An Open Modeling Infrastructure

Olaf Kath
Marc Born

FOKUS
Fraunhofer Institut Offene Kommunikationssysteme

IKV++ TECHNOLOGIES AG

Agenda

- Architecture Guidelines
- Users View
- Medini Toolset
- Model Transformations
- Conclusions

Our Vision on MDA:
A Modeling Infrastructure

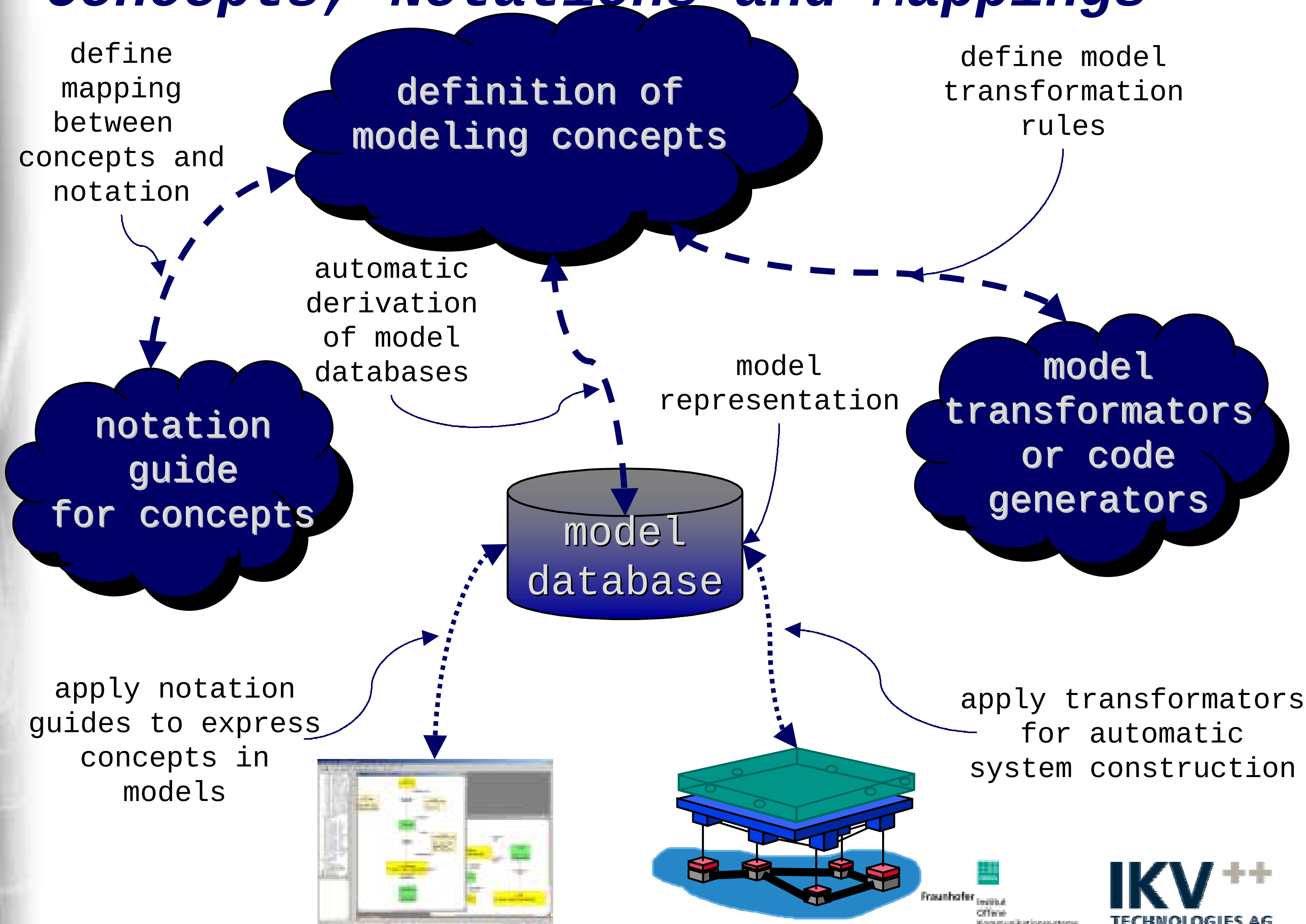
open to a variety of modeling and model transformation techniques

and open to different domains

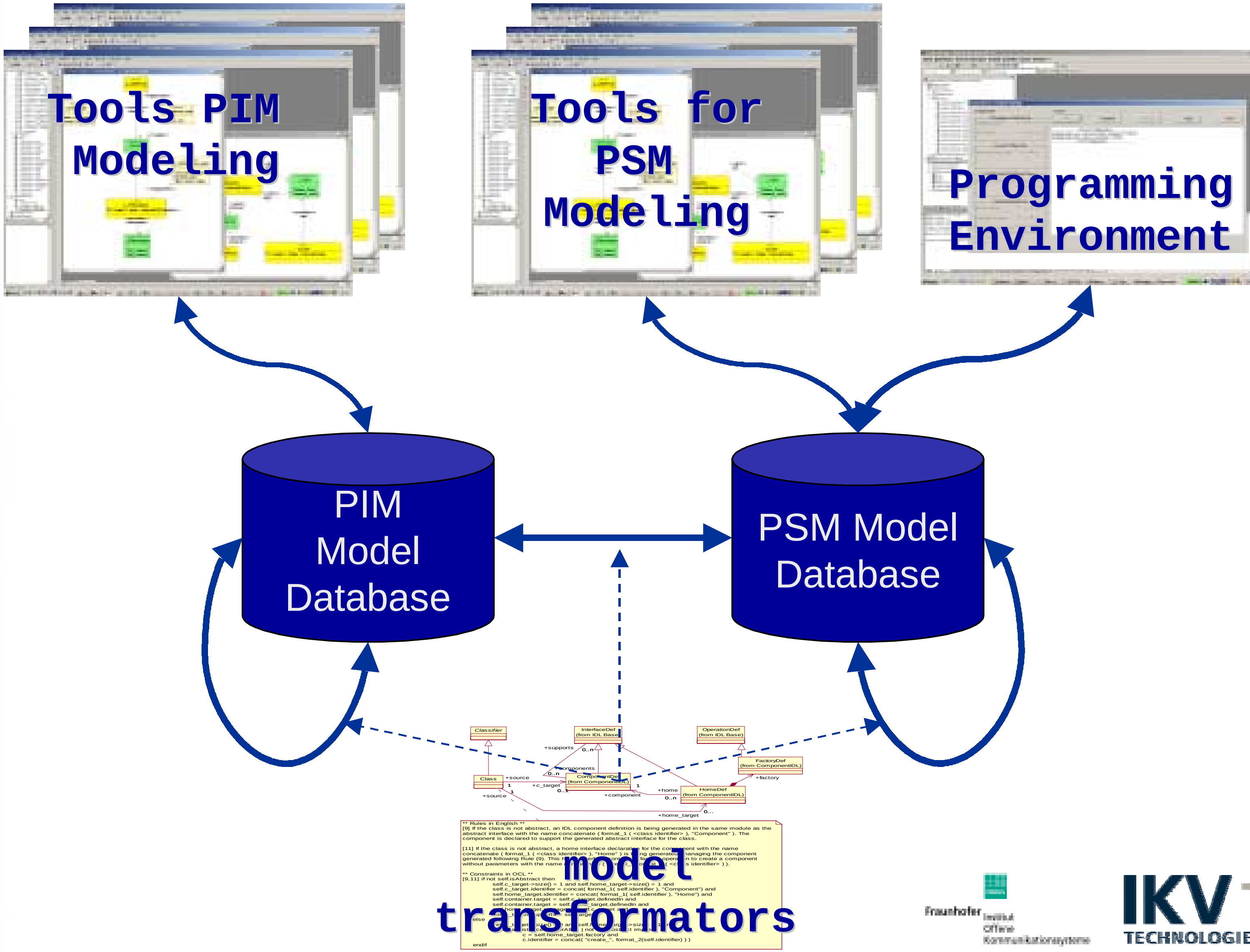
Architecture Guidelines

- All modeling instruments and principles are defined by models themselves, model databases are derived
  - We use the Meta Object Facility (MOF) for this purpose
- Based on these metamodels, domain specific visual modeling languages are defined
  - We use different techniques here, e.g. UML based languages, textual languages, new graphical languages
- Also based on the metamodels, further processes like code generation or model transformations are defined
  - We keep notation and these processes separate

Architecture Guidelines: Concepts, Notations and Mappings



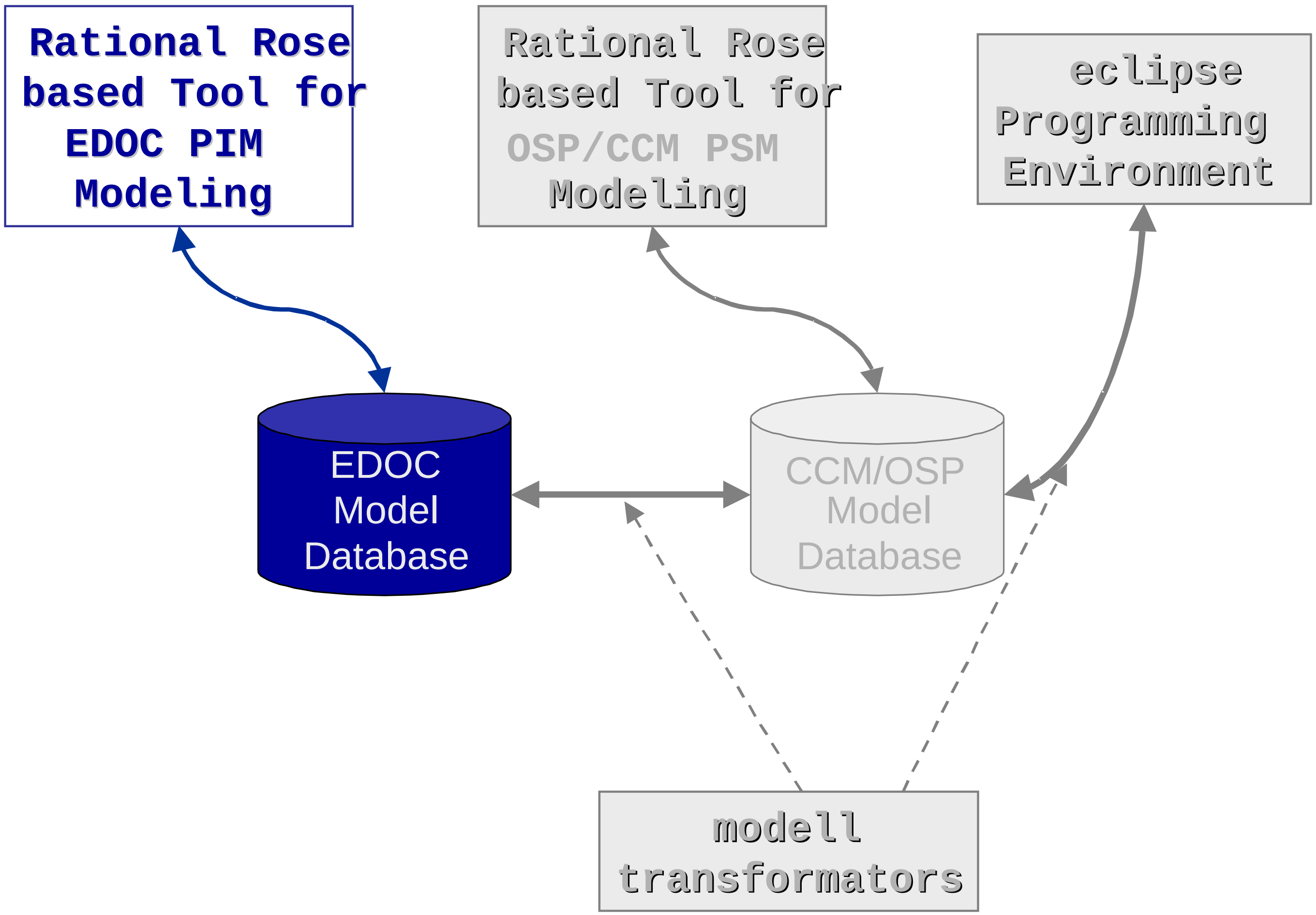
The Big Picture



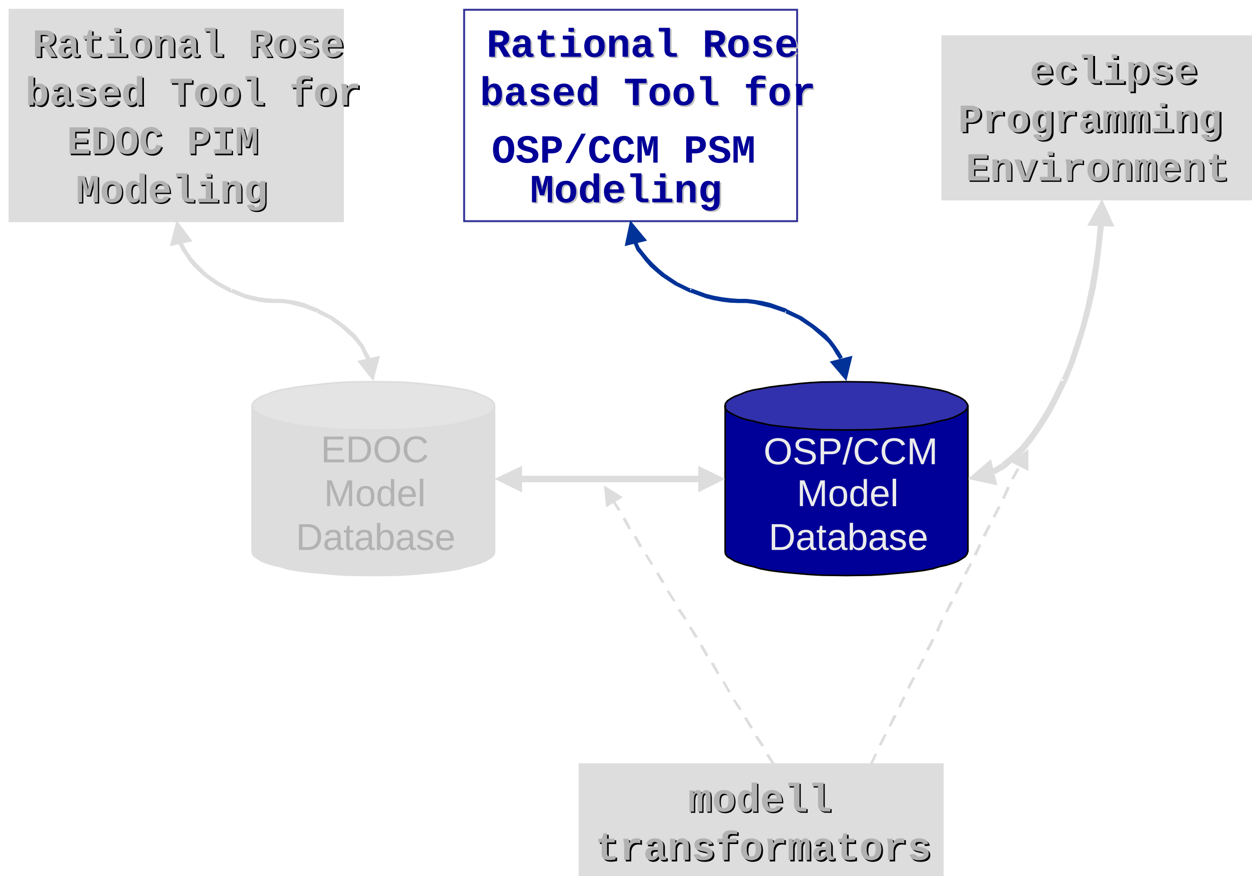
Fraunhofer Institut Offene Kommunikationssysteme

IKV++ TECHNOLOGIES AG

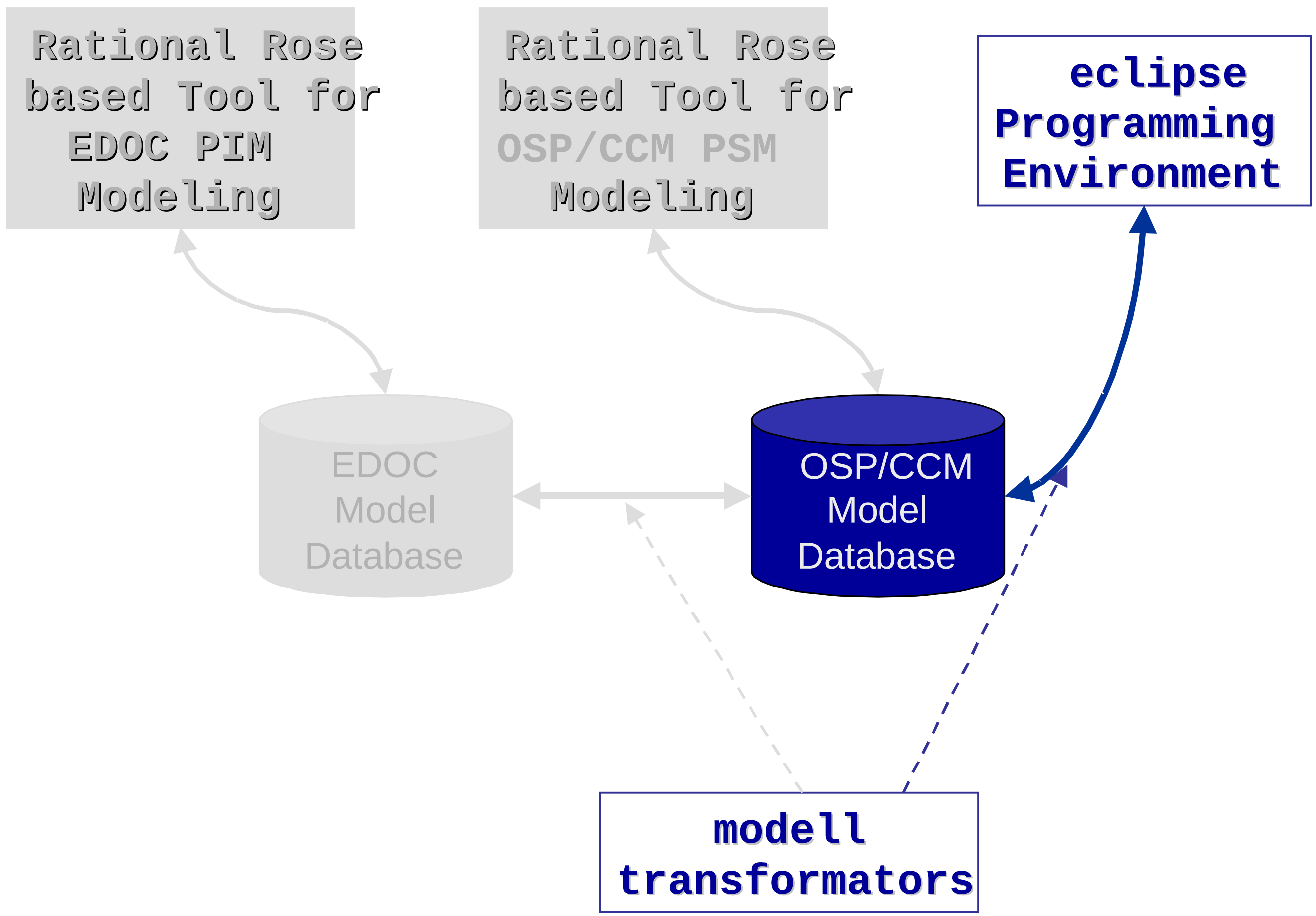
Users View:
PIM Modeling using EDOC



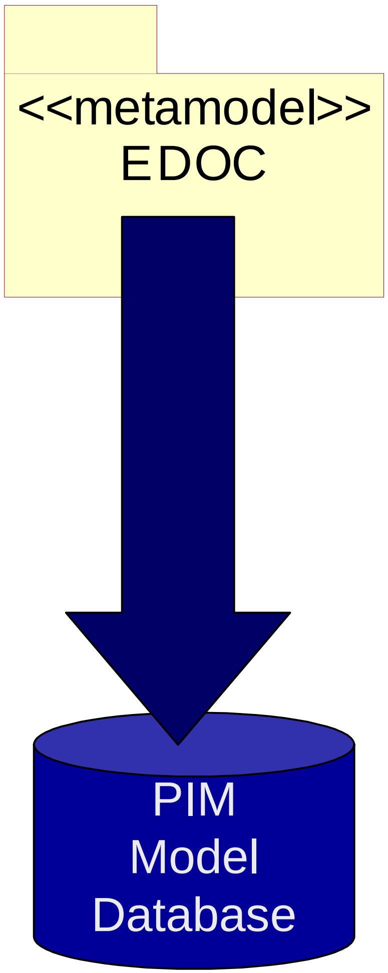
Users View:
PSM Refinement using Modeling Tool



Users View: Generation of Java Code out of the PSM

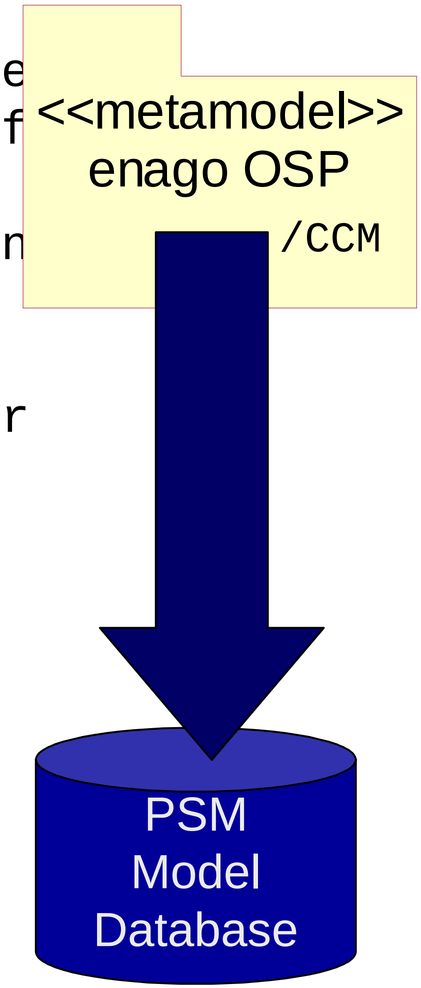


Tools supporting the Modeling Infrastructure

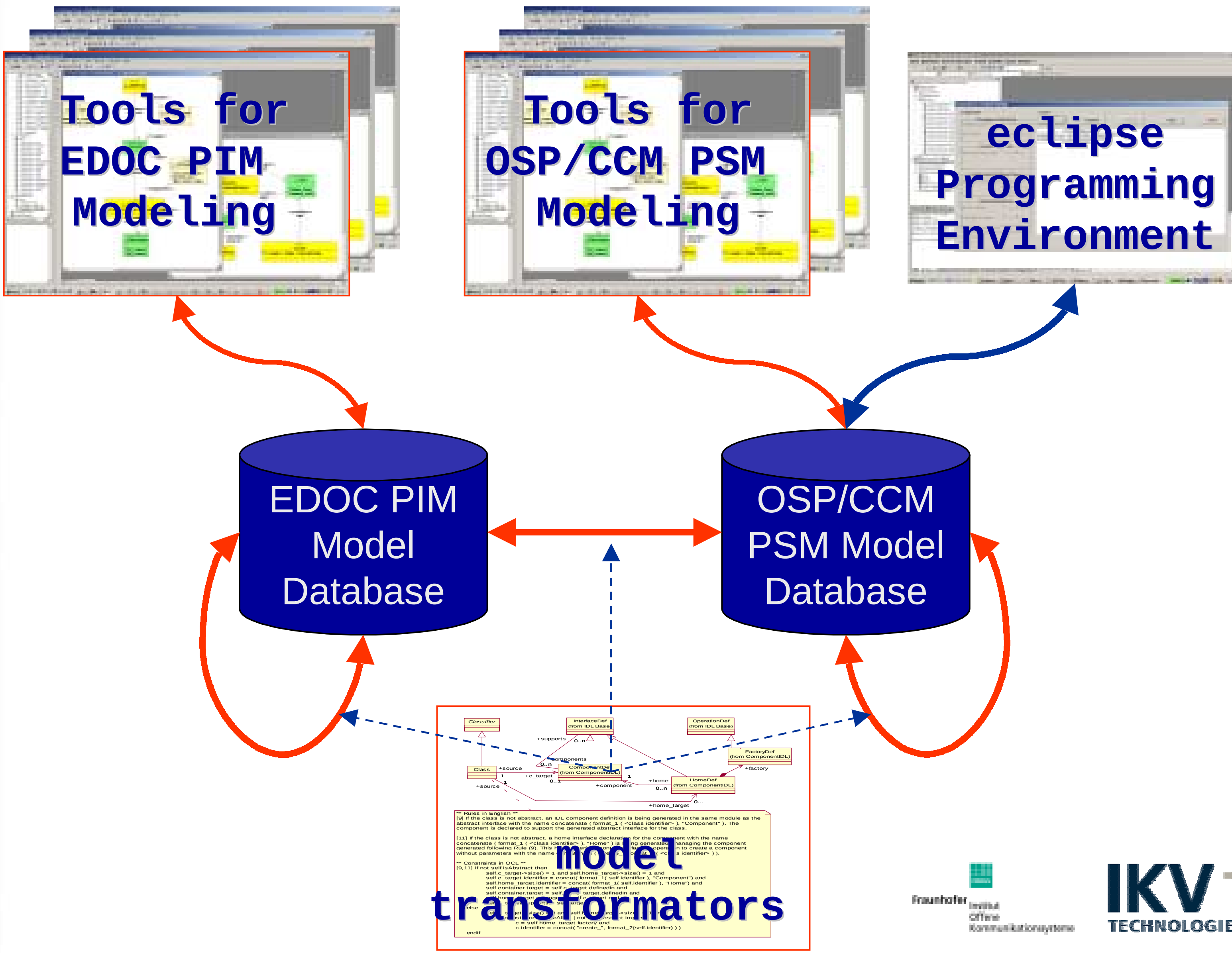


Automatically generate
model databases out of
the metamodels, using
IKVs medini tool chain

These model databases
support mechanisms for
- transactions
- model change notifications
- XML import/export
- ...



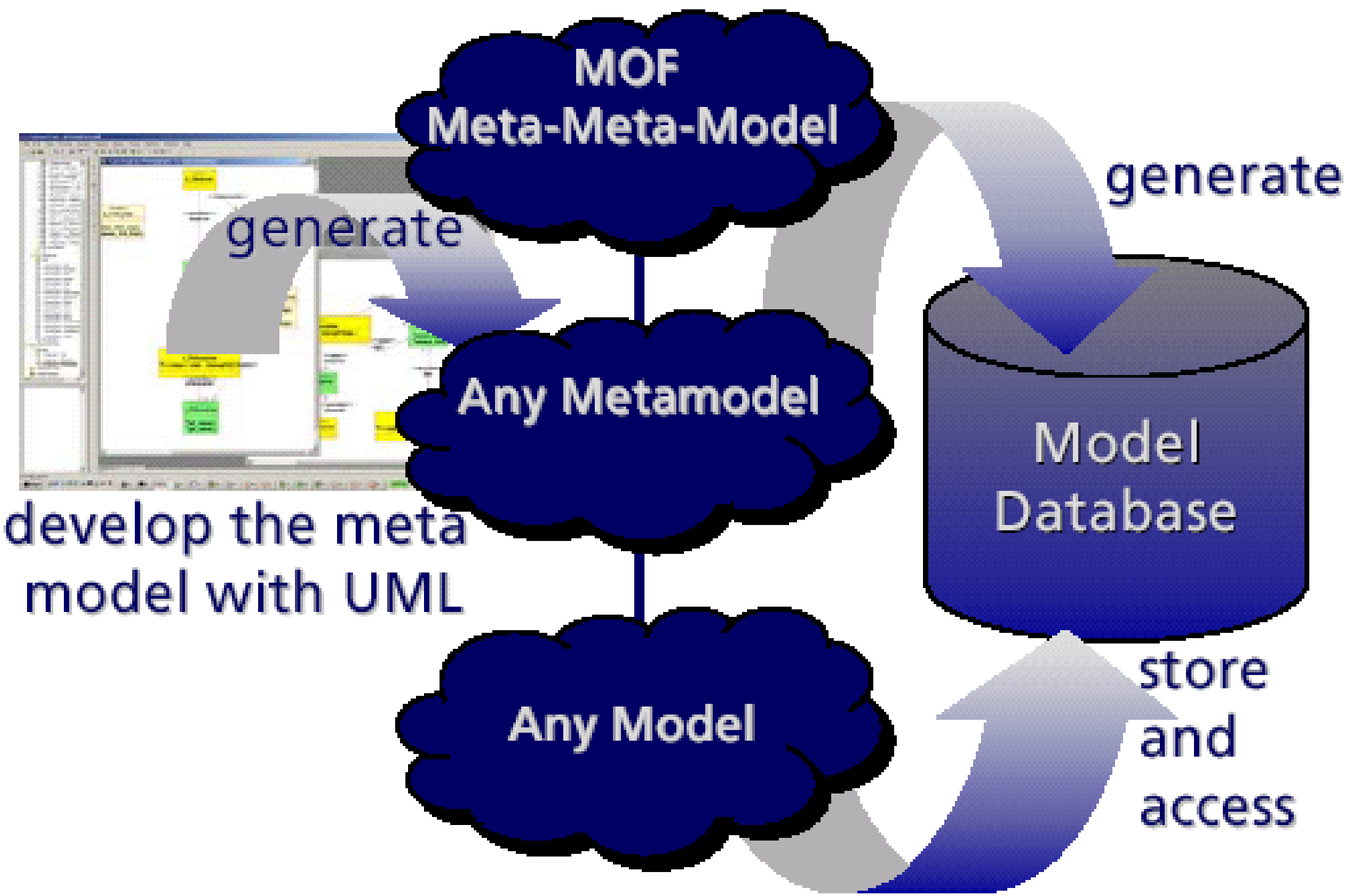
Applying the Medini tool set



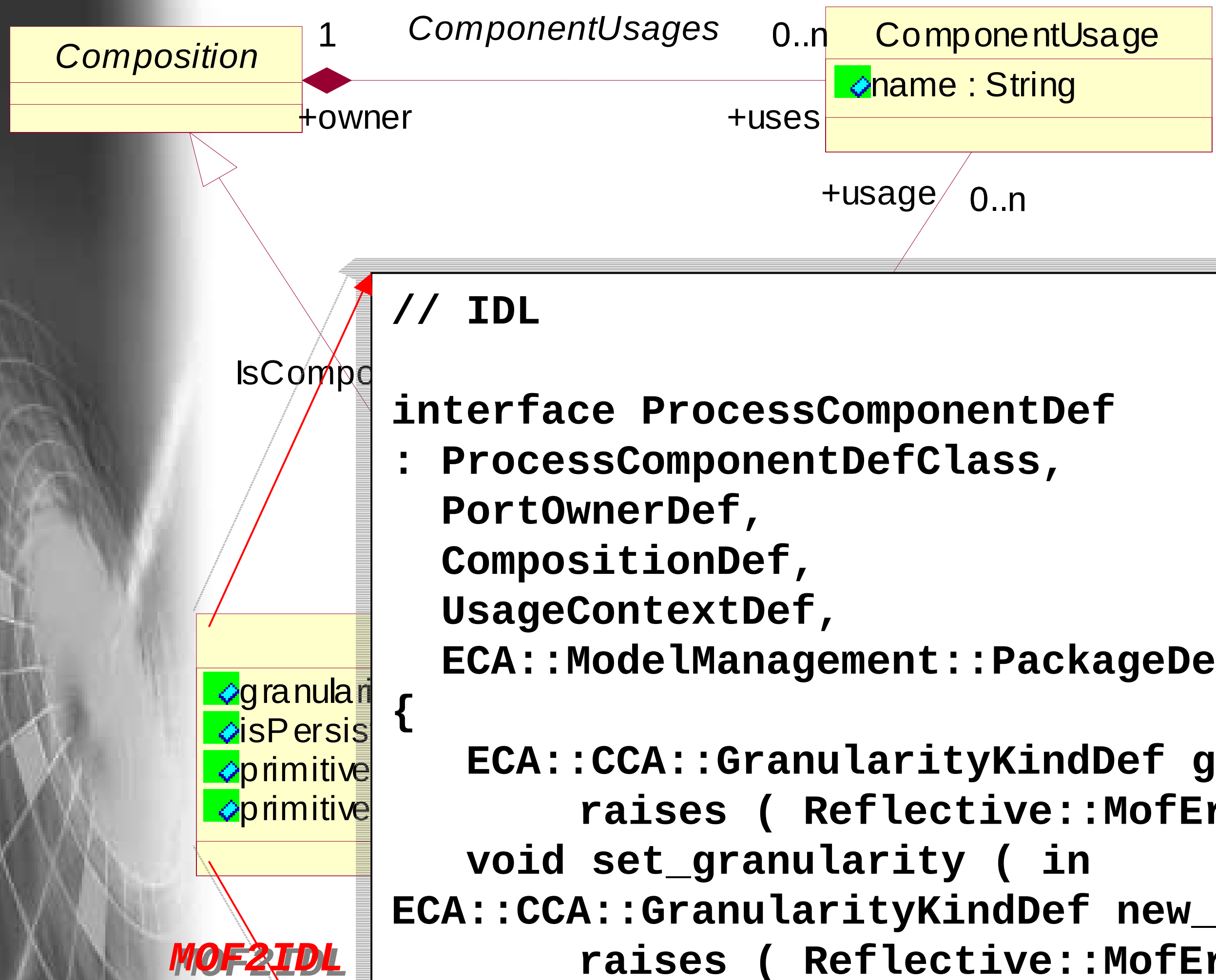
Fraunhofer Institut Offene Kommunikationssysteme

IKV++ TECHNOLOGIES AG

Infrastructure: Generation of Repositories



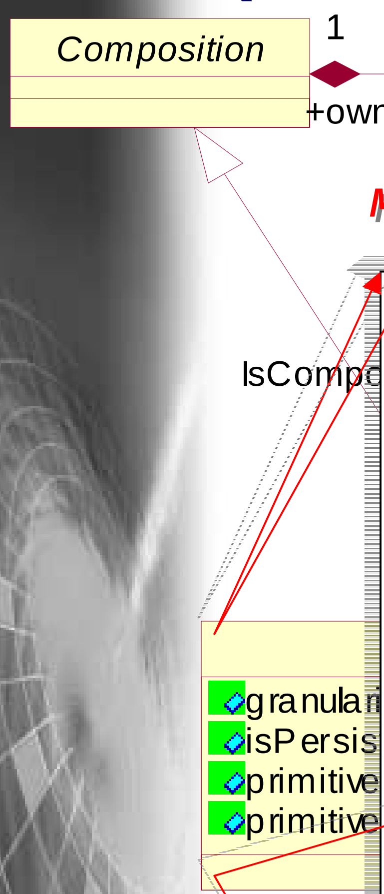
Generation of the EDOC Repository




```
// IDL

interface ProcessComponentDef
: ProcessComponentDefClass,
  PortOwnerDef,
  CompositionDef,
  UsageContextDef,
  ECA::ModelManagement::PackageDef
{
   ECA::CCA::GranularityKindDef granularity ()
        raises ( Reflective::MofError );
   void set_granularity ( in
ECA::CCA::GranularityKindDef new_value)
        raises ( Reflective::MofError );
   boolean is_persistent ( )
        raises ( Reflective::MofError );
// ...
};
```

Generation of the EDOC Repository (cont.)




```
// IDL

interfa
: Proce
   PortO
   Compo
   Usage
   ECA::
{
   ECA::CCA::
      raises ( Reflective::MofError );
   void set_granularity ( in ECA::CCA::GranularityKindDef
      raises ( Reflective::MofError );
void set_is_persistent ( in boolean new_value)
      raises ( Reflective::MofError );
   // ...
};
```

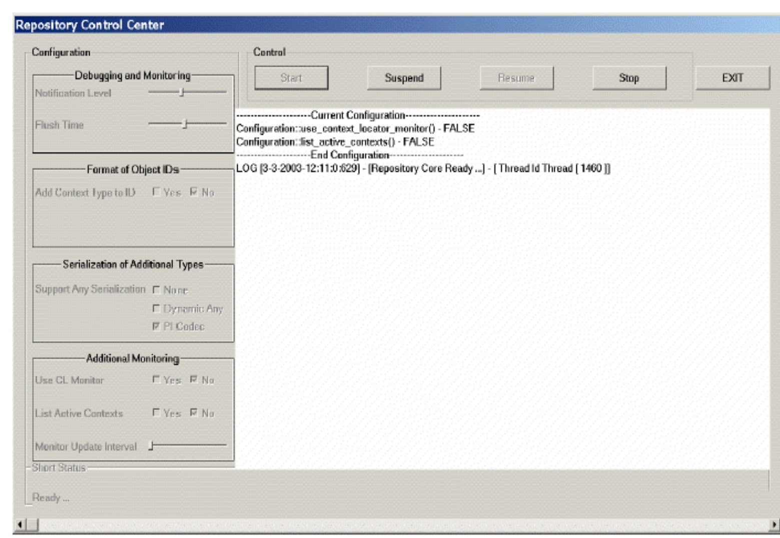
```
/* C++ */

class ProcessComponentDefServant
: virtual public
ProcessComponentDefClassServant,
 virtual public PortOwnerDefServant,
 virtual public CompositionDefServant,
 virtual public UsageContextDefServant,
 virtual public
ModelManagement::PackageDefServant
{
public:
   virtual ECA::CCA::GranularityKindDef_Ptr
granularity ();
   virtual bool is_persistent ( );
   virtual void set_is_persistent (bool
new_value);
// ...
};
```

EDOC Model Repository

- Repositories are generated with:
  - MOF2IDL Generator
  - MOF2C++ Generator

- EDOC repository implements the generated CORBA IDL interfaces
- ***Repository Control Center*** provides
  - Debugging and Monitoring capabilities
  - Serialization to XML
  - In the future: Control over Medini extensions



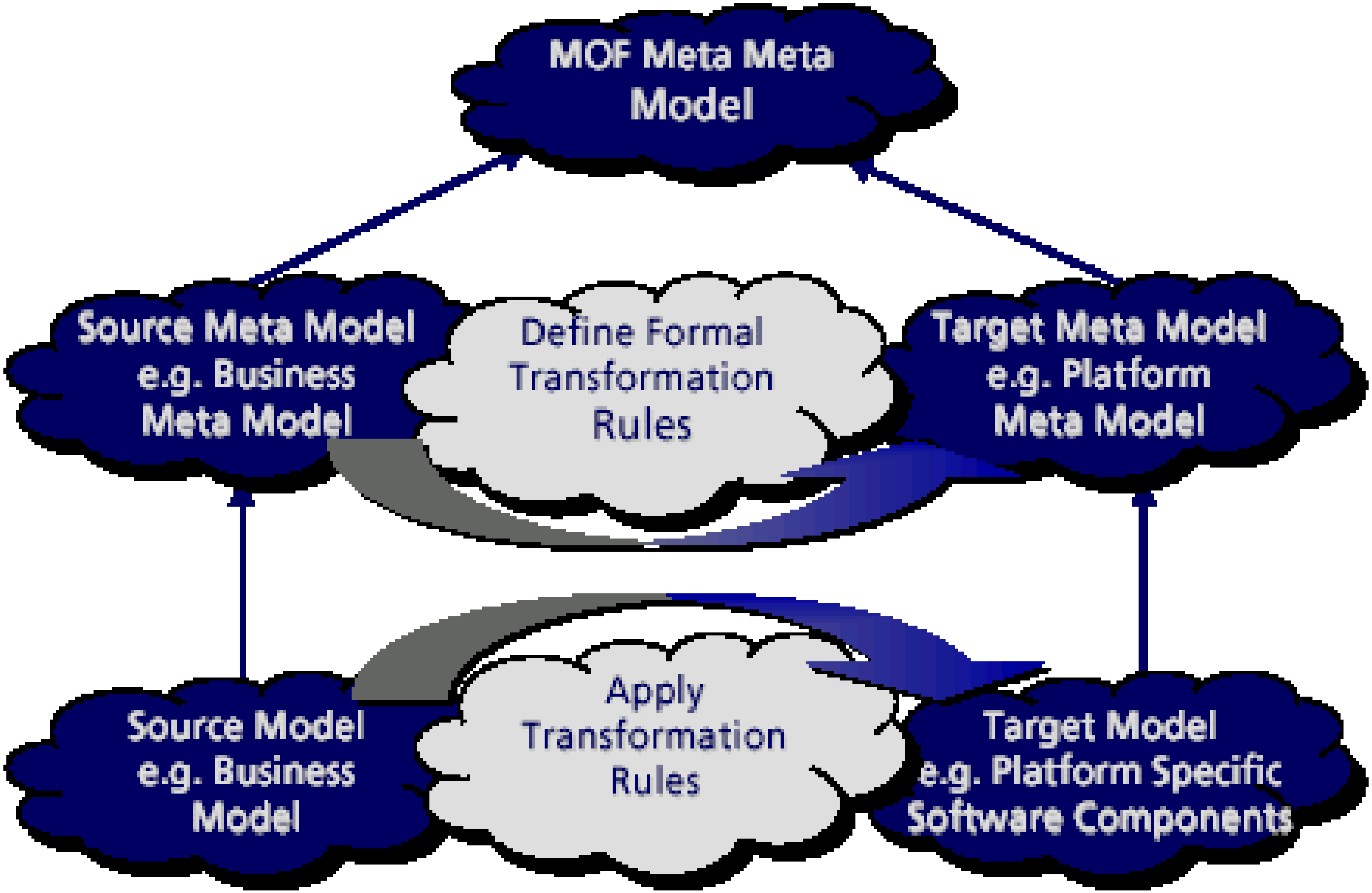
Medini Extensions to repositories

- Repository implements the generated CORBA IDL interfaces plus:
  - Transaction capabilities
  - Persistency
  - Medini Extensions under development
    - ***Consistency Management*** through active communication of model changes as Notifications
      - *Medini ActiveRepository Service* (MARS)
    - ***Validity Maintenance:*** repository checks meta-model constraints at run-time

Fraunhofer Institut Offene Kommunikationssysteme

IKV++ TECHNOLOGIES AG

Model Transformations



Fraunhofer Institut Offene Kommunikationssysteme

IKV++ TECHNOLOGIES AG

Transformations from EDOC to OSP/CCM

- Transformations are defined between meta-models
  - EDOC meta-model <> enago OSP/CCM meta-model
  - They specify, how *any* EDOC compliant model (i.e. instances of the EDOC meta-model) is transformed to the OSP/CCM platform
- Transformations are by itself instances of a (transformation-)meta-model
  - this meta-model determines, how to define the model-transformations
  - currently no OMG standard exists (initial submission deadline has just passed)

Approaches on transformations

- ***Rule based transformations***
  - Specifiy *how* to construct a target from a source model
  - Example language: *UML ActionSemantics*

- ***Declarative transformations***
  - Specify only the *result* of a transformation (i.e. as invariants and postconditions)
  - Example language: *Object Constraint Language (OCL)*
- Other approaches use templates or scripting/semiformal languages

Approaches on transformations (cont.)

- ***Rule based transformations***
  - Specifiy *how* to construct a target from a source model
  - Example language: *UML ActionSemantics*

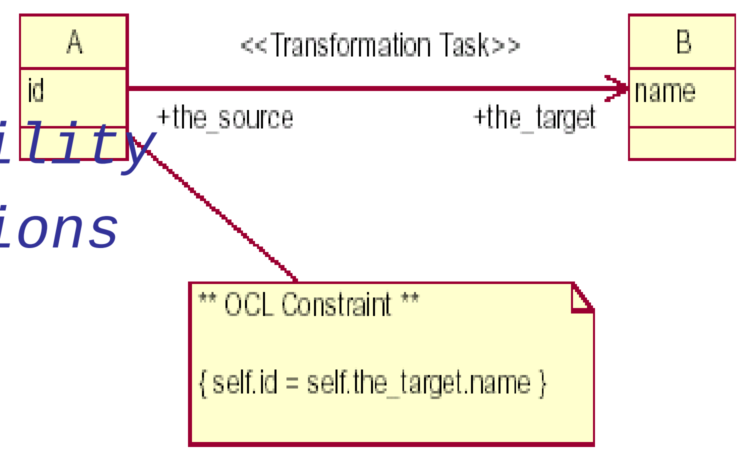
- *Advantages:*
  - *Significant and intuitive formalism*
  - *Simple implementation derivation*
- *Disadvantages:*
  - *Lack of (automatic) testability*
  - *Rules are only „one-way“*
    - *changes late in the software process can not be propagated back*

Fraunhofer Institut Offene Kommunikationssysteme

IKV++ TECHNOLOGIES AG

Approaches on transformations (cont.)

- ***Declarative transformations***
  - Specify only the *result* of a transformation (i.e. as invariants and postconditions)
  - Example language: *Object Constraint Language*



- *Advantages:*
  - *Testability & Traceability*
  - *„Two-way“ transformations*
- *Disadvantages:*
  - *Complex formalism (error-prone)*
  - *Implementation more difficult*

Our Approach for Transformations

- Three steps towards a ***MOF Transformator Generator*** (MTG):
  1. Specification and development of a Transformator Core Generator (”Skeleton Generator”)
     - independent of any transformation concepts
     - produces meta-model specific transformators (for EDOC and OSP)
  2. Development of an EDOC to OSP/CCM transformator based on generated skeleton
     - Parallel implementation of code generators
     - Realization of traceability, testability and parameterization (or interaction)

The MOF Transformator Generator (MTG)

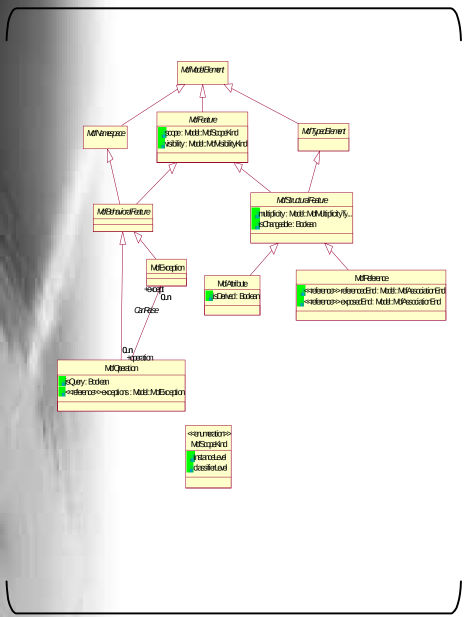
- Specification and development of a Transformator Core Generator (*"**Skeleton Generator**"*)
  - *First part of the whole Generator*
  - *Input:*
    - *Source (EDOC) or target (OSP/CCM) meta-models*
  - *Output:*
    - *meta-model specific Skeletons („Walker") for model-transformations (EDOC) or code-generation (OSP/CCM)*
  - **Status:** The *Skeleton Generator* produces already utilizable transformators for half-automated development

Fraunhofer Institut Offene Kommunikationssysteme

IKV++ TECHNOLOGIES AG

Operation of the generated Transformator Skeleton

Skeleton „walks“ over source tree



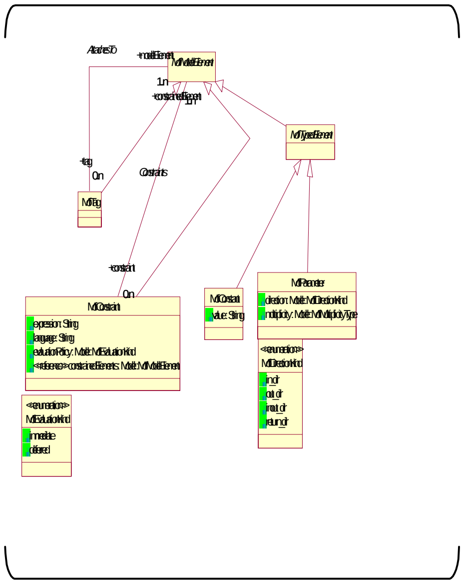
EDOC Source Model M1-level Instance (in repository)

transformation_procedure_1 ()

transformation_procedure_2 ()

transformation_procedure_3 ()

⋮



OSP/CCM Target Model M1-level instance (in repository)

Fraunhofer Institut Offene Kommunikationssysteme

IKV++ TECHNOLOGIES AG

Transformations: final step (vision)

- Replacement of hand-written code with evaluated transformation rules
  - *Rule based or declarative approach for the transformations (possibly a combination)*
  - *Gained experience from FhG FOKUS / IKV++ joint submission to the MOF2.0-IDL Request For Proposal and OMG process will influence the decision*

Advantages of our approach

- ***Platform independence*** through a formal specification of the transformations
- ***Testability***
  - rule and/or constraint based specification of transformations allow for validation and/or verification
- ***Traceability***
  - the trace of elements is easily extractable from transformation models
- Highest ***flexibility***, late determination of transformation model

Conclusion

- Medini tool set supports infrastructure
  - Integration of modeling tools, transformations
  - Extensions during the project needed
- Transformations will be one essential
  - Already realized part of the
  - Possibly combined approach of rule and constraint based specification in future

Fraunhofer Institut Offene Kommunikationssysteme

IKV++ TECHNOLOGIES AG