Free your work.

Free your work.

Wir wollen Ihnen die Freiheit geben,
sich auf Ihr Geschäft zu konzentrieren.

LIANTIS
IT-Consulting

Design und Implementierung von Codegeneratoren

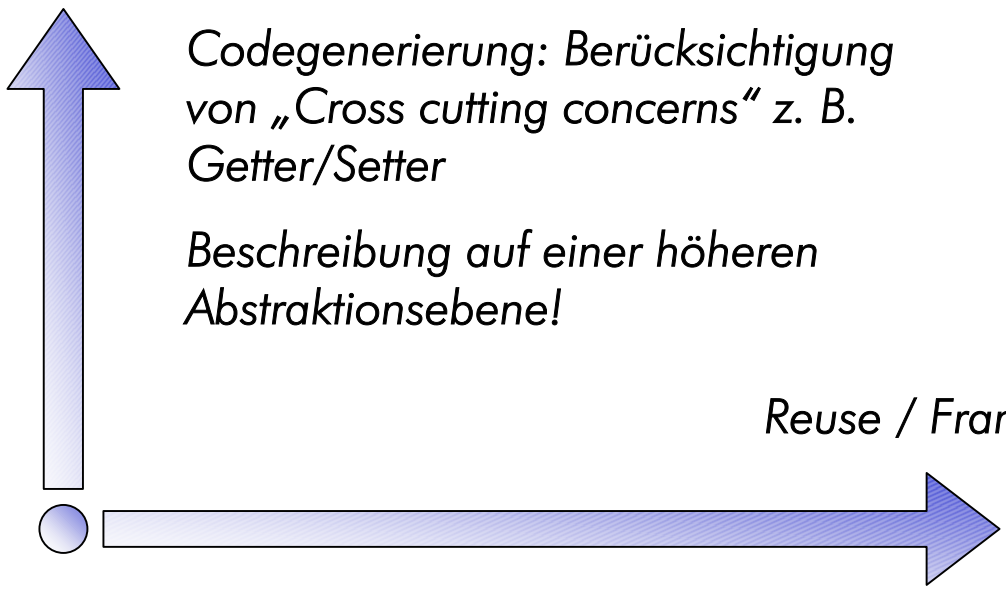
Am Beispiel der MDA-Suite XCoder

Dipl.-Inf. Constantin Szallies

LIANTIS
IT-Consulting

Warum MDA?

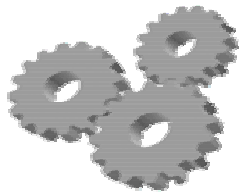
- Softwareentwicklung automatisieren!
- Zwei orthogonale Ansätze kombinieren:



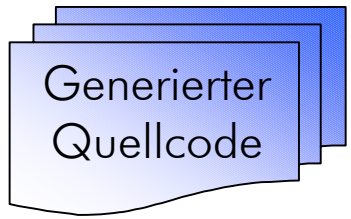
© LIANTIS

Softwareentwicklung mit der MDA

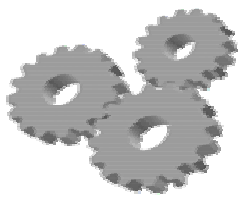
Modellierung



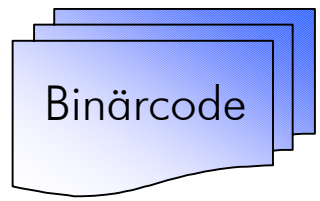
Generator



Individueller Code =
nicht automatisierte Bearbeitung



Compiler



Konsequentes Forward Engineering

- Modellierung
- Generierung
- Ergänzen des individuellen Codes

Modell und Code jederzeit synchron!

Ermöglicht die Erstellung eines Modells auf höherer Abstraktionsebene!

Roundtrip Engineering bedeutet: Modell und Code äquivalent

© LIANTIS

LIANTIS IT-Consulting

Die XCoder-Suite

- Wiederverwendbare Komponenten und anpassbare Standard-Generatoren
- Open Source: http://sourceforge.net/projects/xcoder
- In Projekten von Liantis verwendet und weiterentwickelt, wenn Standard-Lösungen ungeeignet
- Das komplette Framework ist vollständig modelliert und generiert sich selbst: MDA²
- Leichtgewichtig, schnell und einfach zu erweitern

© LIANTIS

LIANTIS IT-Consulting

XCoder-Architektur (hier: für C++)

Modell-Transformation

UML-Metamodell-Instanz

C++-Metamodell-Instanz

XMI-Export

XMI-Reader

Generate

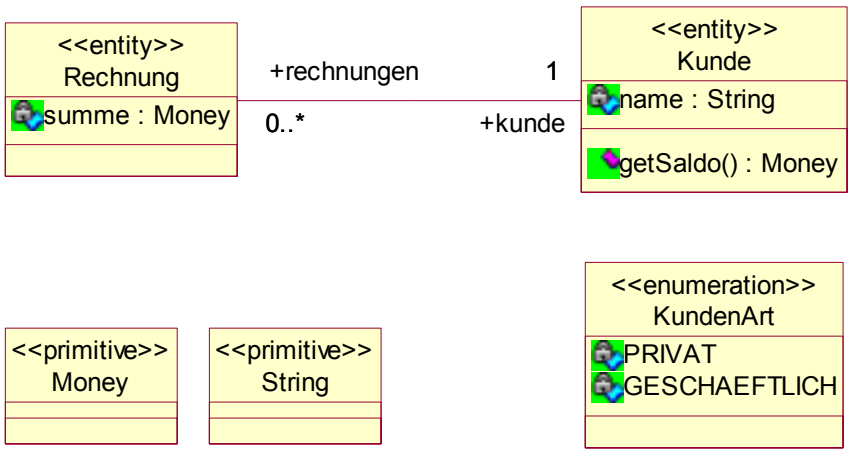
Model.xml

C++ Quellen

© LIANTIS

LIANTIS IT-Consulting

Platform Independent Model (PIM)



© LIANTIS

LIANTIS IT-Consulting

Platform Specific Model (PSM)

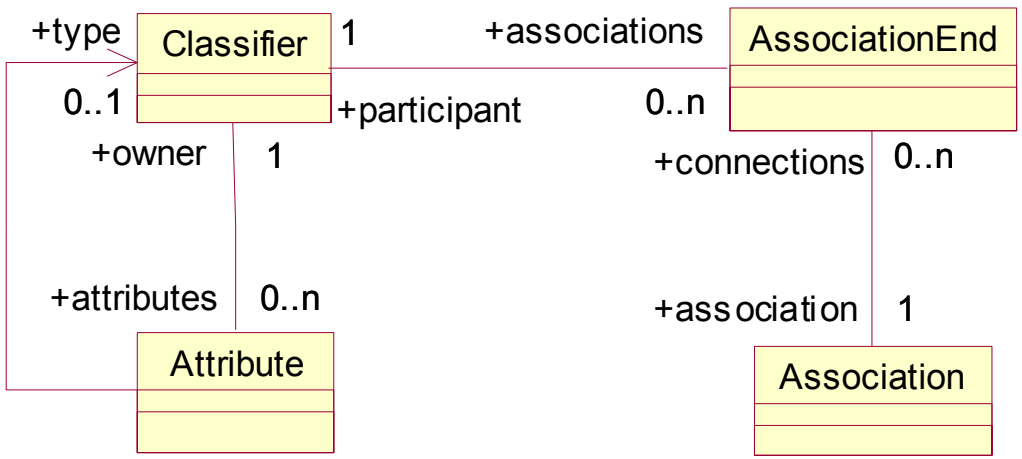
- Durch Transformation des PIM erzeugt

© LIANTIS

LIANTIS IT-Consulting

UML-Metamodell

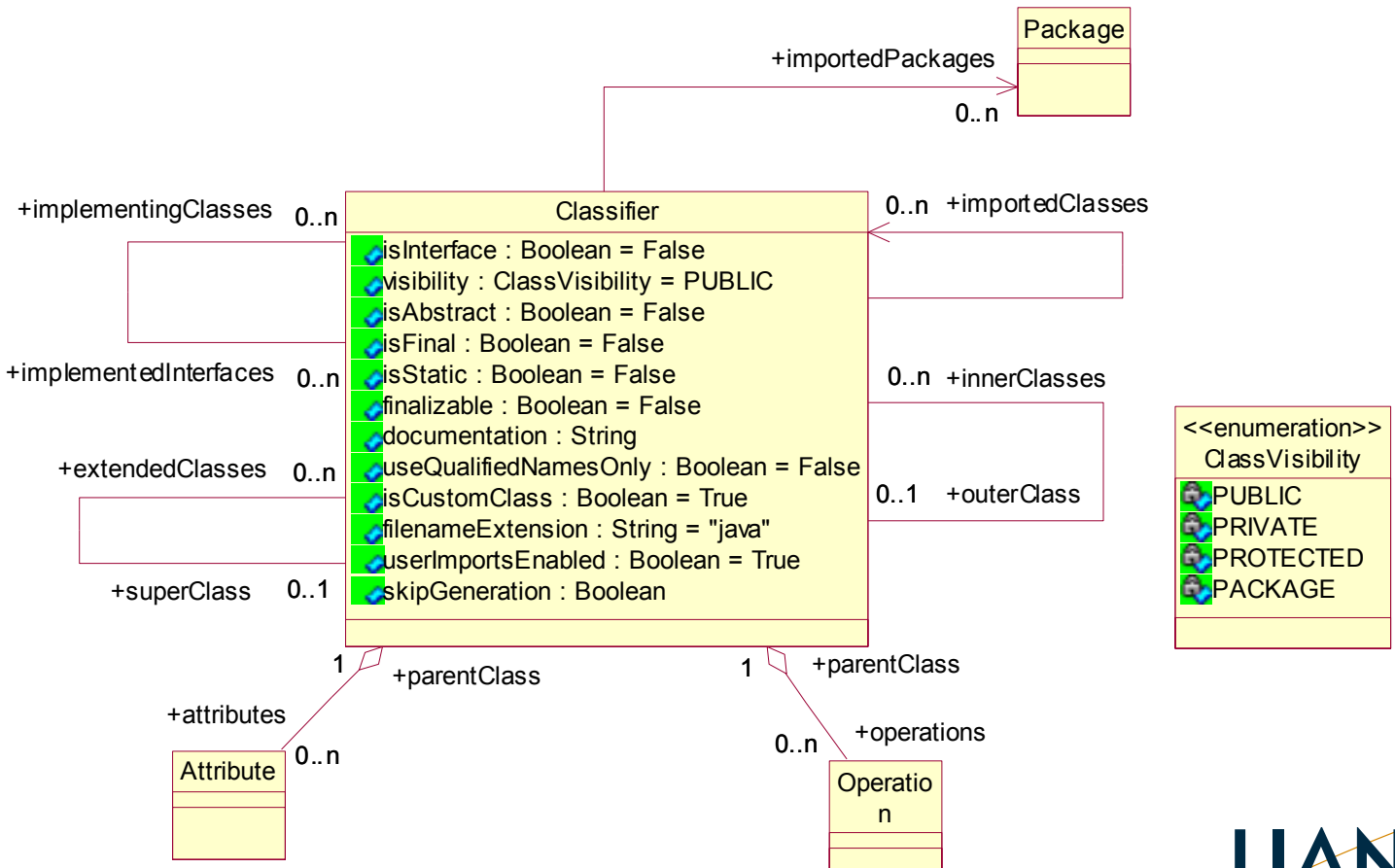
- Eine Instanz des Metamodells ist das im UML-Werkzeug erstellte Modell
- Einlesen des Modells im XMI-Format



Präsentation
© LIANTIS

Java-Metamodell

- Metamodell der Programmiersprache Java



© LIANTIS

LIANTIS IT-Consulting

Das Java-Metamodell verwenden

```
// Create java model
Model javaModel = new Model();

// create class
Classifier javaClassifier = new Classifier();
javaModel.addClassWithQualifiedName(javaClassifier,
                                    "example.Kunde");
// add attribute
Attribute someAttribute = new Attribute();
someAttribute.setName("myAttribute");
someAttribute.setType(Type.getType("int"));
javaClassifier.addToAttributes(someAttribute);
```


© LIANTIS

LIANTIS IT-Consulting

Abstraktes Dateisystem

```
// Create file system
TransactionFileSystem fileSystem = new TransactionFileSystem();
// Create java model
Model javaModel = new Model();
// create java classes
...
// Generate java class
javaModel.generate(fileSystem);

fileSystem.commit();
```

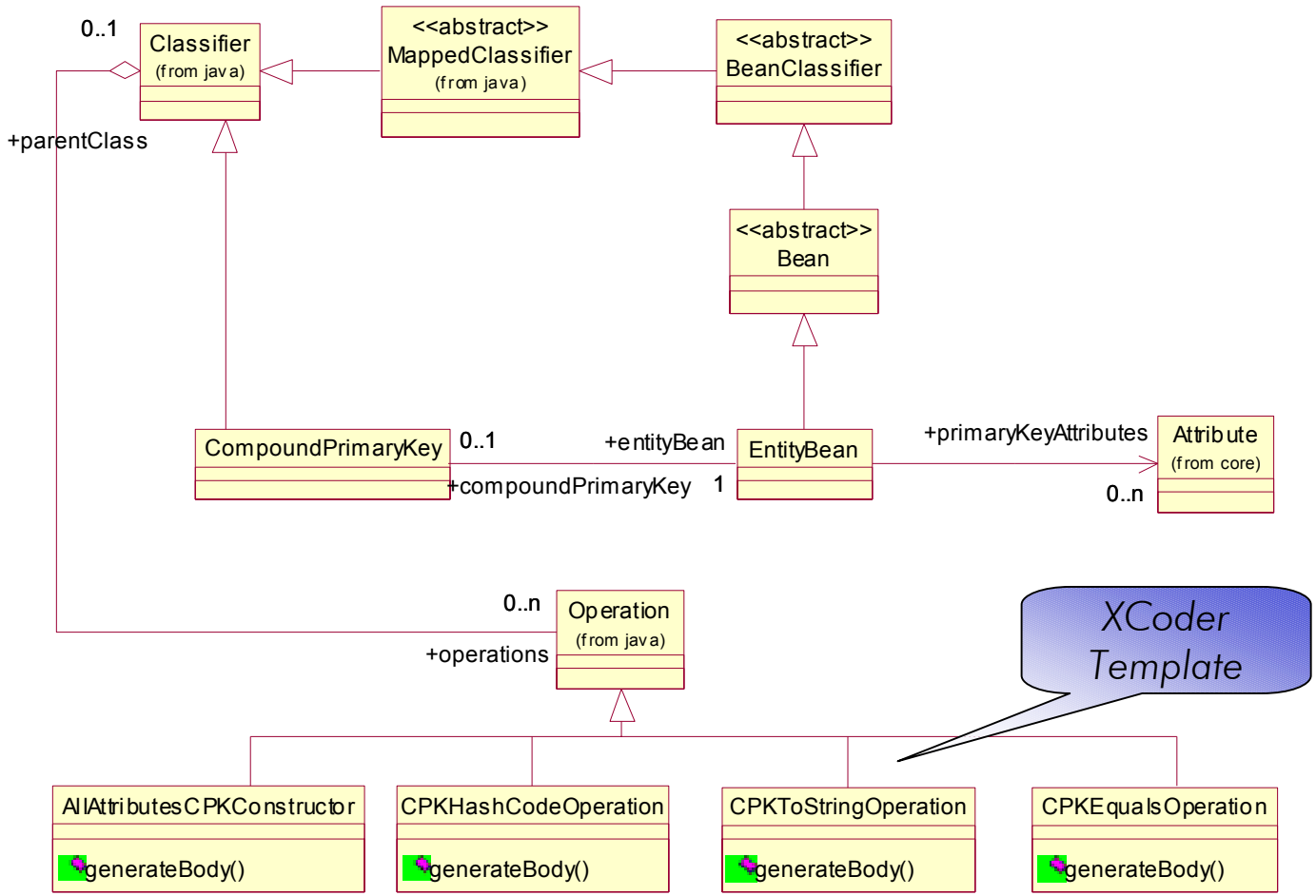
Kapselung des Dateisystems

Transaktionsschutz für Dateiänderungen

© LIANTIS

LIANTIS IT-Consulting

Das EJB-Metamodell als Erweiterung des Java-Metamodells



© LIANTIS

LIANTIS
IT-Consulting

Template für „toString“-Methode

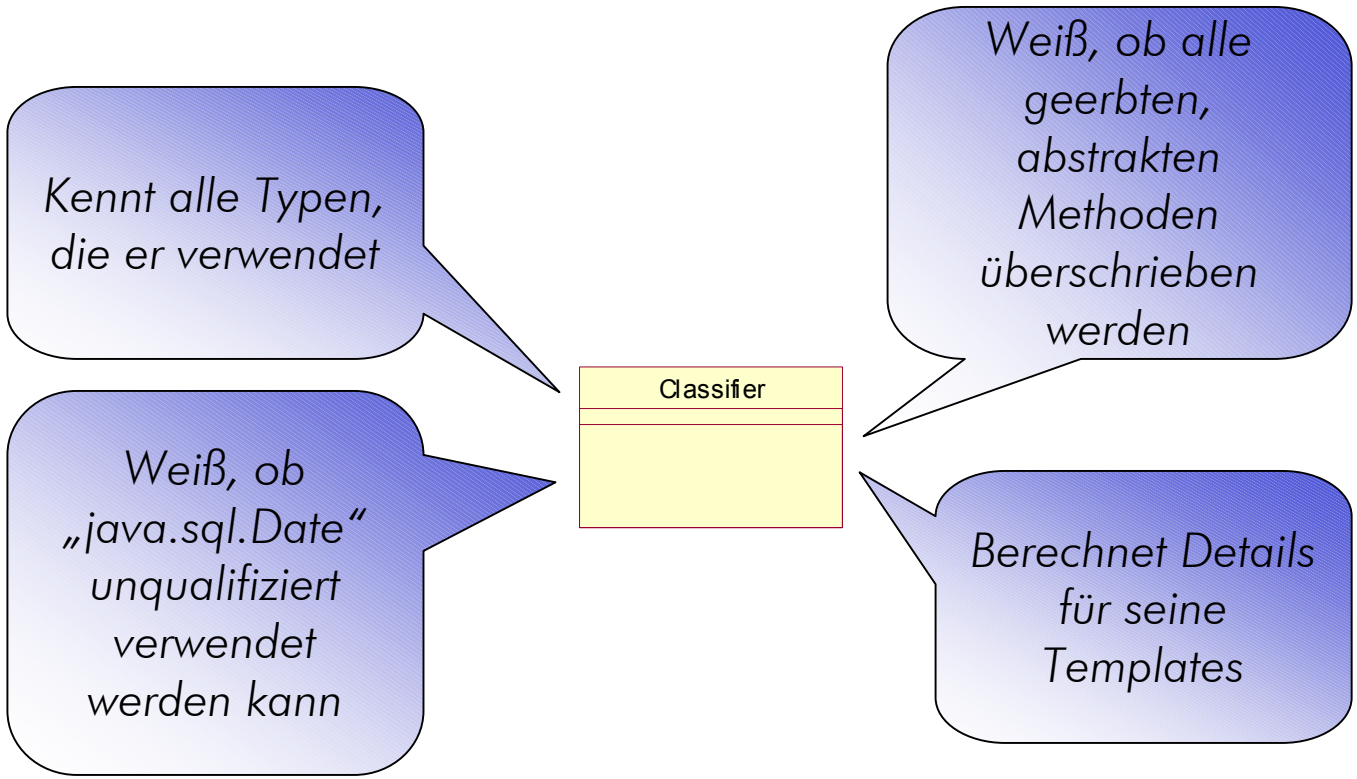
```
public void generateBody(com.liantis.io.Printer printer)
 {
   // @BEGINPROTECT _3C8E2EF60260
   java.util.Vector keys = ((CompoundPrimaryKey)this.getParentClass()).
                              getEntityBean().getPrimaryKeyAttributes();

   Attribute firstAttribute = (Attribute)keys.elementAt(0);

   [
   return "(" + [[firstAttribute.getName()]] +
   ]
   for(int i=1,j=keys.size();i<j;i++)
   {
     Attribute anAttribute = (Attribute)keys.elementAt(i);
     [
       "," + [[anAttribute.getName()]] +
     ]
   }
   [
     ")";
   ]
   // @ENDPROTECT
 }
```

Umwandlung mittels Präprozessor

LIANTIS IT-Consulting

Vorteile durch Metamodellierung

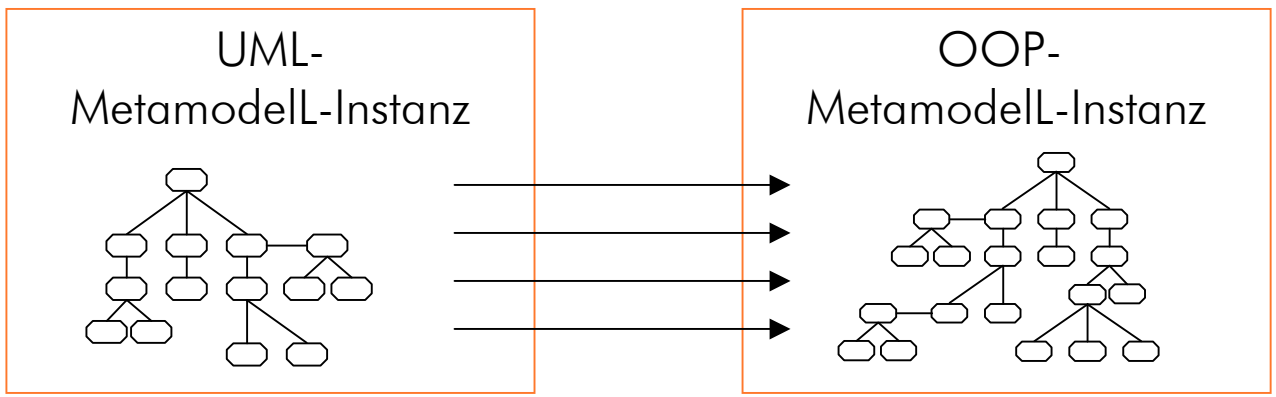
- Die Struktur des zu erzeugenden Codes liegt vollständig vor



© LIANTIS

Modelltransformation

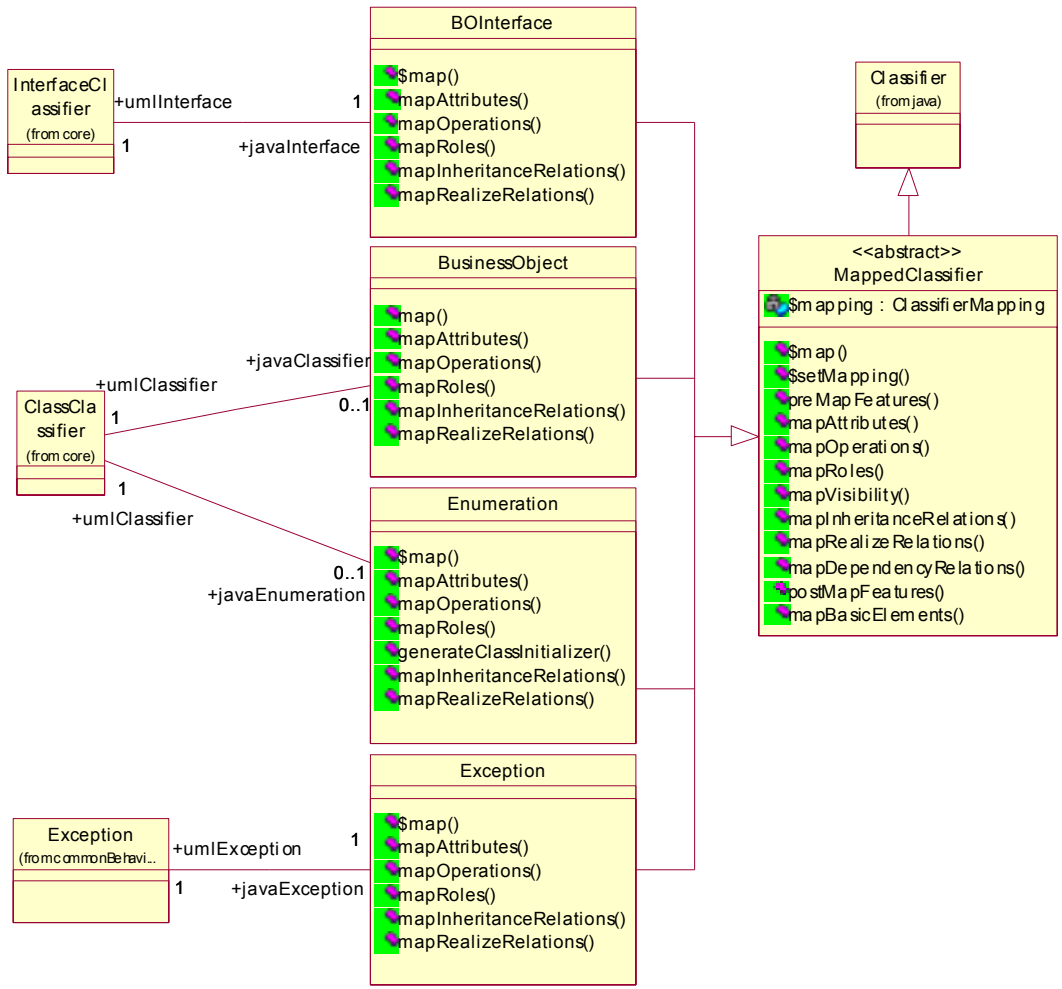
- Aus einer Metamodell-Instanz eine zweite Metamodell-Instanz erzeugen
- Eine oder mehrere Iterationen über das Eingabe-Modell bauen das Ausgabe-Modell auf
- Am Ende: „Model to Text“-Transformation



© LIANTIS

LIANTIS IT-Consulting

Modellierung einer Transformation



© LIANTIS

LIANTIS IT-Consulting

Implementierung einer Transformation

```
public static Enumeration map(ClassClassifier aClass)
{
    // @BEGINPROTECT _3C73A384029F
    Enumeration e = new Enumeration();
    e.setUmlClassifier(aClass);
    e.setId(aClass.getId());
    String name = new StringTransformer().transform(aClass.getName());
    e.setName(name);
    e.setIsFinal(true);
    e.setUserImportsEnabled(false);

    // add attributes as enumeration constants
    java.util.Vector attribs = aClass.getAttributes();
    for (int i = 0; i < attribs.size(); i++)
    {
      Attribute at = attribs.elementAt(i);
      Attribute javaAttribute = new Attribute();
      e.addToAttributes(javaAttribute);
      ....
      if(at.hasType())
        new TransformationError(at,"no attribute types allowed here");
      ....
    }
    return e;
    // @ENDPROTECT
 }
```


© LIANTIS

LIANTIS IT-Consulting

Standardkomponenten

- UML-Metamodell (OMG-Standard)
- Java-Metamodell
- EJB-Metamodell
- C++-Metamodell
- SQL-Metamodell
- C#-Metamodell
- UML → Java-Transformation
- UML → EJB-Transformation
- UML → C++-Transformation
- UML → SQL-Transformation
- UML → C#-Transformation
- Transaktionsgesicherte Generierung unter Erhalt von manuell geschriebenem Code

© LIANTIS

LIANTIS IT-Consulting

Beispielprojekt

- Abrechnungssystem für Telekommunikationsunternehmen und Energieversorger
- Produktiv seit 1998
- Bestehende C++-Architektur
- Modellierung mit Rational Rose
- Generierung mit XCoder/C++
- Koexistenz von manuell erstelltem und generiertem Code
- Schrittweise Migration zu vollständiger Generierung
- Aufwand für XCoder/C++-Anpassung: 20 PT

© LIANTIS

LIANTIS IT-Consulting

Was bedeutet ... für Sie?

- ✓ Schnelle Einführung durch leichtgewichtigen Ansatz
- ✓ Leistungsfähige Lösungen durch Metamodelle und Modelltransformation
- ✓ Flexible und einfache Erweiterung durch modellierten Generator

© LIANTIS

LIANTIS IT-Consulting

Liantis: Kompetenz im Bereich MDA

- 1997
- 1998 - 1999
- 1999
- 1999 - 2001
- 2001 - ...
- 2002 - 2003
- 2004

1. Einsatz v. MDA-Standardwerkzeugen
2. Entwicklung v. MDA-Individuallösungen

- Modellgetriebene Entwicklung seit 1997
  - COBOL/Java: Versicherungsbranche
  - COBOL: Finanzdienstleistungsbranche
  - Java/EDIFACT: Standardisierungsorg.
  - C++: Immobilienwirtschaft
  - Java/C++/C#/J2EE: SE-Werkzeuge
  - C++: Telekommunikation
  - J2EE: Energiewirtschaft
  - J2EE: Bank

© LIANTIS

Geschäftsfelder Liantis

- Beratung und Training
- Software-Entwicklung

- Liantis berät und trainiert zu folgenden Themen:
  - Objektorientierte Software-Entwicklung, insb. UML
  - Software-Architekturen, insb. MDA, EJB, CORBA
  - XML
  - Generierungsverfahren
  - Anforderungsanalyse
  - Vorgehensmodelle
  - Projektmanagement

© LIANTIS

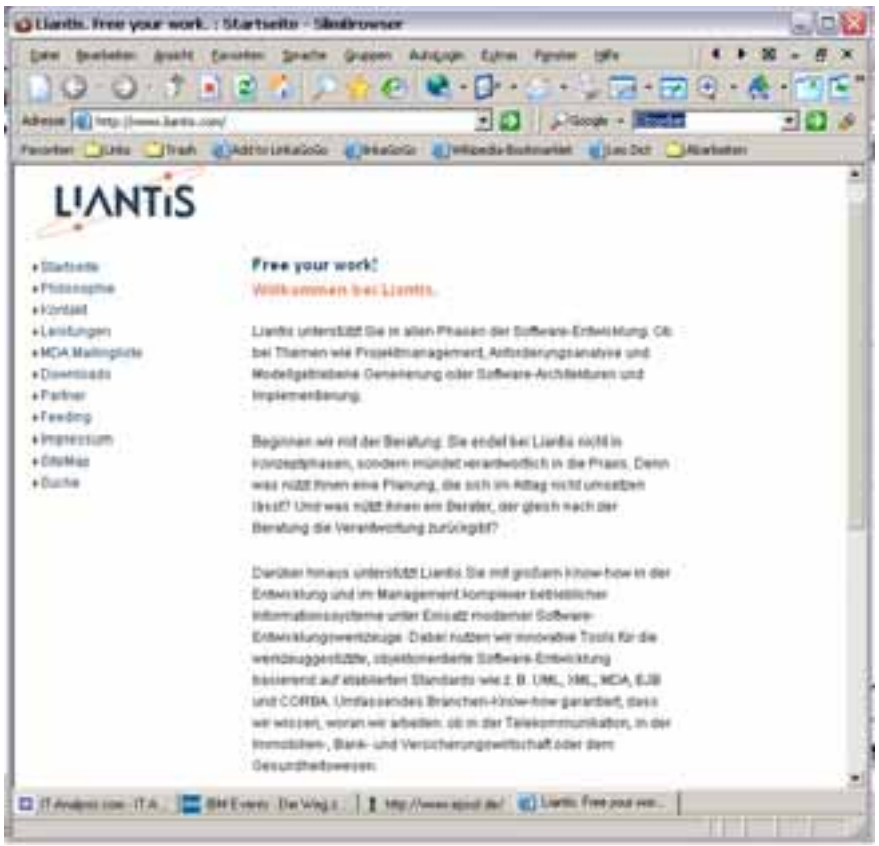
Geschäftsfelder Liantis

- Beratung und Training
- Software-Entwicklung

- Liantis verfügt über langjähriges Know-how in der Entwicklung komplexer Informationssysteme.
- Wir unterstützen Unternehmen bei der Realisierung ihrer Software-Entwicklungsprojekte.
- Dabei werden modernste Softwaretechnologie und Entwicklungswerkzeuge eingesetzt.

© LIANTIS

Free your work.



LIANTIS
IT-Consulting

Free your work.

Durch:

Constantin Szallies
constantin.szallies@liantis.com

Liantis GmbH & Co. KG
St.-Anton-Straße 69 - 71
47798 Krefeld
Fon: 0 21 51 / 931 86-60
Fax: 0 21 51 / 931 86-61
info@liantis.com
www.liantis.com

LIANTIS
IT-Consulting

Free your work.

Wir wollen Ihnen die Freiheit geben,
sich auf Ihr Geschäft zu konzentrieren.