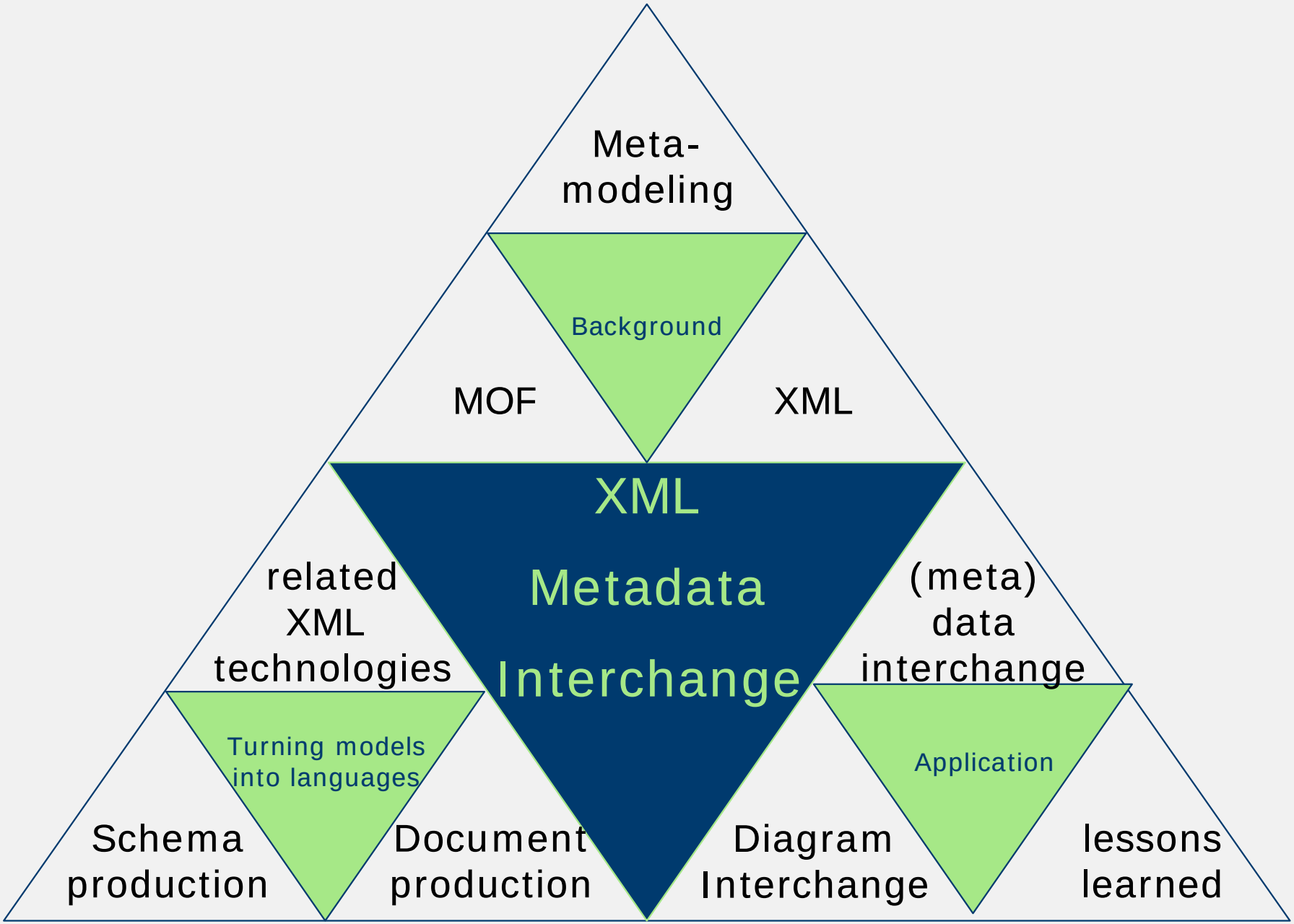
OMG's XML Metadata Interchange Format XMI

Prof. Mario Jeckle

Fachhochschule Furtwangen
mario@jeckle.de
http://www.jeckle.de

XML4BPM, Marburg, 2004-03-25


Meta-
modeling
Background
MOF
XML
XML
Metadata
Interchange
related
XML
technologies
(meta)
data
interchange
Turning models
into languages
Application
Schema
production
Document
production
Diagram
Interchange
lessons
learned

XMI for Interchanging Business Process Management Data

Application of XMI can be twofolded:

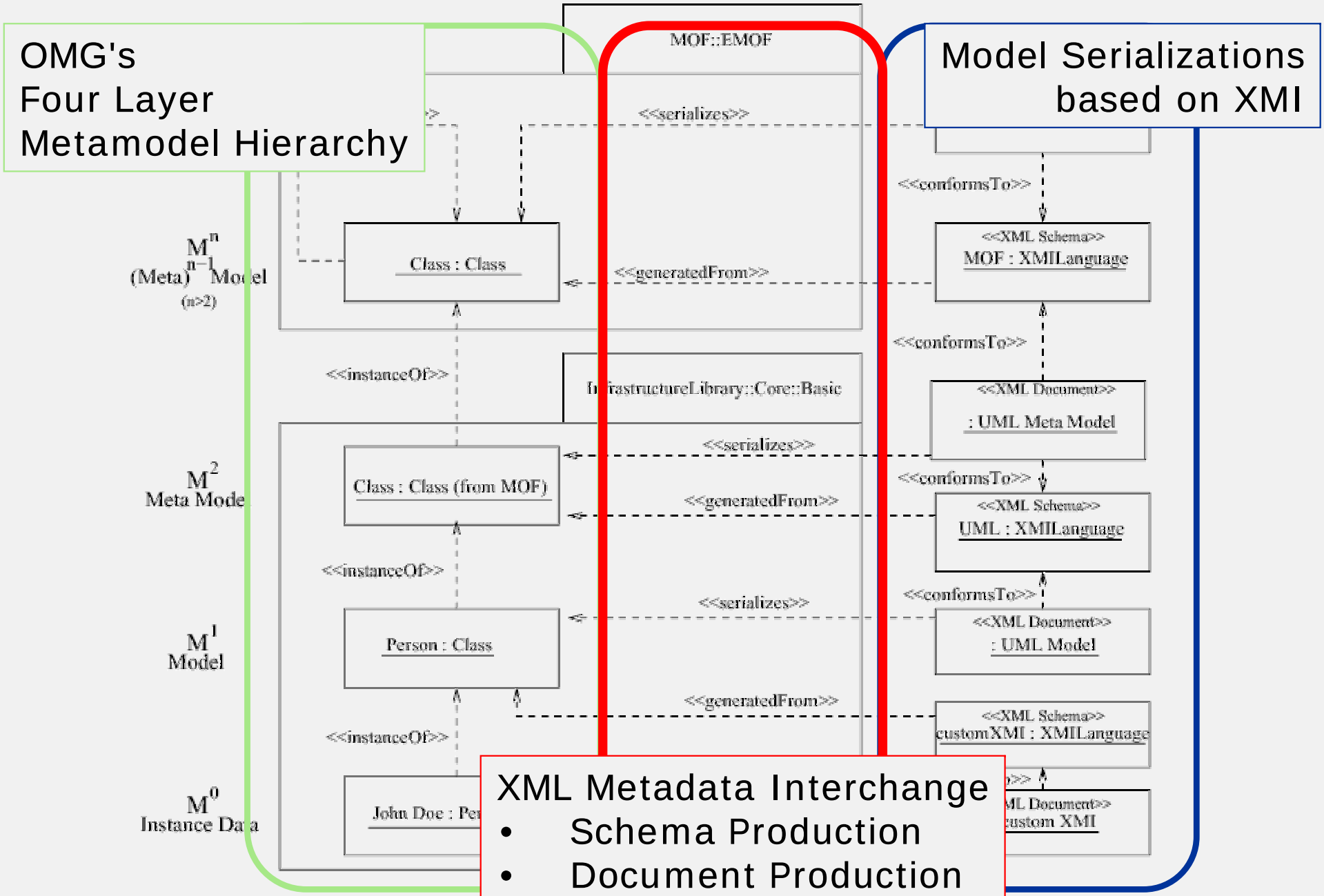
1. Storing descriptions of concrete business processes, i.e. instances of a single meta model

```
<process name="consume" id="p1"/>
<process name="produce" id="p2"/>
<sequence>
    <process id="p2"/>
    <process id="p1"/>
</sequence>
```

2. Storing a business process management model, i.e. the metamodel itself as an instance of a common business process management metamodel

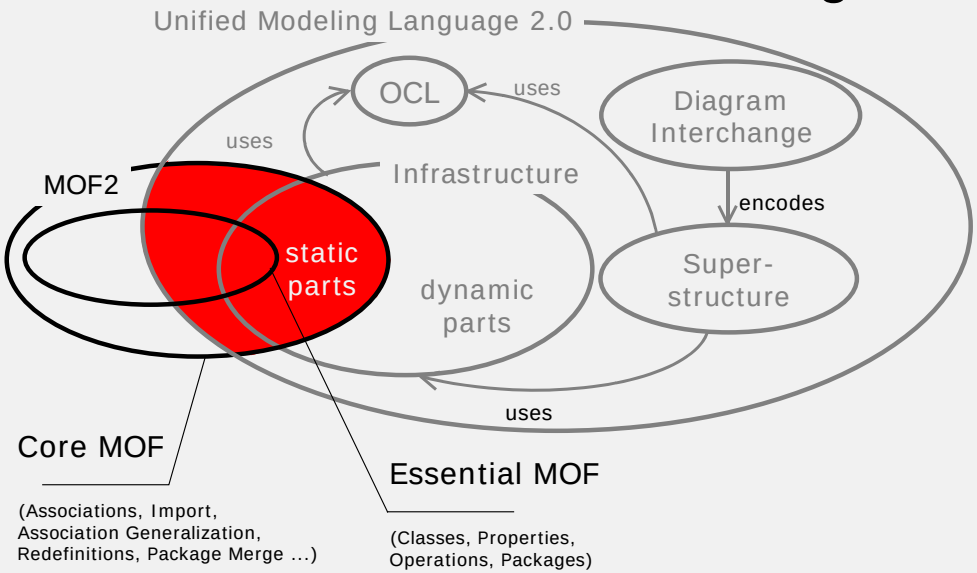
```
<active_instance type="process">
<characteristics>
    <characteristic>name</characteristic>
    <characteristic>id</characteristic>
</characteristics>
...
```

Background: Metamodeling



Background: Meta Object Facility

- Platform-independent metadata management foundation of the Model Driven Architecture
- MOF 2.0 unifies
  - MOF 1.4
  - XMI 1.2
  - XMI production principles for XML Schema
  - JMI 1.0
- Strongly influenced by UML 2.0's infrastructure (shared vision of reusing core modeling concepts)



Background: Extensible Markup Language

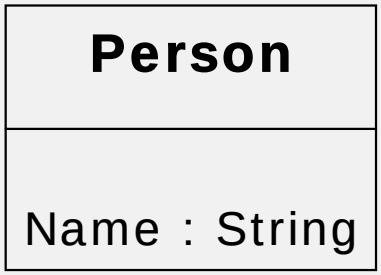
1. XML textually encodes structured data
2. XML looks similar to HTML
3. XML is text, but not meant for reading
4. XML provides a verbose textual serialization
5. XML is a family of techniques
6. XML is a modular approach
7. XML is well-introduces, but conceptually not that new
8. XML transfers HTML to XHTML
9. XML forms the basis of RDF and the Semantic Web
10. XML is a vendor and platform neutral standard

Turing Models into Languages: XML Technologies Involved

- XML Core Standard (XML)
  - Provides encoding of (meta) data
    ```
    <element attribute="value">value</element>
    ```
- XML Schema (XSD)
  - Provides structural rules for encoding schemata describing data and metadata
    ```
    <element name="element" type="string"/>
    ```
- XML Linking (XLink)
  - Provides mechanisms for general purpose links among XML entities
    ```
    <element xlink:href="#destination"/>
    ```

Turing Models into Languages: Producing XML Schemata and Documents

- **EMOF Element**
  Object (from the viewpoint of MOF) with a primitive typed property



- **XML Serialization**
  - XML elements
  - XML attributes

- **XML Serialization of an model instance**

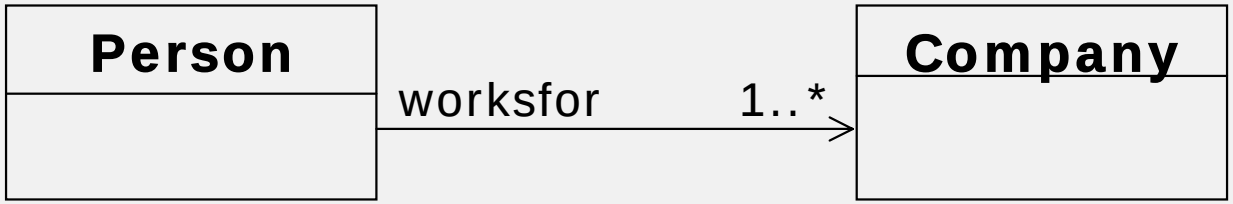
```
<tns:Person xmi:id="d10..." name="John Doe"/>
```

- **XML Serialization of the metamodel**

```
<emof:Class xmi:id="f31..." name="Person">
    <ownedAttribute xmi:id="e56..." name="Name"
                    type="String"/>
</emof:Class>
```

Turing Models into Languages: Producing XML Schemata and Documents

- **EMOF Element**
  Object (from the viewpoint of MOF) with a object-typed property



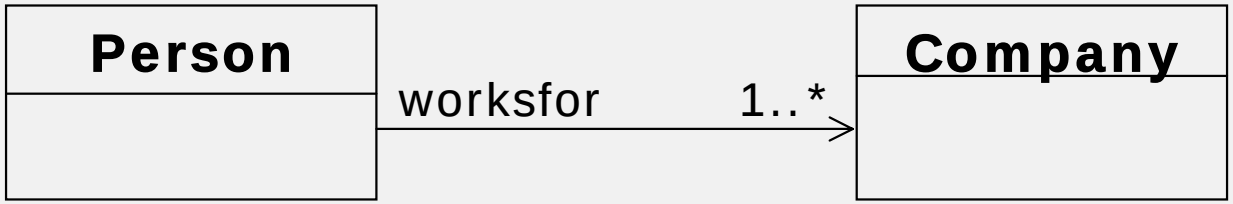
- **XML Serialization**
  - Reference attribute
  - Nested reference element using `href`
  - Nested XML element (is `aggregationKind` is `composite`)

- **XML Serialization of the metamodel**

```
<emof:Class xmi:id="a69..." name="Person">
     <ownedAttribute xmi:id="b72..." name="worksFor"
                     type="Company" lower="1" upper="*"/>
</emof:Class>
<emof:Class xmi:id="c82..." name="Company"/>
```

Turing Models into Languages: Producing XML Schemata and Documents

- **EMOF Element**
  Object (from the viewpoint of MOF) with a object-typed property



- **XML Serialization of an model instance**

```
<Person xmi:id="d49..." worksFor="f92..."/>
<Company xmi:id="f92..."/>
```

alternatively (using hyperlinks):

```
<Person xmi:id="d49...">
     <worksFor xlink:href="#f92..."/>
</Person>
```

alterantively (if association where composite):

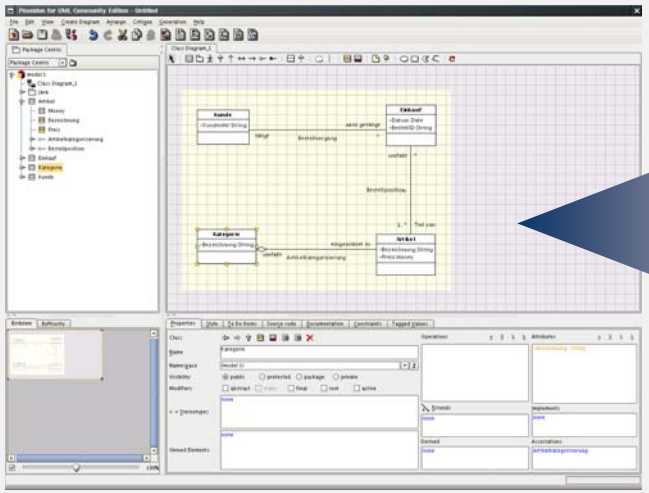
```
<Person xmi:id="d49...">
     <worksFor xmi:id="f92..."/>
</Person>
```

XMI Applications: (Meta)Data Interchange

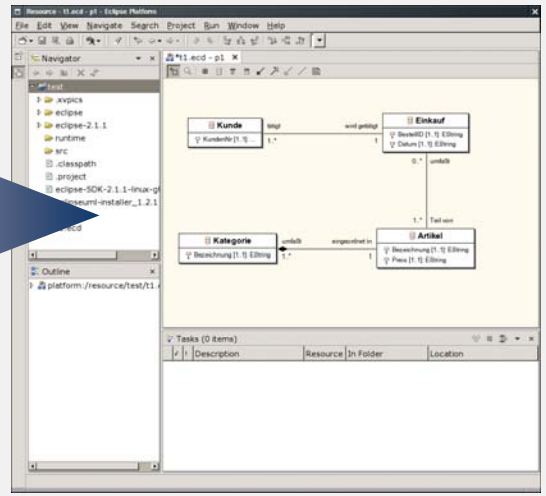
- XMI is currently to some extend successfully deployed for interchanging UML-based models (import and export facilities are provided by CASE tools)
- XMI is also used for transferring MOF-based metamodels (e.g., CWMI, SPEM ...)
- XMI schema and document production rules are implemented on a prototype basis
- Proof of concept: UML 2.0's and MOF 2.0's XMI serializations are produced by applying XMI's production rules

XMI Applications: Diagram Interchange

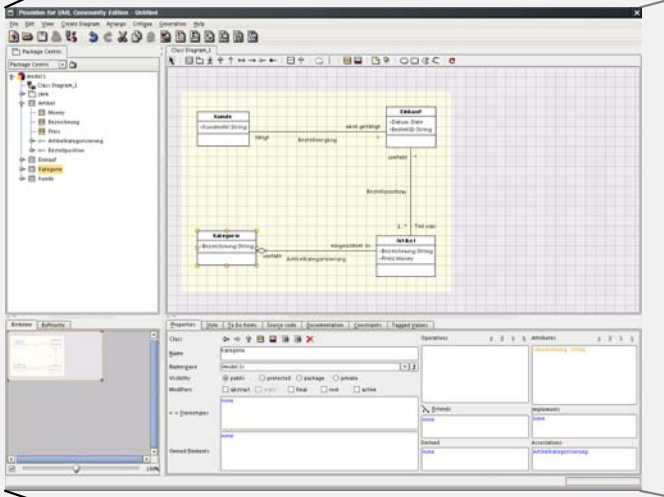
- UML 1.x's XMI serialization lacks support of encoding information on the graphic representation of a model
- UML 2.0 adds new package to the metamodel for encoding graphical information
- XMI/XML encoded information may be processed further by XML general purpose tools such as XSLT



XMI
+
Diagram
Interchange



XMI Applications: Diagram Interchange



```
<XMI xmi.version = '1.2' xmlns:UML = 'org.omg.xmi.namespace.UML' timestamp = 'Wed Dec 31 12:49:37 CET 2003'>
  <XMI.header>
    <XMI.documentation>
      <XMI.exporter>Netbeans XMI Writer</XMI.exporter>
      <XMI.exporterVersion>1.0</XMI.exporterVersion>
    </XMI.documentation>
  </XMI.header>
  <XMI.content>
    <UML:Model xmi.id = 'lsm:aa10fc:f9cbbf26cb:-7ffa' name = 'model 1' isSpecification = 'false'
      isRoot = 'false' isLeaf = 'false' isAbstract = 'false'>
```

"classical" model data

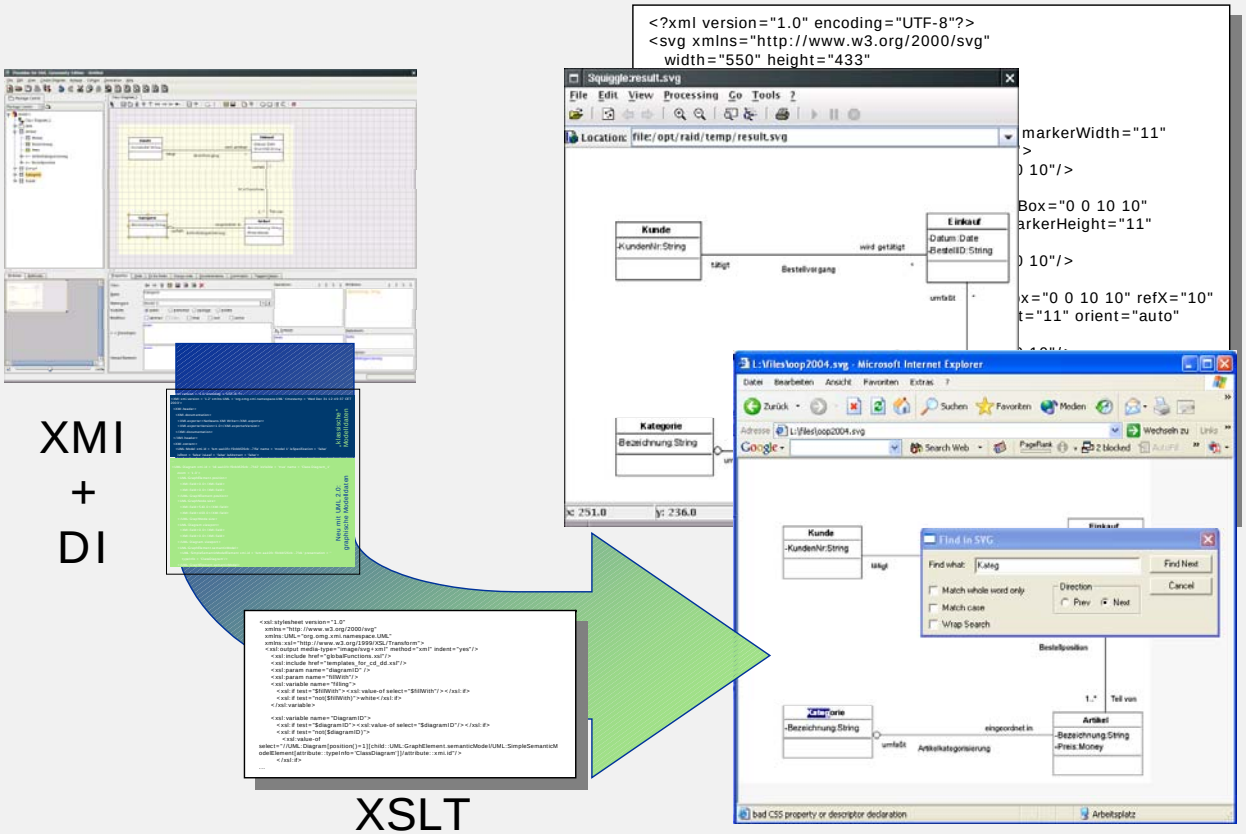
```
...
<UML:Diagram xmi.id = 'ldi:aa10fc:f9cbbf26cb:-7fd2' isVisible = 'true' name = 'Class Diagram_1'
    zoom = '1.0'>
    <UML:GraphElement.position>
      <XMI.field>0.0</XMI.field>
      <XMI.field>0.0</XMI.field>
    </UML:GraphElement.position>
    <UML:GraphNode.size>
      <XMI.field>540.0</XMI.field>
      <XMI.field>403.0</XMI.field>
    </UML:GraphNode.size>
    <UML:Diagram.viewport>
      <XMI.field>0.0</XMI.field>
      <XMI.field>0.0</XMI.field>
    </UML:Diagram.viewport>
    <UML:GraphElement.semanticModel>
      <UML:SimpleSemanticModelElement xmi.id = 'lsm:aa10fc:f9cbbf26cb:-7fd1' presentation = ''
        typeInfo = 'ClassDiagram'/>
    </UML:GraphElement.semanticModel>
...
```

UML 2.0 adds data describing graphicsl representation

- UML 2.0 adds new package to the metamodel for encoding graphical information

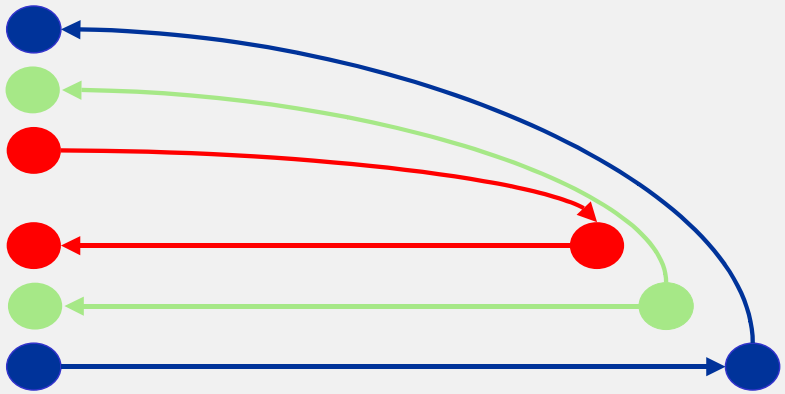
XMI Applications: Diagram Interchange

- Prototype implemented by FHF
- Generating vector-based graphical representation of UML model data using XSLT to transfer XMI into SVG



XMI Applications: Lessons Learned

- **XMI's Inherent Heterogeneity Problem**
  Flexible serialization of tree structures from arbitrary nets may lead to incompatibility if not implemented properly



Person
|
Project
|
Employee

Employee
|
Project
|
Person

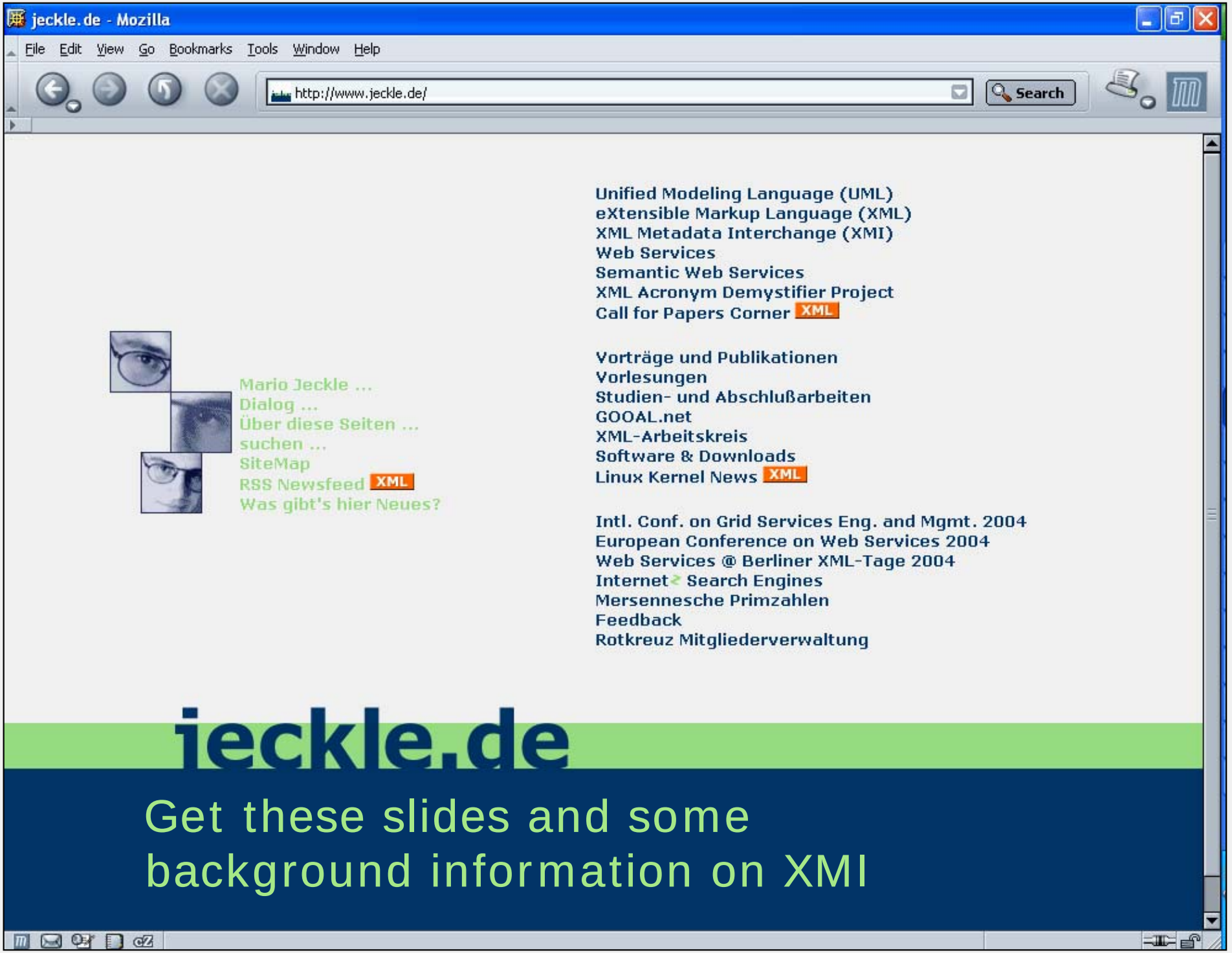
Project

Person Employee

XMI Applications: Lessons Learned

- **XMI's Schema Production Rules**
  Well-working as proved by prototype implementations but currently not widely accepted by the marked

- **Interchange of Visual Model Data**
  - Initially highly desired by the market
  - Now standardized
  - Some unsolved issues
  - Currently only one implementation available (Poseidon)


jeckle.de - Mozilla
File Edit View Go Bookmarks Tools Window Help
http://www.jeckle.de/
Search
Unified Modeling Language (UML)
eXtensible Markup Language (XML)
XML Metadata Interchange (XMI)
Web Services
Semantic Web Services
XML Acronym Demystifier Project
Call for Papers Corner XML
Vorträge und Publikationen
Vorlesungen
Studien- und Abschlußarbeiten
GOOAL.net
XML-Arbeitskreis
Software & Downloads
Linux Kernel News XML
Intl. Conf. on Grid Services Eng. and Mgmt. 2004
European Conference on Web Services 2004
Web Services @ Berliner XML-Tage 2004
Internet² Search Engines
Mersennesche Primzahlen
Feedback
Rotkreuz Mitgliederverwaltung
Mario Jeckle ...
Dialog ...
Über diese Seiten ...
suchen ...
SiteMap
RSS Newsfeed XML
Was gibt's hier Neues?
jeckle.de
Get these slides and some
background information on XMI