Diplomarbeitspräsentationen der Fakultät für Informatik

Architekturzentrierte Modellgetriebene Softwareentwicklung

Fallbeispiel und Evaluierung

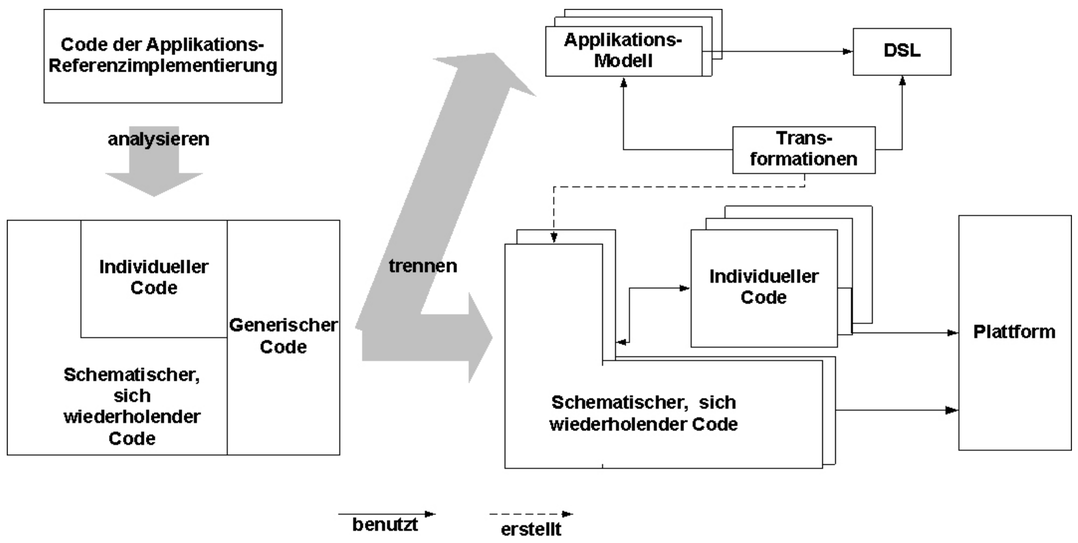
Magisterstudium:
Wirtschaftsinformatik

Benedikt,Weismann

Technische Universität Wien
Institut für Softwaretechnik und Interaktive Systeme
Arbeitsbereich: Business Informatics
BetreuerIn: o.Univ.- Prof. Mag. Dipl.-Ing. Dr. Gerti Kappel
Univ.-Ass. Dipl.-Ing. Dr. Gerhard Kramler

Was ist MDSD?

- **Modell = Sourcecode**
- Sourcecode wird aus einem Modell durch einen **Generator** erstellt
- Modelle orientieren sich am Problemraum = **Domänen-orientiert**
- Domänenspezifische **Plattformen** bilden das Fundament



Quelle: [SV05] Seite 17

Die Ziele der Methodik

- Steigerung der **Entwicklungsgeschwindigkeit**
- Steigerung der **Softwarequalität**
- Steigerung des Grads an **Wiederverwendung**
- Hersteller- und Plattformunabhängikeit **nicht** im Vordergrund (vgl. MDA)

FAZIT: Pragmatischer Ansatz der nur Ziele verfolgt, die mit heute verfügbaren Technologien realisierbar sind!

Merkmale von AC-MDSD

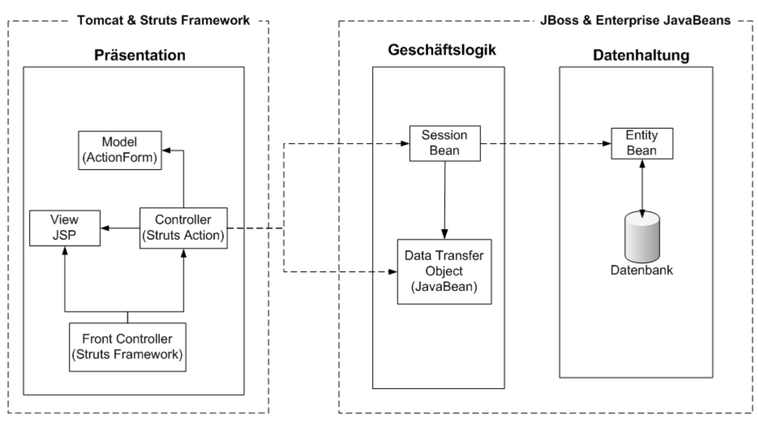
- Domäne = **Softwarearchitektur**
- Architektur trägt **Anwendungsfamilie**
- Generierungsfaktor = **60-80%**
- Generierung des **Infrastrukturcodes**
- **Fachlicher Code** händisch ergänzt

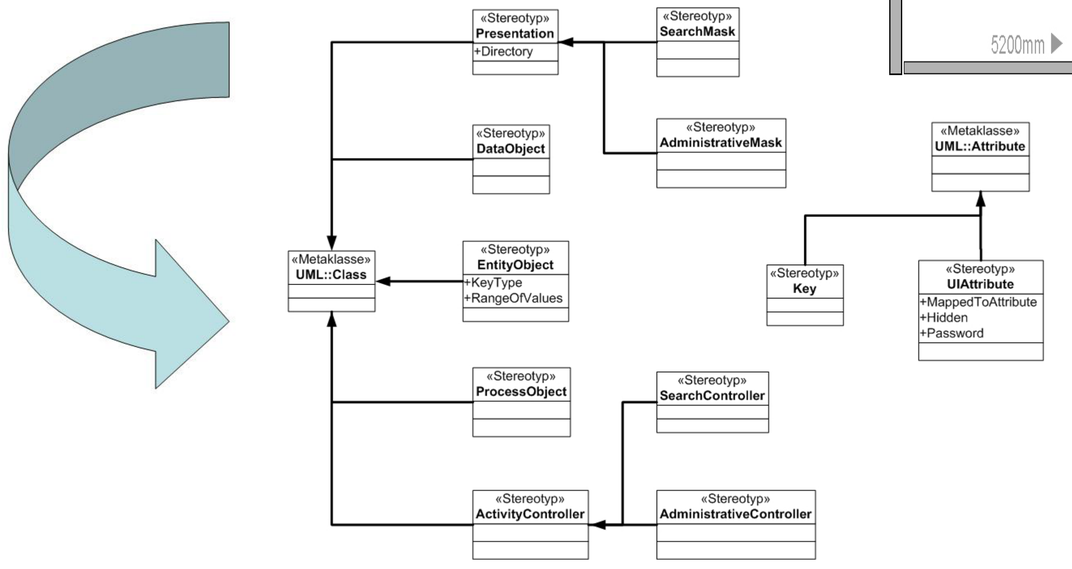
Toolkit: openArchitectureWare

- *openArchitectureWare is a **„tool for building MDSD/MDA tools“***
- Unterstützt bei der Entwicklung domänenspezifischer Generatoren
- **Open source;** Eclipse Plugin
- **http://www.openarchitectureware.org/**
- Modelle und Metamodelle können in **diversen Formaten** vorliegen z.B.: UML, EMF, XML
- **Templatesprache** zur Entwicklung der Transformationsvorschriften
- Möglichkeiten der **Modellverifikation**

Fallbeispiel

- Domäne: Softwarearchitektur für E-Businesssysteme
- Konkrete Anwendung: Onlinebuchhandlung „Bücherwurm“
- Backend auf Basis von EJBs realsiert; Frontendimplementierung durch das Struts-Webframework
- Umsetzung des Metamodells mit Hilfe eines UML-Profils
- Erreichtes **Generierungspotential: 70,8%**
  - Stark schematisiertes Backend: 93%
  - Frontend mit erhöhtem individuellen Codeanteil: 45,2%





Herausforderungen bei der Umsetzung des Fallbeispiels

- **Frage nach der Art der Domäne:**
  - Sind sowohl technische als auch fachliche Domänen geeignet?
- **Metamodellierung & Modellierung:**
  - Drücken die Modelle sowie das Metamodell die zu beschreibenden Sachverhalte korrekt aus?
- **Mangelnde Dokumentation** von openArchitectureWare in Bezug auf die Mächtigkeit

Evaluierung

- **Effizienzsteigerung bei der Entwicklung ähnlicher Applikationen** durch Bündelung von Gemeinsamkeiten in Plattform und Transformationsvorschriten
- **Kostensenkung** durch Automatisierung und Wiederverwendung (vgl. Industrielle Fertigung)
- Sicherstellung von **Annwendungsqualität**:
  - Wohldefinierte Architektur ist eine Grundvorraussetzung!
  - Unabhängigkeit von den Fertigkeiten einzelner Entwickler ist gegeben!
  - Wiederverwendbare Bestandteile werden über mehere Produkte hinweg verbessert!
- Dem Entwickler bleibt **mehr Zeit für neue Technologien und individuelle Anwendungsaspekte.**

Kosten der Einführung

- Anpassung der Organisation
- Schulung von Mitarbeitern
- Entwicklung der Domänenarchitektur
- Kosten der eigentlichen Anwendungsentwicklung stark reduziert

Quellenverzeichnis:

[SV05] Stahl, Thomas; Völter Markus: *Modellgetriebene Softwareentwicklung: Techniken, Engineering, Management.* 1. Auflage, dpunkt.verlag, 2005

Kontakt: benedikt_weismann@gmx.at