

# Using Domain Specific Languages for Product Line Engineering

SPLC 2009 Tutorial

Markus Voelter  
Independent/Itemis  
[www.voelter.de](http://www.voelter.de)  
[voelter@acm.org](mailto:voelter@acm.org)  
**itemis**



Variability      MDD Tooling  
Configuration    Model Variability  
Customization    Transformation Var.  
MDD Intro        Summary & Wrapup



**Variability**      MDD Tooling  
Configuration      Model Variability  
Customization      Transformation Var.  
MDD Intro          Summary & Wrapup



# Variability

... differences among  
products in PL

## Variation Point

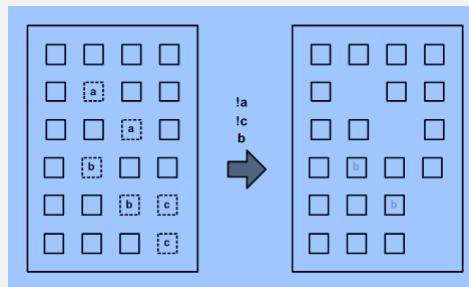
- ... a point where a variation can occur
- ... must be bound for each product
- ... bind when?
- ... bind how?

## Binding Time

	flexibility	performance	code size	complexity
source time	-	+	+	-
compile time	+	+	+	-
link time	+	+	+	-
load time	++	+	+	+
run time	+++	-	-	+

## Variability Mechanisms Removal

... optionally take away  
from overall whole



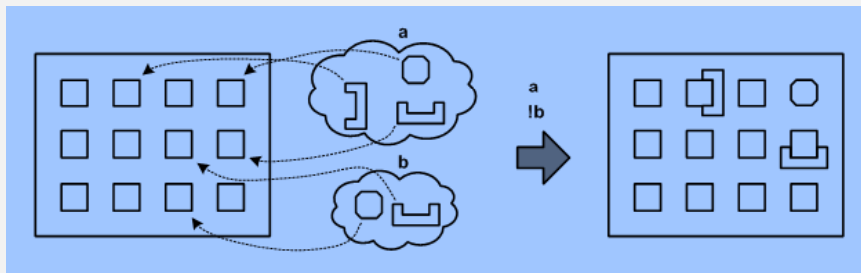
## Variability Mechanisms Removal

... optionally take away  
from overall whole

**Challenge:**  
overall whole can  
get big and unwieldy

## Variability Mechanisms Injection

... optionally add to  
minimal core



## Variability Mechanisms Injection

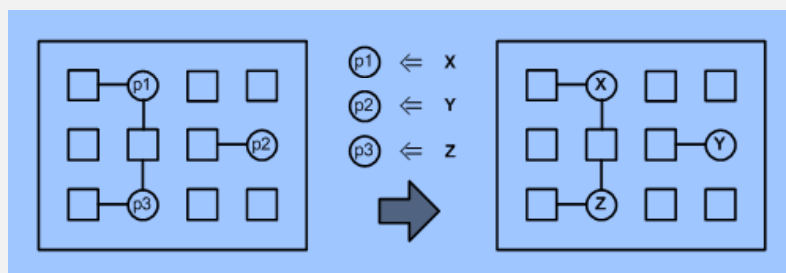
... optionally add to  
minimal core

### Challenge:

how to point into the core  
and add something to it

## Variability Mechanisms Parametrization

... define values for predefined params

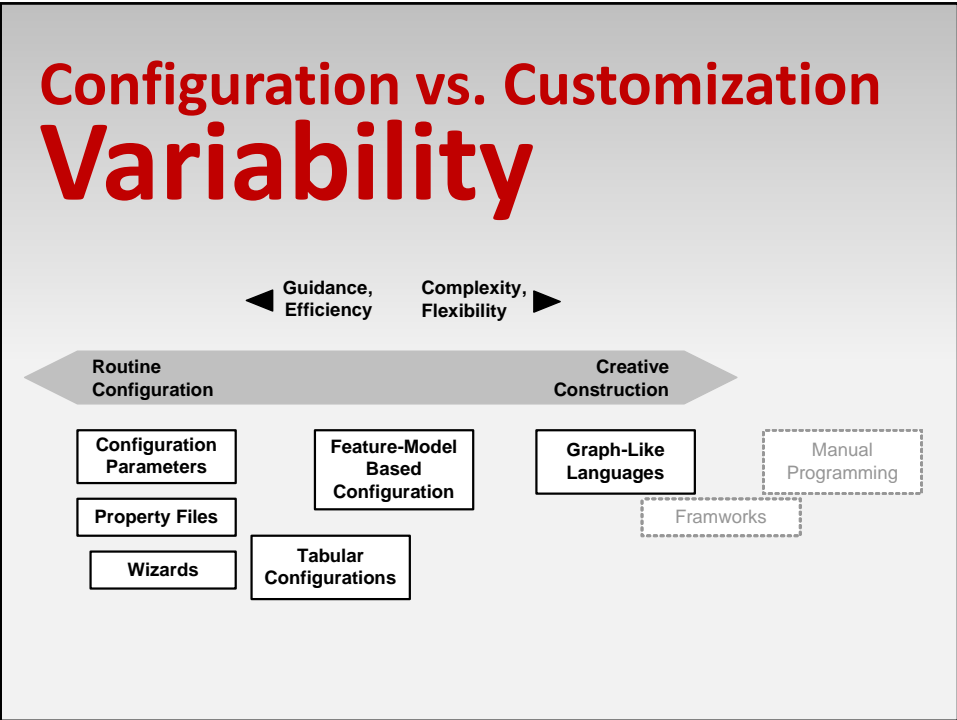


## Variability Mechanisms Parametrization

... define values for predefined params

### Challenge:

types for parameters can be non trivial (DSLs)



Variability      MDD Tooling  
Configuration      Model Variability  
Customization      Transformation Var.  
MDD Intro      Summary & Wrapup

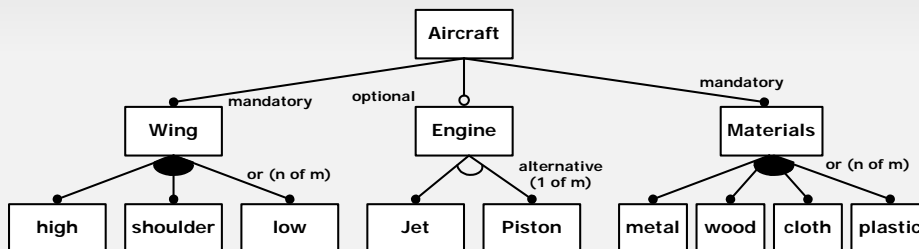
A close-up photograph of a professional audio mixing console, showing several white sliders and knobs on a dark surface, with a blurred background of more controls.

# Configuration

... selecting options  
... setting param values

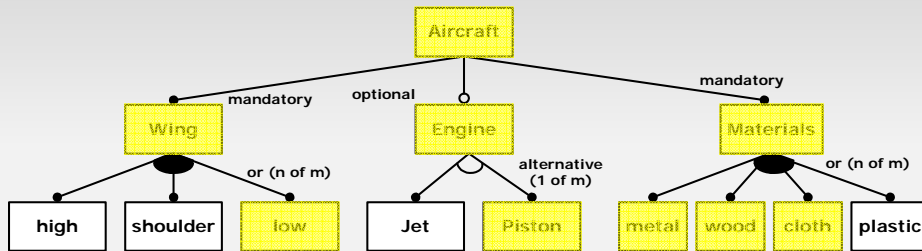


# Configuration Feature Models





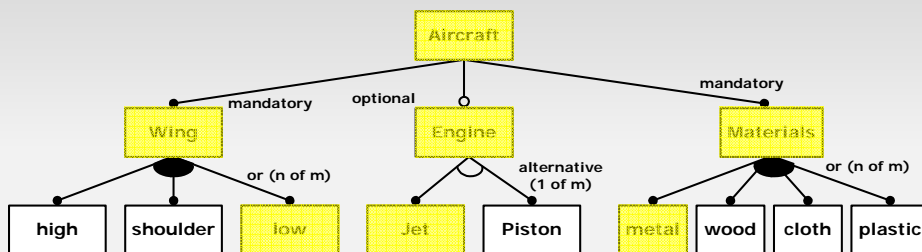
# Configuration Feature Models



## Robin DR-400

An aircraft with a low wing, piston engine and made of metal, wood and cloth

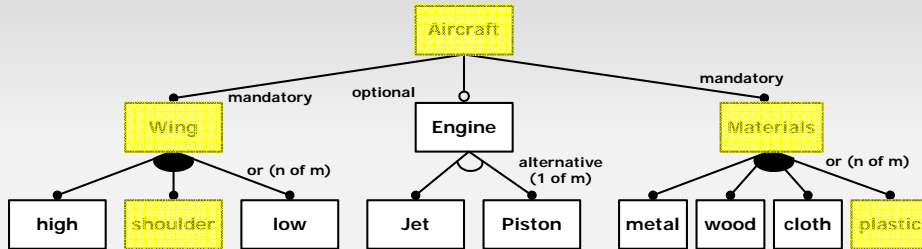
# Configuration Feature Models



## Airbus A 320

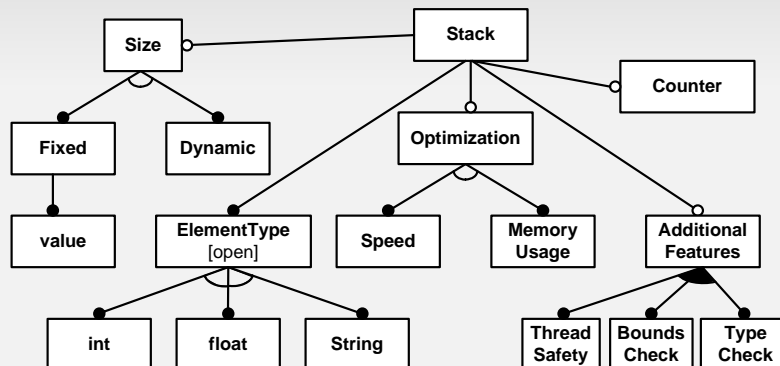
An aircraft with low wing, jet engine(s) and made of metal.

# Configuration Feature Models

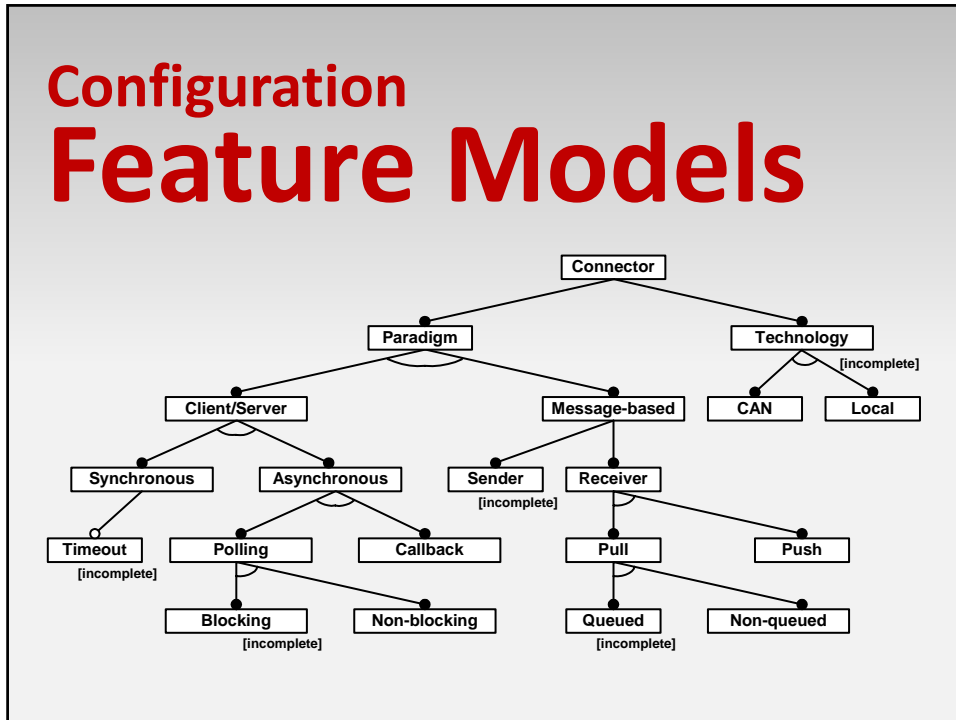


**Schleicher ASW 27**  
An aircraft with shoulder wing, no engine and made of plastic

# Configuration Feature Models



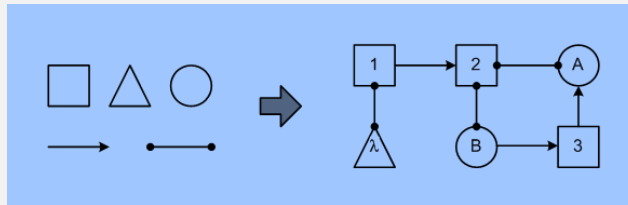
# Configuration Feature Models



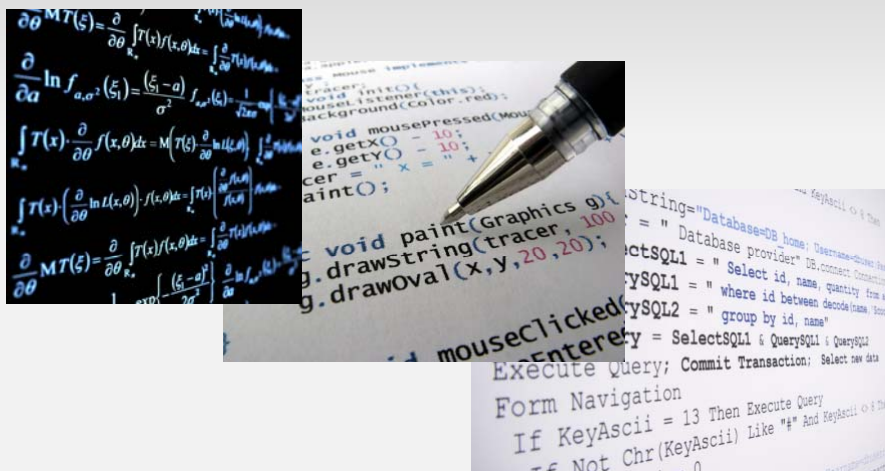
Variability      MDD Tooling  
 Configuration      Model Variability  
**Customization**      Transformation Var.  
 MDD Intro      Summary & Wrapup

# Customization

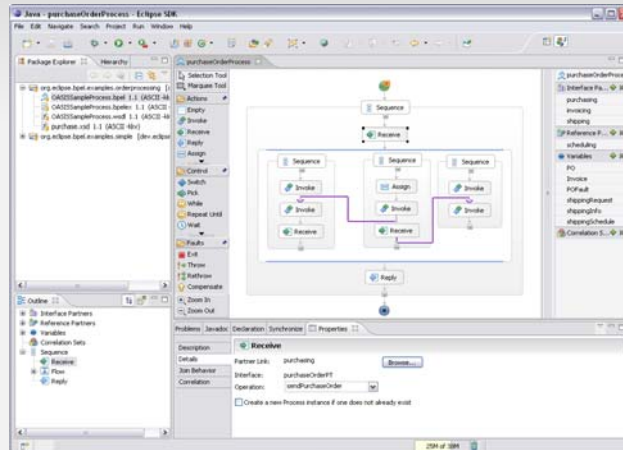
... „real languages“  
 ... instantiation  
 ... connections



# Customization Languages



# Customization Languages



# Customization Languages

```

data (
  data Patient persistent (
    name: String
    age: int range 0..120
    firstName: String(20) regex "[a-z]+"
    birthDate: String
    insuranceName: String
    insuranceNumber: String
    -> adr: Address
    -> tel: Telephone*
    -> stays: Stay*
  )
  data Address persistent (
  )
  data Telephone (
    countryCode: String
    cityCode: String
    localNumber: String
  )
  data Stay (
    beginDate: String
    -> stations: Station*
    endDate: String
  )
  data Station (
    date: String
    stationName: String
  )
)

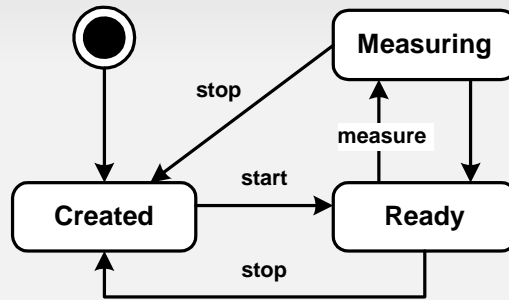
```

```

records (
  record P001 (
    p: Patient
    1 -> p.name
    2 -> p.firstName
    3 -> p.birthDate
  )
  record A001 (
    a: Address
    1 -> a.street
    2 -> a.zip
    3 -> a.city
    p.adr = a
  )
  record V001 (
    1 -> p.insuranceName
    2 -> p.insuranceNumber
  )
  record T001 (
  )
  record S001 (
  )
)

```

# Customization Languages




Variability      MDD Tooling  
 Configuration    Model Variability  
 Customization    Transformation Var.  
**MDD Intro**      Summary & Wrapup

$$\int_{R_n} T(x) \cdot \left( \frac{\partial}{\partial \theta} \ln L(x, \theta) \right) \cdot f(x, \theta) dx = \int_{R_n} T(x) \left( \frac{\frac{\partial}{\partial \theta} f(x, \theta)}{f(x, \theta)} \right) f(x, \theta) dx$$

$$\frac{\partial}{\partial \theta} \mathbf{M}T(\xi) = \frac{\partial}{\partial \theta} \int_{R_n} T(x) f(x, \theta) dx = \int_{R_n} \frac{\partial}{\partial \theta} T(x) f(x, \theta) dx$$


$$1 \quad \int_{R_n} \left\{ -\frac{(\xi_1 - a)^2}{2\sigma^2} \right\} \frac{\partial}{\partial \sigma} \ln f_{\text{norm}}(\xi) = \frac{\xi_1 - a}{\sigma^2}$$



**programming**  
started  
**close to the hardware**

**abstractions**  
~  
**computing**

chips



**abstractions**  
~  
**computing**

bits



```

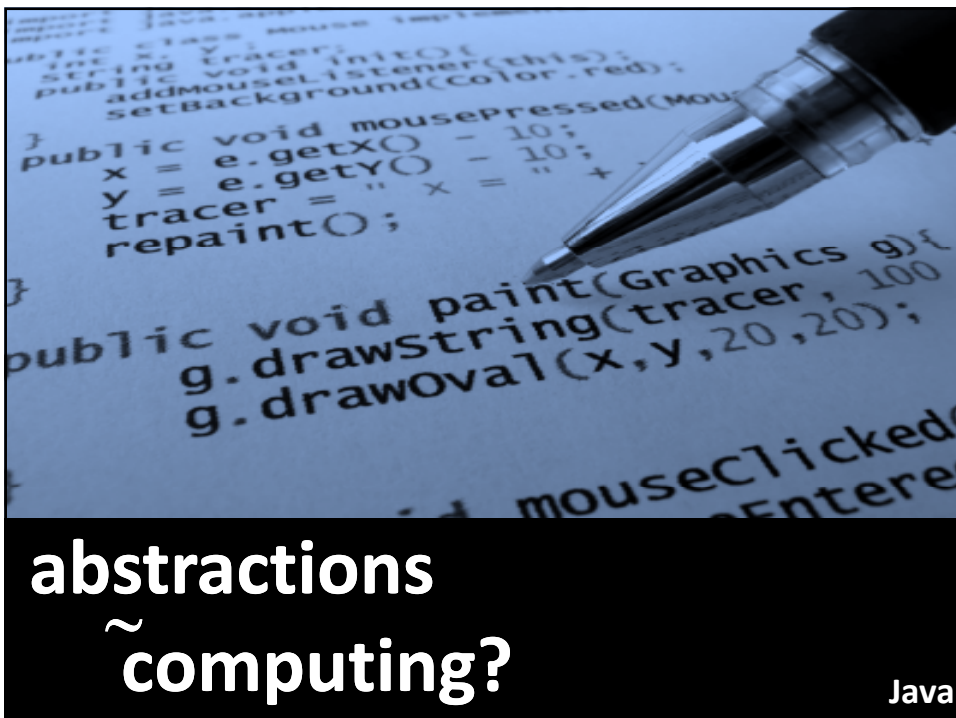
decodeMessage( res ) {
  0; i < MAX_RES_LEN; i++) buf[i] = 0;
  i = 0;
  for (int i = 0; i < res.length; i++) {
    buf[i] = res[i];
  }
}

public int[] extractMessage(int[] res) {
  for (int i = 0; i < MAX_RES_LEN; i++) buf[i] = 0;
  int loc = 0;
  while (i < res.length) {
    ...
  }
}

```

**abstractions**  
~  
**computing**

C



```

public void mousePressed(MouseEvent e) {
  x = e.getX() - 10;
  y = e.getY() - 10;
  tracer = "x = " + x + " y = " + y;
  repaint();
}

public void paint(Graphics g) {
  g.drawString(tracer, 100, 100);
  g.drawOval(x, y, 20, 20);
}

```

**abstractions**  
~  
**computing?**

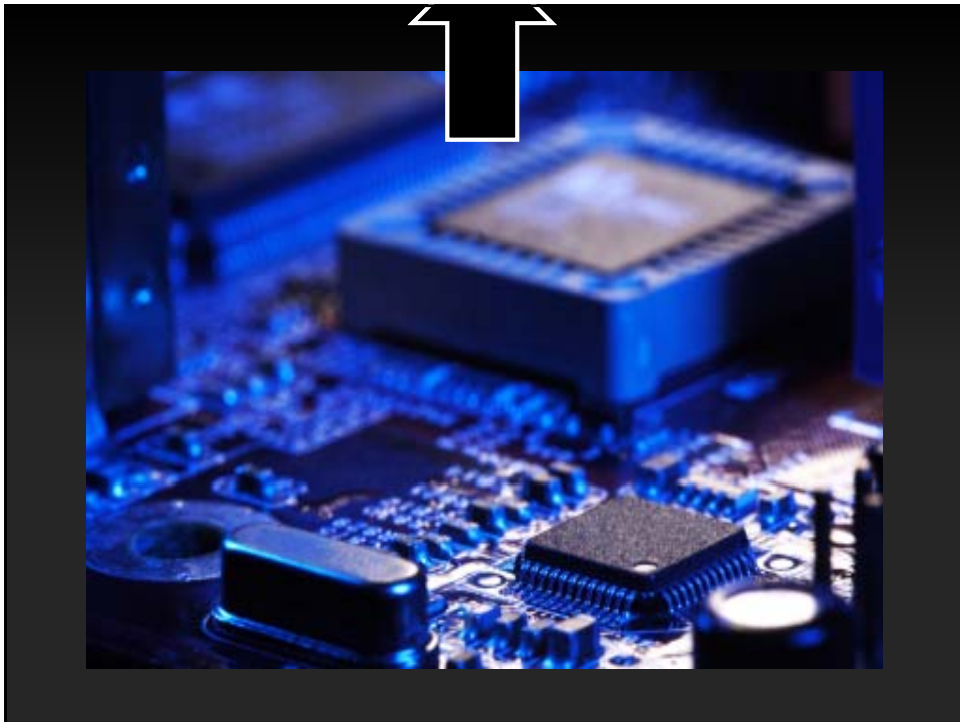
Java

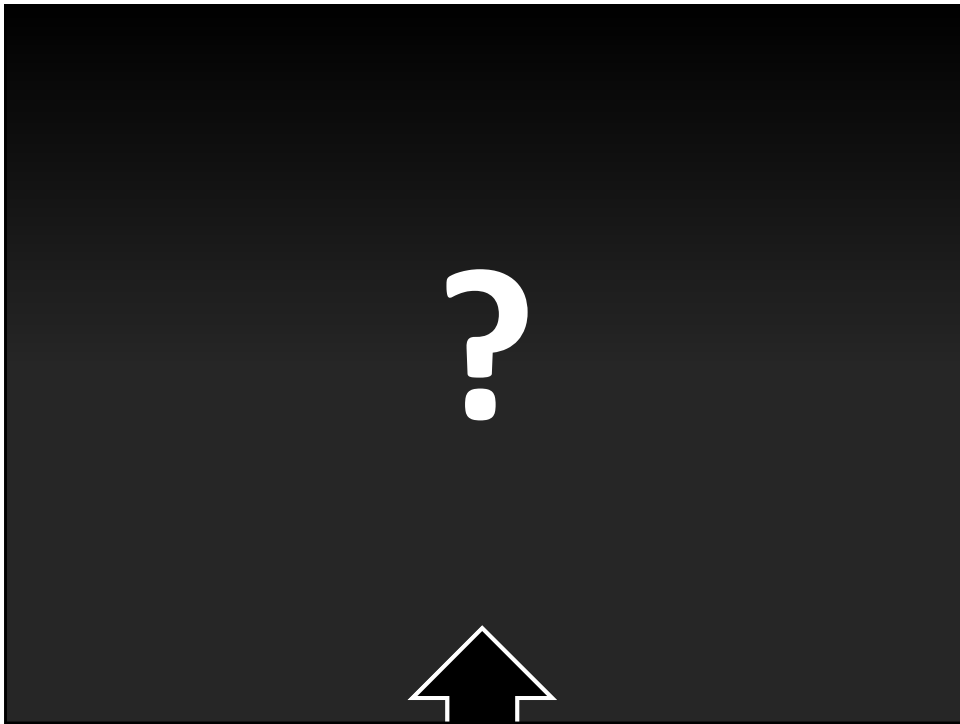


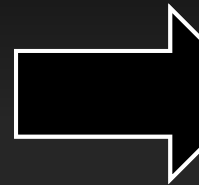
```
ConnectionString="Database=DB_home; Username=duser; Password=  
DBProvider = " Database provider" DB.connect ConnectionStr  
SelectSQL1 = " Select id, name, quantity from all_  
QuerySQL1 = " where id between decode(name,'Scott'  
QuerySQL2 = " group by id, name"  
SelectQuery = SelectSQL1 & QuerySQL1 & QuerySQL2  
Execute Query; Commit Transaction; Select new data  
Form Navigation  
If KeyAscii = 13 Then Execute Query  
If Not Chr(KeyAscii) Like "#" And KeyAscii > 0
```

**abstractions  
~  
computing?**

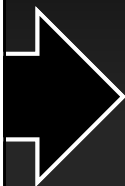
SQL







**general purpose**



**domain specific**

**tailor made**  
**effective++**

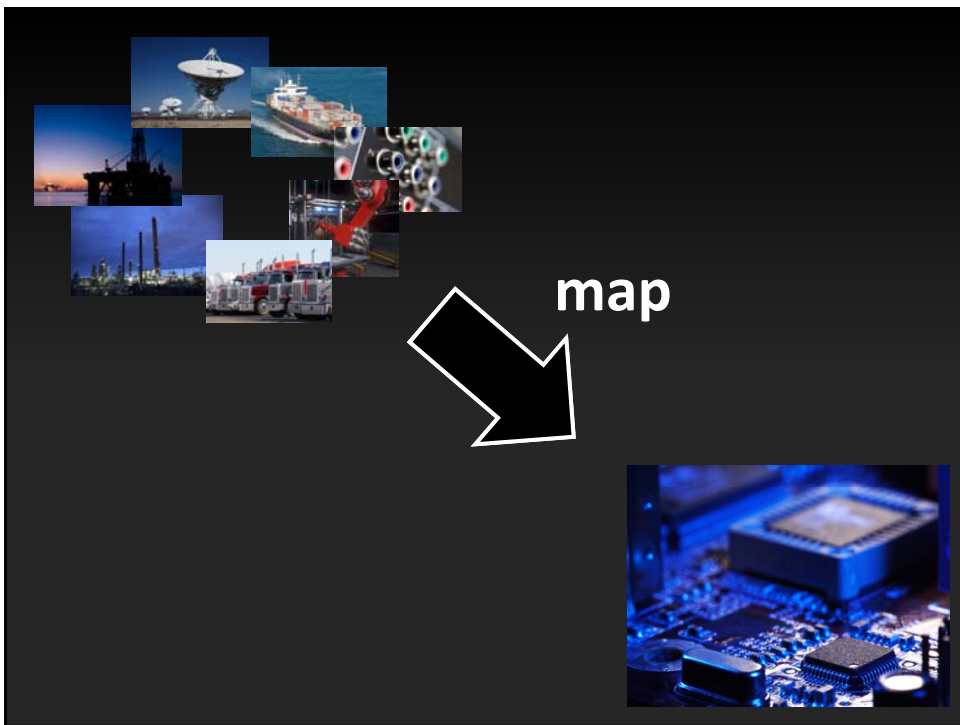


**specialized, limited**

**used by experts**

**together with other**  
**specialized tools**

A DSL is a **focussed, processable language** for describing a specific **concern** when building a system in a **specific domain**. The **abstractions** and **notations** used are natural/suitable for the **stakeholders** who specify that particular concern.



**DSL Program**  
(aka Model)

**automated!**

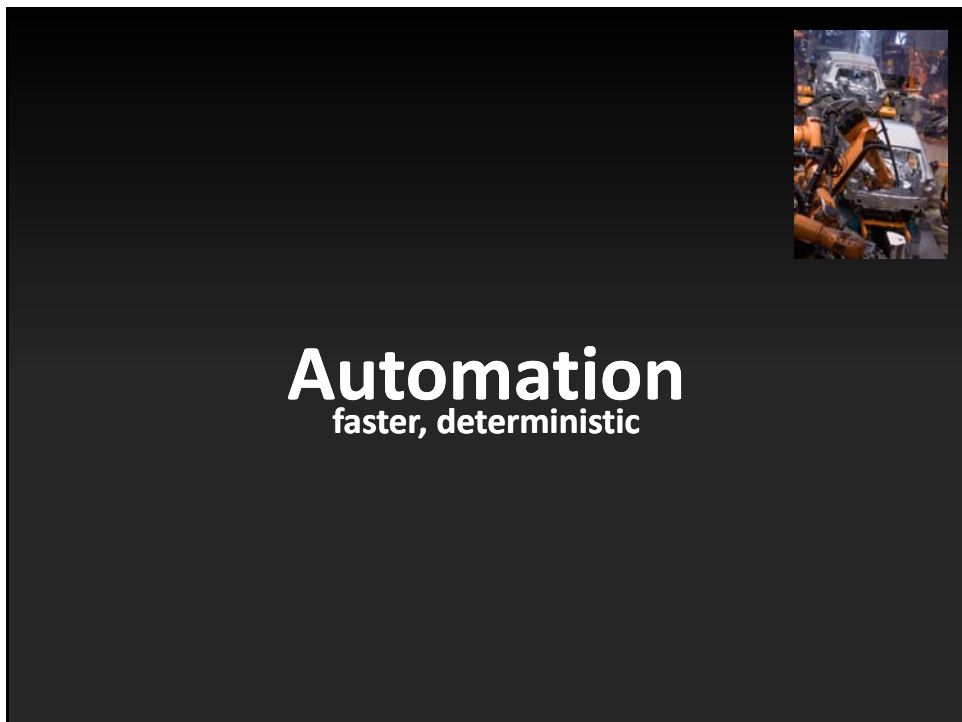


**GPL Program**



**Generation**  
Transformation  
Compilation

**Interpretation**





## Increased Quality

well defined structures allthrough the system



## Meaningful Validation

more semantics in the model





# Capture Domain Knowledge

formalized into languages and models

$$\frac{\partial}{\partial \theta} M T(\xi) = \frac{\partial}{\partial \theta} \int_{\mathcal{X}} f(x) p(x) dx = \int_{\mathcal{X}} \frac{\partial}{\partial \theta} f(x) p(x) dx$$

$$\frac{\partial}{\partial \theta} \ln f_{\theta}(x) = \frac{\partial}{\partial \theta} \ln \left( \frac{1}{Z(\theta)} \int_{\mathcal{Y}} f(x, y) p(y) dy \right) = \frac{\partial}{\partial \theta} \ln \int_{\mathcal{Y}} f(x, y) p(y) dy - \frac{\partial}{\partial \theta} \ln Z(\theta)$$

$$\int_{\mathcal{Y}} \frac{\partial}{\partial \theta} \ln f(x, y) p(y) dy = \int_{\mathcal{Y}} \frac{\partial}{\partial \theta} \ln f(x, y) p(y) dy = \int_{\mathcal{Y}} \frac{\partial}{\partial \theta} \ln f(x, y) p(y) dy$$

$$\frac{\partial}{\partial \theta} M T(\xi) = \frac{\partial}{\partial \theta} \int_{\mathcal{X}} f(x) p(x) dx = \int_{\mathcal{X}} \frac{\partial}{\partial \theta} f(x) p(x) dx$$

# Suitable Notations

textual, graphical, tabular



# Technology Independence

generate „technology glue code“



# Abstraction w/o Runtime Overhead

generator „optimizes away“



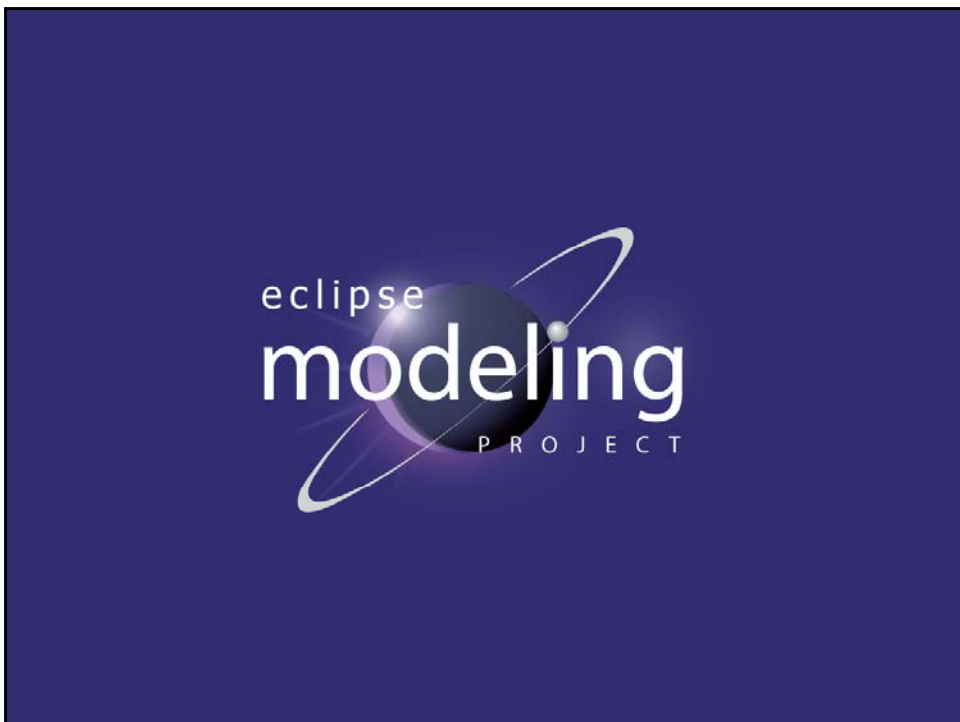
# Capture Implementation Strategy

in the generators



**Everything is a model**  
including for example hardware (some) hardware

Variability      **MDD Tooling**  
Configuration    Model Variability  
Customization   Transformation Var.  
MDD Intro      Summary & Wrapup



EMF



Ecore meta meta model

+

Editing

Transactions

Validation

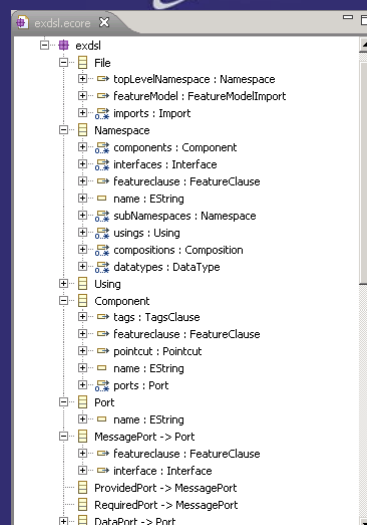
Query

Distribution/Persistence

EMF



Metamodel As Tree

can also be edited  
as UML-like diagram

# EMF



## Constraints

with OCL  
and dialects

```

import exdsl:

extension net::ample::adsl::exdsl::Extensions;
extension org::openarchitectureware::util::stdlib::io;

context Component ERROR "Qualified Name "+qualifiedName()+" must be unique"
allComponents().select( c | c.qualifiedName() == qualifiedName() ).size > 1;

context DataType ERROR "Qualified Name "+qualifiedName()+" must be unique"
allDataTypes().select( c | c.qualifiedName() == qualifiedName() ).size > 1;

context Namespace if !isEmpty() ERROR "Qualified Name "+qualifiedName()+" must be unique"
allNamespaces().select( c | c.qualifiedName() == qualifiedName() ).size > 1;

context emf::EObject if metaType.getProperty("name") != null ERROR "name not defined"
metaType.getProperty("name").get(this) != "Unnamed";

context Interface ERROR "interface names must start with a capital I":
name.startsWith("I");

context MessagePort ERROR "interface not defined. Missing a 'using'? ":
visibleInstancesOfType(this, Interface).contains(interface);

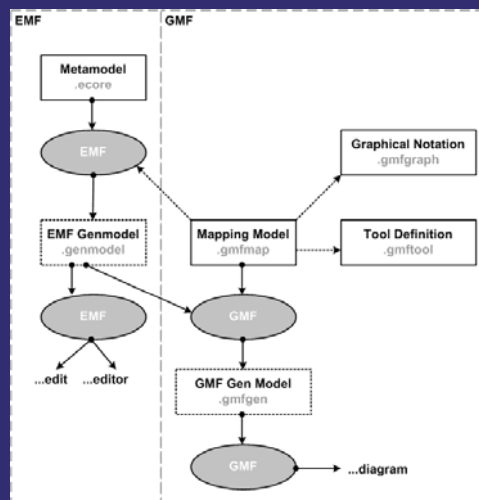
context Attribute ERROR "no type defined: "+type.name:
visibleInstancesOfType(this, DataType).contains(type);

context DataPort ERROR "data not defined: "+type.name:
visibleInstancesOfType(this, ComplexType).contains(type);
    
```

# GMF



Graphical  
Box/Line  
editors based  
on EMF



# TMF / Xtext



## Building Textual Editors

```
Namespace:
  "namespace" name=ID (featureclause=FeatureClause)? "{"
  (usings+=Using) +
  ( subNamespaces+=Namespace |
  components+=Component |
  datatypes+=DataType |
  interfaces+=Interface |
  compositions+=Composition ) *
  "}";

Using:
  "using" namespace=[Namespace|qualID];

Component:
  (pointcut=Pointcut)? "component" name=ID (tags=TagsClause)? (featurec
  (ports+=Port) *
  "}";

Port:
  MessagePort | DataPort;

MessagePort:
  ProvidedPort | RequiredPort;

ProvidedPort:
  "provides" name=ID ":" interface=[Interface] (featureclause=FeatureCl

RequiredPort:
```

# TMF / Xtext



## Building Textual Editors

The screenshot shows the Eclipse IDE with an Xtext editor open. The editor displays the following code:

```
namespace com {
  namespace airwizard {
    using com.airwizard.domaintypes
    using com.airwizard.types

    namespace shared {
      struct aaa {
        x : FlightAIDI
        y : Flights
      }

      typedef String FlightID

      struct FlightStatus {
        f : FlightStatus
        id: FlightID -- com.airwizard.shared
        eta: FlightStatus -- com.airwizard.shared
        flights -- com.airwizard.shared
        s : String -- com.airwizard.types
      }

      interface Time -- com.airwizard.domaintypes
      onew aaa -- com.airwizard.shared
      boolean -- com.airwizard.types
      int -- com.airwizard.types
    }
  }
}
```


The project explorer on the right shows a tree structure with folders like 'airwizard', 'shared', and 'FlightStatus'. The problem view at the bottom shows two errors:

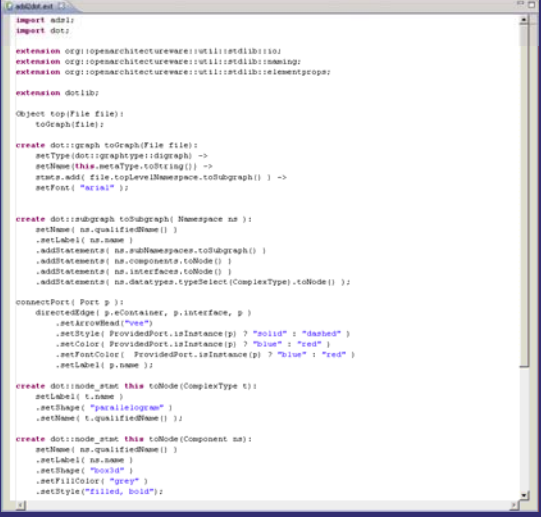
Description	Resource	Path	Location
Couldn't resolve reference to 'FlightAIDI'	components...	net.ample.addl.exidl.samp...	line : 14
no type defined: null	components...	net.ample.addl.exidl.samp...	line : 14

# M2M

## Model-to-Model Transformations

INRIA's ATL  
QVT  
Xtend





```

import atlib;
import dot;

extension org.eclipse.xtext.ecore:util::util::util;
extension org.eclipse.xtext.ecore:util::util::util::util::util;
extension org.eclipse.xtext.ecore:util::util::util::util::util::util;

extension dotlib;

Object top(File file);
toGraph(file);

create dot::graph toGraph(File file);
setType(dot::graphType::digraph) ->
setName(this.set::toGraphing()) ->
state.add( file.toLevelNamespace.toGraph() ) ->
setFont( "arial" );

create dot::subgraph toSubgraph( Namespace ns );
setName( ns.qualifiedName() );
setLabel( ns.name );
addStatements( ns.subNamespace.toGraph() );
addStatements( ns.composition.toNode() );
addStatements( ns.interface.toNode() );
addStatements( ns.datatypes.typeSelect(ComplexType).toNode() );

connectPort( Port p );
directEdge( p.container, p.interface, p );
setArrowHead( "tee" );
setStyle( ProvidedPort.isInstance(p) ? "bold" : "dashed" );
setColor( ProvidedPort.isInstance(p) ? "blue" : "red" );
setFontColor( ProvidedPort.isInstance(p) ? "blue" : "red" );
setLabel( p.name );

create dot::node_atc this toNode(ComplexType t);
setLabel( t.name );
setShape( "triangle-up" );
setName( t.qualifiedName() );

create dot::node_atc this toNode(Component ns);
setName( ns.qualifiedName() );
setLabel( ns.name );
setShape( "rect" );
setFillColor( "grey" );
setStyle( "filled, bold" );

```

# M2T

## Model-to-Text Transformations

JET: Java Emitter  
Templates  
Xpand: oAW's  
template engine





```

1 <<IMPORT imp>>
2
3 <<EXTENSION dataimport::dsl::impUtil>>
4
5 <<DEFINE root FOR DataStructure>>
6 <<FILE filename()>>
7
8 import java.util.List;
9 import java.util.ArrayList;
10 import dataimport.platform.DataBase;
11
12
13 public class <<classname()>> extends dataimport.platform.DataBase
14
15     <<FOREACH attributes AS a>>
16         private <<a.type>> <<a.name>>;
17     <<ENDFOREACH>>
18
19     <<FOREACH references AS r>>
20         <<IF r.isMulti>>
21             private List<<r.type.fqClassname()>> <<r.name>>List = ;
22         <<ELSE>>
23             private <<r.type.fqClassname()>> <<r.name>>;
24         <<ENDIF>>
25     <<ENDFOREACH>>
26
27     <<FOREACH attributes AS a>>
28         public void set<<a.name.toFirstUpper()>>( <<a.type>> value )
29

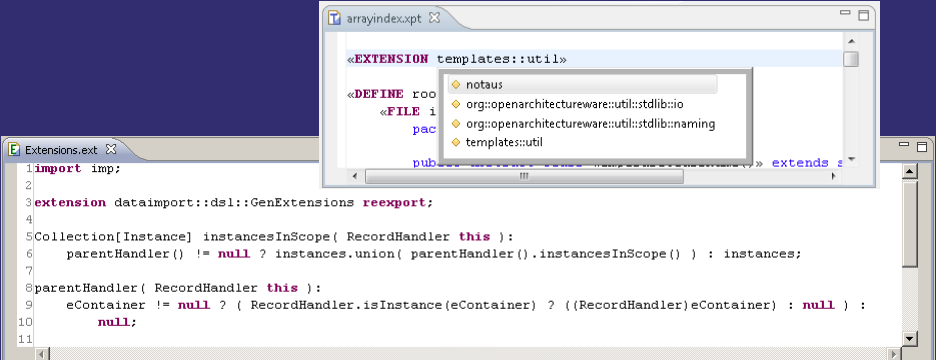
```



**M2T**

eclipse  
**modeling**  
PROJECT

**Model-to-Text Transformations**  
Extensions to modularize complex expressions



```

1 import imp;
2
3 extension dataimport::dsl::GenExtensions reexport;
4
5 Collection[Instance] instancesInScope( RecordHandler this ):
6   parentHandler() != null ? instances.union( parentHandler().instancesInScope() ) : instances;
7
8 parentHandler( RecordHandler this ):
9   eContainer != null ? ( RecordHandler.isInstance(eContainer) ? ((RecordHandler)eContainer) : null ) :
10    null;
11

```

arrayindexxpt

«EXTENSION templates::util»

- notaus
- org::openarchitectureware::util::stdlib::io
- org::openarchitectureware::util::stdlib::naming
- templates::util

openArchitectureWare  
[www.openarchitectureware.org](http://www.openarchitectureware.org)

eclipse  
**modeling**  
PROJECT

**One Stop Toolkit for DSLs + X**

Version 5.0 is current

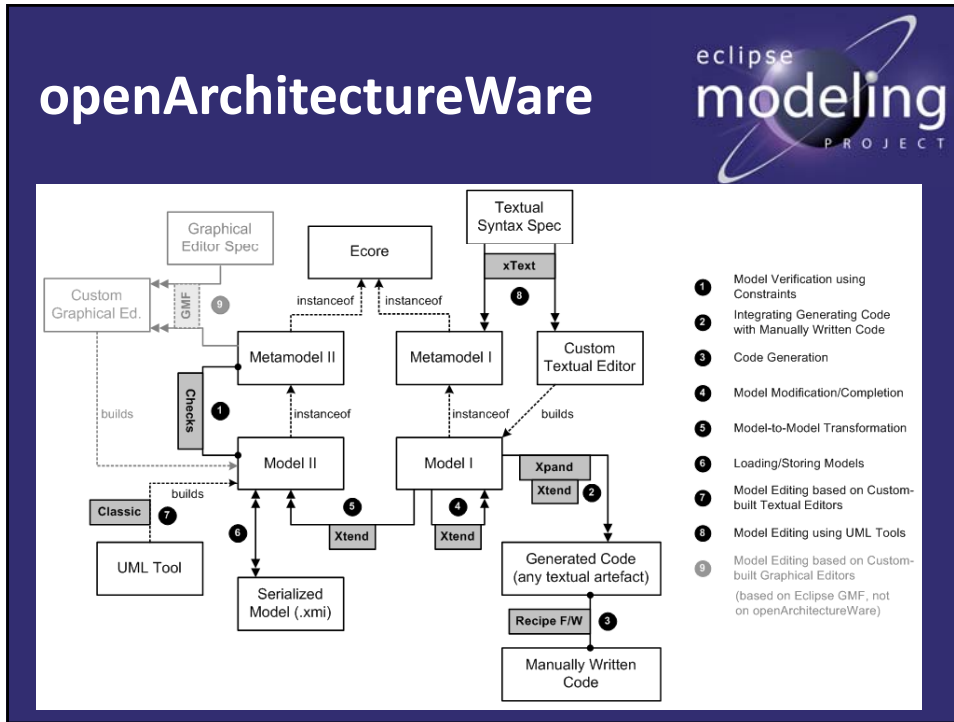
Lively ecosystem of tools and extensions

Proven track record in various domains & project contexts

Stable, productive and helpful developer, support and user communities

Integration with Eclipse:

- Part of Eclipse, Working Group
- Uses EMF as a basis
- Graphical editors based on GMF
- All editors and tooling based on Eclipse



# Specify Grammar

```

exdsl.txt
Namespace:
  "namespace" name=ID (featureclause=FeatureClause)? "{"
  (using+Using) *
  { subNamespaces+Namespace |
    components+Component |
    datatypes+DataType |
    interfaces+Interface |
    compositions+Composition } *
  "}";

Using:
  "using" namespace=[Namespace|qualID];

Component:
  (pointcut=Pointcut)? "component" name=ID (tags=TagsClause)? (feature
  (ports+=Port) *
  "}";

Port:
  MessagePort | DataPort;

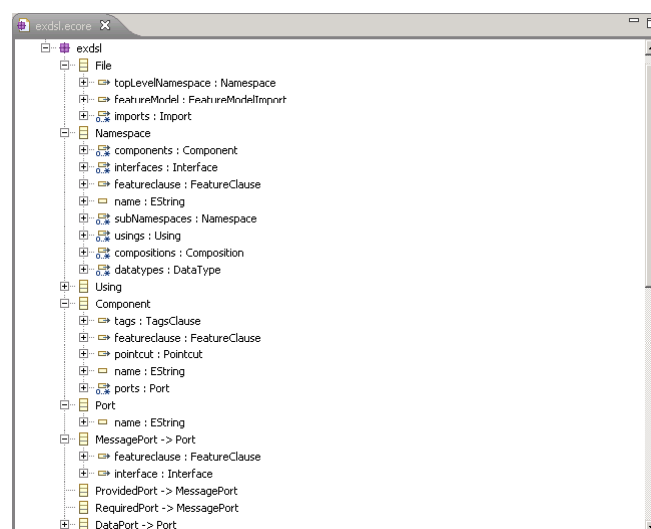
MessagePort:
  ProvidedPort | RequiredPort;

ProvidedPort:
  "provides" name=ID ":" interface=[Interface] (featureclause=FeatureCl

RequiredPort:
  
```

# Antlr Grammar and Parser is generated from this specification

## Generated Metamodel



# Specify Constraints

```

Checks.chk
import exdsl;

extension net::ample::adsl::exdsl::Extensions;
extension org::openarchitectureware::util::stdlib::io;

context Component ERROR "Qualified Name "+qualifiedName()+" must be unique"
  allComponents().select( c | c.qualifiedName() == qualifiedName() ).size == 1;

context DataType ERROR "Qualified Name "+qualifiedName()+" must be unique"
  allDataTypes().select( c | c.qualifiedName() == qualifiedName() ).size == 1;

context Namespace if !isEmpty() ERROR "Qualified Name "+qualifiedName()+" must be unique"
  allNamespaces().select( c | c.qualifiedName() == qualifiedName() ).size == 1;

context emf::EObject if metaType.getProperty("name") != null ERROR "name not defined"
  metaType.getProperty("name").get(this) != "Unnamed";

context Interface ERROR "interface names must start with a capital I":
  name.startsWith("I");

context MessagePort ERROR "interface not defined. Missing a 'using'?":
  visibleInstancesOfType(this, Interface).contains(interface);

context Attribute ERROR "no type defined: "+type.name:
  visibleInstancesOfType(this, DataType).contains(type);

context DataPort ERROR "data not defined: "+type.name:
  visibleInstancesOfType(this, ComplexType).contains(type);

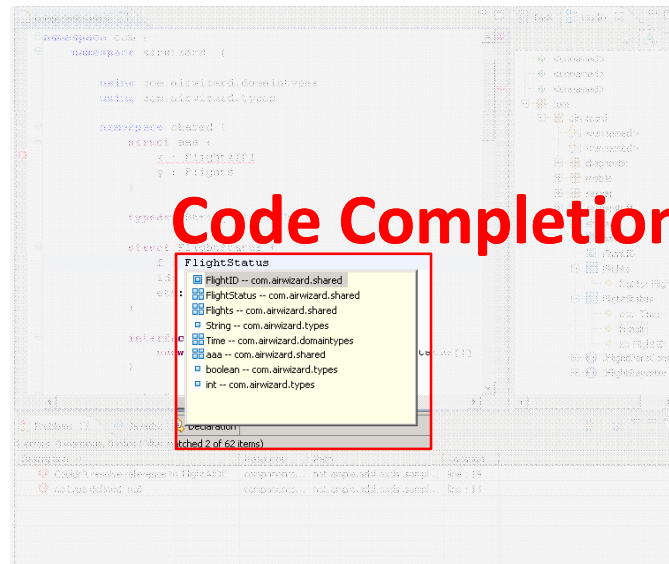
```

# Generated Editor

The screenshot displays the 'Generated Editor' interface. The main window shows XSDL code for a namespace 'com' with sub-namespaces 'airwizard' and 'shared'. The code defines a 'FlightStatus' struct, a 'FlightID' typedef, and an 'IFlightReporter' interface. A tooltip is visible over the 'FlightID' typedef, showing its definition and associated types. The right-hand side of the editor shows a tree view of the project structure, including folders for 'airwizard', 'shared', and 'FlightID'. At the bottom, the 'Problems' window is open, displaying two errors: 'Couldn't resolve reference to 'FlightAIDT'' and 'no type defined: null', both located at line 14 of the file.

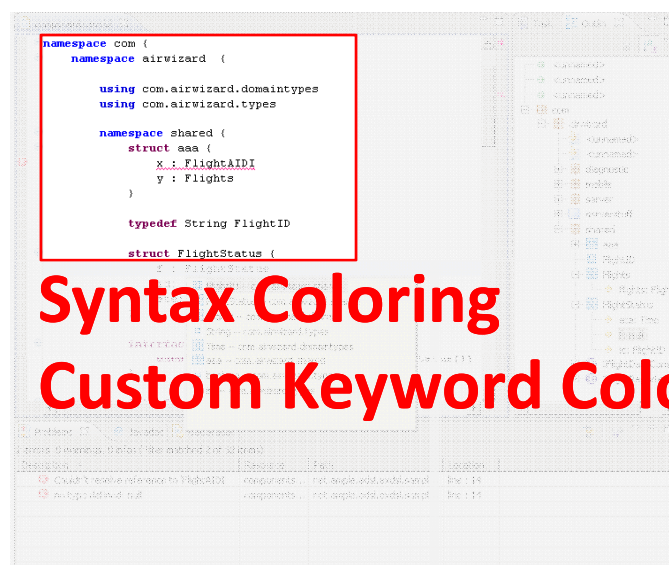
Description	Resource	Path	Location
Couldn't resolve reference to 'FlightAIDT'	components...	net.ample.adsl.exdsl.samp...	line : 14
no type defined: null	components...	net.ample.adsl.exdsl.samp...	line : 14

# Generated Editor



**Code Completion**

# Generated Editor



**Syntax Coloring**  
**Custom Keyword Coloring**

# Generated Editor

**Realtime Constraint Validation**

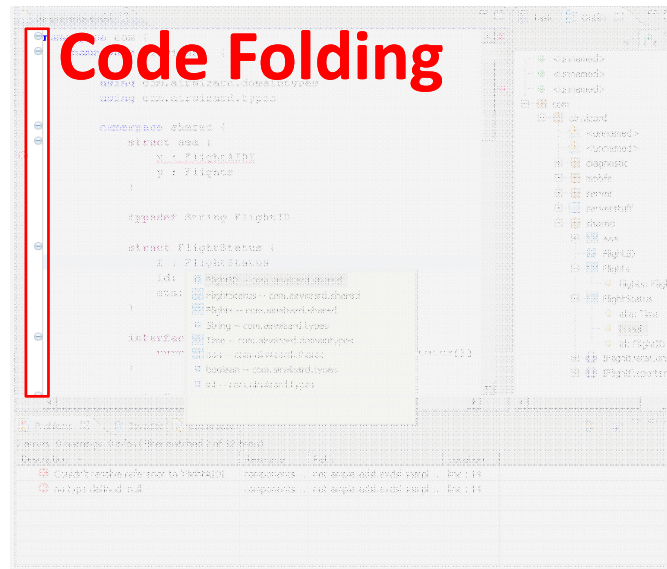
2 errors, 0 warnings, 0 infos (Filter matched 2 of 62 items)			
Description	Resource	Path	Location
Couldn't resolve reference to "FlightAID1"	components...	net:ampl:exdl:sampl...	line : 14
no type defined: null	components...	net:ampl:exdl:sampl...	line : 14

# Generated Editor

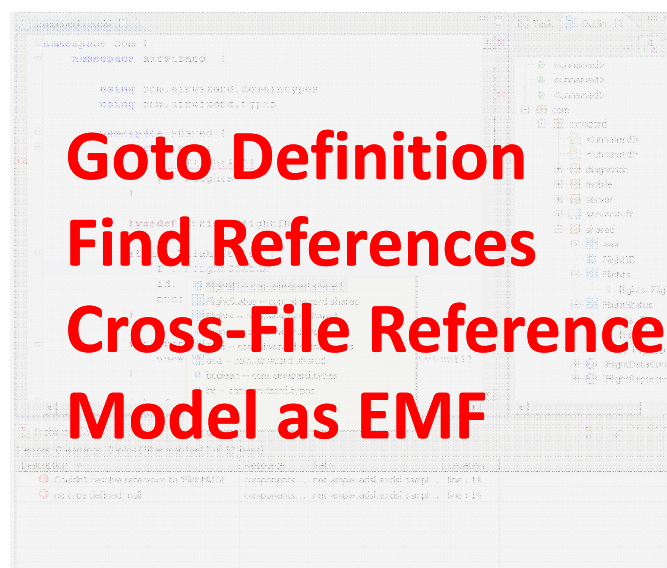
**Customizable Outlines**

- <unnamed>
- <unnamed>
- <unnamed>
- com
  - airwizard
    - <unnamed>
    - <unnamed>
  - diagnostic
  - mobile
  - server
  - serverstuff
  - shared
    - aaa
    - FlightID
    - Flights
      - Flights: Flight
      - FlightStatus
        - eta: Time
        - fn: null
      - id: FlightID
      - IFlightDataConsu
      - IFlightReporter

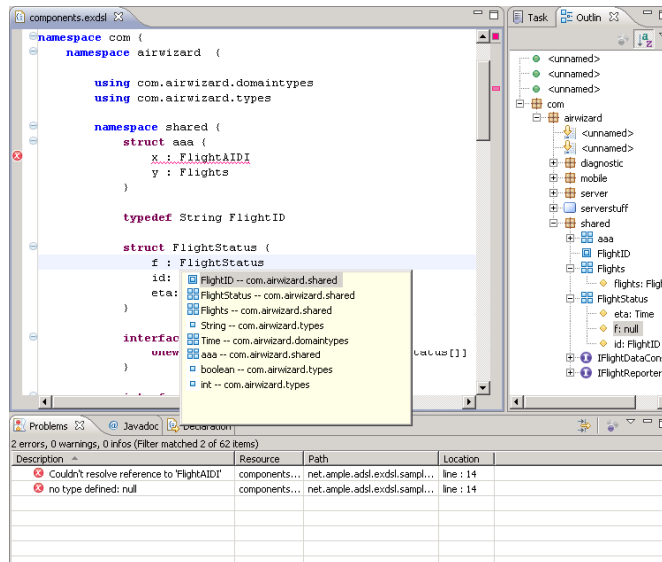
# Generated Editor



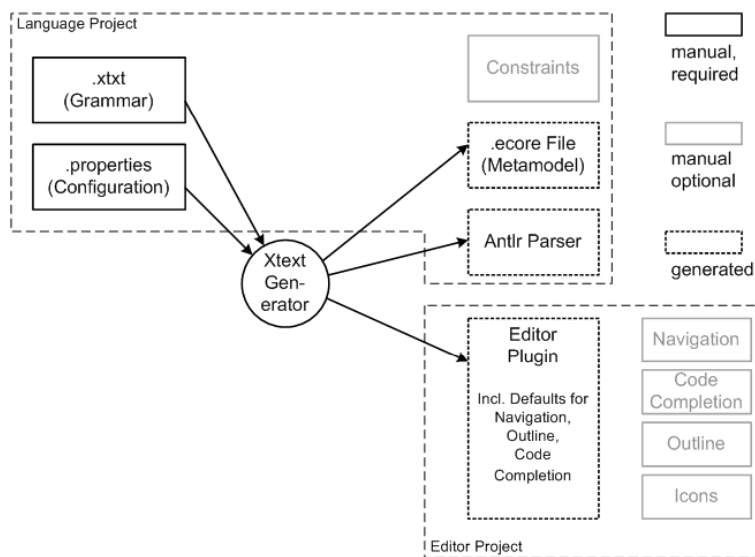
# Generated Editor



# Generated Editor



# Xtext Overview





# DEMO



Building a sample textual DSL  
and code generator for a  
simple domain using Eclipse  
TMF/openArchitectureWare

## Another Tool...?



# Another Tool...?



MS  
Meta Programming System





JetBRAINS

also do...

# IntelliJ IDEA

# Resharper



released in  
**Q3 2009**

licensed under  
**Apache 2.0**

Build new **standalone** DSLs  
Build DSLs that **reuse** parts  
of other languages

(MPS comes with **BaseLanguage**<sup>Java++</sup>)

**extend** base language  
build DSLs that **reuse** parts  
of **BaseLanguage**

# Language Extension Example

## Old

```
ReadWriteLock l = ...
l.readLock().lock();
try {
    //code
} finally {
    l.readLock().unlock();
}
```

## New

```
ReadWriteLock l = ...
lock (l) {
    //code
}
```

## Structure ♦ Editor ♦ Typesystem ♦ Generator

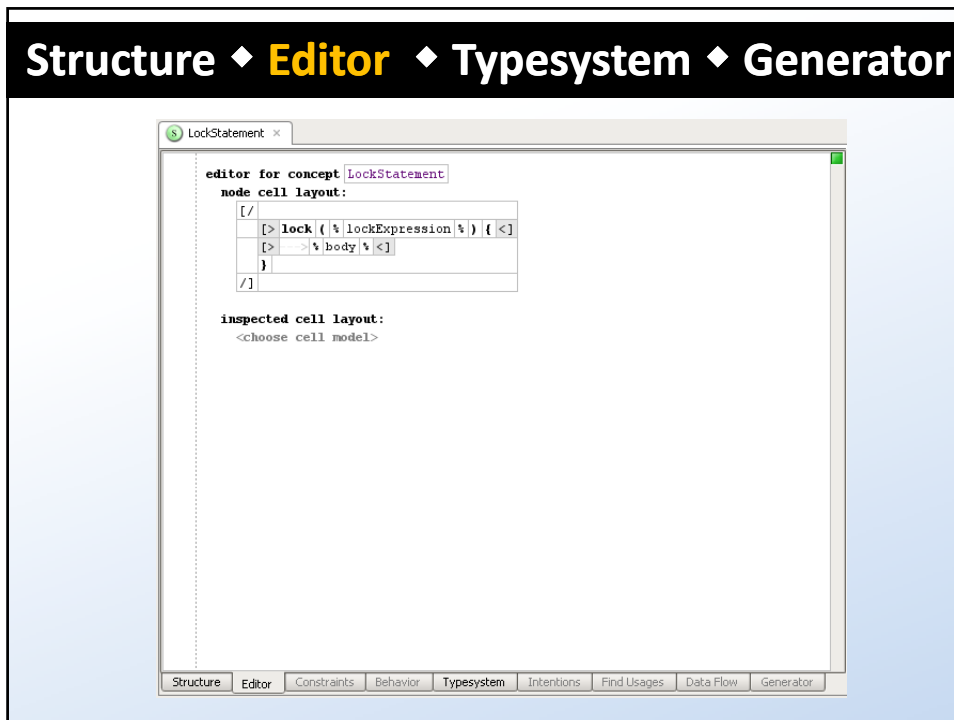
The screenshot shows the Structure view for a concept named `LockStatement`. The view is organized into several sections:

- concept LockStatement** extends `Statement` implements `<none>`
- instance can be root:** false
- properties:** << ... >>
- children:**

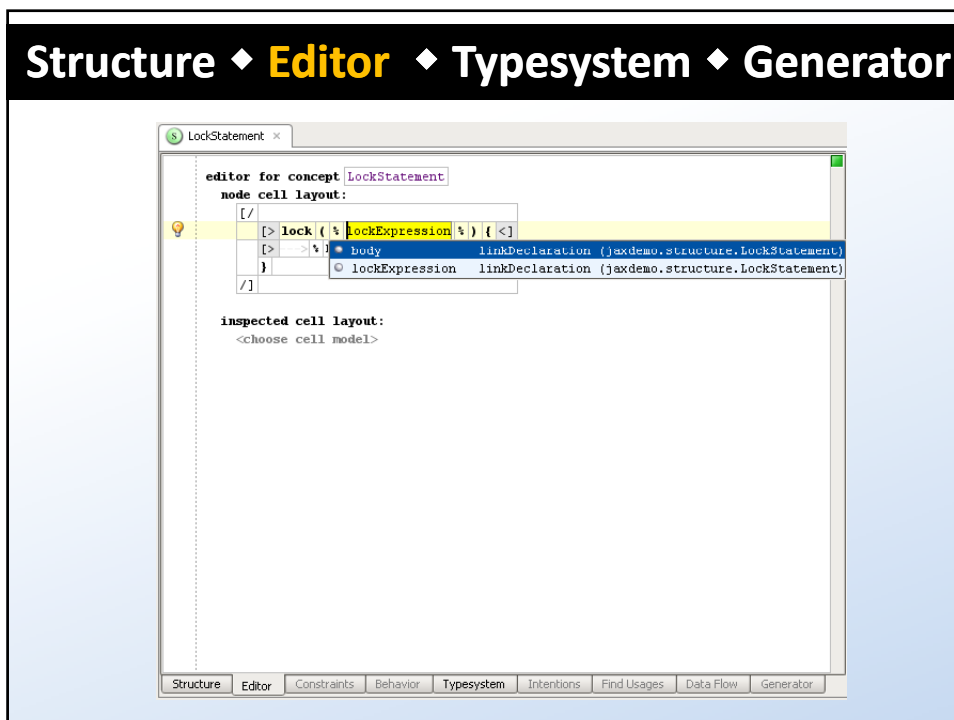
<code>StatementList</code>	body	1	specializes:	<none>
<code>Expression</code>	lockExpression	1	specializes:	<none>
- references:** << ... >>
- concept properties:**
  - `alias = lock`
- concept links:** << ... >>
- concept property declarations:** << ... >>
- concept link declarations:** << ... >>

The bottom of the window shows a tabbed interface with the following tabs: Structure, Editor, Constraints, Behavior, Typesystem, Intentions, Find Usages, Data Flow, Generator.

## Structure ♦ Editor ♦ Typesystem ♦ Generator



## Structure ♦ Editor ♦ Typesystem ♦ Generator



## Structure ♦ Editor ♦ Typesystem ♦ Generator

The screenshot shows the Structure Editor with a rule definition for `typeof_LockStatement`. The rule is defined as follows:

```
rule typeof_LockStatement {
  applicable for concept = LockStatement as lockStatement
  overrides false
  child type restrictions << ... >>

  do {
    typeof(lockStatement.lockExpression) :<<=: Lock;
  }
}
```

The `Lock` type is highlighted, and the Inspector window shows the following information:

```
Inspector
jetbrains.mps.baseLanguage.structure.ClassifierType
[quotedNode] ClassifierType <no name>[1242291552632] in jaxdemo.typesystem
```

The Inspector also shows the following information:

```
Lock ^interface (j.u.concurrent.locks@j)
LockSupport ^class (j.u.concurrent.locks@j)
```

## Structure ♦ Editor ♦ Typesystem ♦ Generator

The screenshot shows the Structure Editor with a list of rules and a reduction rule for `LockStatement`. The rules are:

- conditional root rules:** << ... >>
- mapping rules:** << ... >>
- weaving rules:** << ... >>
- reduction rules:**

```
concept LockStatement --> reduce_LockStatement
inheritors false
condition <always>
```
- abandon roots:** << ... >>
- pre-processing scripts:** << ... >>

## Structure ♦ Editor ♦ Typesystem ♦ Generator

```

content node:
public class someclass extends <none> implements <none> {
  <<static fields>>

  <<static initializer>>
  <<fields>>
  <<properties>>
  <<initializer>>
  public someclass() {
    <no statements>
  }

  public void somemethod() {
    Lock l = null;
    <TF> try { </TF>
      $COPY_SRC[1].lock();
      $LOOP[$COPY_SRC[null:]]
    } finally {
      $COPY_SRC[1].unlock();
    }
  }

  <<static methods>>
  <<static inner classifiers>>
}

```

## Structure ♦ Editor ♦ Typesystem ♦ Generator

```

content node:
public class someclass extends <none> implements <none> {
  <<static fields>>

  <<static initializer>>
  <<fields>>
  <<properties>>
  <<initializer>>
  public someclass() {
    <no statements>
  }

  public void somemethod() {
    Lock l = null;
    <TF> try { </TF>
      $COPY_SRC[1].lock();
      $LOOP[$COPY_SRC[null:]]
    } finally {
      $COPY_SRC[1].unlock();
    }
  }

  <<static methods>>
  <<static inner class>>
}

```

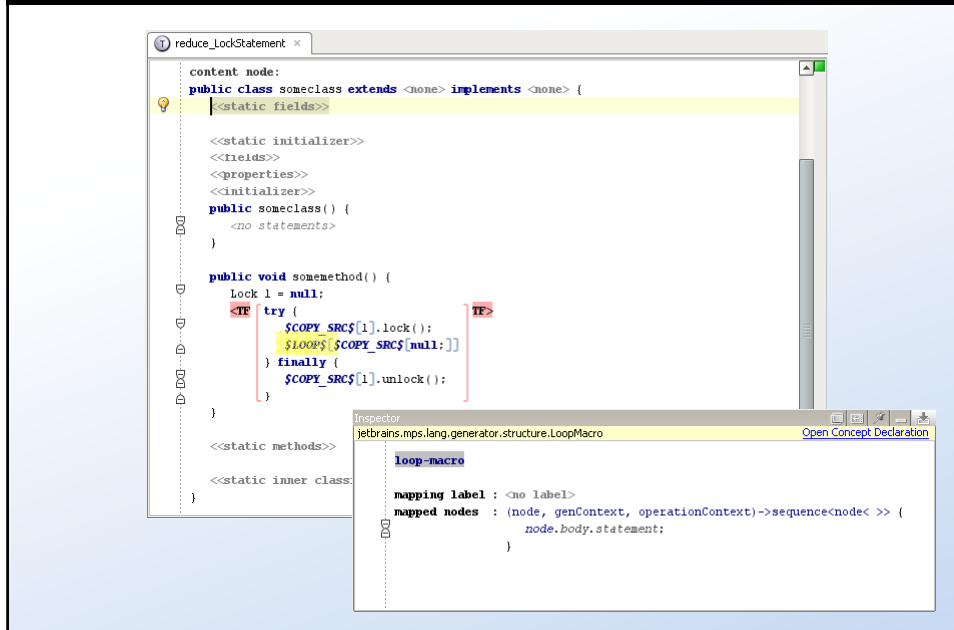
```

Inspector
jetbrains.mps.lang.generator.structure.CopySrcNodeMacro
Open Concept Declaration

copy/reduce node macro
mapping label : <no label>
mapped node : (node, genContext, operationContext)->node< > {
  node.lockExpression;
}

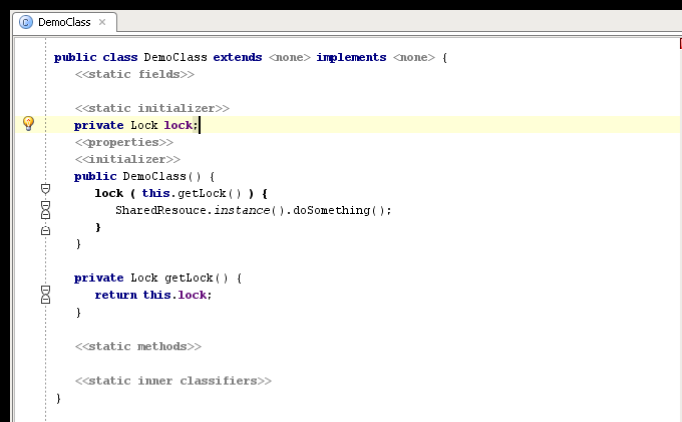
```

## Structure ♦ Editor ♦ Typesystem ♦ Generator



## Language Extension Example

**Result behaves like a native  
base language construct**





# Language Extension Example

Result behaves like a native  
base language construct

```

public class DemoClass extends <none> implements <none> {
  <<static fields>>

  <<static initializer>>
  private Lock lock;
  <<properties>>
  <<initializer>>
  public DemoClass() {
    lock ( 1 ) {
      SharedResource.instance().doSomething();
    }
  }

  private Lock getLock() {
    return this.lock;
  }

  <<static methods>>

  <<static inner classifiers>>
}

```

# Language Extension Example

Translated to regular Java code  
based on the generator

```

package jaxdemo.sandbox.sandbox;
import java.util.concurrent.locks.Lock;

public class DemoClass {
  private Lock lock;

  public DemoClass() {
    try {
      this.getLock().lock();
    } finally {
      this.getLock().unlock();
    }
  }

  private Lock getLock() { return this.lock; }
}

```

# Example Languages

## UI Language

```

component NewLanguageDialog
  IDEDialog
    left: 100 top: 100 width: 600 height: 200
    title = New Language

    content pane:
      Grid
        Row
          [Label
            text : "Language Namespace:"]
        Row
          [TextField ( name )
            text : { this.LanguageNamespace } ]
        Row
          [Label
            text : "Language Path:"]
        Row
          [PathField ( path )
            path : { this.LanguagePath } ]

      buttons
        [text OK
          default true
          handler this.onOk() ] [text Cancel
          default false
          handler this.onCancel() ]

```

# Example Languages

## HTML Templates

```

html template MessageHeader(Message message, boolean hideUser, boolean userLast)

is not refreshable: Only one root element allowed for refreshable template.

<< variables >>

rss links: << rss links >> << rss template uris>>

<< explicit js import >>

if (!(hideUser) && !(userLast)) {
  <span class: right-margin-span>
    [html include <visible>
      [ def[]: UserLink(user: message.getUser(), bold: true) ]
    ]
  </span>
}
<span class: right-margin-span>
  [html include <visible>
    [ def[]: Date(date: message.created) ]
  ]
</span>
if (null != message.updated) {
  <span class: right-margin-span>
    updated:
    [html include <visible>
      [ def[]: Date(date: message.updated) ]
    ]
  </span>
}

```

# Example Languages

## Persistent Classes

```

public persistent class Forum extends <none> implements <none> {

    features
    save changes history if: false
    save changes history callback: no callback
    version mismatch resolution: default
    invariant: no invariant

    << static fields >>

    public simple string name opts;
    public unordered child FThread[0..n] threads opts;
    public unordered bidirectional association User[0..n] subscribers onDelete(clear);
    public unordered bidirectional association User[0..n] watchers onDelete(clear);

    public Forum(string name, User creator) {
        this.name = name;
        this.watchers.add(creator);
    }

    << destructor >>

```

Variability      MDD Tooling  
 Configuration      Model Variability  
 Customization      Transformation Var.  
 MDD Intro      Summary & Wrapup



# Two Levels

~ problem space  
vs. software space

Problem Space:

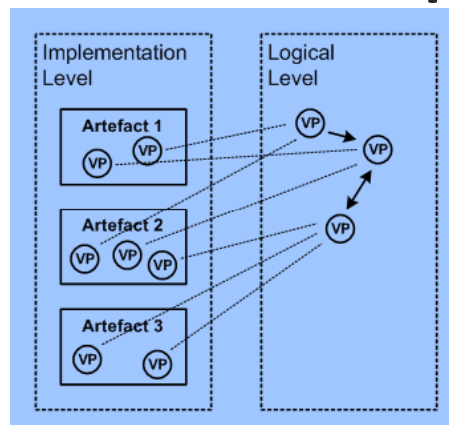
**Configuration**

Software Space:

**Customization**

# Two Levels

~ problem space  
vs. software space



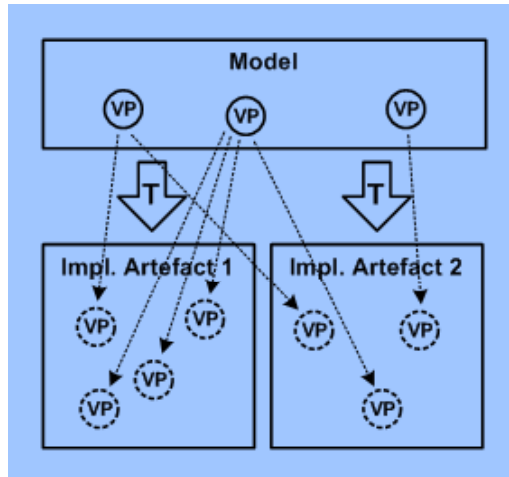
## Two Levels Removal

```
#if defined (ACE_HAS_TLI)
    static ssize_t t_snd_n (
        ACE_HANDLE handle,
        const void *buf,
        size_t len,
        int flags,
        ACE_Time_Value *timeout = 0,
        size_t *bytes_transferred = 0);
#endif /* ACE_HAS_TLI */
```

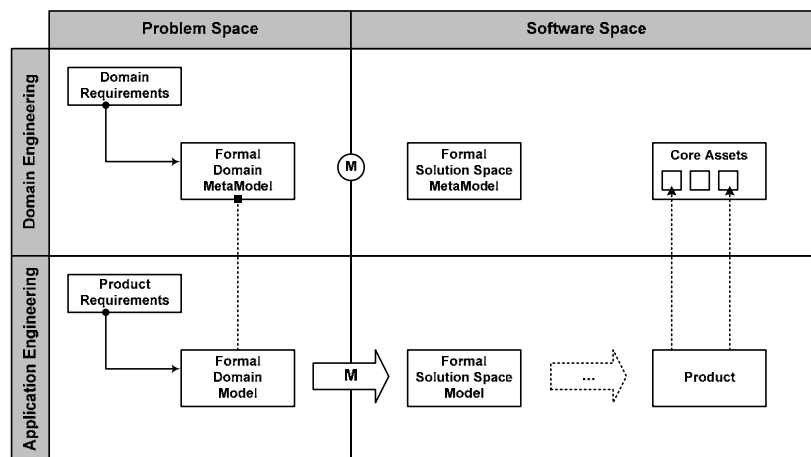
## Model-Based Implementation

- ... customization in  
problem space
- ... Problem-Space DSL

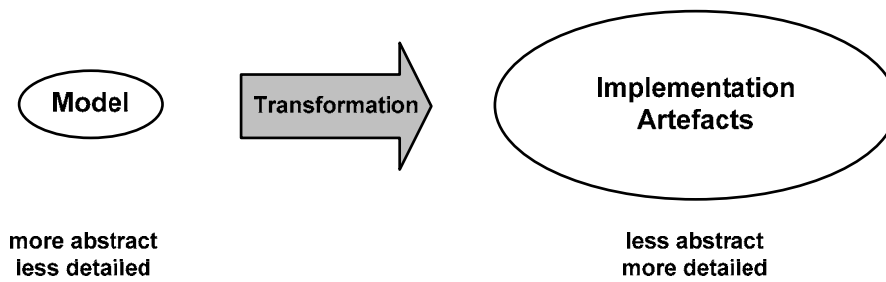
# Model-Based Implementation



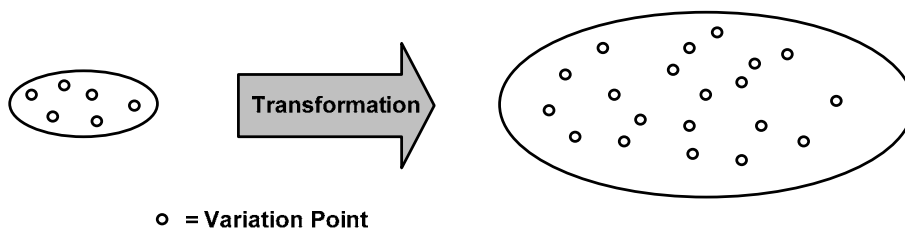
# Model-Based Implementation



## MDSD - Thumbnail

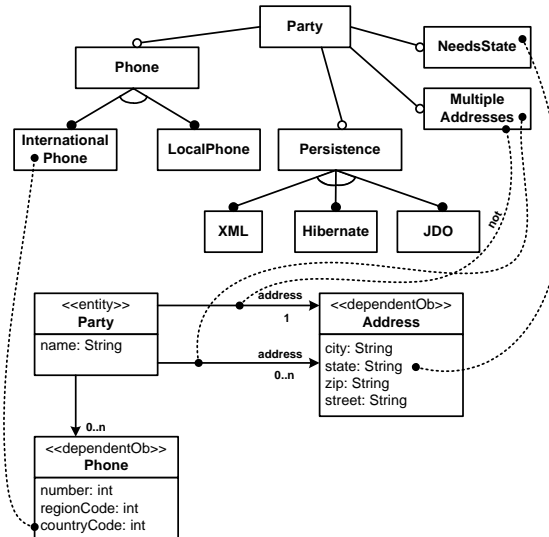


## MD-PLE - Thumbnail



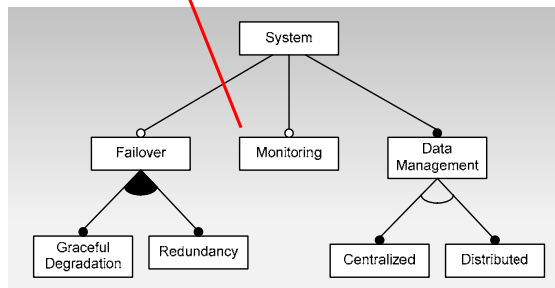
**fewer!**

# Two Levels Removal



# Two Levels Removal

```
component DelayCalculator {
  provides default: IDelayCalculator
  requires screens[0..n]: IInfoScreen
  provides mon: IMonitoring feature monitoring
}
```





# Two Levels Removal

```

namespace monitoringStuff feature monitoring {

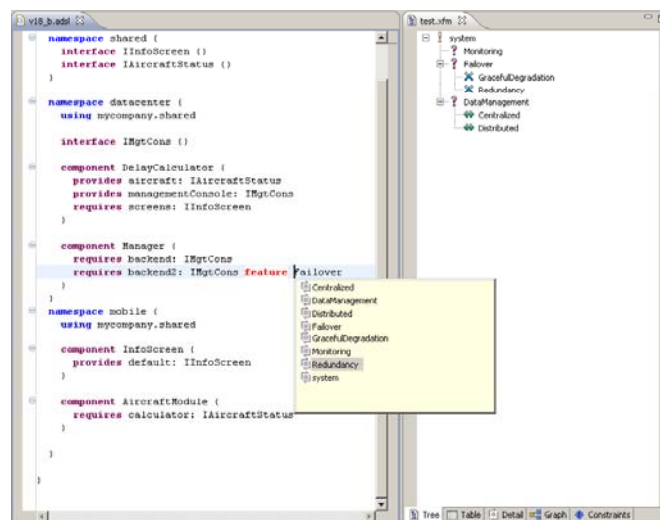
  component MonitoringConsole {
    requires devices:[*]: IMonitor
  }

  instance monitor: MonitoringConsole

  dynamic connect monitor.devices query {
    type = IMonitor
  }
}

```

# Two Levels Removal



# Two Levels Injection

```
namespace monitoring {
    component MonitoringConsole ...
    instance monitor: ...
    dynamic connect monitor.devices ...

    aspect (*) component {
        provides mon: IMonitoring
    }
}
```

# Two Levels Injection

```
component DelayCalculator {
    ...
}
component AircraftModule {
    ...
}
component InfoScreen {
    ...
}
```

```
aspect (*) component {
    provides mon: IMonitoring
}
```

```
component DelayCalculator {
    ...
    provides mon: IMonitoring
}
component AircraftModule {
    ...
    provides mon: IMonitoring
}
component InfoScreen {
    ...
    provides mon: IMonitoring
}
```

# Two Levels <sup>+</sup> Removal Injection

```
namespace monitoring feature monitoring {
    component MonitoringConsole ...
    instance monitor: ...
    dynamic connect monitor.devices ...

    aspect (*) component {
        provides mon: IMonitoring
    }
}
```

# Manual Code Variability

```
public class LightDriverImplementation extends LightDriverImplBase {
    @Override
    protected String getIdInternal() {
        return getConfigParamValueForId();
    }
    ...
    //# dimmableLights
    @Override
    protected int setLightLevelInternal(int level) {
        state().setEffectiveLightLevel(level);
        return level;
    }
    //~# dimmableLights
}
```

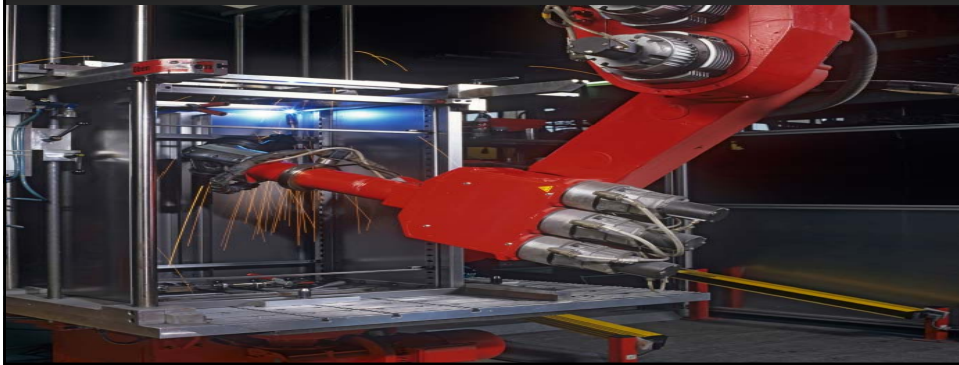


# DEMO

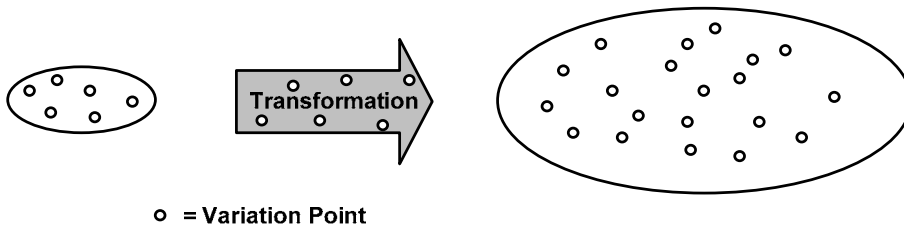


Extending the sample DSL to include feature-based variability connected to an external feature model

Variability MDD Tooling  
Configuration Model Variability  
Customization Transformation Var.  
MDD Intro Summary & Wrapup



## MD-PLE – Thumbnail II



**more options**

# Transformation Variability

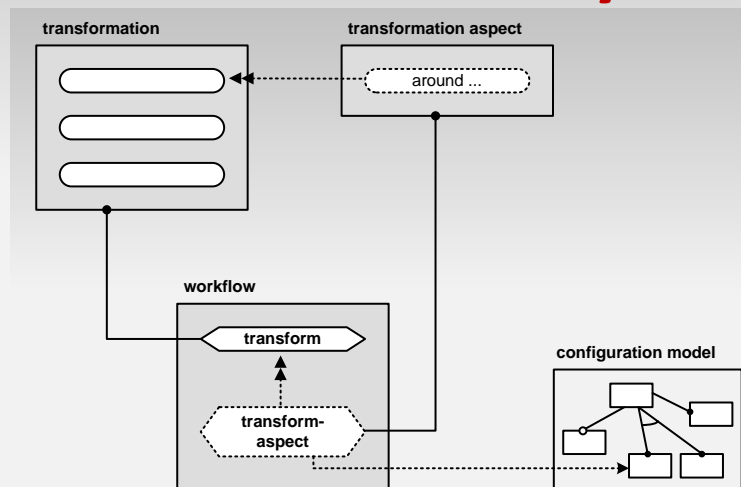
```

create System transformPs2Cbd( Building building ):
...
  hasFeature( "burglarAlarm" ) ? ( handleBurglarAlarm() -> this ) : this;

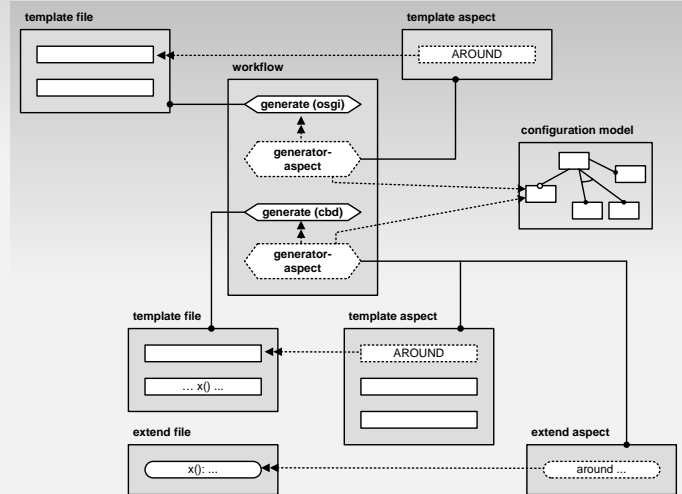
handleBurglarAlarm( System this ):
  let conf = createBurglarConfig(): (
    configurations.add( conf ) ->
    ...
    conf.connectors.add( connectSimToPanel( createSimulatorInstance(),
      createControlPanelInstance() ) ) ->
    hasFeature( "siren" ) ? conf.addAlarmDevice("AlarmSiren") : null ->
    hasFeature( "bell" ) ? conf.addAlarmDevice("AlarmBell") : null ->
    hasFeature( "light" ) ? conf.addAlarmDevice("AlarmLight") : null
  );

```

# Transformation Variability



# Generator Variability



# DEMO



Introducing Variability into the  
Code Generator built before

Variability      MDD Tooling  
Configuration    Model Variability  
Customization    Transformation Var.  
MDD Intro        **Summary & Wrapup**



**DSLs can be used to effectively  
describe customization variab.**

**Transformation and Generation  
can be used to map PS to SS**

**Configuration and Customization  
can be sensibly combined**

**Various Tools are available,**

<http://eclipse.org/modeling>

<http://dslvariantmanagement.googlecode.com/>



Using  
**Domain Specific Languages**  
for  
**Product Line Engineering**

Variability MDD Tooling  
Configuration Mod. Variability  
Customization Transformation Var.  
MDD Intro Summary & Wrapup

**THE END.**  
**Thank you.**  
**Questions?**

Markus Voelter  
Independent/Itemis  
[www.voelter.de](http://www.voelter.de)  
[voelter@acm.org](mailto:voelter@acm.org)  
**itemis**

