

# Informatics Seminar

Representation vs. Content

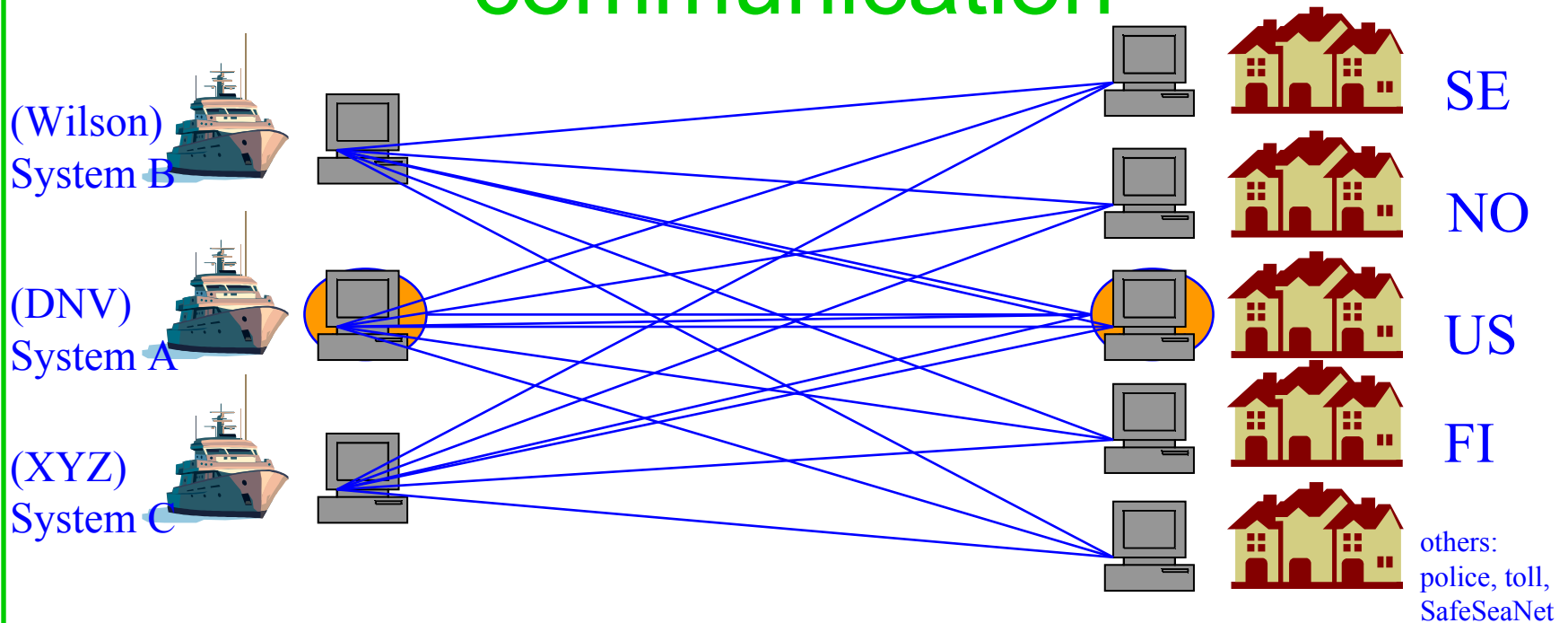
Prof. Andreas Prinz



# Content

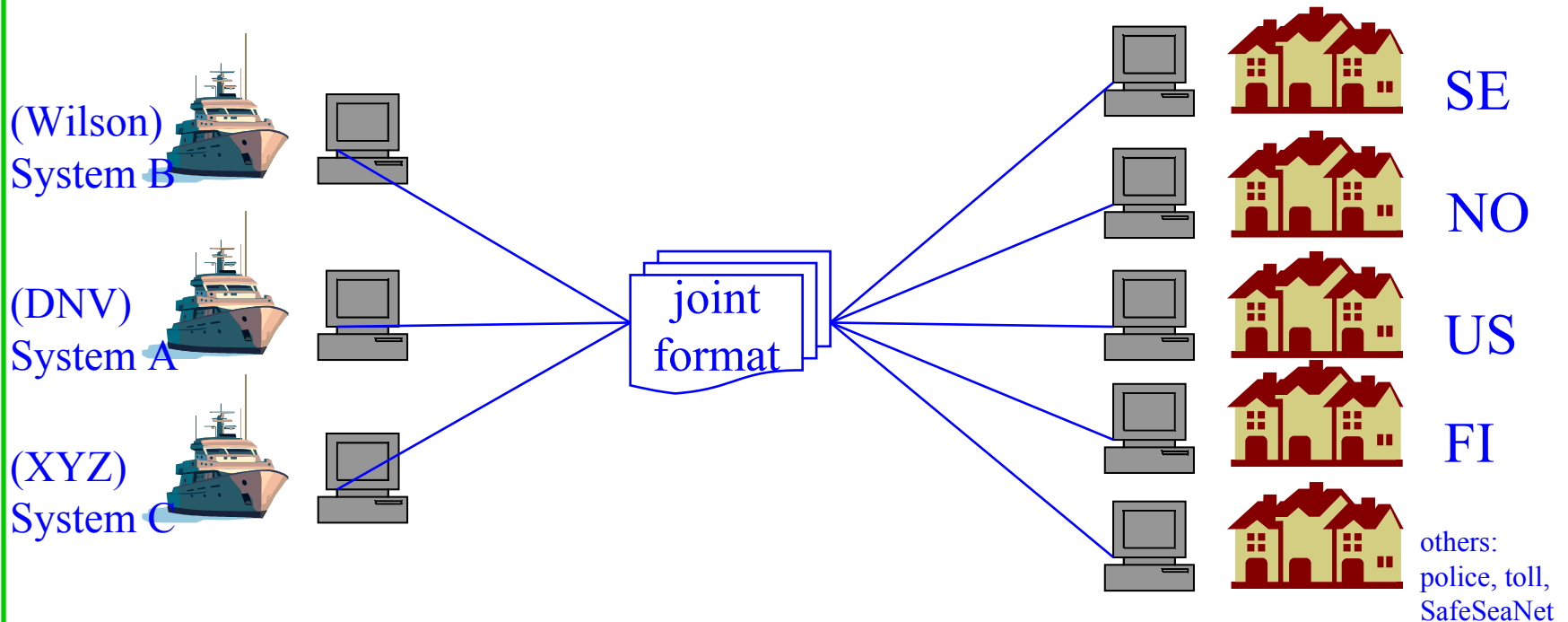
- Introduction
- Models and Meta-models
- Model-driven technologies
- Examples
- Summary
- Exercise

# Sample Problem: Ship-land communication



- improvement of routines on board
- improvement of routines in ports
- Who is going to integrate those and how?

# Sample solution



- Joint information model for information representation and information exchange
- independent of communication technology
- works as a reusable interface for the local formats

# Numbers

11111010110

3726

MMVI

6018 / 3

2006

6018 : 3

the current year

7D6

1003 \* 2

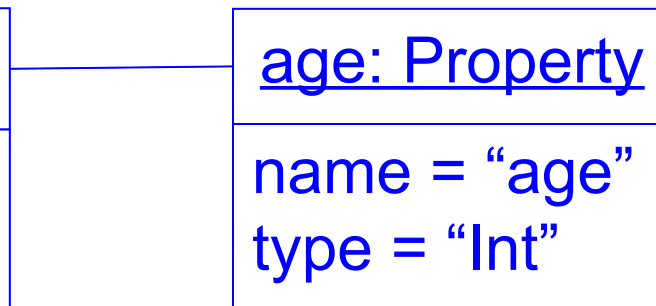
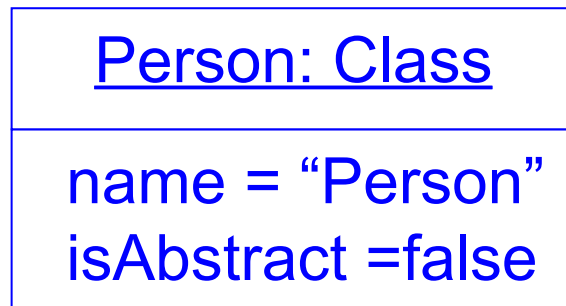
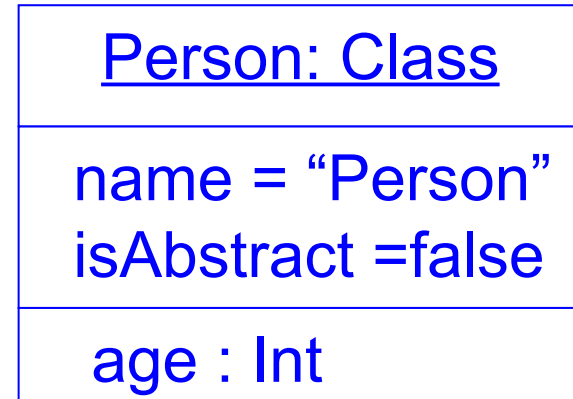
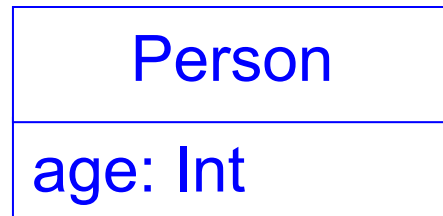
1003 · 2

two thousand and six

6018

3

# UML example - clabjects



# Content

- Introduction
- Models and Meta-models
- Model-driven technologies
- Examples
- Summary
- Exercise

# What is a Model?

- A model is an abstraction of a (part of a) system.
  - one model describes several systems, one system can have several models
  - simplified view of a system with respect to criteria
  - can answer questions about the system if related to the view
  - needs a representation, e.g. using a modelling language
- Models on different abstraction levels
  - Models of the real Bits: Assembler
  - Models of the Control Flow: Prog. Lang.
  - Models of data storage: Database descriptions
  - Models of access: Interface languages
- What is the best model of a cat? → It is a cat. But it has to be the same cat!
- A model has aspects like a language.

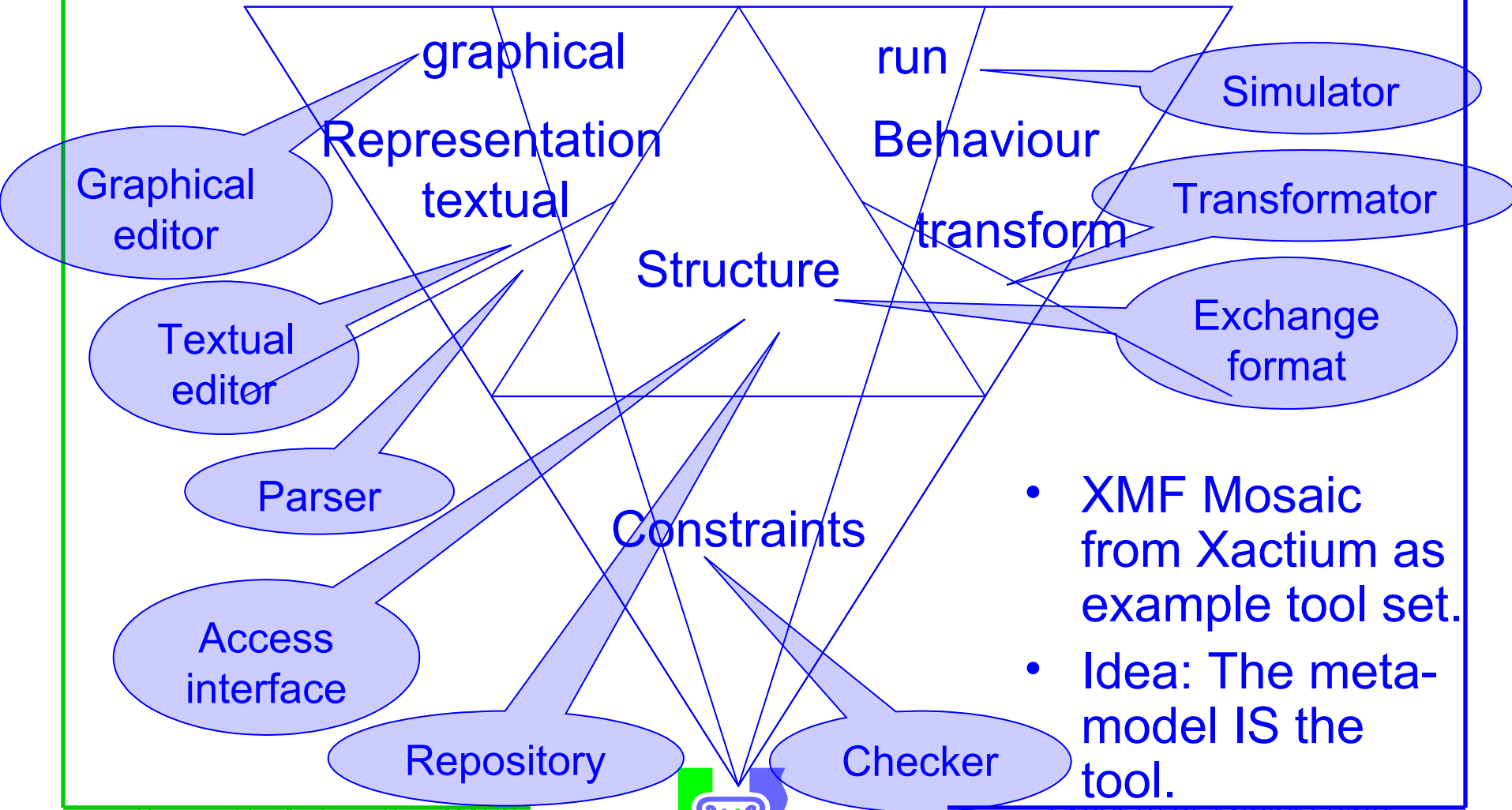
# What are Meta-Models?

- A description of a class of models
- Models / high-level descriptions of the modelling language
  - narrow view: structure of the modelling language
  - wider view: all important aspects of the language, i.e. structure, presentation, static and dynamic semantics
- Meta-models (languages) can have several aspects.

# Aspects of Compilers/Languages

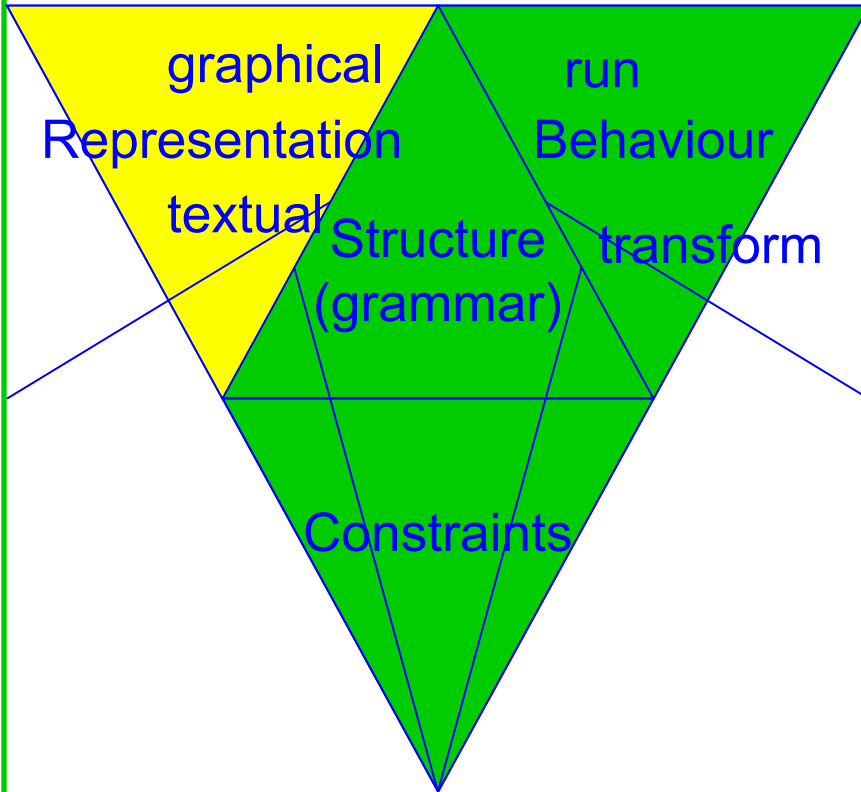
- Language structure: What are the concepts? How are they related?
- Static semantics: additional conditions, what is allowed?
- Representation: How are programs written? -> graphical vs. textual
- Dynamic semantics: What do the programs mean? How to generate code for them?

# Aspects of a meta-model / language

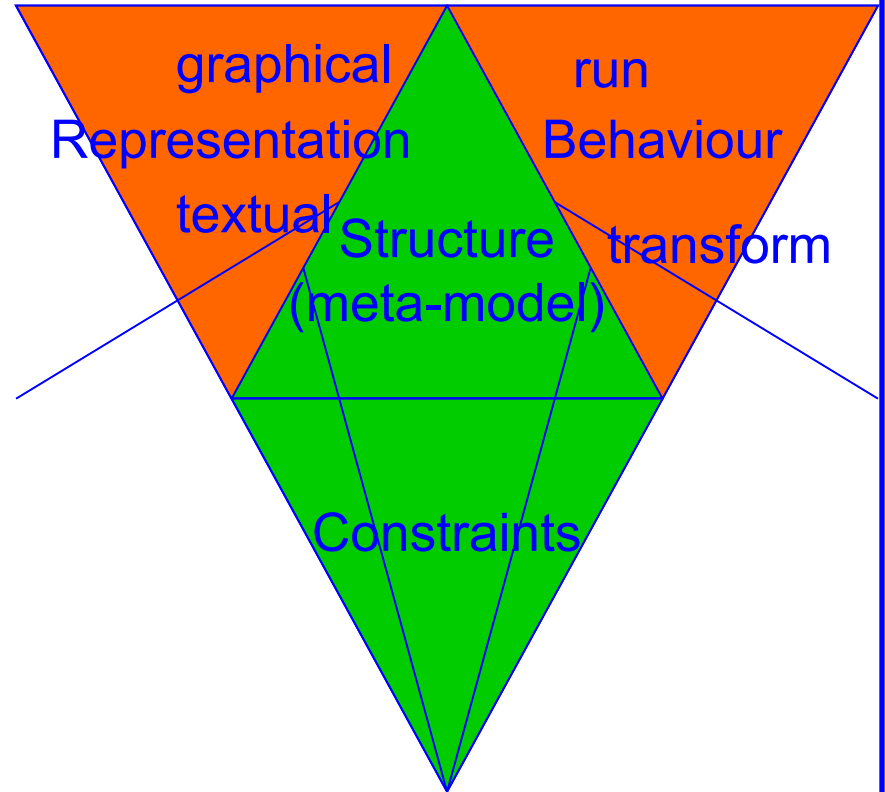


- XMF Mosaic from Xactium as example tool set.
- Idea: The meta-model IS the tool.

# Meta-models for SDL and UML

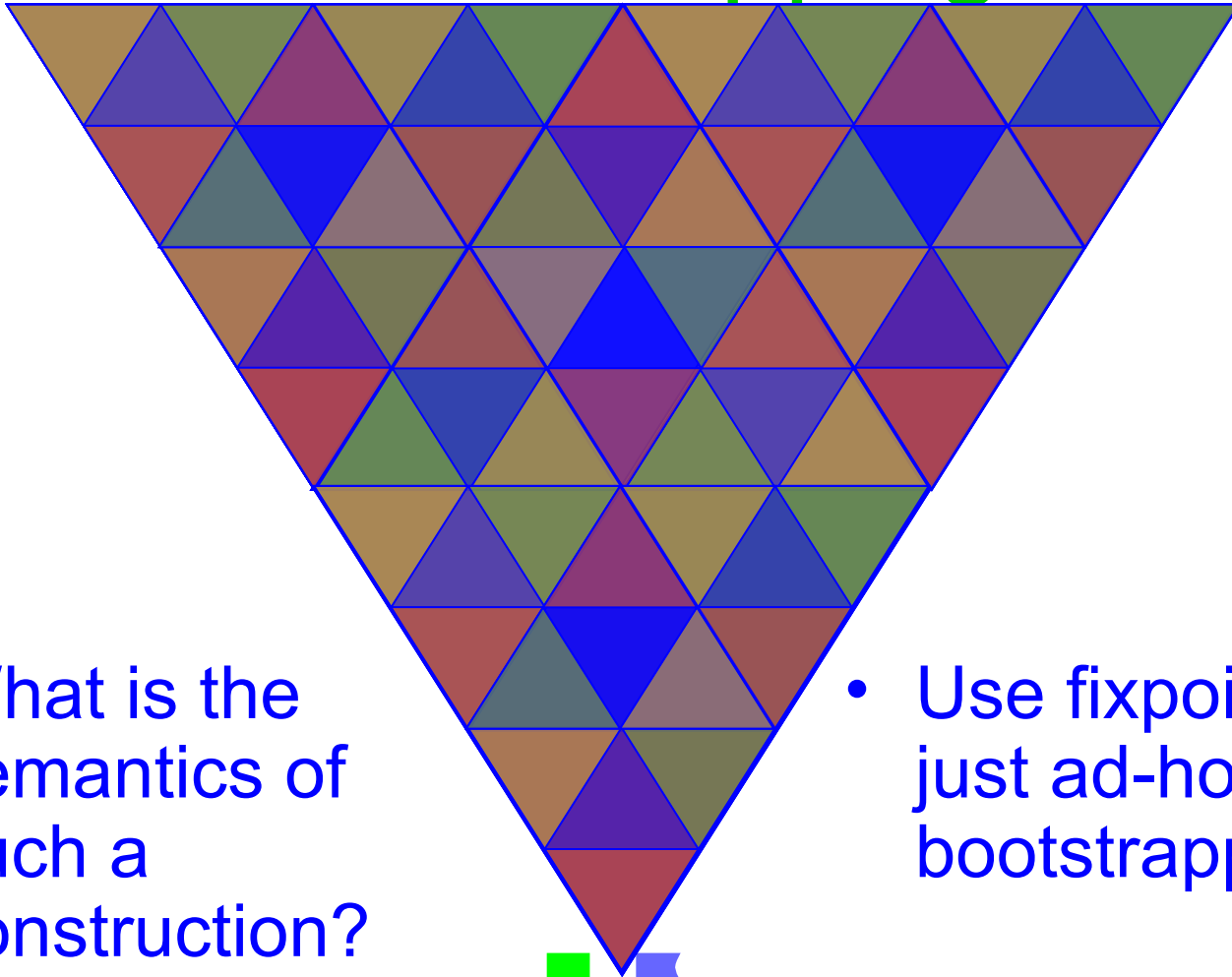


SDL



UML

# Bootstrapping



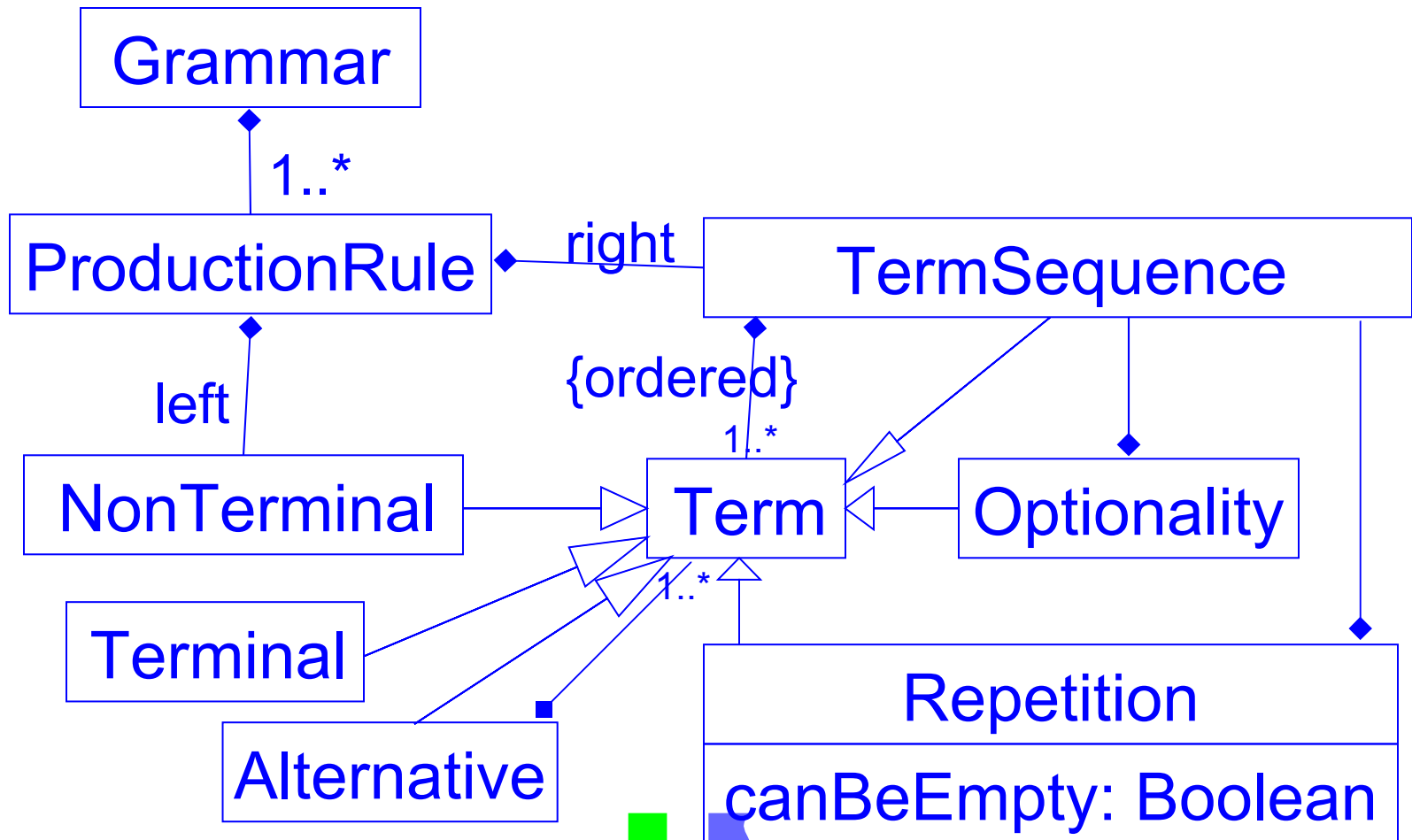
- What is the semantics of such a construction?

- Use fixpoints or just ad-hoc bootstrapping

# Which presentation to take?

- The one that fits best, but ...
- We need tools for the presentation!
- Better take a predefined one?
- Define your own language and use MDA for the presentation description
- This is called Language Driven Development = LDD

# Simple sample structure



# Simple sample constraints

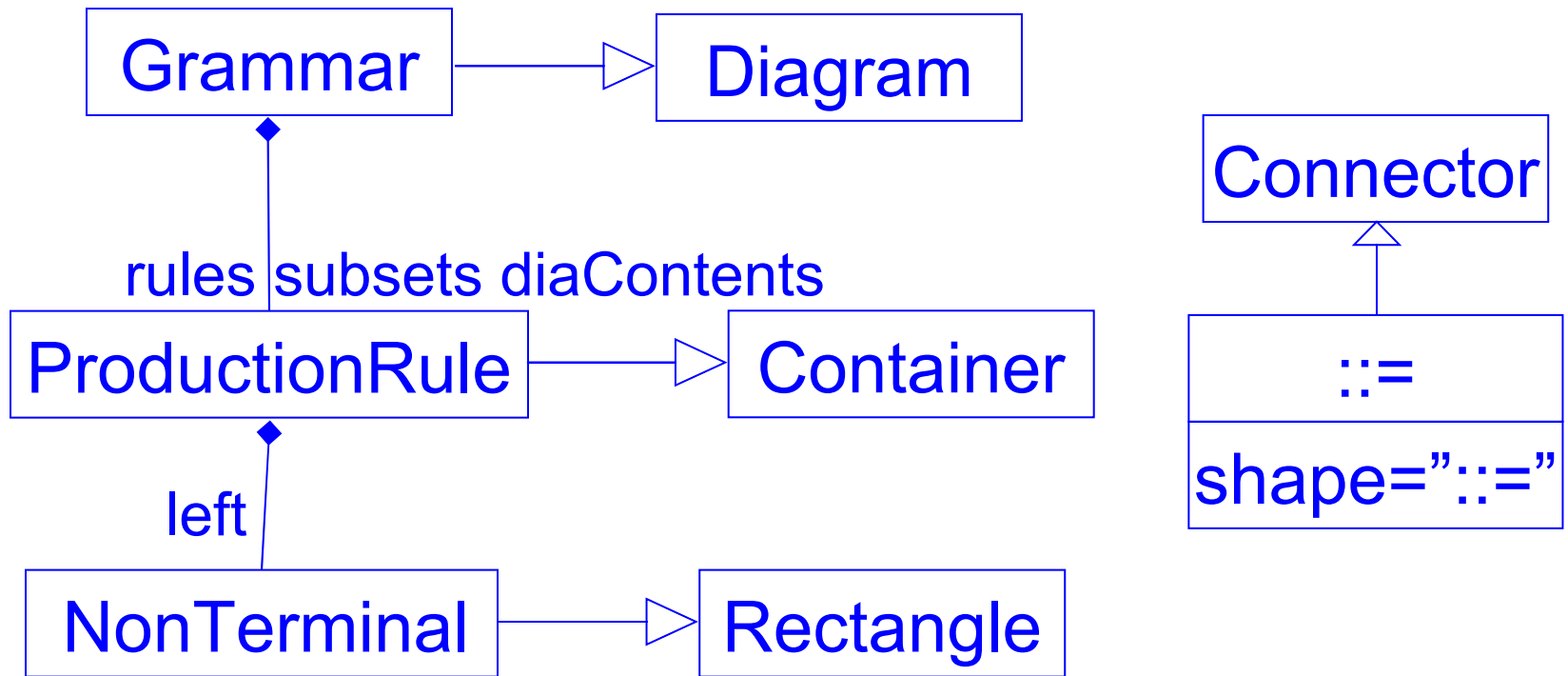
context = Repetition

inv not self.canBeEmpty implies  
self.exists(TermSequence)

context = NonTerminal

inv theGrammar.ProductionRule.exists (p |  
p.name = self.name)

# Simple sample graphics



ProductionRule.contents = left - ::= -> right

# Simple sample text syntax

Grammar : {rules=ProductionRule}\*;

ProductionRule : left:NonTerminal “::=”  
right:TermSequence “.”;

NonTerminal : name=ID;

TermSequence : {term:Term}\*;

Term : NonTerminal | Terminal | Optionality

Terminal : name=ID;

Optionality : “[” opt:TermSequence “]”

# Simple sample transformation

removeAlternatives:

ProductionRule(nt, Alternative(set a))

--> set ProductionRule(nt, a)

removeOptional:

Optional(x)

--> Alternative({x,Nothing})

# Simple sample execution

Run(a:NonTerminal) =<sub>def</sub>

case a.rule of

NonTerminal: Run(a.rule)

Terminal: Print(a.rule.value)

Repetition:

choose n:Natural

foreach x:1..n Run(a.rule.sequence)

Optional:

choose b:Boolean

if b then Run(a.rule.term) else Skip

TermSequence:

foreach n:1..length(a.rule) Run(a.rule[n])

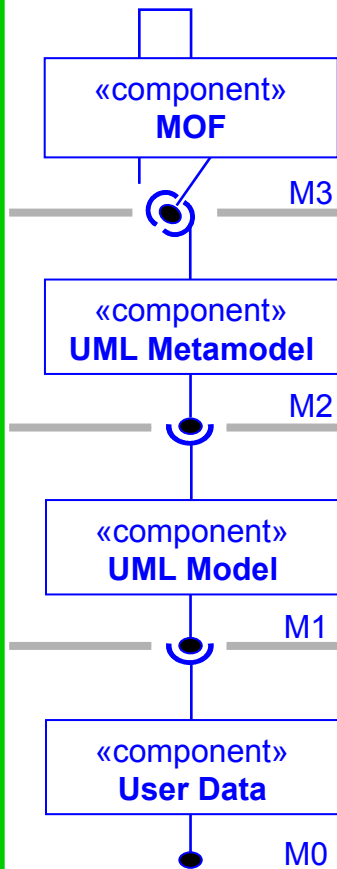
# Content

- Introduction
- Models and Meta-models
- Model-driven technologies
- Examples
- Summary
- Exercise

# MDA: <http://www.omg.org/mda/>

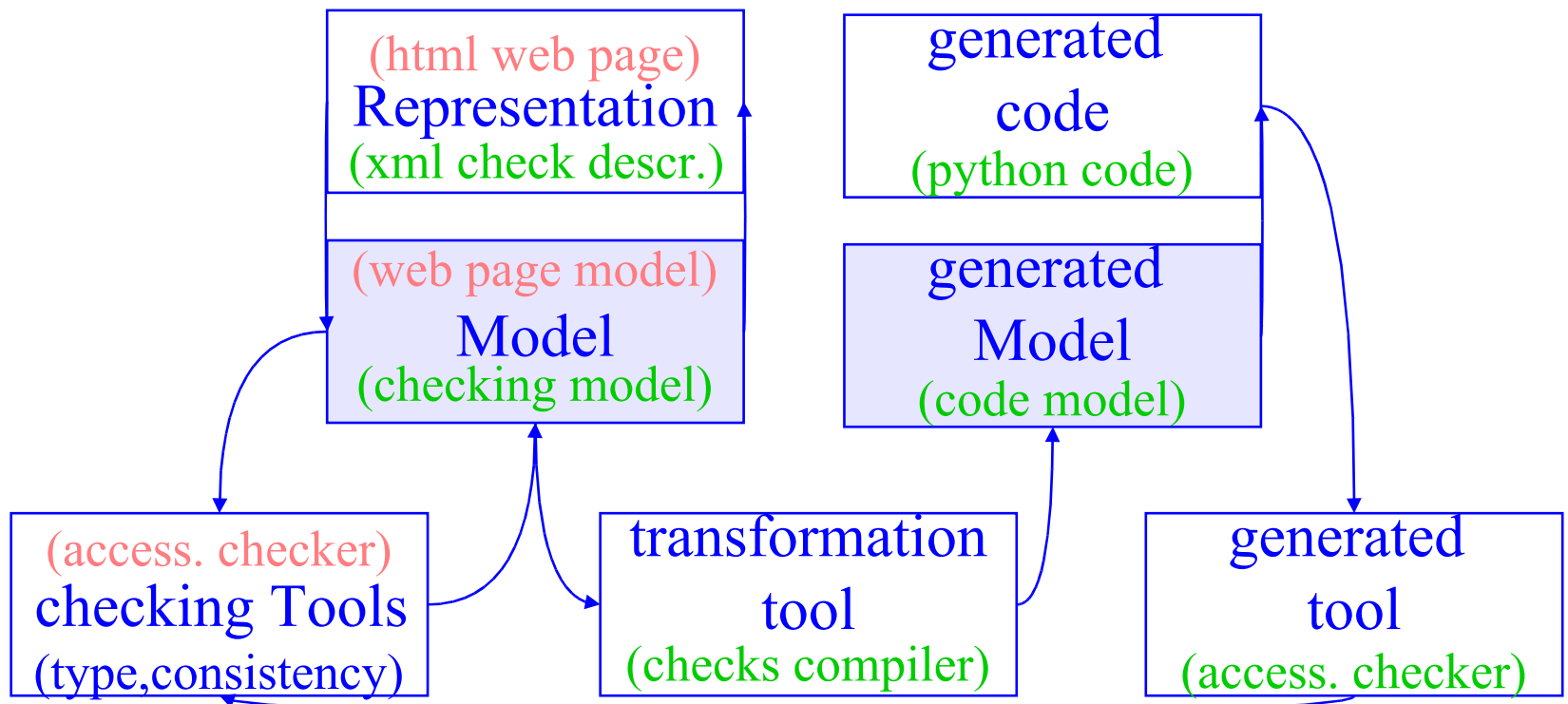
- MDA (Model Driven Architecture) is a way to specify and build systems
  - Often based on UML modeling
  - Supports full lifecycle: analysis, design, implementation, deployment, maintenance, evolution & integration with later systems
  - Applicable to different programming languages, networks, operating systems, middleware
- Explicit separation of three abstraction layers.
  - CIM: The most abstract model, the **Computational Independent Model (CIM)**, deals only with concepts of the application domain.
  - PIM: A CIM is refined into a **Platform Independent Model (PIM)** that contains already computational information about a system but is free from platform specific realization details. A PIM is usually written in UML.
  - PSM: Platform details are part of a **Platform Specific Model (PSM)**, which is a refinement of the PIM. PSM can be diverse: Java, J2EE, SQL, C++, .NET, COBOL, C#, CORBA, XML, etc. etc.

# OMG 4-level architecture

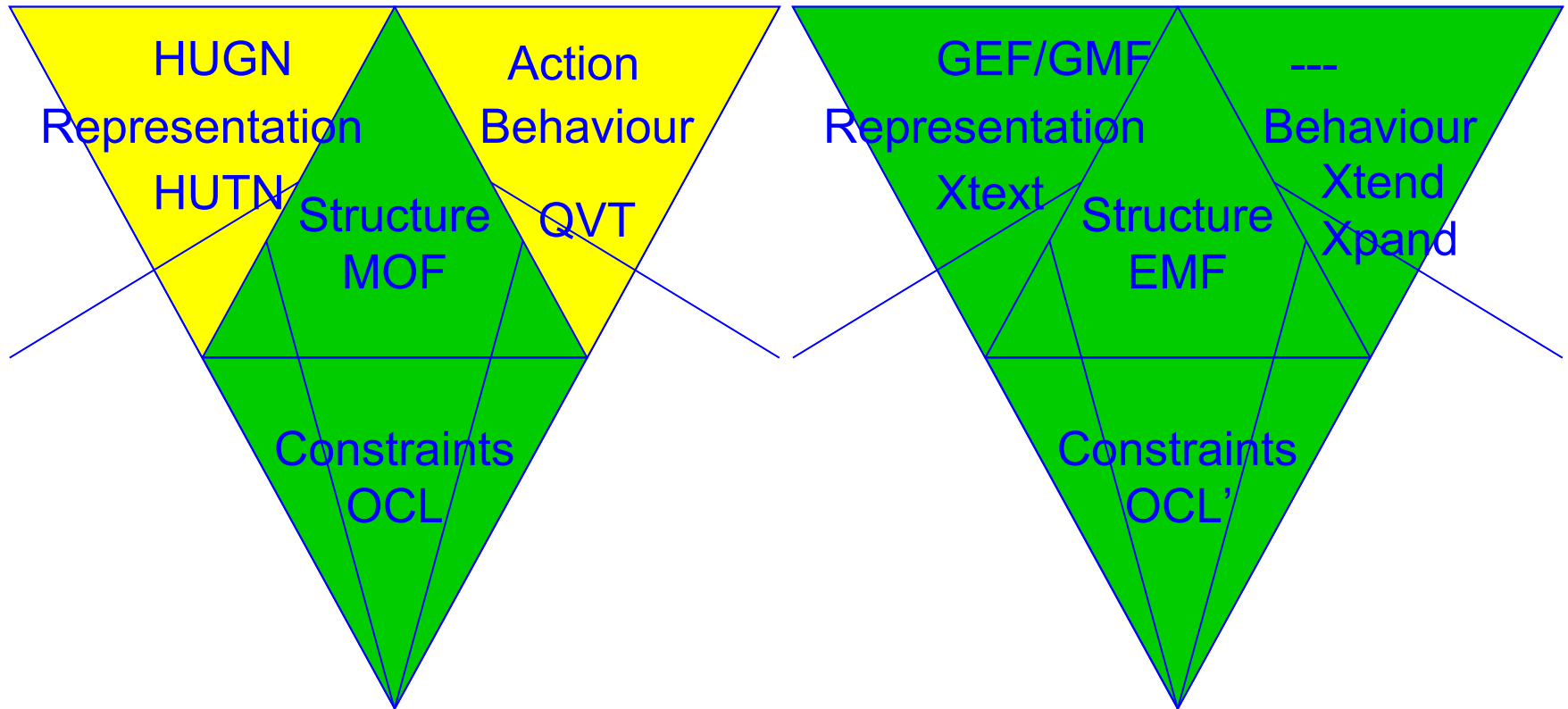


OMG Level	Examples	Class example	OCL example
3 = meta meta model	MOF	class concept	general languages (grammars)
2 = meta model	UML MM	class concept	OCL language
1 = model	UML Model	a class	a formula
0 = instances	real objects	an object	a truth value

# Importance of internal structure



# Language support MDA and Eclipse



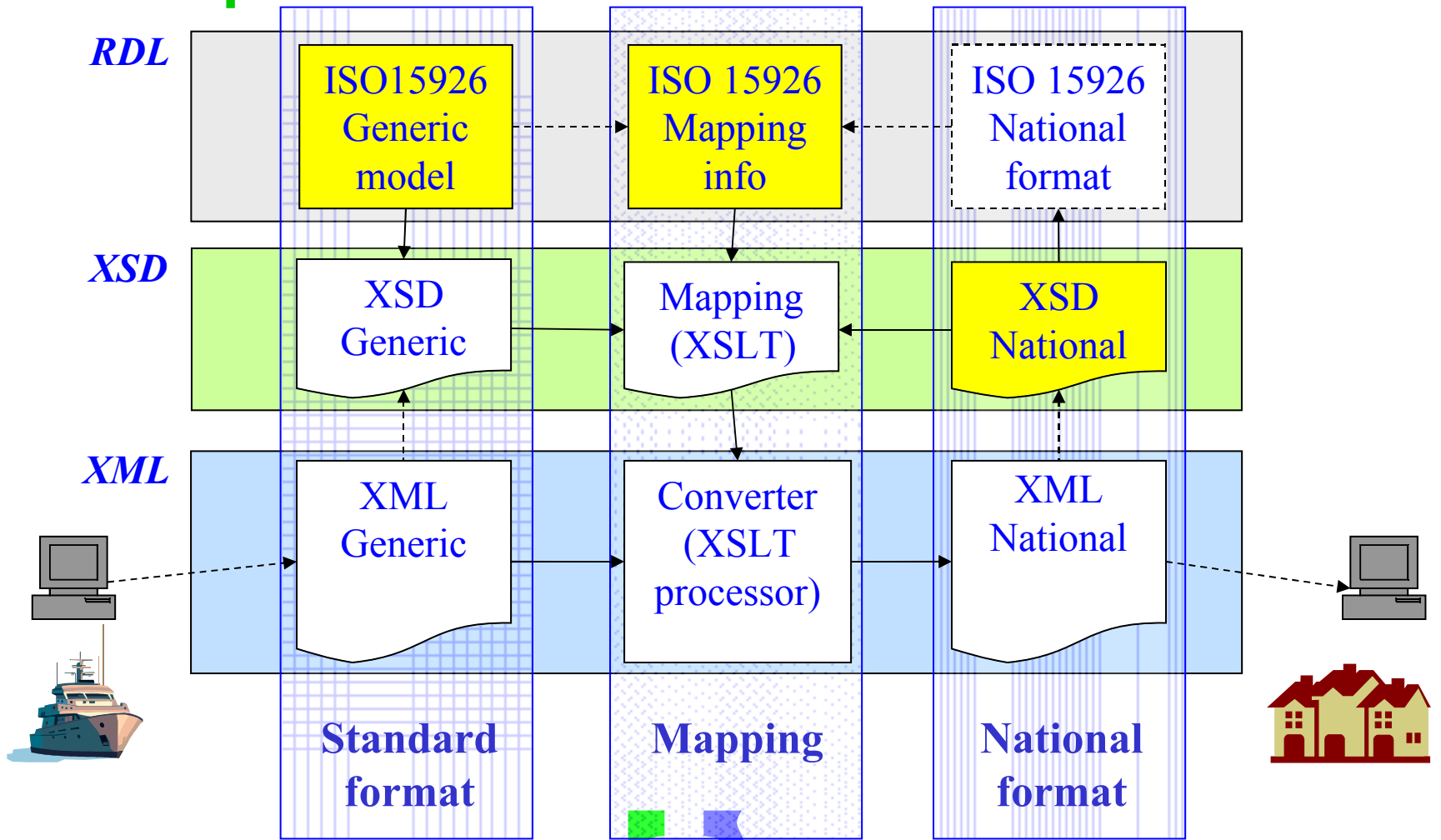
MDA

Eclipse (oaw)

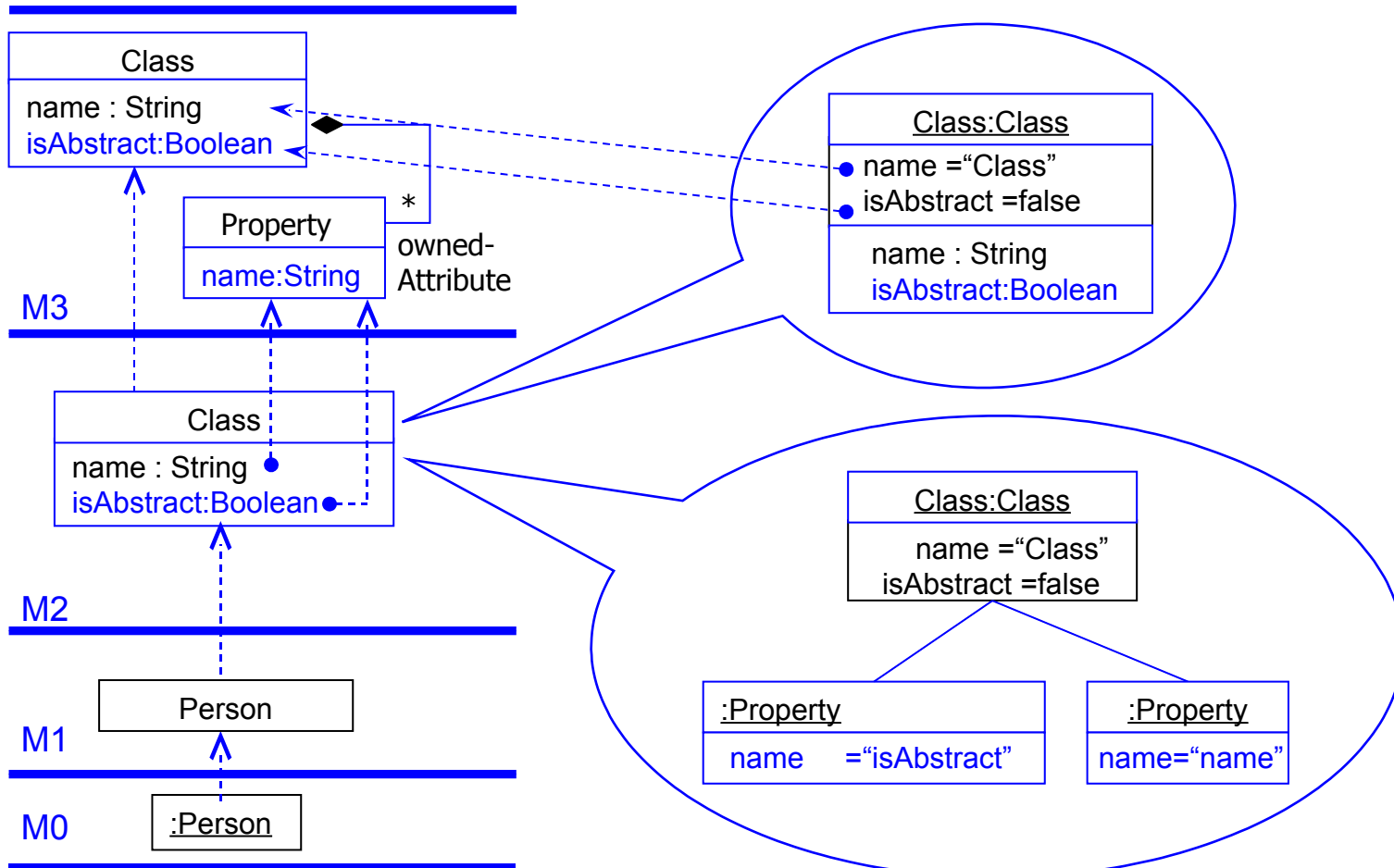
# Content

- Introduction
- Models and Meta-models
- Model-driven technologies
- Examples
  - Ship-Shore Communication
  - MDA
  - EIAO
  - MS Word: Discussion
  - MS PowerPoint: Discussion
  - Latex: Discussion
  - CSS
  - CMS
- Summary
- Exercise

# Ship-Shore: Detailed solution



# Instances on several levels



# EIAO WAM Concept

- A Web Accessibility Metric (WAM) is a formal rule specifying how to make a statement about accessibility barriers in a web resource. Goal of EIAO: make them automatically.
- Define several layers of rules based on the measurable properties of the web resource: A-WAM (analytic), B-WAM (barrier reporting), C-WAM (composing).
- Described in an unambiguous way to ensure that the results are repeatable.
- Furthermore, the meaning of the end result (i.e. the accessibility statement) has to be clear.
- This is achieved by providing a consistent interpretation of the intermediary results.

# A10.01.01.005.001-HTML01.001

- Does the inspected <area> element have an alt attribute?

- **Schematron description:**

```
<sch:rule context="html:area">
```

```
<sch:report test="@alt">1</sch:report>
```

```
</sch:rule>
```

- **OCML description:**

```
context=area
```

```
inv self.exists(alt)
```

# A10.03.03.001.001-HTML03.001

- Is the deprecated attribute bgcolor used for the inspected element?

- **Schematron description:**

```
<sch:rule context="html:body | html:table | html:tr | html:th |  
  html:td">
```

```
<sch:report test="@bgcolor">1</sch:report>
```

```
</sch:rule>
```

- **OCLE description:**

```
context=body inv self.exists(bgcolor)
```

```
context=table inv self.exists(bgcolor)
```

```
context=tr inv self.exists(bgcolor)
```

```
context=th inv self.exists(bgcolor)
```

```
context=td inv self.exists(bgcolor)
```

# Further examples: Discuss

- MS Word: Discussion
- MS PowerPoint: Discussion
- Latex: Discussion
- CSS:
  - [http://www.maxdesign.com.au/presentation/page\\_layouts/](http://www.maxdesign.com.au/presentation/page_layouts/)
  - <http://www.benmeadowcroft.com/webdev/>
  - [http://www.fu2k.org/alex/css/layouts/3Col\\_NN4\\_FFFF.mhtml](http://www.fu2k.org/alex/css/layouts/3Col_NN4_FFFF.mhtml)
  - <http://webhost.bridgew.edu/etribou/layouts/rMenu/index.html>
- CMS
  - OSYS home: <http://osys.grm.hia.no/>
  - OSYS admin: [http://osys.grm.hia.no/osys\\_admin/](http://osys.grm.hia.no/osys_admin/)

# Conclusions / Summary

- It is important to focus on the content / structure of the information
  - definition of good structures is difficult
  - patterns for good structures needed
  - formal high-level descriptions allow the generation of tools
- Meta-models / Languages have several aspects:
  - content (meta-model)
  - presentation (textual, graphical)
  - constraints (static restrictions)
  - (dynamic) semantics (transformations, execution)
- *Models* (High-level descriptions) used for automatic generation of real code, thus achieving more productivity, quality, longevity = Model-driven technology
  - Direct access to the models
  - Easy exchange of representation or several of them
  - Combination of tools handling the language
  - Description of relations between languages
- This is a very active area and many tools exist or are being created.

# Content

- Introduction
- Models and Meta-models
- Model-driven technologies
- Examples
- Summary
- Exercise

# Exercise

- Define for ROTOS
  - structure model
  - 2 structure instances
  - textual representation model
  - 2 textual instances
  - graphical representation model
  - 2 graphical instances

# ROTOS

