www.novatec-gmbh.de

NOVATEC
make IT happen!

EJB 3 modellgetrieben – ein Praxisbericht

15. Java Forum Stuttgart
5. Juli 2012

// NovaTec – Ingenieure für neue Informationstechnologien GmbH
// Leinfelden-Echterdingen, München, Frankfurt am Main, Jeddah / Saudi-Arabien

Vorstellung Christian Schwörer

NOVATEC
make IT happen!

NOVATEC
make IT happen!

Christian Schwörer
Senior Consultant

NOVATEC // Ingenieure für neue Informationstechnologien GmbH
Dieselstraße 18/1
D-70771 Leinfelden-Echterdingen

phone +49 711 22040-700
fax +49 711 22040-899
mobile +49 151 1614 7759
christian.schwoerer@novatec-gmbh.de

- seit 2003: Software-Entwickler und Technischer Projektleiter
- seit 2009: Senior Consultant bei der NovaTec GmbH / Leiter Competence Group *Model-Driven Development*
- Beratungsschwerpunkte:
  - Modellgetriebene Softwareentwicklung
  - Java EE, JPA / Hibernate

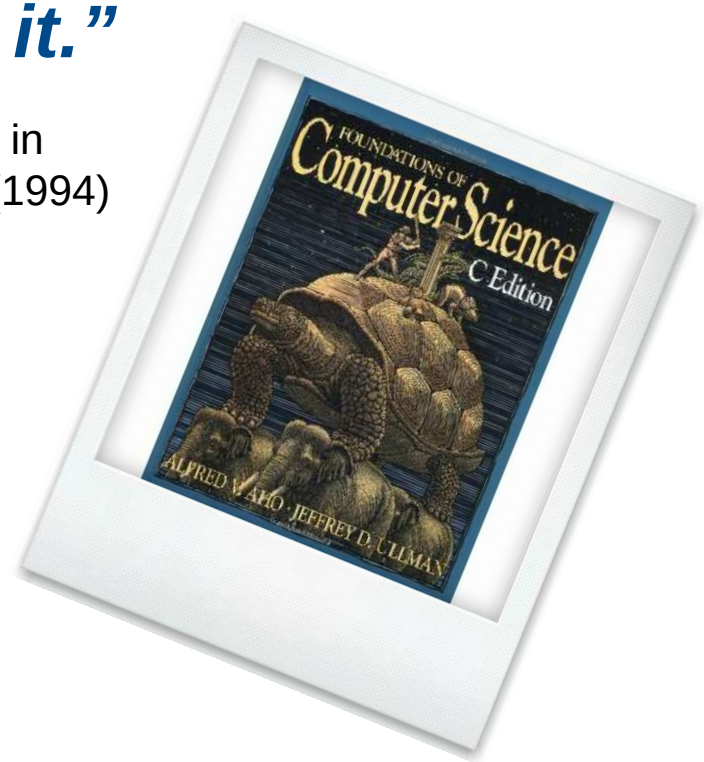
Agenda

- **Kurzeinführung *Modellgetriebene Softwareentwicklung***
- **Migrationsprojekt**
  - Modellierung
  - Generierung
  - Manuelle Vervollständigung
- **Lessons Learned und Best Practices**
- **Fazit und Ausblick**

NOVATEC
make IT happen!

„Computer Science is a science of abstraction – creating the right model for a problem and devising the appropriate mechanizable techniques to solve it.”

Alfred V. Aho & Jeffrey D. Ullman in
„Foundations of Computer Science“ (1994)



NOVATEC
make IT happen!

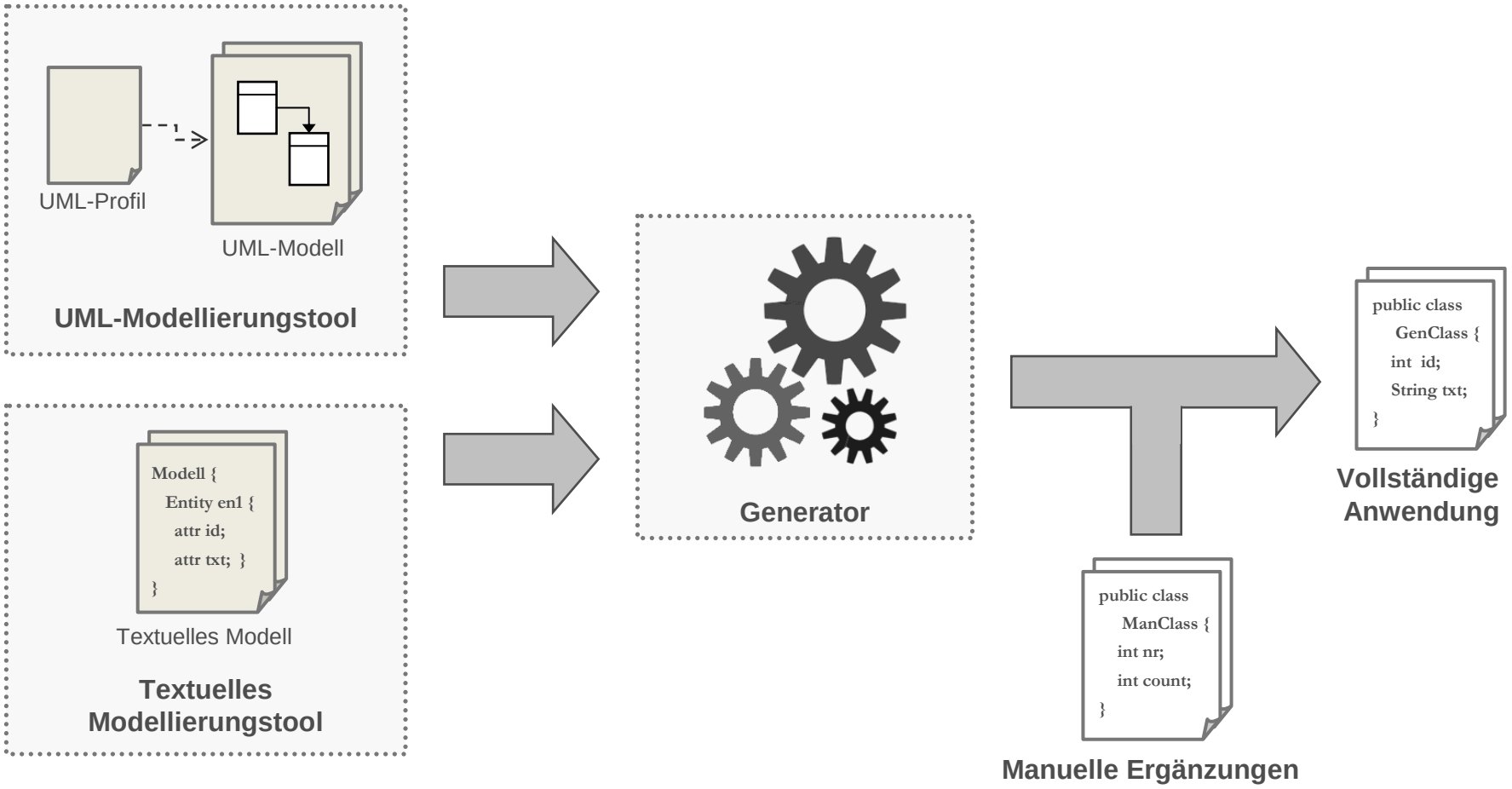
Grundprinzip

- **„Code-zentrierte“ Softwareentwicklung**
  - Modelle dienen ausschließlich zu Dokumentationszwecken
  - Die Anwendungslogik wird vollständig in einer Programmiersprache ausformuliert

- **Modellgetriebene Softwareentwicklung / Model Driven Development (MDD)**
  - Große Teile der Anwendung werden im Modell statt im Quellcode definiert
  - Modelle nehmen somit eine zentrale, treibende Rolle ein und sind in ihrer Bedeutung dem Quellcode gleichgestellt

Ziel: Möglichst viele **Artefakte** eines zu erstellenden Softwaresystems werden **generativ** aus **formalen Modellen** erzeugt

Exemplarische MDD-Werkzeugkette



NOVATEC
make IT happen!

(Erhoffte?) Vorteile durch MDD

- Die Anwendungsentwicklung erfolgt auf einer **höheren Abstraktionsebene**
- **Architekturvorgaben** lassen sich besser durchsetzen
- Zentrale **Framework- oder Architekturänderungen** lassen sich besser ausrollen
- **Entwicklungseffizienz und der Automatisierungsgrad** können erhöht werden
- Die **Wiederverwendung** wird vereinfacht
- Die **Softwarequalität** kann gesteigert werden
- Eine **Plattformunabhängigkeit** lässt sich einfacher realisieren

Agenda

- **Kurzeinführung *Modellgetriebene Softwareentwicklung***
- **Migrationsprojekt**
  - Modellierung
  - Generierung
  - Manuelle Vervollständigung
- **Lessons Learned und Best Practices**
- **Fazit und Ausblick**

NOVATEC
make IT happen!

Rahmenbedingungen MDD-Migrationsprojekt

- **Auftraggeber:** Großer IT-Dienstleisters für Versicherungsunternehmen
- **Ziel:** Anhebung der Zielplattform eines etablierten modellgetriebenen Entwicklungs-prozesses auf eine neue Technologie
  - Ablösung der EJB 2-basierten durch eine EJB 3-basierte Anwendungsplattform
  - Optimale Kombination des „ease of use“ von EJB 3 mit den Vorteilen des modellgetriebenen Vorgehens
  - Entwicklungsplattform wird von 200 Entwicklern zur Anwendungsentwicklung für mehr als 4000 Nutzer genutzt

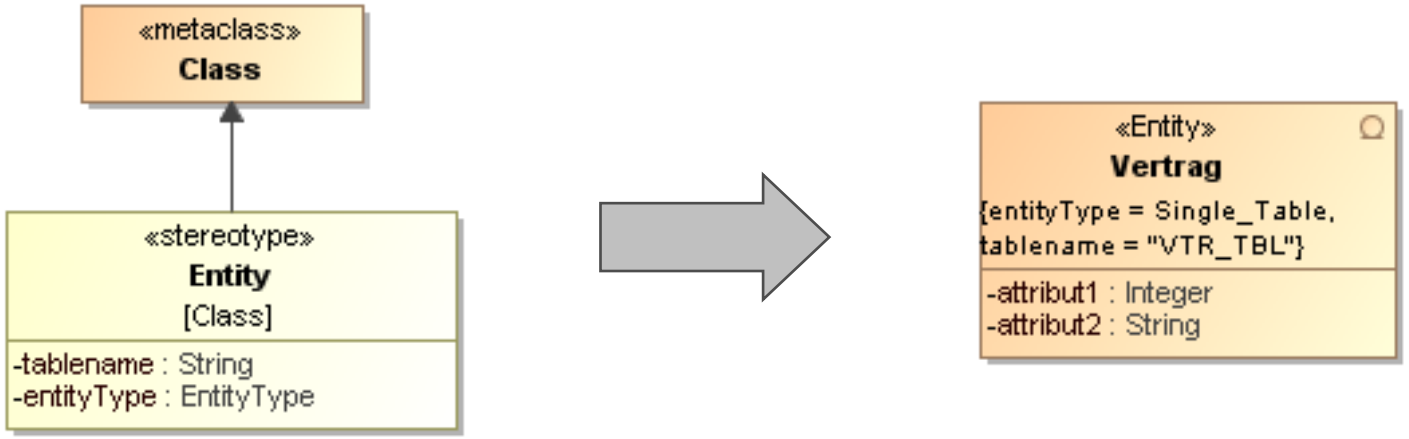
Modellgetriebener Entwicklungsprozess

NOVATEC
make IT happen!

MDD-spezifische Prozessschritte

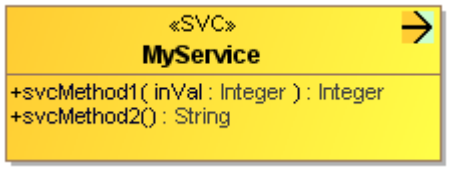
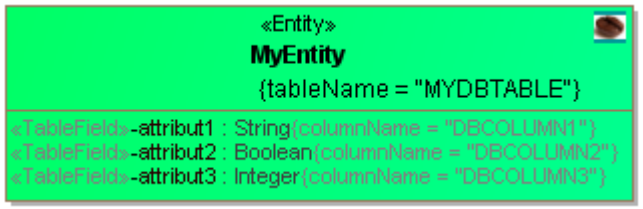
Domänenspezifische Modellierungssprache (DSL) basierend auf der UML

- Domänenspezifische Erweiterung über den *Profil*-Mechanismus
- *Stereotypen* zur Modellierung
- Hinzufügen von spezifischen Informationen über *Tagged Values*



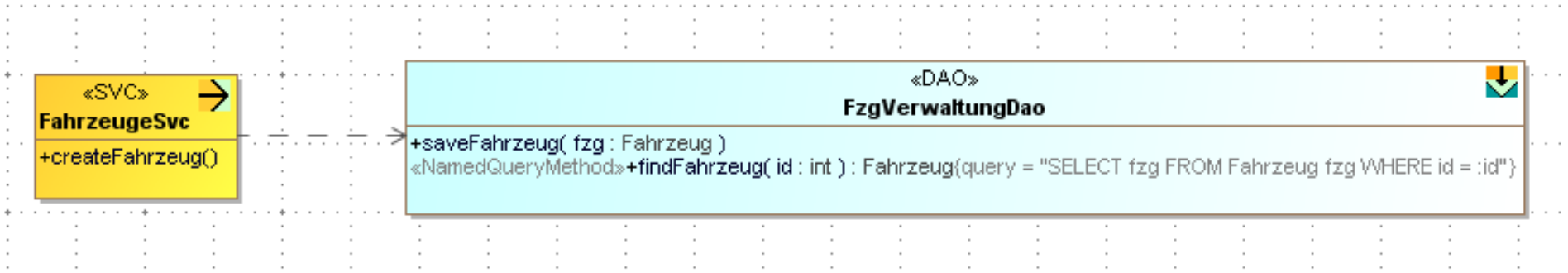
NOVATEC
make IT happen!

Modellierungs- / Architekturstereotypen (Auszug)

Stereotyp	Beschreibung	UML-Visualisierung
SVC	Repräsentiert einen **Service**, der Funktionalität nach außen bereitstellt, bspw. für die Präsentationsschicht	
DAO	Ein **Data Access Object** kapselt den Datenbank-Zugriff	
Entity	Ein fachliches Geschäfts-objekt, d.h. eine persistente **Entity** repräsentiert die Objektsicht auf gespeicherte Daten	

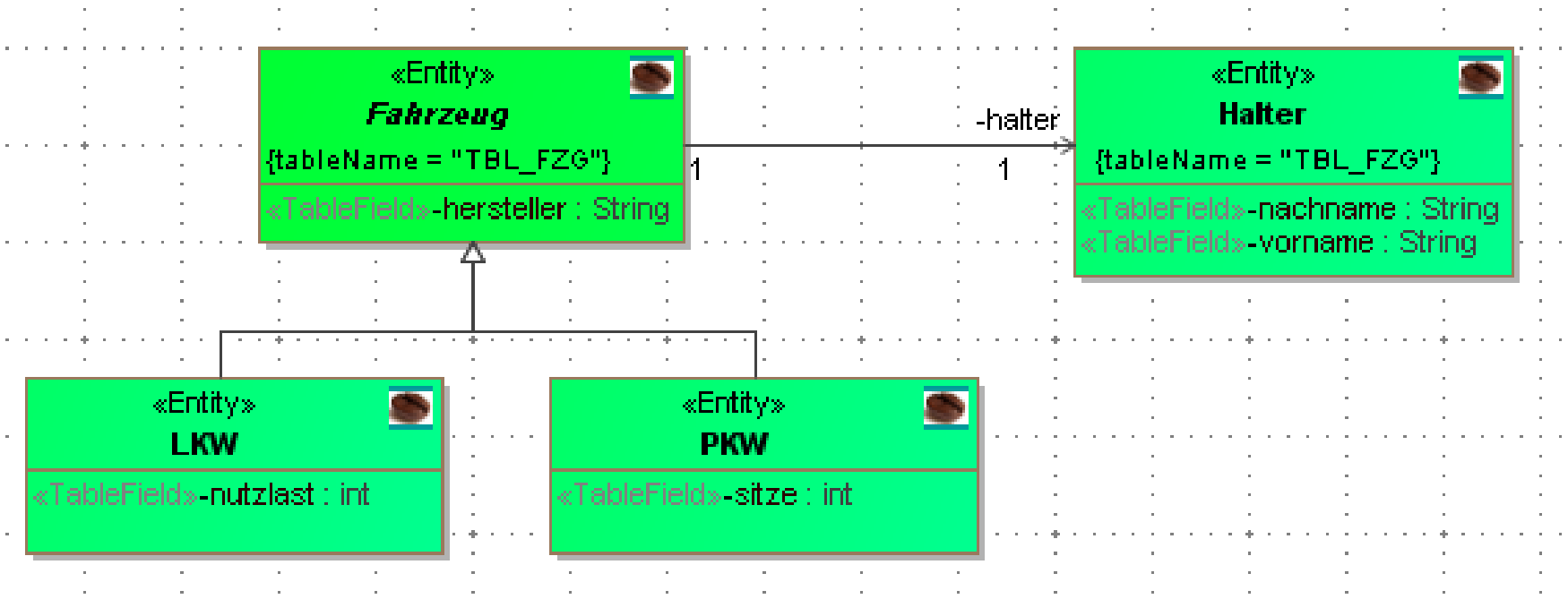
NOVATEC
make IT happen!

Komponente services



NOVATEC
make IT happen!

Komponente `entities`

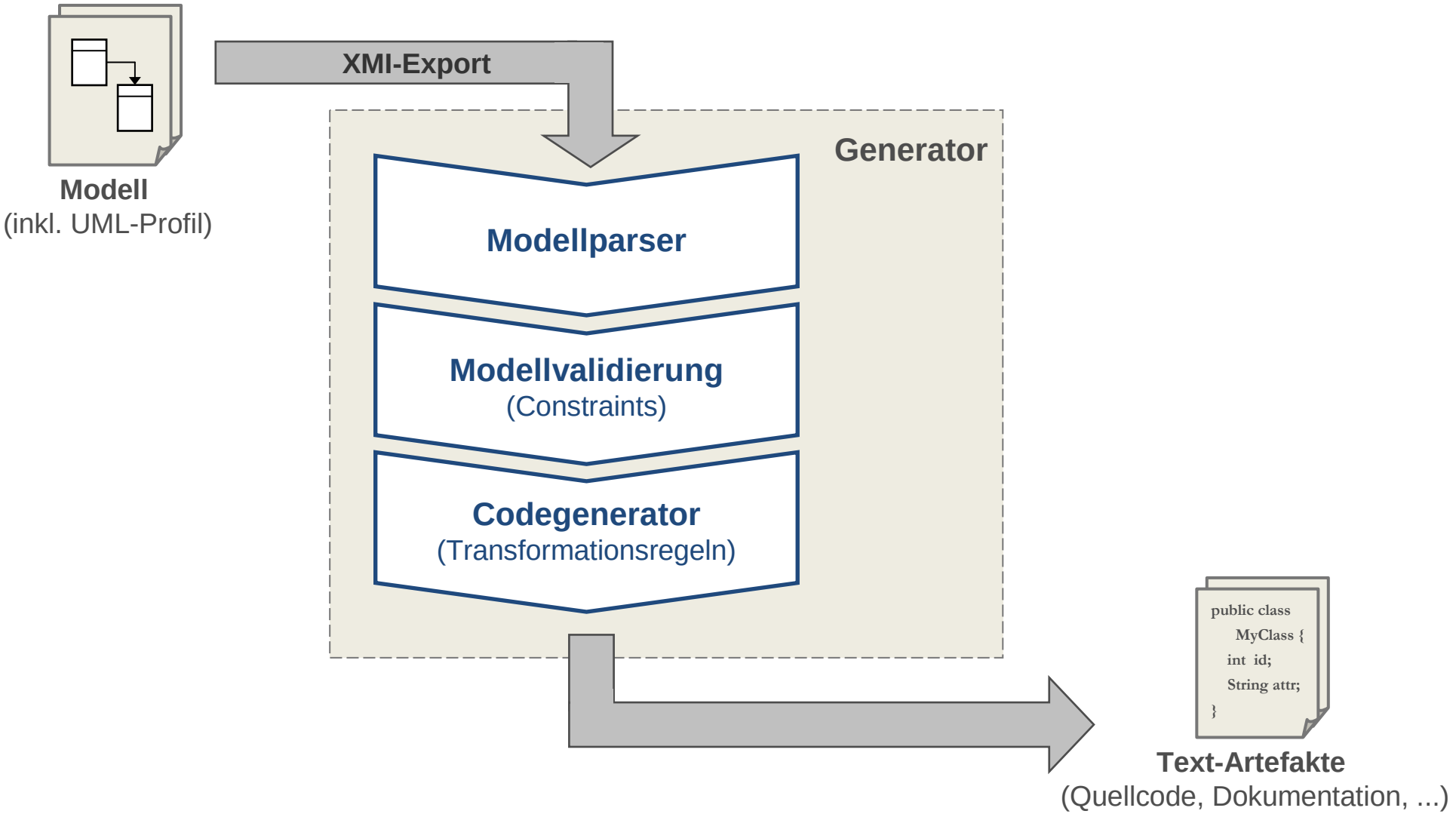


Modellgetriebener Entwicklungsprozess

NOVATEC
make IT happen!

MDD-spezifische Prozessschritte

Generatorprozess



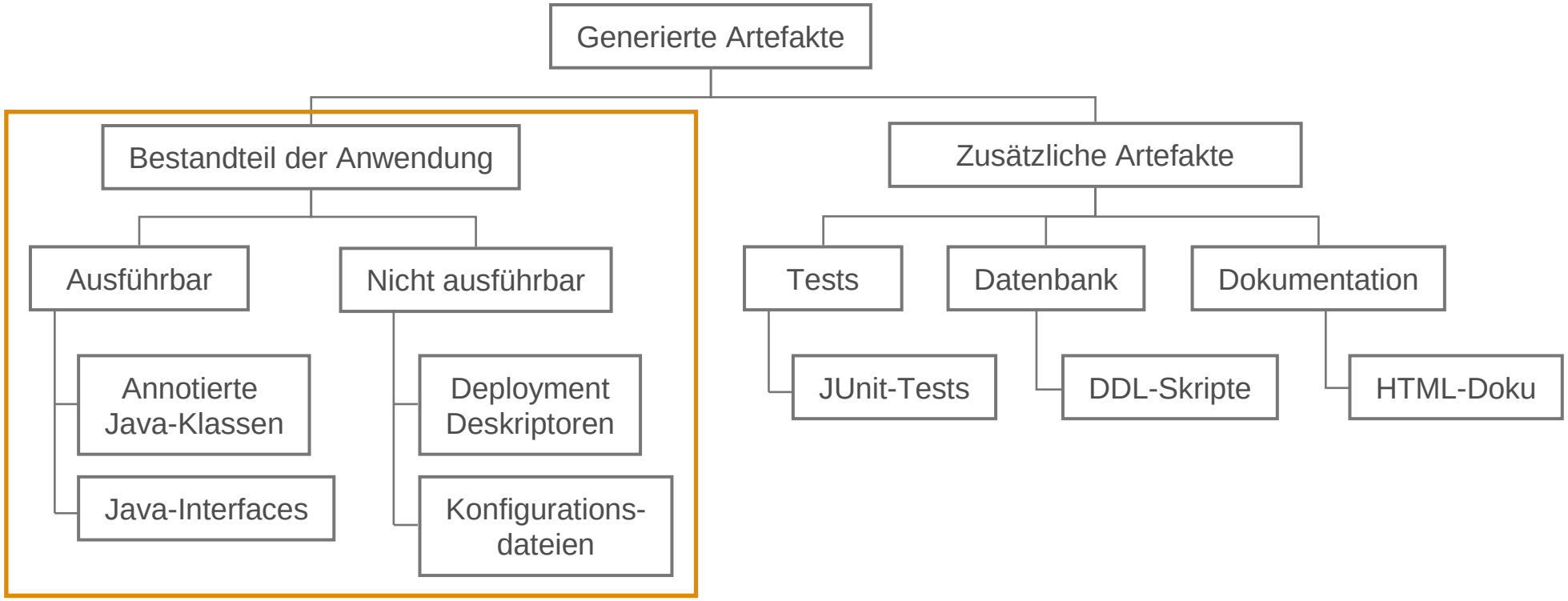
Generator-Erstellung

- Verwendetes Generator-Framework: **openArchitectureWare (oAW)**
  - Bestandteile würden inzwischen ins *Eclipse Modeling Project* übernommen
  - Stellt eigene Sprachen für **Modellvalidierungen** und **Modelltransformationen** bereit

- **Codegenerierung bzw. Modell-zu-Text-Transformation (M2T)**
  - Statisch typisierte Templatesprache ***XPand***

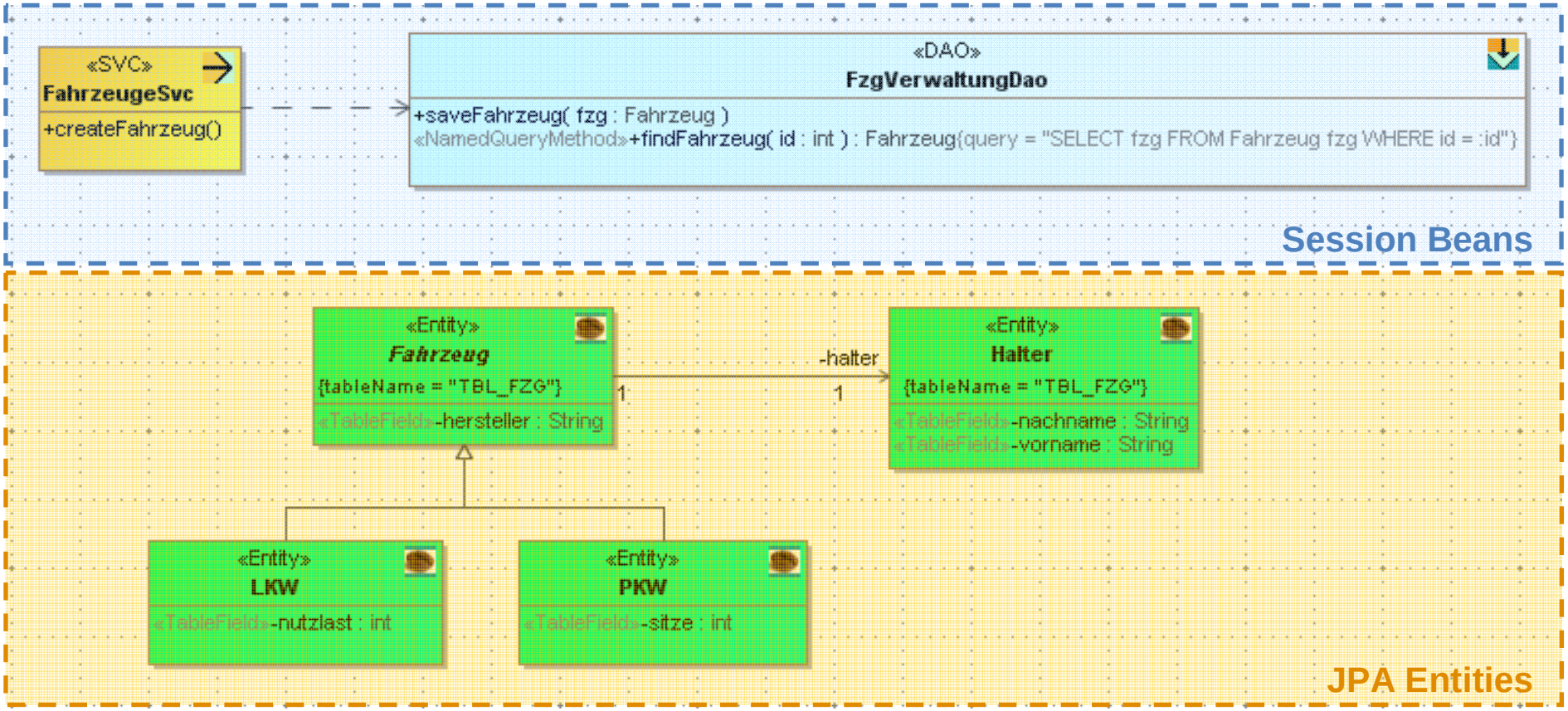
```
«DEFINE javaClass FOR Entity»
 «FILE name + ".java"»
    package «javaPackage()»;
    public class «name» {
      //TODO: implementation
    }
  «ENDFILE»
«ENDDEFINE»
```

Übersicht: Generierte Artefakte



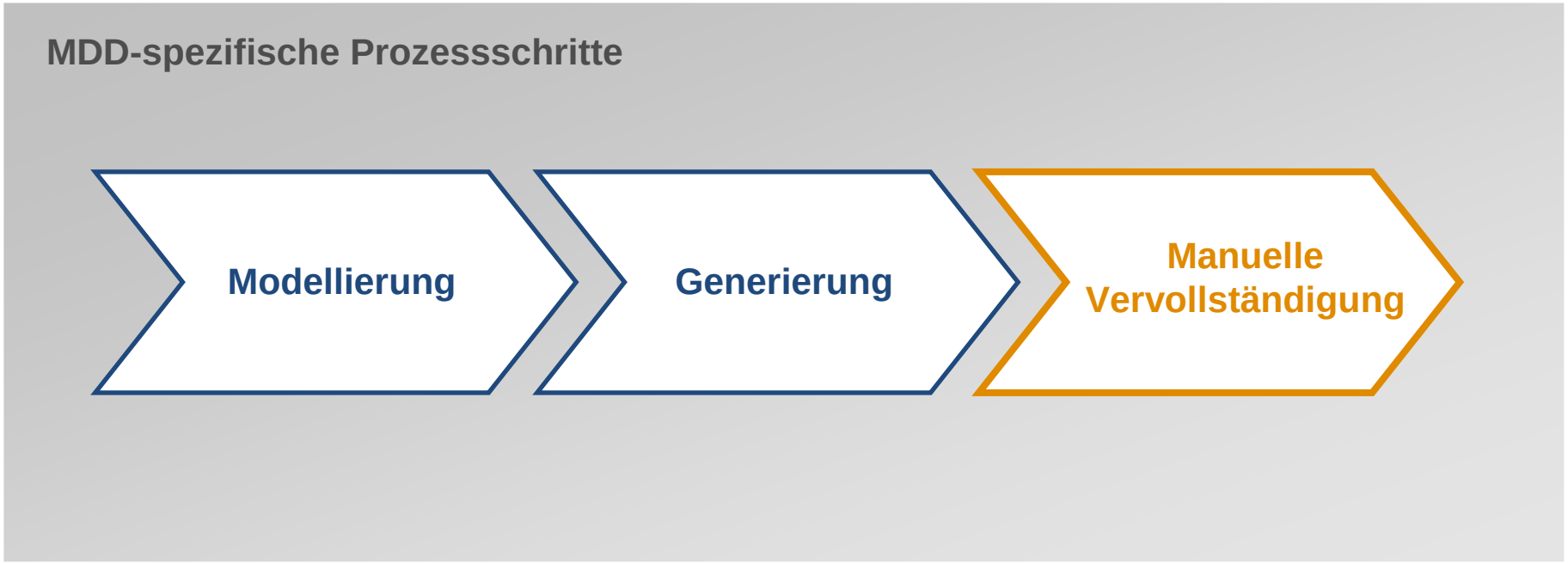
Don’t Repeat Yourself (DRY): Eine Änderung im Modell wirkt sich auf **alle** daraus generierten Artefakte aus.

Zu generierende JEE-Artefakte



Modellgetriebener Entwicklungsprozess

NOVATEC
make IT happen!



Manuelle Vervollständigung

NOVATEC
make IT happen!

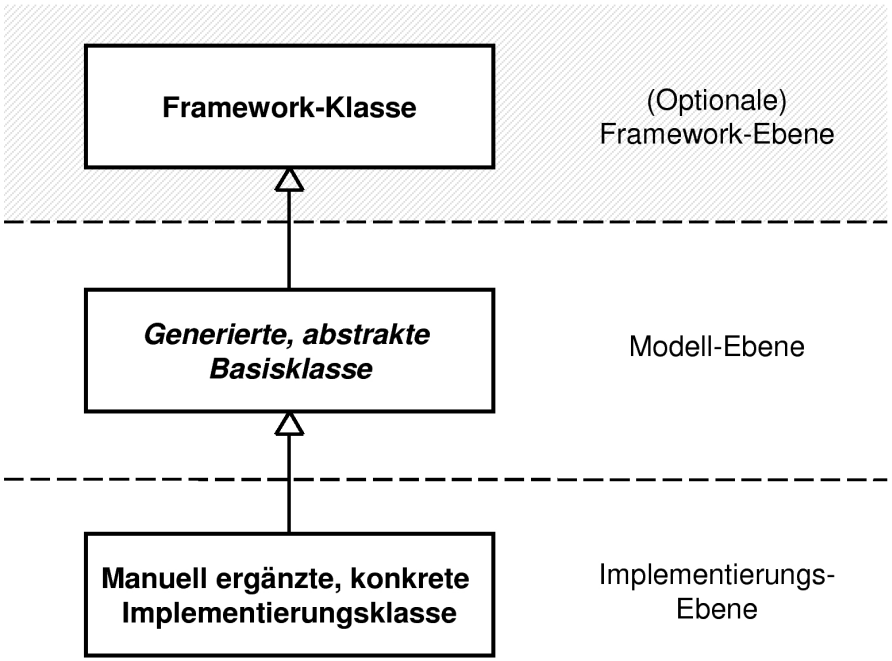
- **Pragmatische MDD: Nicht 100% einer Anwendung werden generiert**
  - Erzeugt wird ein „Implementierungsrahmen“, der große Teile des Infrastrukturcodes enthält
  - Dieser Code kann bis zu zwei Drittel der Anwendung ausmachen
- **Strategie zur Kombination von Generat und manuellen Ergänzungen erforderlich**
  - Artefakte, die Modellinformationen enthalten, werden beim nächsten Generatorlauf überschrieben
  - Manuelle Code-Ergänzungen müssen erhalten bleiben
- **Wichtiger Erfolgsfaktor**
  - Bei einer falsch gewählten / unausgereiften Strategie wir der Entwicklungsprozess starr und unflexibel

Strategien zu Kombination

Verschiedene Strategien denkbar

- Protected Regions, Delegation, Vererbung, Nutzung von Deployment Deskriptoren, …*

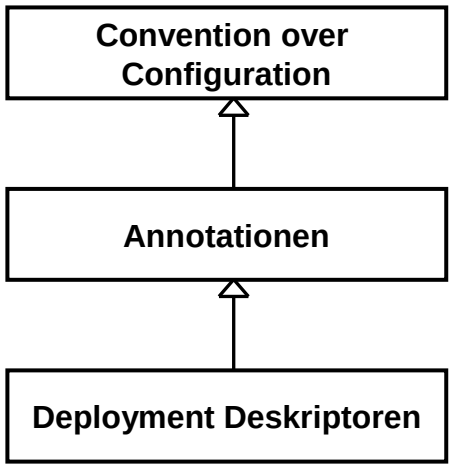
Am besten geeignet: Vererbungsansatz



* Vgl. "Model-driven Beans: EJB 3 modellgetrieben entwickeln" im Java Magazin 3.2012

Realisierte Strategie

- **EJB 3-Metadaten ergeben sich aus Modellinformationen**
  - JPA-Mapping-Informationen
  - Ausprägung einer Session Bean, Angaben zur Transaktionalität, …
- **Problem: Java-Annotationen sind nicht alle ableitbar, bspw:**
  - Ausprägung einer Session Bean (*@Stateful* bzw. *@Stateless*)
  - Definition einer persistenten Entity (*@Entity*)
- **Vererbungsansatz kann nicht 1:1 für EJB 3-Artefakte adaptiert werden**
  - Nutzung der JEE-„Hierarchie“ zur Angabe von Metadaten:



Strategie für EJB 3-Artefakte

NOVATEC
make IT happen!

1. Verwende den Vererbungsansatz soweit wie möglich!

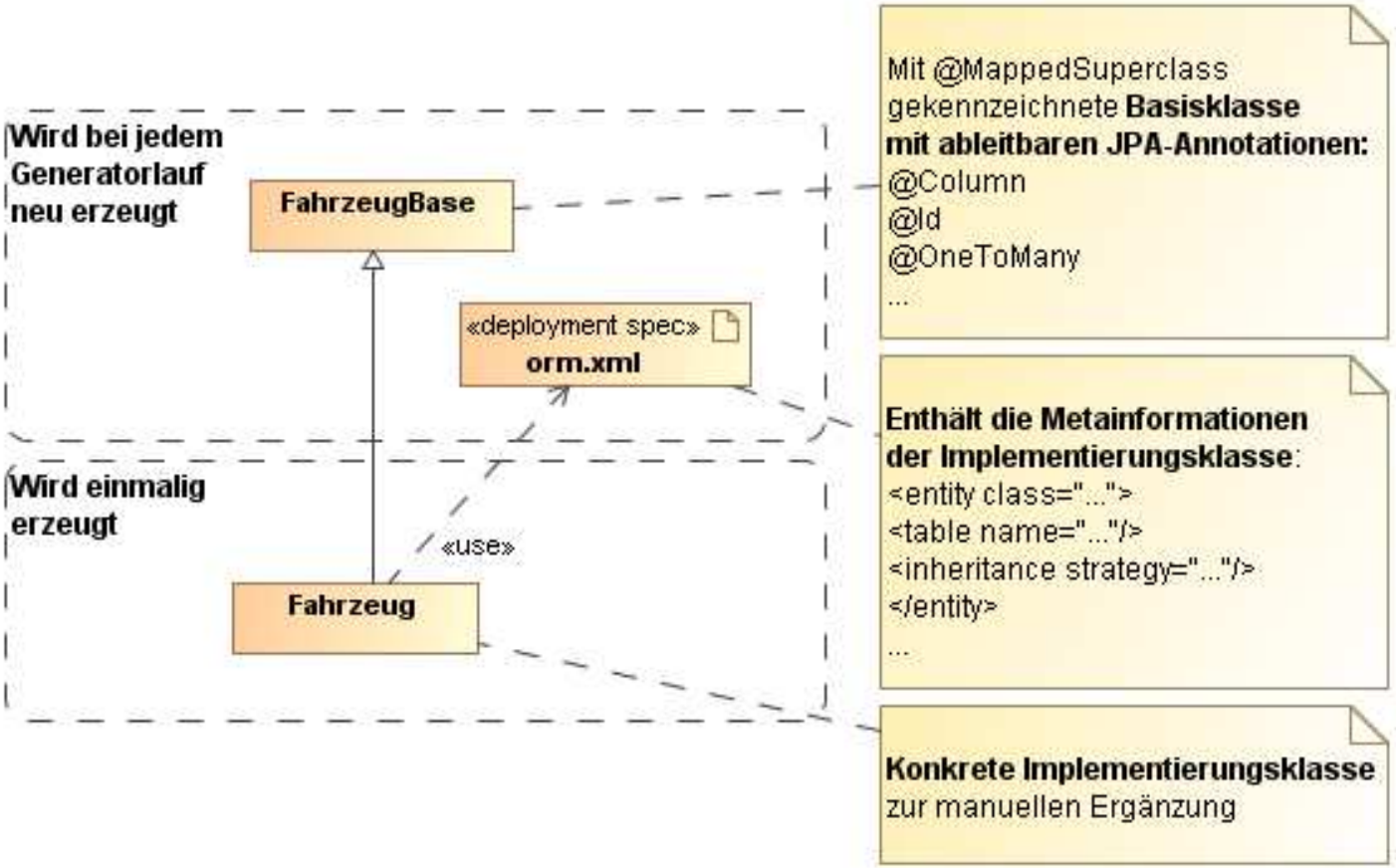
- Abstrakte Basisklasse mit allen sinnvoll durch Vererbung ableitbaren Modellinformationen als Annotationen
  - Bei Session Beans bspw. Injektionen per **`@EJB`** oder **`@PersistenceContext`**
  - Bei Entities bspw. JPA-Mappinginformation (**`@Id`**, **`@Column`**, **`@OneToOne`**, **`@OneToMany`**, …)
  - Alle sich aus dem Modell ergebenden Methoden werden abstrakt vorgegeben
- Konkrete Implementierungsklasse
  - Einmalig erzeugt, nicht mehr überschrieben
  - Enthält lediglich die Methodenrümpfe zur manuellen Vervollständigung
  - Frei von EJB 3-Metadaten

2. Nutze den Deployment Deskriptor (`ejb-jar.xml` bzw. `orm.xml`)!

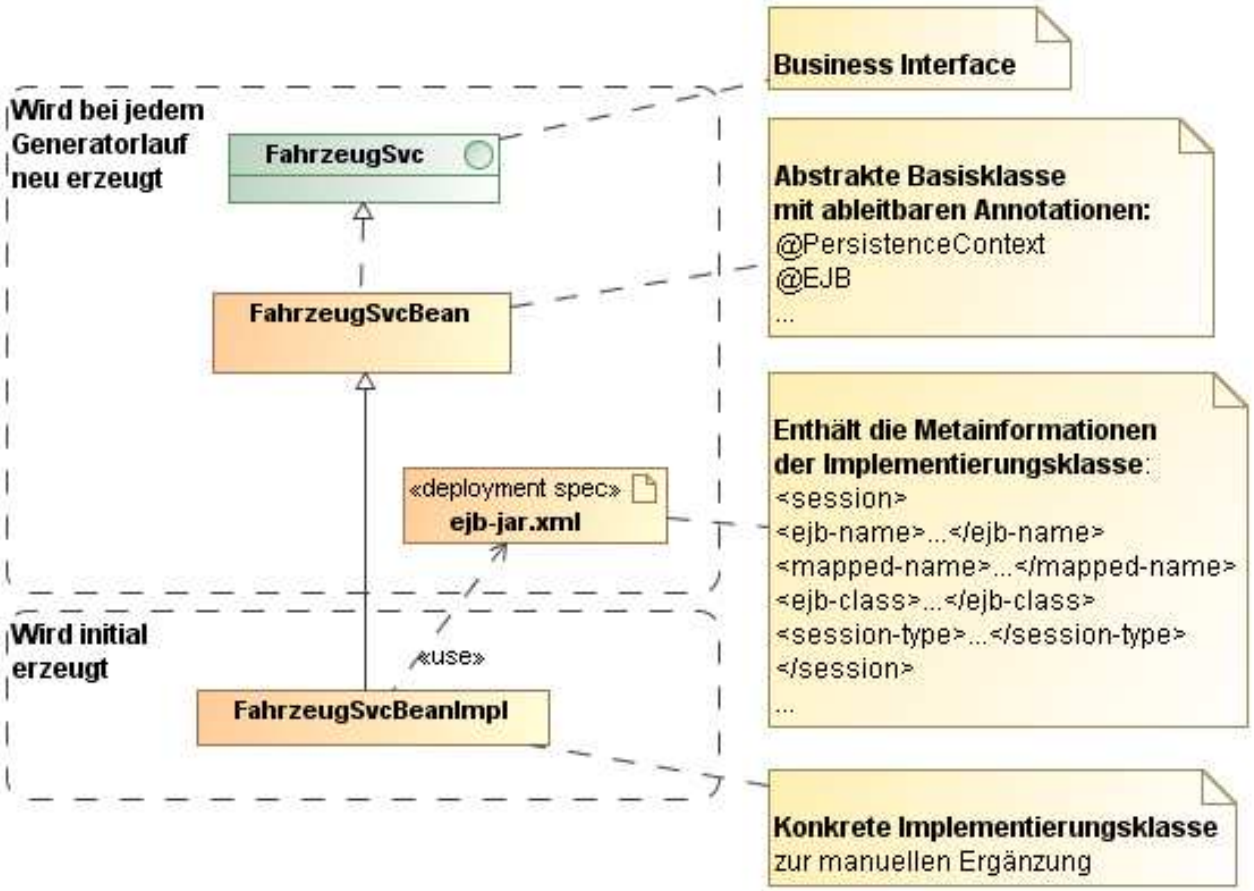
- Enthält alle Metadaten, die von der Implementierungsklasse nicht sinnvoll aus der Basisklasse abgeleitet werden können, bspw.
  - Session Bean-Ausprägung per **`<session-type>Stateless</session-type>`**
  - Kennzeichnung als persistente Entity durch **`<entity .../>`**

NOVATEC
make IT happen!

Mikroarchitektur am Beispiel der Entity Fahrzeug

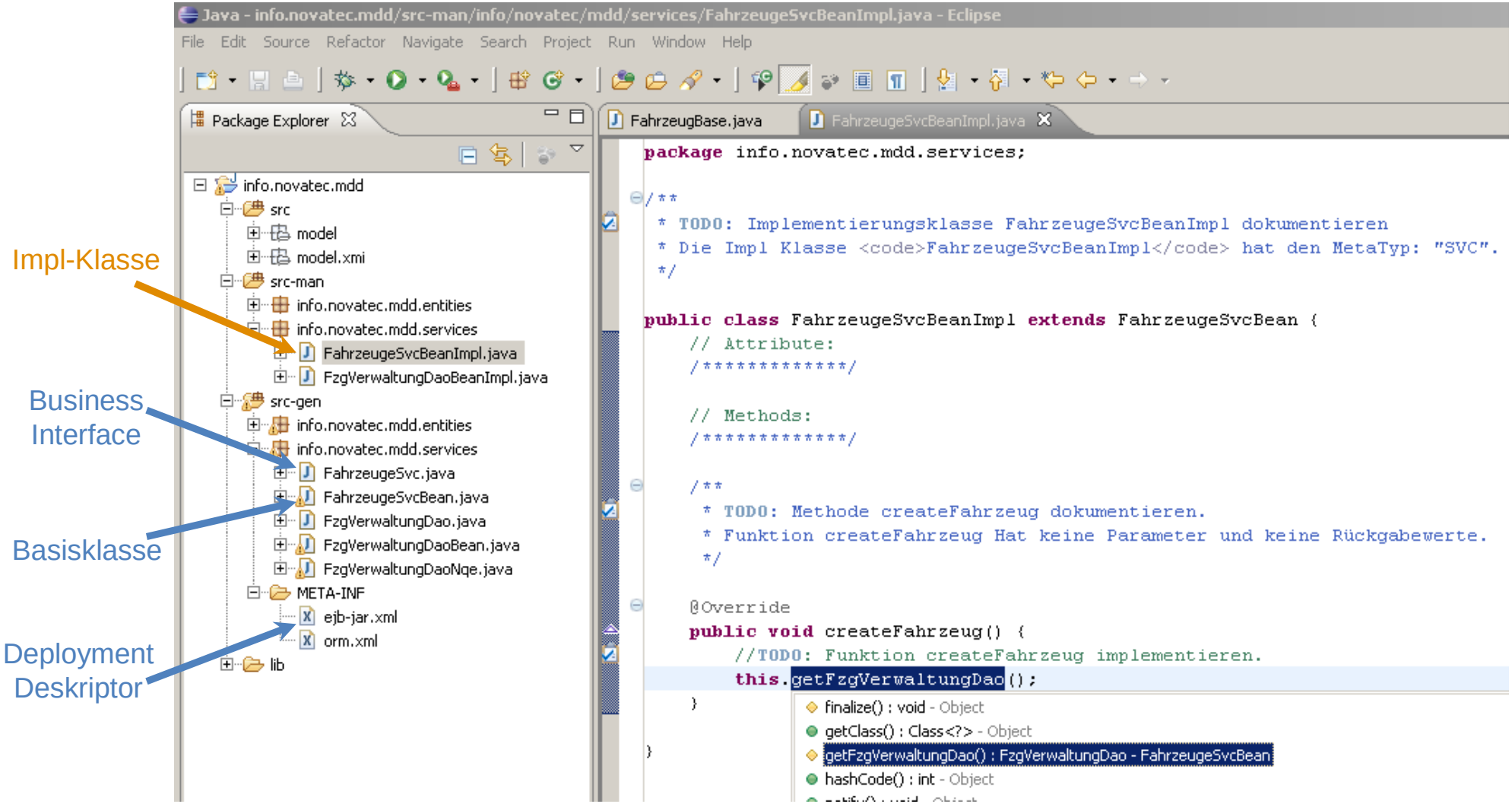


Mikroarchitektur am Beispiel des Services FahrzeugSvc



Entwicklersicht: Session Bean-Artefakte in der IDE

NOVATEC
make IT happen!



Agenda

- **Kurzeinführung *Modellgetriebene Softwareentwicklung***
- **Migrationsprojekt**
  - Modellierung
  - Generierung
  - Manuelle Vervollständigung
- **Lessons Learned und Best Practices**
- **Fazit und Ausblick**

Lessons Learned: Modellierung

NOVATEC
make IT happen!

DSL-Erstellung auf Basis der UML

- Gute Erweiterbarkeit anhand des Profil-Mechanismus
- Eignet sich besonders gut für statische Strukturen, weniger für logische Abläufe
- UML(-Metamodell) ist teilweise zu umfangreich

XMI-Export

- Darf sich nicht zum „Bottleneck“ entwickeln

Versionsverwaltung und kollaboratives Arbeiten mit grafischen Modellen

- Zentrales Repository mit dem Prinzip: *Lock* → *Modify* → *Unlock* („pessimistic“)
- Modell muss in sinnvolle Teile fragmentiert werden

Lessons Learned: Generierung & Vervollständigung

NOVATEC
make IT happen!

- **openArchitectureWare / Eclipse Modeling Project**
  - Sehr ausgereiftes Framework
  - Alle Anforderungen an den Generator konnten umgesetzt werden
- **Generierungsanteil**
  - Entities: bis zu 100%
  - Session Beans: ca. 50–70%
- **Strategie zur Kombination von Generaten und manuellen Ergänzungen**
  - Alle Stereotypen konnten darüber abgebildet werden
  - Wichtiger Erfolgsfaktor

Best Practices

NOVATEC
make IT happen!

Generiere gutaussehenden Code!

- Einrückungen, Klammern, Imports, etc. berücksichtigen
- Generierter Code sollte die Codier-Richtlinien erfüllen, die auch für manuell erstellte Quelltexte gelten (Checkstyle, Findbugs, ...)

Integriere und Automatisiere den Build-Prozess!

- Zusätzliche Schritte *Modellierung* → *Export* → *Generierung* möglichst automatisieren
- Einbindung in die IDE bzw. Nutzung bestehender Build-Tools (Ant, Maven, …)

Gib bei Validierungsfehlern schlüssige Meldungen aus!

- Validierung möglichst bereits bei der Modellierung
- Validierung bei der Generierung: Klare Angabe des ungültigen Modellelements, ggf. mit Korrekturvorschlag

Best Practices

Nutze generische Frameworks!

- Generierung gegen eine reichhaltige, vollständig definierte, dokumentierte und getestete Zielarchitektur
- Generisches Framework statt übermäßiger Generierung

Setze MDD pragmatisch ein!

- Versuche nicht *alles* sondern möglichst *sinnvoll* und *effizient* zu generieren
- Keine „Weltmodelle“

Agenda

- **Kurzeinführung *Modellgetriebene Softwareentwicklung***
- **Migrationsprojekt**
  - Modellierung
  - Generierung
  - Manuelle Vervollständigung
- **Lessons Learned und Best Practices**
- **Fazit und Ausblick**

NOVATEC
make IT happen!

(Erreichte!) Vorteile durch MDD

- Die Anwendungsentwicklung erfolgt auf einer **höheren Abstraktionsebene** ✓
- **Architekturvorgaben** lassen sich besser durchsetzen ✓
- Zentrale **Framework- oder Architekturänderungen** lassen sich besser ausrollen ✓
- **Entwicklungseffizienz und der Automatisierungsgrad** können erhöht werden ✓
- Die **Wiederverwendung** wird vereinfacht ✓
- Die **Softwarequalität** kann gesteigert werden ✓
- Eine **Plattformunabhängigkeit** lässt sich einfacher realisieren ✗

Ausblick

Grafische Modelle bei verteilter / dezentraler Versionsverwaltung

- Distributed Version Control Systems: *Git*, *Mercurial*, *Bazaar*, …
- Prinzip: *Copy* → *Modify* → *Merge* („optimistic“)
- Erfordern Merge-Mechanismen für grafische Modelle
- Bisher nur toolspezifische Lösungen

MDD in Verbindung mit Continous Integration (CI)

- CI-Server muss Zugriff auf Modellrepository haben
- Anstoßen des Modell-Exports und der Generierung
- Klare Definition eines *Changes*

„Ich sage nur ein Wort: Vielen Dank!“

Horst Hrubesch, Europameister
und Kopfballungeheuer

www.novatec-gmbh.de

NOVATEC

make IT happen!

- Wir führen IT-Projekte zum Erfolg -

Weitere Informationen:
Gleich am **NovaTec-Stand** oder per Mail:
christian.schwoerer@novatec-gmbh.de

NovaTec – Ingenieure für neue Informationstechnologien GmbH

Hauptniederlassung
Dieselstr. 18/1
D-70771 Leinfelden-Echterdingen

Telefon: +49 (0)711 220 40-700
Fax: +49 (0)711 220 40-899

Niederlassung München
Landsberger Straße 439
D-81241 München

E-Mail: info@novatec-gmbh.de
Internet: www.novatec-gmbh.de

Niederlassung Frankfurt
Friedrich-Ebert-Anlage 36
D-60325 Frankfurt am Main

Niederlassung
Mittlerer Osten
P.O. Box 140611
Jeddah 21333
Saudi-Arabien