Randbedingung wie die Häufigkeit von Erweiterungen kann eine Architektur völlig umkrempeln

- Mögliche Design-Konsequenz:
  - Erweiterungen hardcodiert einbauen
  - Polymorphie nutzen
  - Komponenten-Technologien nutzen
  - Meta-Ansatz

NICOLAI M. JOSUTTIS
solutions in time

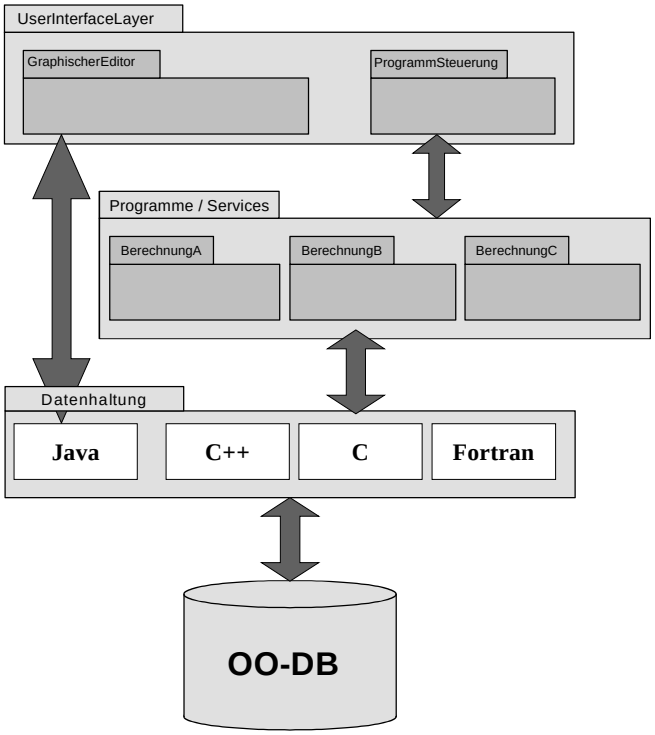
SYSTEM BAUHAUS

Einzelstück-Design mit Meta-Ansatz

Präsentation

Funktionalität

Datenhaltung



NICOLAI M. JOSUTTIS
solutions in time

SYSTEM BAUHAUS

Zusammenfassung

These:

Objektorientierte Systeme sind nicht objektorientiert

denn:

- Die Behandlung von Anforderungen ist weitgehend unabhängig von der Frage, ob objektorientiert designt wird
- System-Komponenten sind unabhängige Subsysteme mit einfachen Daten-Schnittstellen
- Das gilt auch für verteilte Systeme (CORBA, EJB/AS)

NICOLAI M. JOSUTTIS
solutions in time

SYSTEM BAUHAUS

Zusammenfassung

These:

Objektorientierte Systeme sind nicht objektorientiert

Doch das ist nicht die ganze Wahrheit, denn:

- Objekttechnologien werden lokal verwendet
- Objekttechnologien helfen auch übergreifend
- Jedes nicht-triviale System ist einzigartig

Zusammenfassung

Tendenz zu verteilter n-Tier-Architektur:

<table>
<tr><td rowspan="2">Präsentation</td><td>User Interface Layer
(Client Side Presentation)</td><td>Client</td></tr>
<tr><td>View Layer
(Server Side Presentation)</td><td rowspan="4">Server</td></tr>
<tr><td rowspan="2">Kernfunktionalität</td><td>Business Layer
(Session Layer)</td></tr>
<tr><td>Entity Layer</td></tr>
<tr><td rowspan="2">Datenhaltung</td><td>Connectivity Layer</td></tr>
<tr><td>Data Layer</td><td>Backend</td></tr>
</table>

NICOLAI M. JOSUTTIS
solutions in time

SYSTEM BAUHAUS

Ausblick

- „I built a lot of large systems,
  but I never built a complex system“
  [Kerth, Meszaros, Doble]
- „Start stupid and evolve“
  [Kent Beck]
- „Abstract is not vague“
  [Kerth, Meszaros, Doble]
- „When it is not important to make a decision,
  it is important not to make a decision“
  [Desmond D‘Souza]

NICOLAI M. JOSUTTIS
solutions in time

SYSTEM BAUHAUS

Noch Fragen, Hauser?

http://www.josuttis.de

solutions@josuttis.de

Nicolai Josuttis
Berggarten 9
D-38108 Braunschweig

Tel.: **05309 / 5747**
0700 / 5678 8888
0700 / JOSUTTIS