Thoughts on the Collaboration between Software Engineers and Domain Experts

Markus Völter
Software Engineers

Domain Experts
1 History and State of the Art
2 Language Workbenches
3 JetBrains MPS
4 Example Systems
5 Challenges for Education
6 Summary
History & State of the Art
Every person who does his thinking with symbolized concepts, whether in the form of the English language, pictographs, formal logic or mathematics should be able to benefit significantly.
The Mother of All Demos

Dec. 9, 1968
Every person who does his thinking with symbolized concepts, whether in the form of the English language, pictographs, formal logic or mathematics should be able to benefit significantly.
symbolized concepts

==

language
Every person who does his thinking with symbolized concepts, whether in the form of the English language, pictographs, formal logic or mathematics should be able to benefit significantly.
diverse notations
Every person who does his thinking with symbolized concepts, whether in the form of the English language, pictographs, formal logic or mathematics should be able to benefit significantly.
thinking [writing, analyzing, running...]
thinking [writing, analyzing, running...]
thinking [writing, analyzing, running...]

```
Figure Palette
New Subplots
- 2D Axes
- 3D Axes

Variables
- ans: 1x1
- fh: 1x1
- x: 1x41
- X: 41x41
- y: 1x41
- Y: 41x41
- Z: 41x41

Annotations
- Line
- Arrow

Property Editor - Text
- Line Style: -
- Line Width: 0.5
- Edge Color: 
- Font: Helvetica
- Background: 

Plot Browser
- Normal Response
```

Figure: Surface Contour Plot
- Title: Normal Response
- Axes: X, Y, Z
- Data: ans, fh, x, X, y, Y, Z
- Colors: Blue, Yellow, Green
- Grid: X, Y, Z

[Image of Surface Contour Plot]
thinking [writing, analyzing, running...]
Every person who does his thinking with symbolized concepts, whether in the form of the English language, pictographs, formal logic or mathematics should be able to benefit significantly.
Every person who does his thinking with symbolized concepts, whether in the form of the English language, pictographs, formal logic or mathematics, should be able to benefit significantly.

engineers
scientists
medical doctors
finance experts
insurance experts
lawyers
if, in any incoming or outgoing messages of type Email, the message has keyword Financial and the message contains special Credit Card # and the recipients in department Treasury and the message contains all 

then

send a notification to compliance_officer@ja and archive in the mail store.
Every person who does his thinking with symbolized concepts, whether in the form of the English language, pictographs, formal logic or mathematics should be able to benefit significantly.
Every person who does his thinking with symbolized concepts, whether in the form of the English language, pictographs, formal logic or mathematics should be able to benefit significantly.
2

Language

Workbenches
Every person who does his thinking with symbolized concepts, whether in the form of the English language, pictographs, formal logic or mathematics should be able to benefit significantly.
Expressivity for Core Domain Knowledge
User-Friendly Notation
Testing
Meaningful Analyses
Synthesis of Software

Symbolized Concepts, English language, pictographs, formal logic or mathematics
„thinking“
benefit
An old idea from the 1970s.

BUT...
Language Workbench

(Martin Fowler)
Language Workbench

(Martin Fowler)

Freely define languages and integrate them
Freely define languages and integrate them + more detailed criteria I will omit here
Language Workbench
(Martin Fowler)

powerful editing testing refactoring debugging teamwork

language definition implies IDE definition
Open Source
Apache 2.0
http://jetbrains.com/mps
+ Refactorings, Find Usages, Syntax Coloring, Debugging, ...
Projectional Editing
[Projectional Editing]

Parsing

Projectional Editing

Concrete Syntax

Abstract Syntax Tree

Concrete Syntax

Abstract Syntax Tree
Projectional Editing
Syntactic Flexibility

Regular Code/Text

Mathematical

Tables

Graphical
**[Projectional Editing]**

**Syntactic Flexibility**

**Regular Code/Text**

```c
// A documentation comment with references
void aSummingFunction(int8[] data, int8 dataLen) {
    int16 sum;
    for (int8 i = 0; i < dataLen; i++) {
        sum += data[i];
    }
} aSummingFunction (function)
```

**Mathematical**

```latex
\[
\text{double midnight2}(\text{int32} \ a, \ \text{int32} \ b, \ \text{int32} \ c) = \frac{-b + \sqrt{b^2 - 4 \sum_{i=1}^{a} a \cdot c}}{2 \cdot a};
\]
```

**Tables**

```
int16 decide(int8 spd, int8 alt) {
    return spd > 0 spd > 100 otherwise 0;
    alt < 0  1   1
    alt == 0 10  20
    alt > 0  30  40
    alt > 100 50  60
} decide (function)
```

**Graphical**

[Diagram showing a decision process with nodes labeled `Cst.Customer`, `Contract`, `Tariff`, and `Customer attributes`.

- cust 1
- starts: date
- ends: date
- trf 1
- attributes]
[Projectional Editing] Language Composition

- Separate Files
  - Type System
  - Transformation
  - Constraints
- In One File
  - Type System
  - Transformation
  - Constraints
  - Syntax
  - IDE

50+ extensions to C
10+ extensions to requirements lang.
Example Systems
Insurance Configuration 1
1 Overview

1.1 Description


1.2 Selling Period and Holder

This product can be sold from 9/9/9 until 9/9/9. The holder of the product can be a Person. Specifying the beneficiary is optional.

1.3 Covers

This product includes the following covers:

- Financial cover
Life Product

Insurance Configuration Tool

1 Overview

1.1 Description


1.2 Selling Period and Holder

This product can be sold from 9/8/9 until 9/9/9.
The holder of the product can be a Person.
Specifying the beneficiary is optional.

1.3 Covers

This product includes the following covers:
Financial cover
Business Application

Insurance Configuration Tool

Example product
- Financial cover
  - Premium waiver
  - Total permanent disability

Example product

- Premium waiver
- Total permanent disability
- Death in Accident
- Natural Death
- Death by Terrorist Attack
Business Application
Insurance Configuration Tool

Formula Library  FormulaLibrary01 for Rule Set Type IEEE_RST

numeric formulaOne = 3 < 4  \[12 + 7 + \text{CATV} - \sum_{\text{in} = 1}^{6} (\text{in} + 9 + \sum_{\text{out} = 1}^{\text{in}} (\text{in} + \text{out}))\]

Testsuite
Show Test Suite

numeric dummy = 3 > 2  \[\sum_{\text{index} = 1}^{6} (\text{index} + 1)\]

Testsuite
Show Test Suite
[Business Application]

Insurance Configuration Tool

Formula Library FormulaLibrary01 for Rule Set Type IEEE_RST

numeric formulaOne = 3 < 4  
12 + 7 + CATV - \sum_{in = 1}^{6} (in + 9 + \sum_{out = 1}^{in} (in + out))

Testsuite
Show Test Suite  Run All  Clear Evaluation

Input Variables  Output Variables
<no name>  CATV  result
0  Expected 9  Run  Run and Debug

<<Test Case>>

numeric dummy = 3 > 2  
\sum_{index = 1}^{6} (index + 1)

Testsuite
Show Test Suite  Run All  Clear Evaluation

test
Input Variables  Output Variables
result  0  Run  Run and Debug

Expected 0  
Result 0

<<Test Case>>
Rule Set Type DemoRuleSetType

Business objects

person : Person

Variables:  Parent

PRMI : int  <no parent>
FR : int
NN : int
TT : int
J : int
A3 : int
G3 : int
ANUI : int
X : int
**[Business Application]**

**Insurance Configuration Tool**

---

**Rule Set Type** DemoRuleSetType

---

**Business objects**

person : Person

---

**Variables:**

- PRMI : int
- FR : int
- NN : int
- TT : int
- J : int
- A3 : int
- G3 : int
- ANUI : int
- X : int

**Parent**

<no parent>

---

**Libraries**

- Standard
- Extra

---

**Rule Set Type** DemoRuleSetType

---

**Business objects**

<no business objects>

---

**Variables:**

<no variables>

---

**Parent**

<no parent>

---

**Libraries**

<no libraries>
rule set DemoRuleSet2 is of type DemoRuleSetType

EU0 : int [ save false print false ]
CATEG : string [ save false print false ]
CATEG1 : double [ save true print true ]

PREMIO = [ A1 > 10 => EU0
  <always> => FLAG
]

FLAG = [ CATEG1 equals 60 or CATEG1 equals 63 or CATEG1 equals 64 => 160
  PREMIO equals 0 => 162
  CATEG1 > 0 or substr(inga[4], 1, 1) equals "V" => 163
  <always> => PREMIO + FLAG
]

PREMIO = [ <always> => round(PREMIO * (1 + factacer), 0) ]
== Insurance Specialist
== Insurance Mathematician
3.6 Calculating the value of a life/ 2 lives

\[ D_x = v^x \frac{x}{100} \]  

Implemented in \( v^{2.0} \).

\[ \omega - x \]

\[ N_x = \sum_{t=0}^{l} D_{x+t} \]  

3.6 Contante waarde 1 leven/ 2 levens

\[ E_x = \frac{x+n}{x} \]  

19 Dec (4)

\[ a_x = \bar{a} - 1 \]  

21 Dec (3)

\[ \bar{a}_x = \bar{a} - 0.5 \]  

22 Dec (3)

\[ \bar{a}_x = \bar{a} - 0.5 + 0.5 \cdot E_x \]  

25 Dec (3)

4 BN(_ris) koopsommen
## Elements...

### Rules

- **Rule Bereken Mutatieperiode**
  - **Result:** Mutatieperiode
  - **Name:** Bereken Mutatieperiode
  - **Documentation:**
    - Het vaststellen van de periode tussen de huidige en de vorige mutatie in dagen.
    - De mutatieperiode kan niet meer dan 360 dagen bedragen omdat elk jaar een begin- en eindmutatie kent i.v.m. het openen en sluiten van het verslagjaar.
    - Dit wordt niet afgevangen omdat het uitvoeren van de begin- en eindmutatie verantwoordelijkheid zijn van de pensioenadministratie.
  - **Tags:** Basisberekening
  - **Algorithm:**
    
    \[
    \text{if } \text{maximum(Mutaties per datum) == 1 then daysof(duration(valid(Mutaties per datum))) else 0}
    \]
  - **Test cases:**

<table>
<thead>
<tr>
<th>Name</th>
<th>Valid time</th>
<th>Transaction time</th>
<th>Fixture</th>
<th>Product</th>
<th>Element</th>
<th>Expected value</th>
<th>Actual value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gelijke datums</td>
<td>03/01/2008</td>
<td></td>
<td>Mutatieperiode - Mutatiedatum = Mutatiedatum Vorig</td>
<td>3</td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Periode &lt; 30</td>
<td>03/01/2008</td>
<td></td>
<td>Mutatieperiode - Mutatiedatum &gt; Mutatiedatum Vorig (binnen 1 maand)</td>
<td>15</td>
<td></td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>Periode &gt; 30</td>
<td>03/01/2008</td>
<td></td>
<td>Mutatieperiode - Mutatiedatum &gt; Mutatiedatum Vorig (meerdere maanden)</td>
<td>60</td>
<td></td>
<td>60</td>
<td>60</td>
</tr>
</tbody>
</table>
Tax/Benefits
bloedverwanten: lijst van Burgers zijn gedefinieerd als {
    Een bloedverwant is een Burger die
    bloedverwant in rechte lijn is of die
    bloedverwant in tweede graad zijlijn is
    Einde declaratie
}

bloedverwanten in rechte lijn: lijst van Burgers zijn gedefinieerd als {
    Een bloedverwant in rechte lijn is een Burger die
    nakomeling is of die
    voorouder is
    Einde declaratie
}

bloedverwanten in tweede graad zijlijn: lijst van Burgers zijn gedefinieerd als {
    Een bloedverwant in tweede graad zijlijn is een ouder.kind met
    ouder.kind ongeveer het actuele voorkomen
    Einde declaratie
    ' dus: broer of zus (incl. erkend kind van ouder)
}

bloed- of aanverwanten in rechte lijn: lijst van Burgers zijn gedefinieerd als {
    Een bloed- of aanverwant in rechte lijn is een Burger die
    bloedverwant in rechte lijn is of die
    aanverwant in rechte lijn is
    Einde declaratie
Telco Demo
== (Imagined) Telecoms Product Manager
proxy for Customer.Customer

core data entity BillingRegion

code [key]: string references:
name: string
baseMinPrice: float
maxRebateFactor: float

entity Contract

starts: date customer: Customer 1 \textarrow{\rightarrow} contracts 0..*
ends: date applicableTariff: Tariff 1

entity Tariff

attributes: references:
[Business Application] Telco Demo App

Contract
- starts: date
- ends: date

Tariff
- applicableTariff: integer

Customer
- customer: integer

BillingRegion
- code [key]: string
- name: string
- baseMinPrice: float
- maxRebateFactor: float
## Telco Demo App

### Core Data  Default Regions for entity Billing Region

<table>
<thead>
<tr>
<th>Code</th>
<th>Name</th>
<th>Base Price</th>
<th>Min Price</th>
<th>Max Price</th>
<th>Rebate Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>BW</td>
<td>Baden Württemberg</td>
<td>0.20</td>
<td></td>
<td>0.8</td>
<td></td>
</tr>
<tr>
<td>BY</td>
<td>Bayern</td>
<td>0.20</td>
<td></td>
<td>0.8</td>
<td></td>
</tr>
<tr>
<td>BE</td>
<td>Berlin</td>
<td>0.15</td>
<td></td>
<td>0.7</td>
<td></td>
</tr>
<tr>
<td>BB</td>
<td>Brandenburg</td>
<td>0.10</td>
<td></td>
<td>0.7</td>
<td></td>
</tr>
<tr>
<td>HB</td>
<td>Bremen</td>
<td>0.20</td>
<td></td>
<td>0.7</td>
<td></td>
</tr>
<tr>
<td>HH</td>
<td>Hamburg</td>
<td>0.15</td>
<td></td>
<td>0.7</td>
<td></td>
</tr>
<tr>
<td>HE</td>
<td>Hessen</td>
<td>0.15</td>
<td></td>
<td>0.7</td>
<td></td>
</tr>
<tr>
<td>MV</td>
<td>Mecklenburg-Vorpommern</td>
<td>0.10</td>
<td></td>
<td>0.7</td>
<td></td>
</tr>
<tr>
<td>NI</td>
<td>Niedersachsen</td>
<td>0.15</td>
<td></td>
<td>0.7</td>
<td></td>
</tr>
<tr>
<td>NW</td>
<td>Nordrhein-Westfalen</td>
<td>0.15</td>
<td></td>
<td>0.7</td>
<td></td>
</tr>
<tr>
<td>RP</td>
<td>Rheinland-Pfalz</td>
<td>0.15</td>
<td></td>
<td>0.7</td>
<td></td>
</tr>
<tr>
<td>SL</td>
<td>Saarland</td>
<td>0.15</td>
<td></td>
<td>0.7</td>
<td></td>
</tr>
<tr>
<td>SN</td>
<td>Sachsen</td>
<td>0.10</td>
<td></td>
<td>0.7</td>
<td></td>
</tr>
<tr>
<td>ST</td>
<td>Sachsen-Anhalt</td>
<td>0.10</td>
<td></td>
<td>0.7</td>
<td></td>
</tr>
<tr>
<td>SH</td>
<td>Schleswig-Holstein</td>
<td>0.15</td>
<td></td>
<td>0.7</td>
<td></td>
</tr>
<tr>
<td>TH</td>
<td>Thüringen</td>
<td>0.10</td>
<td></td>
<td>0.7</td>
<td></td>
</tr>
</tbody>
</table>
Calculations CustomerBasic for Customer

imports: 0 TimeUnits
2 BusinessRequirements

Node: isRebated [FlagVar]
Kind: implements
1st Target: Users should be rebated
[ Some users should get cheaper phone calls. The reasons for the rebates are outlined below. ]

flag isRecentlyActive := entity.calls.last.startTime.isOlderThan(30 day)
flag isRebated := magic of type boolean[T]

[A couple of statistics about the last month's activity ]
value callsLastMonth := entity.calls.where(!it.startTime.isOlderThan(30 day))
flag activeThisMonth := !callsLastMonth.isEmpty
value devicesUsedLastMonth := callsLastMonth.select(it.sourceDevice).distinct
callsLastMonth.size
value totalPriceLastMonth := \[ \sum_{i = 0}^{\text{callsLastMonth.size}} \text{callsLastMonth.at}(i).price.value \]
value averageCallPriceLastMonth := \frac{\text{totalPriceLastMonth}}{\text{callsLastMonth.size}}

[ Some random examples. ]
value example := all[Call].first.customer.calls.first.startTime
Calculations CallCalculations for Call

| flag isLocal := magic of type boolean |
| flag isLongDistance := magic of type boolean |
| flag isRoaming := magic of type boolean |

| value cust := entity.customer |
| value pricingFactor := |

<table>
<thead>
<tr>
<th>cust.isRebated</th>
<th>isLocal</th>
<th>isLongDistance</th>
<th>isRoaming</th>
<th>otherwise 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>0.6</td>
<td>0.8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>!cust.isRebated</td>
<td>0.8</td>
<td>0.9</td>
<td>1.0</td>
<td></td>
</tr>
</tbody>
</table>

Here is a comment added in the gutter, just as in MS Word.

22/09/14 08:19 (13 s ago) by markusvoelter
contract BaseContract specializes <no baseContract> imports: << ... >>

Context Objects:
  c: Customer

[final] assign callsThisMonth
callsThisMonth := c.callsLastMonth

assign amountThisMonth
amountThisMonth := 0

[final] store storeBill
    c.bills := new MonthlyBill {
        amount := amountThisMonth
    }
contract FlatrateContract specializes BaseContract

Context Objects:
  c: Customer

conditional assign overrides BaseContract.amountThisMonth as of 16/8/2014
amountThisMonth :=
  c.isRebated 40
  otherwise 50

conditional assign overrides BaseContract.amountThisMonth as of 20/8/2014
amountThisMonth :=
  c.isRebated 40
  otherwise 60
contract FlatrateContract specializes BaseContract

Context Objects:
  c : Customer

[final] assign BaseContract.callsThisMonth
callsThisMonth := c.callsLastMonth

[final] store BaseContract.storeBill
  c.bills := new MonthlyBill {
    amount := amountThisMonth
  }

conditional assign overrides BaseContract.amountThisMonth as of 16/8/2014
  amountThisMonth := c.isRebated 40
  otherwise 50

conditional assign overrides BaseContract.amountThisMonth as of 20/8/2014
  amountThisMonth := c.isRebated 40
  otherwise 60
[Business Application]

Telco Demo App

```
rule checkStuff
  given anything
  when [the customer.calls.size is equal to 10] and [the call.endTime is smaller than 20]
  then [set call.price to 20]
    [execute cancelContract with customer]
```
[Business Application]

Telco Demo App

Assessment: UnusedCode
query: unused code
sorted: must be ok: hide ok ones:
last updated: Sep 18, 2014 (3 days ago) by markusvoelter

BaseContract
  storeBill

CustomerBasic
  example
  isMale
  activeThisMonth

FlatrateContract
  FlatrateContract.amountThisMonth
  FlatrateContract.amountThisMonth

total 11, new 0, ok 1
1. Initially you have no points.

InitiallyNoPoints /functional: tags

When the game starts, you have no points.

workpackage initial scope: 1 responsible: peter prio: 1 effort: 1 days

2. Once a flight lifts off, you get 100 points

PointsForTakeoff /functional: tags


3. The factor of points

PointsFactor /functional: tags

Biological Data Analysis
== Computational Biologist/Geneticist
[Biological Data]
CampagneLab's NYoSh

ExecutionTool
Execute a job with a GobyWeb plugin in this environment: RemoteEnvironment
with owner manuele.simian

Job name: MyFirstAlignment
Select the Plugin to execute: STAR22_GOBY

Assign one or more FileSet Instances to each slot:
INPUT_READS : 849-WT (tag: ULBYRRD )

Options for the job:
Ambiguity threshold : 10
GENOME_REFERENCE_ID : build NCBI37.57 for organism mouse
CHUNK_SIZE : 50000000

Submit  Clear
organism set GobyWeb {
  Genomes:
  Homo sapiens ( human )
  Mus musculus ( mouse )
  Rattus norvegicus ( rat )
  Canis familiaris ( dog )
  Danio rerio ( zebrafish )
  Oryctolagus cuniculus ( rabbit )
  Caenorhabditis elegans ( celegans )
  Xenopus tropicalis ( frog )
  Builds:
  human ( NCBI57 . 37 ) //
  human ( GRCh37 . 70 ) //
  human ( NCBI36 . 54 ) //
  mouse ( NCBI37 . 55 ) // MM9
  mouse ( NCBI37 . 57 ) //
  dog ( BROADD2 . 57 ) //
  zebrafish ( Zv9 . 65 ) // GCA_000002035.2
  rat ( Rnor_5.0 . 76 ) // GCA_000001895.3
  rat ( RGSC3.4.62 . 62 ) //
  rabbit ( oryCun2 . 69 ) //
  celegans ( WBcel215 . 69 ) //
  frog ( JGI_4.2 . 67 ) //
}
CampagneLab’s NYoSh

Sequence Variations Detected per Reader Position

- goby web alignment 849-WT
  - aligned from sample 849-WT
  - number alignment entries: 38288002
  - number of queries: 29019999
goby web read sample 849-WT {
  tag: NGBOXSJ
  number of reads: 29230382
  is paired: false
  is bisulfite: false
  is ready to align: true
  min length: 101
  max length: 101
  organism: mus_musculus
  platform: Illumina
  lib protocol preserve strand: true
}
Embedded Sw Development
== Electrical Engineer
Embedded Sw Dev
struct Trackpoint {
    int8 id;                  // sequence ID of the trackpoint
    int8/s/ timestamp;       // timestamp as taken from GPS time
    int8/m/ x;               // longitude, simplified as a number
    int8/m/ y;               // latitude, simplified as a number
    int8/m/ alt;             // altitude as of the GPS
    int8/mps/ speed;         // current speed, if available
};

derived unit mps = m s   for velocity
int32 sumUpIntArray(int32[] arr, int32 size) {
    return \sum_{i=0}^{size} arr[i];
} sumUpIntArray (function)

int32 averageIntArray(int32[] arr, int32 size) {
    return \frac{\sum_{i=0}^{size} arr[i]}{size};
} averageIntArray (function)

double midnight2(int32 a, int32 b, int32 c) {
    return \frac{-b + \sqrt{b^2 - 4 \cdot a \cdot c}}{2 \cdot a};
} midnight2 (function)

double sumOfProductsOfLogs(int32[] arr, int32 size) {
    return \sum_{i=0}^{size} \frac{k}{\log_2 arr[i]};
} sumOfProductsOfLogs (function)
exported cs interface TrackpointStore1 {
    void store(Trackpoint* tp)
        pre(0) isEmpty()
        pre(1) tp != null
        post(2) !isEmpty()
        post(3) size() == old(size()) + 1
    Trackpoint* get()
        pre(0) !isEmpty()
    Trackpoint* take()
        pre(0) !isEmpty()
        post(1) result != null
        post(2) isEmpty()
        post(3) size() == old(size()) - 1
    query int8 size()
    query boolean isEmpty()
}
[Embedded Software] mbeddr
```java
mock component StorageMock report messages: true {
    provides TrackpointStore1 store
    Trackpoint* lastTP;
    total no. of calls is 5
    sequence {
        step 0: store.isEmpty return true;
        step 1: store.store {
            assert 0: parameter tp: tp != null
        }
        do { lastTP = tp; }
        step 2: store.isEmpty return false;
        step 3: store.take return lastTP;
        step 4: store.store
    }
}
```
exported component Judge extends nothing {
    provides FlightJudger judger
    int16 points = 0;
    void judger_reset() \op judger.reset {
        points = 0;
    } runnable judger_reset
    void judger_addTrackpoint(Trackpoint* tp) \op judger.addTrackpoint {
        points += 0 | tp->alt <= 2000 m | tp->alt >= 2000 m |
        tp->speed < 150 mps 0 10
        tp->speed >= 150 mps 5 20
    } runnable judger_addTrackpoint
    int16 judger_getResult() \op judger.getResult {
        return points;
    } runnable judger_getResult
} component Judge
```plaintext
statemachine FlightAnalyzer initial = beforeFlight {

...  
    state crashed {
        entry { raiseAlarm(); }  
    }

...  

void raiseAlarm() {}  

statemachine FlightAnalyzer initial = beforeFlight {

    out crashNotification() => raiseAlarm  
...  
    state crashed {
        entry { send crashNotification(); }  
    }
```

```
composite state airborne initial = flying {
  on reset [ ] -> beforeFlight { points = 0; }
  on next [tp->alt == 0 m && tp->speed == 0 mps] -> crashed

state flying {
  on next [tp->alt == 0 m && tp->speed > 0 mps] -> landing
  on next [tp->speed > 200 mps]
    -> flying { points += VERY_HIGH_SPEED; }
  on next [tp->speed > 100 mps]
    -> flying { points += HIGH_SPEED; }
}

state landing {
  on next [tp->speed == 0 mps] -> landed
  on next [ ] -> landing { points--; }
}

state landed {
  entry { points += LANDING; }
}
}
**Embedded Software**

```plaintext
[checked]
exported statemachine FlightAnalyzer initial = beforeFlight {
  next(Trackpoint* tp) {
    beforeFlight:
    [tp->alt > 0 m] -> airborne
    airborne:
    [tp->alt == 0 m & tp->speed == 0 mps] -> crashed
    [tp->speed > 200 mps & tp->alt == 0 m] -> airborne
    { points += VERY_HIGH_SPEED; }
    [tp->speed > 100 mps & tp->speed <= 200 mps & tp->alt == 0 m] -> airborne
    { points += HIGH_SPEED; }
    [tp->speed == 0 mps] -> landed
    [tp->speed > 0 mps] -> landing
    { points--; }
    [ ] -> beforeFlight
  } reset() {
    [ ] -> beforeFlight
  }
}
```
section 1.2 existing.comps: Interfaces and Components {

Interfaces declare operations that can be provided or used by components. Each operation can also declare pre- and postconditions as well as protocols. These can be checked either at runtime or statically. The @cm(Components) module contains examples. Below is an interface:

embed as text Components.TrackpointStore1/

The interfaces, components and their relationships in a given module can also be rendered graphically. An example is shown in @fig(ci)

visualize Components.store.TrackpointStore1/
    components + interfaces (grouped) as ci
    location: vis:/
    scaling: width100

The components and their provided (solid lines) and required (dotted lines) ports.

Of course the visualizations are also not just images. In the source to the document, we embed references to \code(IVisualizable) instances. In the doc, one can select the visualization category, and then, during generation, PlantUML automatically rerenders the image. 
}
System: LWB

Lang 1 → Lang 2 ← Lang 3
Lang 4
Lang 5
... Lang N
[LOBA]

- Business Rules
- (Financial) Calculations
- Data Structures
- Mappings or Queries
- Validations
- Scientific Processes
- Contracts
- Processes
- UI

Core Business Logic
Expressivity for Core Domain Knowledge

Build Language for Domain!

User-Friendly Notation
You’ve seen the demos.

Testing
An integrated DSL for testing.

Meaningful Analyses
Types, Consistency, Checking

Synthesis of Software
Code Generation.
Everything is wonderful. Right?
Well, not quite.
5 Challenges for Education
Illustrate the usefulness of language development

Emphasize Language Engineering over Compiler Construction

Make it part of the SE curriculum, not (just) CS
Teach Computational Thinking
Teach Abstraction Techniques
Create an Understanding how Computer Science attacks Problems
Help them overcome anxiety relative to „programming“
Modularize

Procedures, Classes, Components, Services, User Stories
Encapsulate

Private Members
Frameworks
Facade Pattern
Components
Layers/Rings/Levels
Packed Data Wrapper
Contracts

Interfaces
Pre/Post Conditions
Protocol State Machines
Message Exchange Patterns
Published APIs
Handle Crosscuts

Aspect Orientation
Interceptors
Application Servers
Exception Handling
Parametrization

Function Arguments
Command-Line Args
Configuration Files
Standard Library

Lisp (Grow A Language)
Autosar Sys Components
Microkernel OSs
Orthogonality

Closures, Program As Data, Macros, Higher-Order Functions
Identity

Pointers
GUIDs
MAC-Address
URI
Qualified Names
Abstraction

Operating Systems
High-Level Languages
Models, DSLs
Types & Instances

Programming Languages
Components
Models & Metamodels
RDBMS/XML Schemas
Hierarchical Decomposition

Procedures/Methods
State Machines, Components
Specialize

Currying
Inheritance
State Machines
Viewpoints

Configuration Files
4+1 Model
Blackbox/Whitebox
Types/Instances/Deployment
Nota9on

UML
Lisp
Java
Ruby
Declaration

Implementation

App Servers (EJB), Plugin RT (Eclipse) Models, Transactional Memory
Test semantics, not syntax (code gen)
Higher Order Functions (map, foreach)
Lazy Evaluation
Transactional Memory

Don’t Overspecify
Avoid Sideeffects

Functional Programming
Concurrency (Sharing)
Distribution
Summary
Teach Computational Thinking
Teach Abstraction Techniques
Create an Understanding how Computer Science attacks Problems
Help them overcome anxiety relative to „programming“
[Read & Learn]

- **DSL Engineering**
  Designing, Implementing and Using Domain-Specific Languages
  Markus Voelter
  dslbook.org

- **Generic Tools, Specific Languages**

- **The MPS Language Workbench**
  Fabien Campagne
  http://books.campagnelab.org

- **mbeddr**

- **MPS**
  Meta Programming System
Thank you!