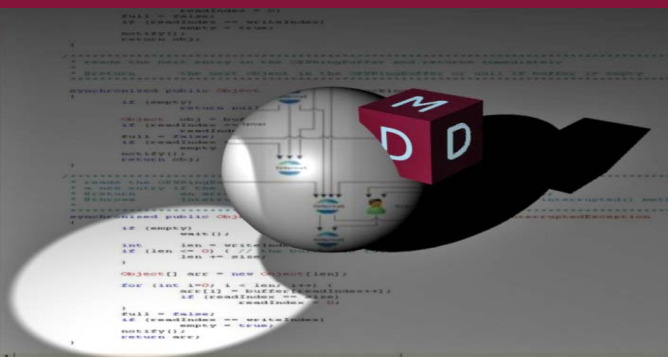


Softwarevarianten im Griff mit textuellen DSLs

Erfahrungsbericht



.consulting .solutions .partnership

SE 2010 - Paderborn
Industrietag 24.02.2010

Johannes Reitzner
Leiter CoC Model Driven Development

Planungssysteme für Importeure zur Unterstützung des Bestellvorschlagsprozesses für Autos des VW-Konzerns



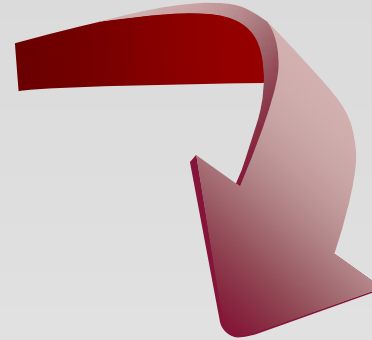
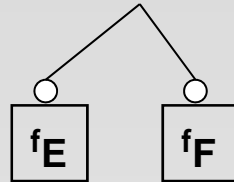
Merkmale

- Weitgehend gleicher Planungsprozess, aber länderspezifisches Spezialverhalten
- Landessysteme sind voneinander unabhängig
- Neue Anforderungen sollen rasch integrier- und ausrollbar sein (time to market)
- Verfolgbarkeit „Was ist installiert?“

Entscheidungen

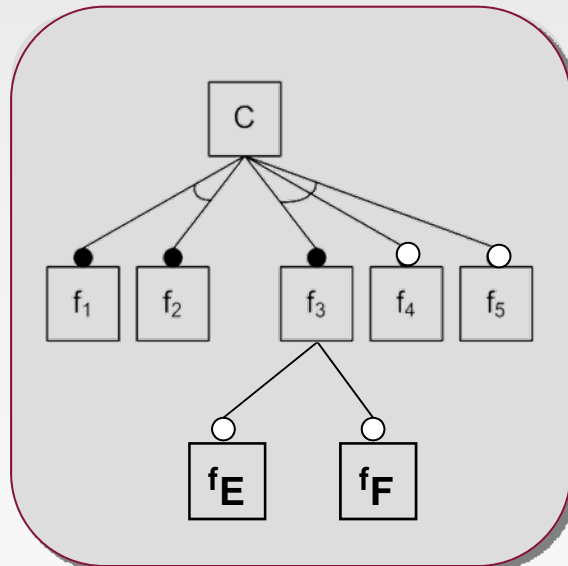
- Produktlinienansatz mit aktivem Variantenmanagement
- Landesindividuell ausrollbare Systeme
- Modellgetriebener Entwicklungsansatz
- Traceability Feature – Modelle – Code

Neue Anforderungen



Kontinuierliche Pflege des Featuremodells

Featuremodell



- Fortlaufende Produktlinienoptimierung
 - Change Requests / neue Anforderungen
 - Prozessvereinheitlichungdenken wird gefördert
 - „Sonderlocken“ brauchen gute Argumente
- Coaching des Fachbereichs sinnvoll
- Basis für Verfolgbarkeit
 - Fachlichkeit / Technik pro System

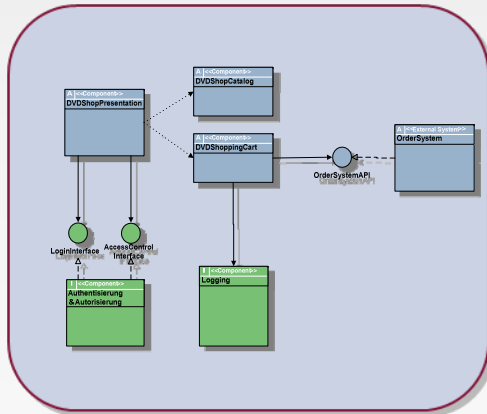
Durchgängige Anforderungsanalyse



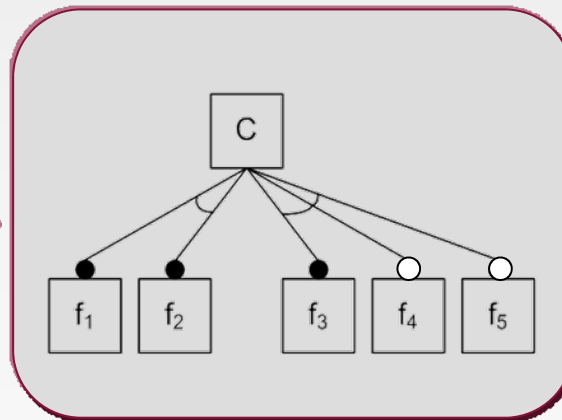
*RE/RM= Requirement Engineering / Requirement Management



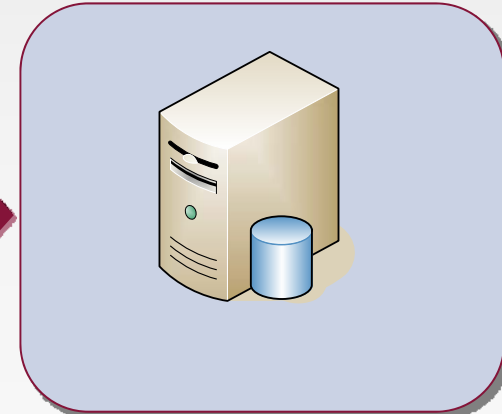
Architekturmodell



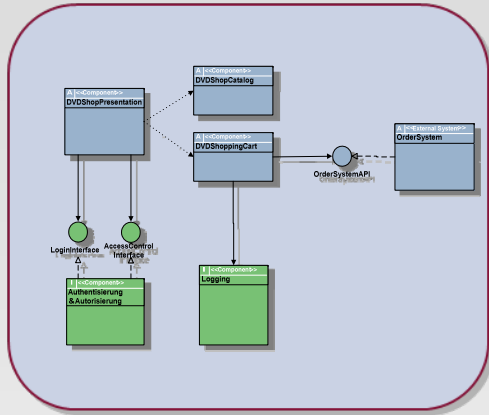
Featuremodell



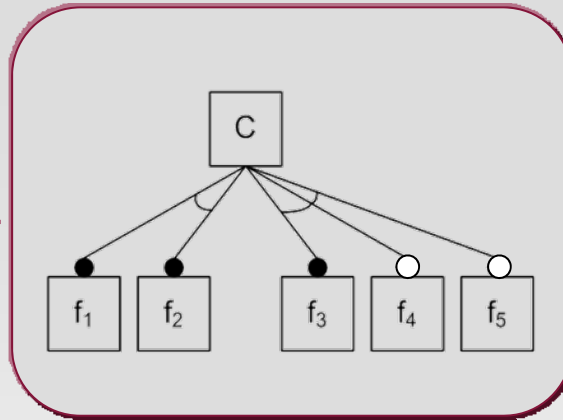
Produktlinien-Code



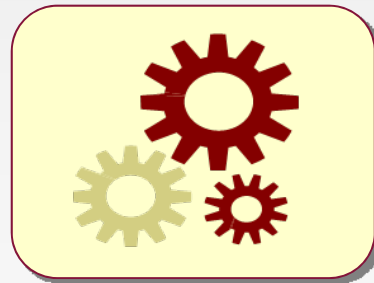
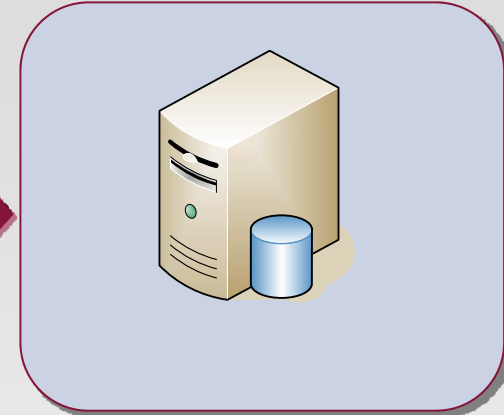
Architekturmodell



Featuremodell



Produktlinien-Code

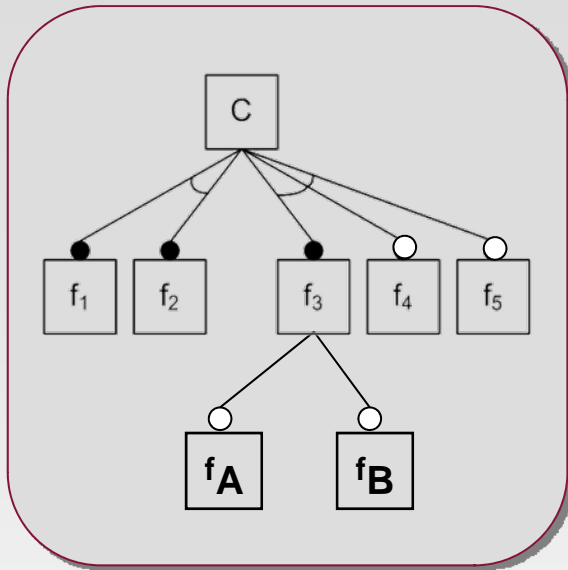


Code-Generatoren



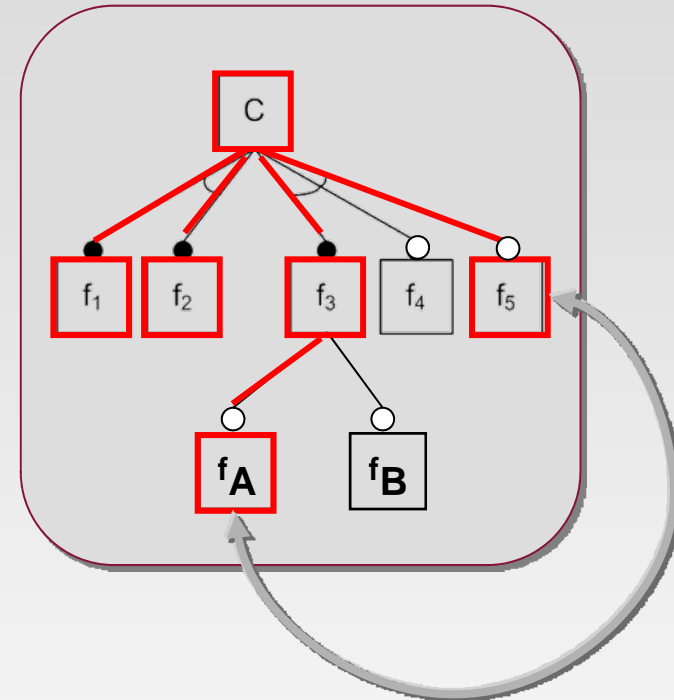
handgeschriebener Code

Featuremodell



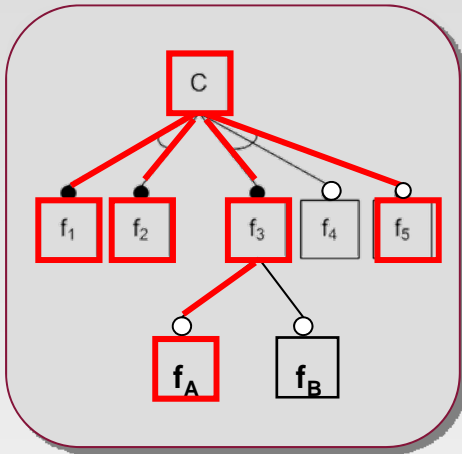
konfigurieren

Variantenmodell

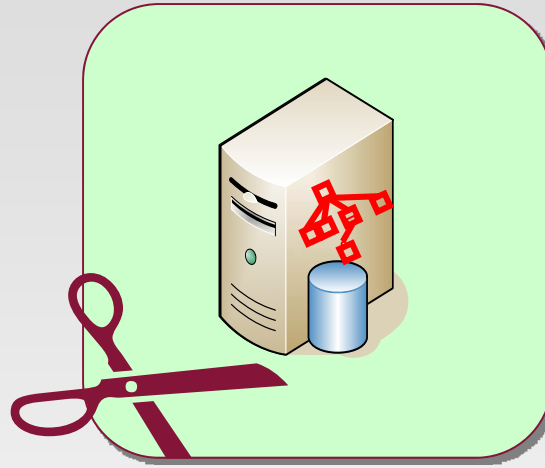


Die Selektion von f_A führt zur erzwungenen Selektion von f_5

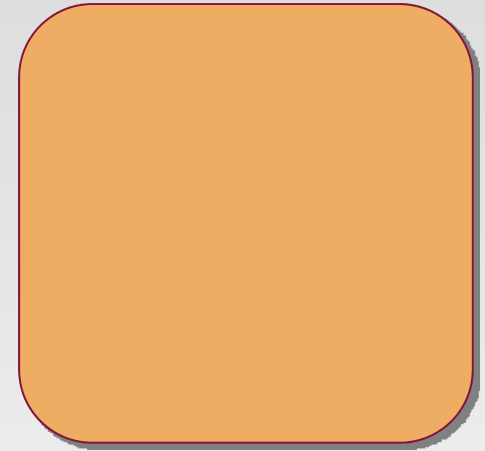
Variantenmodell



Produktlinien-Code



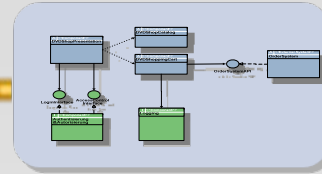
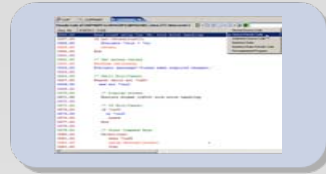
Produkt



**Schneiden des Codes
auf Grundlage des Variantenmodells**

Architecture & Design
Specific Language

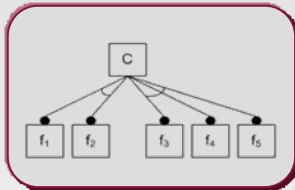
ADSL Visualisierung



Xtext / TMF

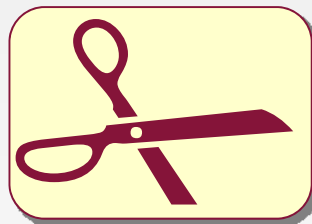
visEMF Plugin

Featuremodel
Variantmodel



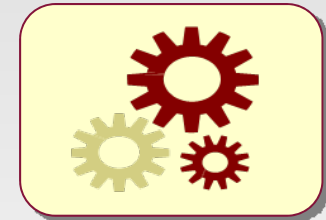
pure::variants

Tailoring
Model / Code



MWE-Components

Code
Generatoren



openArchitectureWare
Xpand / Xtend

Codierung



Eclipse JDT

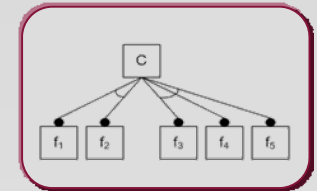
eclipse

Featuremodell

- IPOTFeatures
 - Master Data
 - Conversion Week
 - Brand Administration
 - NADIN configuration
 - Set importers
 - Set relevant importer**
 - Working out of importer stock
 - Model groups
 - Minimum Quota**
 - Calendar
 - Program Planning and Calculations
 - Scenario Support
 - 1 to n Scenarios
 - Quota Planning
 - MO planning
 - PO planning
 - Production distribution
 - Stock Factor Adjustment
 - Sales Target Distribution
 - Importer Dealer production split
 - Changing retails in PO
 - Show Min Quota

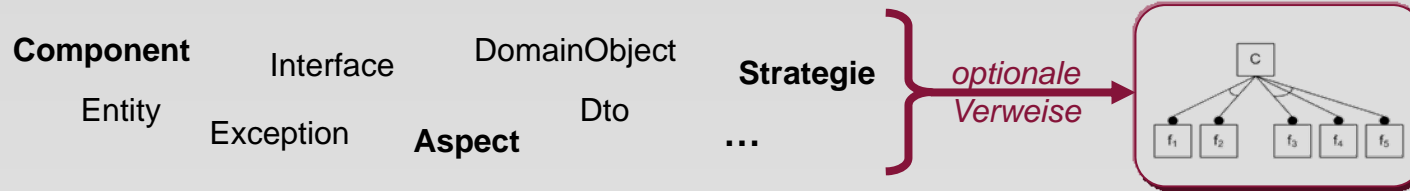
Variantenmodell

- IPOTFeatures
 - Master Data
 - Conversion Week
 - Brand Administration
 - NADIN configuration
 - Set importers
 - Set relevant importer
 - Working out of importer stock
 - Model groups
 - Minimum Quota
 - Calendar
 - Program Planning and Calculations
 - Scenario Support
 - 1 to n Scenarios
 - Quota Planning
 - MO planning
 - PO planning
 - Production distribution
 - Stock Factor Adjustment
 - Sales Target Distribution
 - Importer Dealer production split
 - Changing retails in PO
 - Show Min Quota



pure::variants

- Erste nutzbare Modelle rasch erstellt
- mächtige Funktionalität -> braucht Einarbeitung
- Modelle im Ecore Format
- Auswertbar durch Generatoren



```
entity Broker featureAndList (f1, f3) {  
  name:      string;  
  exporter:  boolean featureExp (f1 and f2 or not f3);  
  optLockingVersion : long;  
}
```

Argumenten

Verweise zum Featuremodell

```
7     featureAndList (f1, f3) {  
8       name:      string;  
9       exporter:  boolean featureExp (f1 and f2 or not f3);  
10    }  
11    aspect [pointcut=*] entity JPA_Aspect feature JPA {  
12      optLockingVersion : long;  
13    }  
14  
15
```

Aspekte:

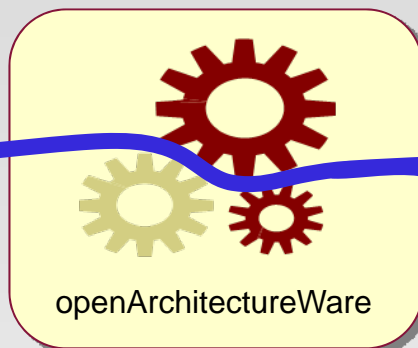
- Feingranulare Elementkomposition
- z.B. technische Attribute

Design-Pattern

ADSL

```
entity Broker featureAndList(f1,f3)  
{  
  name : string;  
  ...  
}
```

Generator

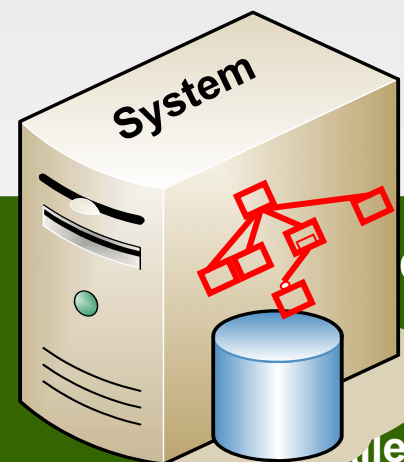
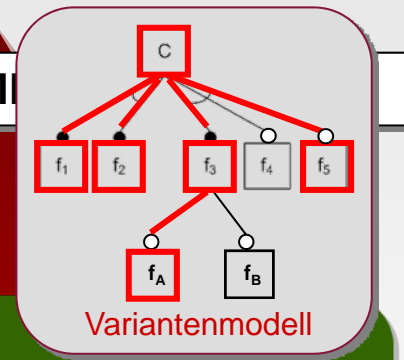


Produktlinien - Code

```
Broker.java  
1 package de.volkswagen.ipot.common.  
2  
3 //#BEGIN featureAndList(f1, f3) #  
4  
5 public class Broker {  
6     private String name = null;  
7
```

Feature-Antailoring auf Basis des

XML



der Feature-Bezeichner
Bezeichner sofort sichtbar
Eclipse-Plugin
alle Artefakte mit Features

The screenshot displays the MDD-Workbench interface with four main panels:

- Featuremodell:** A tree view of features under 'IPOT.xfm'. 'Sales Target Distribution' is highlighted with a red oval. A green arrow points from this feature to the 'Relations' tab in the top-left pane.
- Designmodell:** A code editor showing an ANTLR-style DSL for 'quotaplanning.adsl'. The line `feature sales_target_distribution;` is circled in red. A green arrow points from this line to the 'Relations' tab.
- Java-Code:** A code editor showing Java code for 'SalesCategoryParameter.java'. The line `//#BEGIN feature sales_target_distribution #` is circled in red. A green arrow points from this line to the 'Relations' tab.
- Persistenz:** An XML editor showing 'persistence.xml'. The line `!-- #BEGIN feature sales_target_distribution # -->` is circled in red. A green arrow points from this line to the 'Relations' tab.

At the bottom left, there is a diagram titled 'IncrementalHierarchicLayouter' showing a hierarchical structure of nodes.

DSL Modelle



Graphische Darstellung

The screenshot displays the MDD-Workbench interface. The top-left pane shows the DSL code for a component named 'MarketStructure'. The code defines an entity 'Dealer' with attributes like 'name', 'importer', and 'minimum_quota', and an operation 'doNothing()'. The top-right pane shows a 'yFiles Overview' diagram with a complex network of nodes and edges. The bottom pane shows a detailed UML class diagram with the following elements:

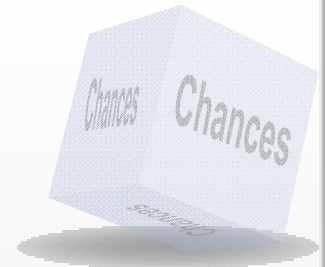
- marketstructure**: A class implementing `<<PartialComponent>>` with attributes `id: Long`, `dealerIdentifier: String`, `area: String`, `region: String`, `maxImportVersion: Integer`, `name: String`, `parent: MarketUnit`, and `brand: Brand`. It has methods `+getLeafChildren(): List<MarketUnit>`, `#collectMarketUnitChildren(marketUnits: List<MarketUnit>): List<MarketUnit>`, `#collectMarketUnitNotImporterChildren(marketUnits: List<MarketUnit>): List<MarketUnit>`, and `+hasParent(): boolean`.
- ItemPlanningService**: An interface with methods `+releaseMarketOrientedItemPlanning(scenario: Scenario): Scenario` and `+releaseProductionOrientedItemPlanning(scenario: Scenario): Scenario`.
- Dealer**: An entity implementing `<<Entity>>` with attributes `importer: boolean` and `importerRelevant: boolean`, and a method `+doNothing(): void`.

The diagram shows relationships between these classes, including inheritance and associations.

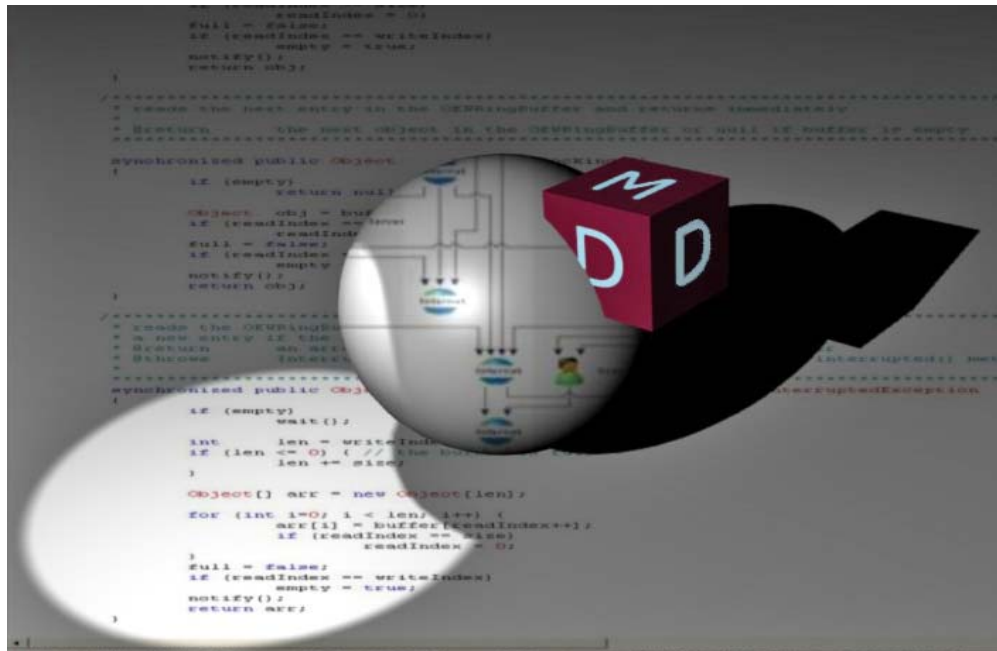
- Textuelles Modell und Grafik nebeneinander
- Konfigurierbare Views auf DSL Modelle (auch UML)
- Manuelle Layoutänderungen bleiben erhalten

Immer wieder nachgefragt ...

- **Variantensimulation in der Entwickler-IDE**
schnelles Ein-/Ausblenden von Varianten, MPS/JetBrains?
- **SPL-Teststrategien Erfahrungen sammeln**
- **Ausbau DSL-Visualisierung**
wichtig für Akzeptanz textueller DSLs



Herzlichen Dank für Ihre Aufmerksamkeit !



msg Applied Technology Research
johannes.reitzner@msg-systems.com

Domänen-spezifische Modelle

ADSL

M2M

xDSL

M2M

CA3

M2M

oAW
Xtend

Sprach-Entwicklung

Anschauliche graphische Darstellung

Parametrisierung *interaktiv pro Diagramm*

graphmm
(Metamodell / Xtext)

Generische Schnittstelle
für Graphendarstellungen
(Knoten, Kanten, Farben, ...)

graphmm
yFilesViewer

yFiles
(Grafikframework)

visEMF Eclipse-Plugins