

**MDDRW –
Model-Driven Development
in the Real World**

FZI Karlsruhe

Markus Voelter
(voelter@acm.org, www.voelter.de)

Prolog:

I thought we should **answer
the questions** specifically.

Prolog:

I thought we should **answer
the questions** specifically.

I would have talked about
other stuff otherwise.

Prolog:

such as:

- Architecture Modeling
- Textual DSLs
- Product Lines, Feature Modeling and the
integration with “models”
- AO Modeling, Model Weaving
- Eclipse Modeling, openArchitectureWare

1

**Comments
on the CFP**

” ... MDSD is a clear
opponent to a community
which declares software
engineering should be
code-centric. “

Isn't MDSD (or DSLs) just a **different kind** of code?

” ... and there are no further implementation decisions, as they are already fixed by the existing transformations “


But you can **parametrize** or **configure** the transformation.

” ... unclear how to best specify the executable semantics of software in a universal manner than in code. “

How do we handle (formal) semantics of today's **programming languages**?

Do we?

2 Some general thoughts

So, are we doing the
same-old  stuff?
Programming == Modeling?

I don't think so.

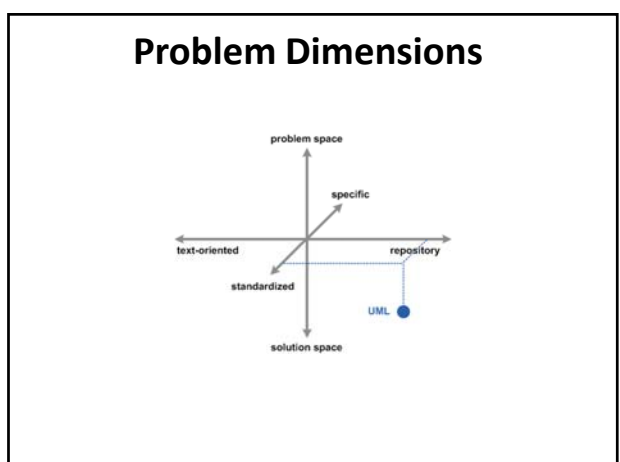
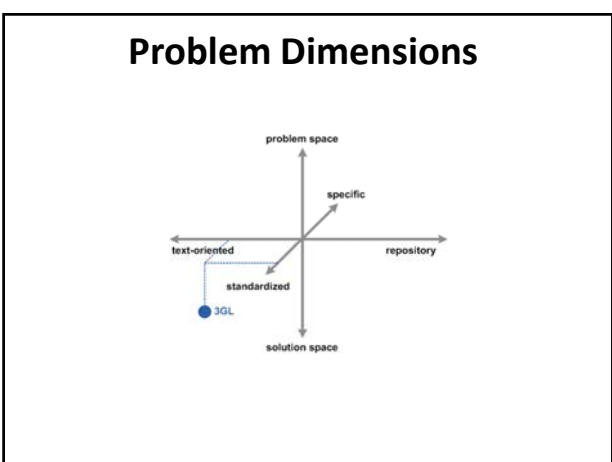
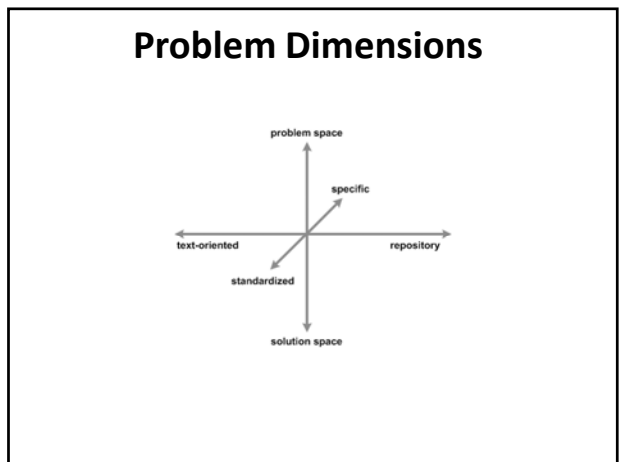
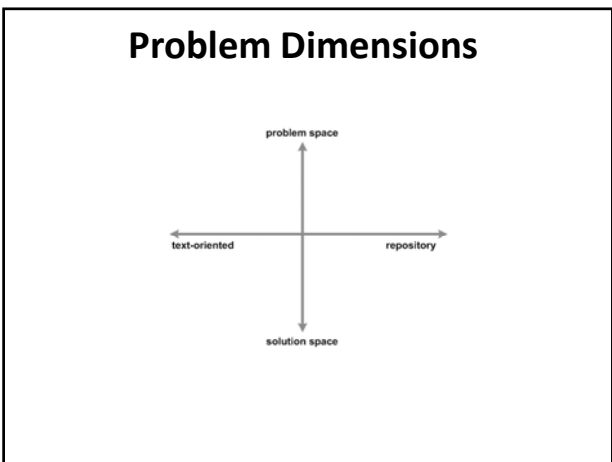
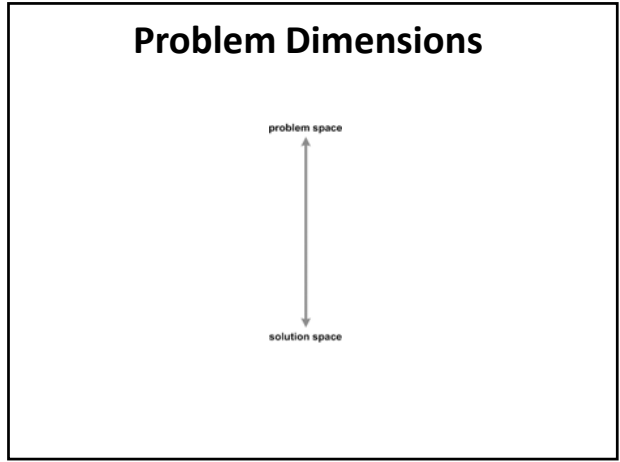
Domain orientation
(# solution orientation)

Domain orientation
(# solution orientation)
Precision
(# turing/algorithmic)

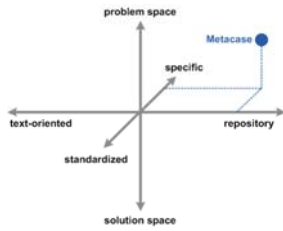
Domain orientation
(# solution orientation)
Precision
(# turing/algorithmic)
Adaptable syntax
(# predefined syntax)

Domain orientation
(# solution orientation)
Precision
(# turing/algorithmic)
Adaptable syntax
(# predefined syntax)
Different stakeholders
(# just programmers)

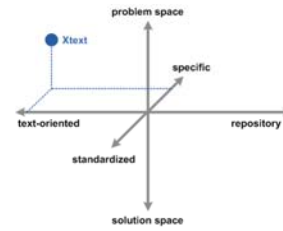
Domain orientation
 (# solution orientation)
Precision
 (# turing/algorithmic)
Adaptable syntax
 (# predefined syntax)
Different stakeholders
 (# just programmers)
We build languages ourselves
 (# buying a "standard")



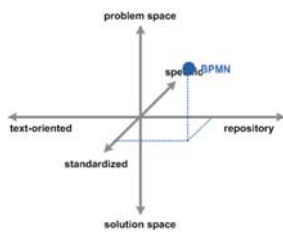
Problem Dimensions



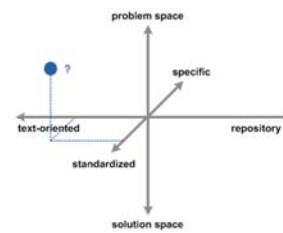
Problem Dimensions



Problem Dimensions



Problem Dimensions



3 Answers to the questions

? How to come to the meta-model and to the transformations?

Iteration
Experience
Inspiration
Feedback



What means software
verification?

Meta Models
Models
Transformations
Generators

For models:
constraints and
simulations



For the rest?
Do we really need it?

Do we really need it?
Where?

Where?

Automotive?

Space?

Intensive Care?

AFAIK, no!



How to evolve meta models and their related instance models?

Conceptual Challenges

vs.

Tooling Challenges

No problems!

Deprecation

Sensible Defaults

Errors in case of conflict

Graceful!

Automatic Migration?

Do we have that for code? Not really.

Good:

MetaEdit+

Intentional

Xtext

Bad:

EMF?

GMF!

**Storage Format
!=
Metamodel**

**make version
a first class
concept**

**Don't generate
tooling –
interpret!**

**Load
View
Check
Modify
Reason**



**What are suitable ways to
define and maintain large
models (on the meta and
instance level)?**

This includes the question whether a textual syntax
or a graphical notation is more suitable?

**Repository
vs.
Files?**

I am undecided!

**Textual DSLs:
File-based.
Just like code.**

We know how to do that!
Good Starting point.
Simple Integration with Code.

**For other notations:
Repository
Unlike Code.**

We know how to do that...
... but we don't have good tools.

Don't mix!

Textual Stuff in Repositories...
Graph notations in (XML) files.

**What is a repository
in the first place?**

**What is a repository
in the first place?**

**RDBMS? OODBMS?
Btrees? Text files + Index?**

Good starting point:

IDEs

Good starting point:

IDEs

Store models in files
Build (persistent) index file(s)
for searching and linking.

Good starting point:

IDEs

Store models in files
Build (persistent) index file(s)
for searching and linking.

(works especially well for [textual DSLs](#),
but also works for other syntaxes)

Textual DSLs

vs.

Graphical

Textual DSLs

vs.

Graphical

vs.

Visualization

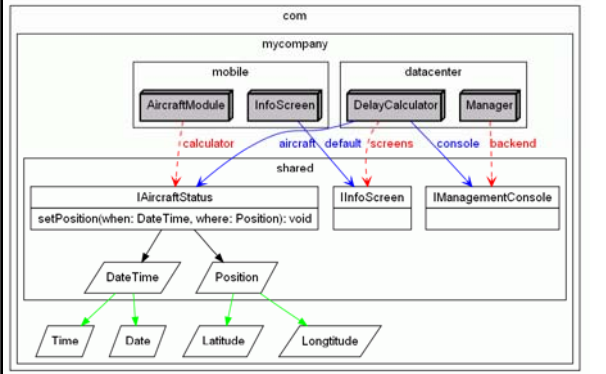
Via M2M

Read-Only

Auto-Layout

Drill-Down

Graphviz



Prefuse



“ ... it becomes clear that to make MDSD working in real-world projects **major research effort** from various fields of software engineering is needed. ”

Really?

Remember:
 Don't have to be better than today with text (code)!

Tooling/Engineering problems – **not** research areas.

**Best Practice
Approaches
available –
IMPLEMENT THEM!**

Thank you.
Questions?
Criticism?
Tomatoes?