MDSD mit Xtext und Xtend

Roger Gilliar

Was ist ein Modell?

Abbildung

Verkürzung

Pragmatismus

Wer?

Wann?

Wozu?

Modellgetriebene Softwareentwicklung

Aus formalen Modellen lauffähige Programme erzeugen

Verwendung einer DSL mit Codegeneratoren oder Interpretern*

Beispiel Modell

Oracle Data Dictionary

select * from user_tables

select * from dba_free_space

Beispiel DSL

Fachsprache

Reguläre Ausdrücke

SQL

XUnit Frameworks

Beispiel DSL

```
<changeSet author="jsmith" id="2">

    <createTable tableName="employees">
        <column name="id" type="number(4,0)">
              <constraints nullable="false" primaryKey="true"
                           primaryKeyName="EMP_PK"/>
        </column>
        <column name="ename" type="varchar2(14)"
                remarks="The first and last name"/>
        <column name="salary" type="number(6,2)"
                remarks="The full remuneration"/>
        <column name="dpt_id" type="number(4,0)"/>
    </createTable>

    <addForeignKeyConstraint baseColumnNames="dpt_id"
                             baseTableName="employees"
                             constraintName="emp_dpt_fk"
                             referencedColumnNames="id"
                             referencedTableName="departments"/>

</changeSet>
```

Beispiel DSL

NeubewertungMesswert(TypA) :
erzeugeNeuenMesswert(TypB) und
erzeugeNeuenMesswert(TypC) wenn
messwertExistiertNicht(TypD)

Beispiel DSL

```
new Html => [

   head [
       it.title [$("HTML with Xtend")]
   ]

   body [
       h1 [$("HTML with Xtend")]
       p [$("this format can be used as an alternative to templates.")]

       a("http://www.xtend-lang.org") [$("Xtend")]

      p [
          $("This is some ")
          b[$("mixed")]
          $(" text. For more see the ")
          a("http://www.xtend-lang.org")[$("Xtend")]
          $(" project")
        ]
        p [$("More text.")]
   ]
]
```

Beispiel DSL

```
Feature: Addition

  In order to avoid silly mistakes
  As a math idiot
  I want to be told the sum of two numbers

  Scenario: Add two numbers
    Given I have entered "50" into the calculator
    And I have entered "70" into the calculator
    When I press "add"
    Then the result should be "120"
```

Warum überhaupt eine DSL verwenden ?

Deklarative Beschreibung

Bessere Lesbarkeit

Weniger technischer Code

Statische Validierung

Leichte Erlernbarkeit

```
ui_table {
  config {Scrollable, MultiSelection}
  content_provider Array
  column "Internal-Id"    width 100  property #internId
  column "Description“    width 500  property #desc
}
```

DSL = 6 Zeilen
Java Code = 56 Zeilen

Nachteile?

Vendor Lockin

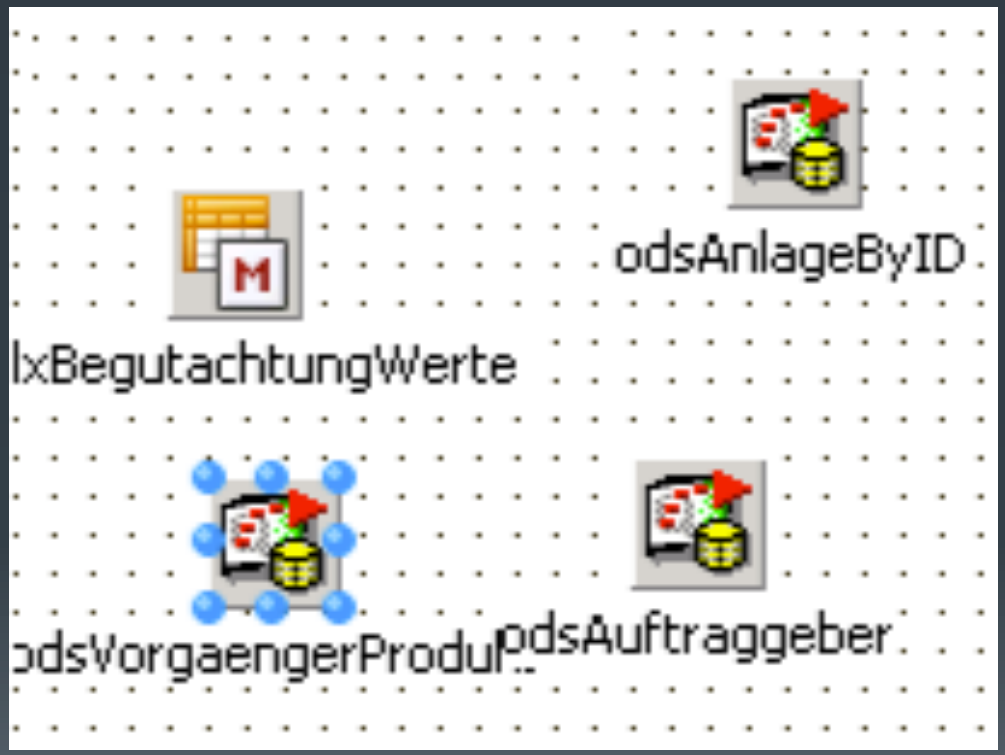
Lernaufwand

Toolsupport für DSL

Finden des Abstraktionsniveaus

Initialaufwand

Motivation?


odsAnlageByID
lxBegutachtungWerte
odsVorgaengerProdul
odsAuftraggeber

```
class Behaelterinfo < ActiveRecordSqlBase

  entity :BehaelterInfo, :preload => true,
        :load_all_order_by => :Beschreibung do

     column :Groesse,                  :IGroesse
     column :VolumenInM3,              :Double, :field_name => "vol"
     column :Special,                  :IJaNein
     column :Kurzbez,                  :string
     column :KurzbezFakturierung,      :string
     column :Beschreibung,             :string
     column :LeergewichtInKg,          :Double

     find_by :columns => [:=> :Groesse],
           :result => :record, :IBehaelterInfo
           :verify => :only_one_found

  end

end
```

Java Code Generierung

Vorteil

Deklarative Beschreibung

Bessere Lesbarkeit

Weniger technischer Code

Statische Validierung

Leichte Erlernbarkeit

Nachteil

Weitere Sprache Ruby/ JRuby

Toolsupport

Alternativen ?

http://createyourproglang.com/

JetBrains MPS

http://www.heise.de/developer/artikel/mbeddr-Embedded-Entwicklung-mit-erweiterbarem-C-1647220.html

Spring ROO

(Source is the model)

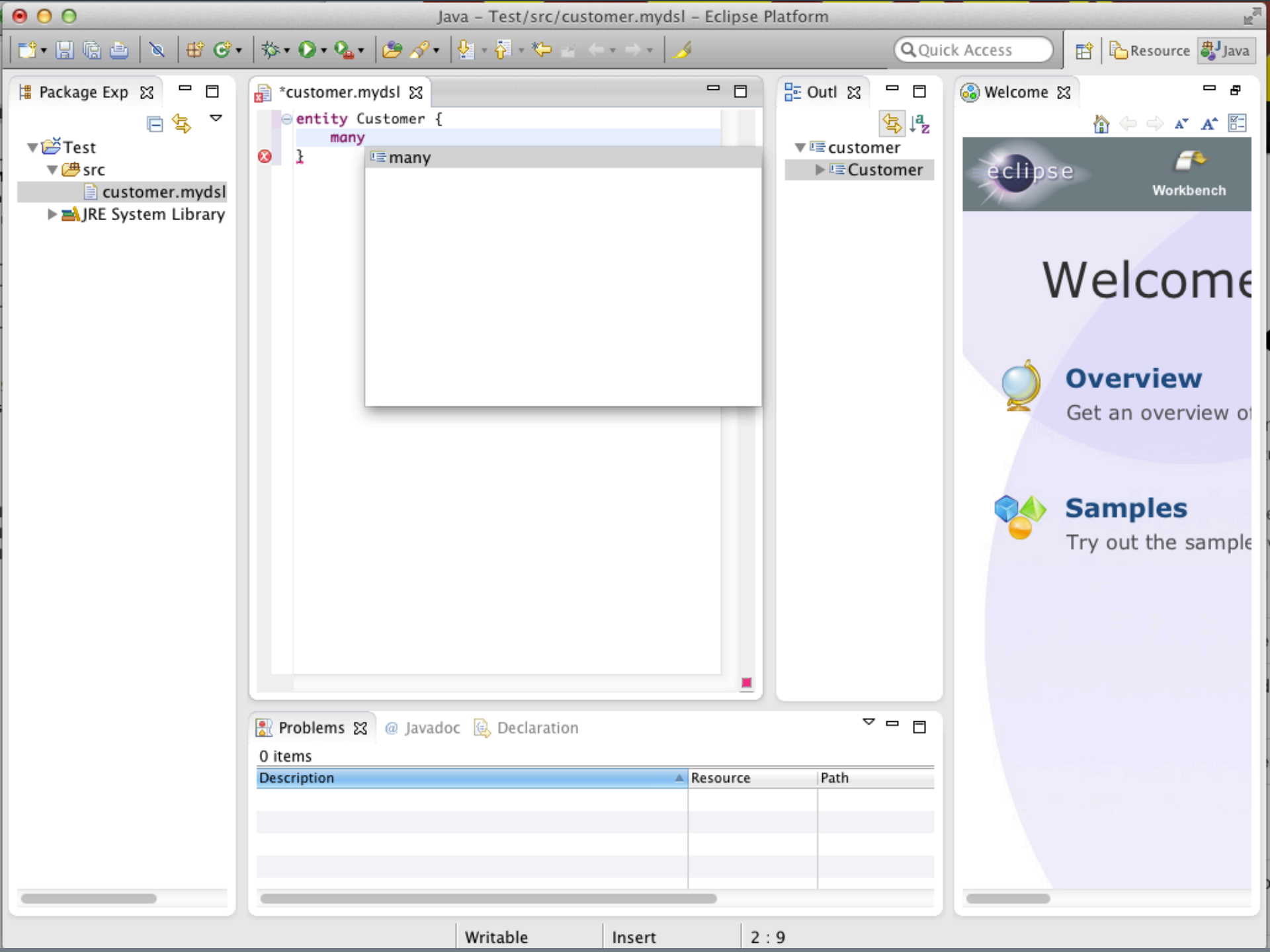
JaMoPP

(Source is the model)

EMF

Xtext/Xtend

```
grammar org.example.domainmodel.Domainmodel with
                                org.eclipse.xtext.common.Terminals

generate domainmodel “example.domainmodel”

Domainmodel :
  elements += Type*
;

Type:
  DataType | Entity
;

DataType:
  'datatype' name = ID
;

Entity:
  'entity' name = ID ('extends' superType = [Entity])? '{’
     features += Feature*
  '}’
;

Feature:
  many?='many'? name = ID ':' type = [Type]
;
```


Java – Test/src/customer.mydsl – Eclipse Platform
Quick Access
Resource
Java
Package Exp
Test
src
customer.mydsl
JRE System Library
*customer.mydsl
entity Customer {
many
}
many
Outl
customer
Customer
Welcome
eclipse
Workbench
Welcome
Overview
Get an overview of
Samples
Try out the sample
Problems
Javadoc
Declaration
0 items
Description
Resource
Path
Writable
Insert
2 : 9

Xtend

Coffeescript for Java

Spracheigenschaften

Keine Semikolons

```
val page = PlatformUI::workbench.activeWorkbenchWindow.activePage
Environment::enableTest()
display = Display::current
```

Everything is an expression

```
val data = try {
  fileContentsToString('data.txt')
} catch (IOException e) {
  'dummy data'
}
```

var/val

```
val combo = widget1 as Combo

var data = "String”
data = "New Data"
```

Typeinference

```
val greetings = newHashMap(
  "german"  -> "Hallo",
  "english" -> "Hello"
)

val jobManager = Job::jobManager
```

Typeinference Methods

```
def rowCount(EventDestination ed) {
    return ed.wf.doCommandReturnInt(
        new Property(ed.propertyName),
        [widget |
            val table = widget as Table
            table.itemCount])
}
```

Keine Checked Exceptions

Dispatch Methods

```
def dispatch printType(Number x) {
  "it's a number"
}

def dispatch printType(Integer x) {
  "it's an int"
}
```

Extended Switch Expression

```
val myString = "Hello“
switch myString {
  case myString.length > 5 : "a long string."
  case 'some'              : "It's some string."
  default                  : "It's another short string."
}
```

```
val x = list(1, 2, 3) as Object
switch x {
  String case x.length > 0 : x.length
  List<?>                  : x.size    List
  default : -1
}
```

Extension Methods

```
def removeVowels (String s){
  s.replaceAll("[aeiouAEIOU]", "")
}

removeVowels("Hello")
"Hello".removeVowels
```

Template

```
def someHTML(List<Paragraph> paragraphs) '''
                <html>
                  <body>
                    «FOR p : paragraphs BEFORE '<div>'
                       SEPARATOR '</div><div>' AFTER '</div>'»
                      «IF p.headLine != null»
                        <h1>«p.headline»</h1>
                      «ENDIF»
                      <p>
                        «p.text»
                      </p>
                    «ENDFOR»
                  </body>
                </html>
              '''
```

@Property / @Data

@Property String name

Functional Programming

Functional Programming I

```
val colors = list("red", "blue", "green")
colors.head => "red"
colors.tail => iterable("blue", "green")
colors.last => "green"
colors.empty => false
```

Functional Programming II

```
var (String)=>int lambda = [length]
lambda.apply("hello")
=> 5
```

Functional Programming III

```
list("a string", 42, true).filter(typeof(String))
 =>  iterable("a string")

list("red", "blue", "green").filter[startsWith("b")]
 => list("blue")

val strings = list("red", "blue", "green")
val charCount = strings.map[s|s.length]
          .reduce[sum, size | sum + size]

charCount => 12
```

Xtext

```
grammar org.xtext.example.hdsl.HibernateDsl with org.eclipse.xtext.xbase.Xbase
generate hibernateDsl "http://www.xtext.org/example/hdsl/HibernateDsl"

DomainModelFile:
        ('package' name=QualifiedName)?
        imports+=Import*
        entities+=Entity*
;

IMPORT:

  'import' importedNamespace=QualifiedName
;

Entity:
    'entity' name=ValidID ('extends' superType=JvmTypeReference)? '{'
        features+=Property*
    '}'
;

Property:
   name=ValidID ':' type=JvmTypeReference (notNull?='NotNull')?
        (columnName=ValidID)?
;
```

```
import de.mcs.company.*

/*
A customer
*/
entity Customer {
     name              : String notNull
     lastname          : String
     company           : Company
     age               : Integer
}
```

```
class HibernateDslJvmModelInferrer extends AbstractModelInferrer
{

  @Inject extension JvmTypesBuilder
  @Inject extension IQualifiedNameProvider

  extension TypesFactory = TypesFactory::eINSTANCE

  def dispatch infer(Entity entity, IJvmDeclaredTypeAcceptor
       acceptor, boolean prelinkingPhase) {

       acceptor.accept(
              entity.toClass( entity.fullyQualifiedName )
       ).initializeLater [

              documentation = entity.documentation
              annotations +=
                entity.toAnnotation("javax.persistence.Entity")
```

```
members += entity.toConstructor() []
```

```
for ( f : entity.features ) {

    switch f {

        Property : {
         val a = f.toAnnotation("javax.persistence.Column")
         val v  = createJvmStringAnnotationValue
         val op = a.annotation.members
             .filter(typeof(JvmOperation))
              .filter(o|o.simpleName == "name").head

         v.operation = op; v.values += f.name; a.values += v

         val field = f.toField(f.name, f.type) [
               annotations += a
               if (f.notNull) {
                 annotations +=
                    f.toAnnotation
                      ("javax.validation.constraints.NotNull")
               }
        ]
        members += field
```

```
members += entity.toMethod("get" + f.name.toFirstUpper, f.type) [
        body = [append('''
            return this.«f.name»;''')]
                                                ]


members += entity.toMethod("set" + f.name.toFirstUpper,
        entity.newTypeRef(Void::TYPE)) [
                parameters += toParameter(f.name, f.type)
                body = [append('''
                        columnValues.put("«f.name»", «f.name»);
                        this.«f.name» = «f.name»;''')]
                                                ]
```

```
val stringType = entity.newTypeRef(typeof(String))
val objectType = entity.newTypeRef(typeof(Object))

members += entity.toMethod("setKey", entity.newTypeRef(Void::TYPE))
[
   parameters += toParameter("key", stringType)
   parameters += toParameter("value", objectType)
   body = [append('''
   «FOR f : fields»
        if (key.equals("«f.simpleName»")) {
          set«f.simpleName.toFirstUpper»(
               («f.type.simpleName»)value);
               return;
        };
   «ENDFOR»''')]]]
```

Resultate

Episode 36: Modellierst du schon oder programmierst du noch?

roger.gilliar@mcs.de

Quelle:
http://www.eclipse.org/xtend/
http://www.eclipse.org/Xtext/
http://jnario.org/org/jnario/jnario/documentation/20FactsAboutXtendSpec.html
http://de.wikipedia.org/wiki/Modell