Persisting Model Data

Metamodel Instances or Human-Readable Textual Syntax?

Ericsson Modeling Days

2016 September 13



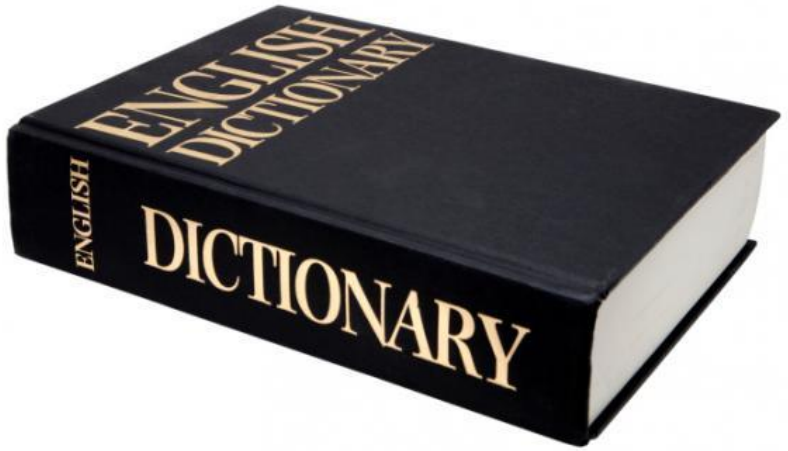
Outline

- Welcome and Introductions
- Definition of Terms
- MDA Flow
- Examples: Papyrus-RT and Papyrus-xtUML (BridgePoint)
- Underlying Persistence
- Advantages of Two Ways of Persisting
- Conclusions



Definition of Terms

- syntax
- semantics
- human-readable parsable textual concrete syntax
- grammar
- abstract syntax tree (AST)
- metamodel

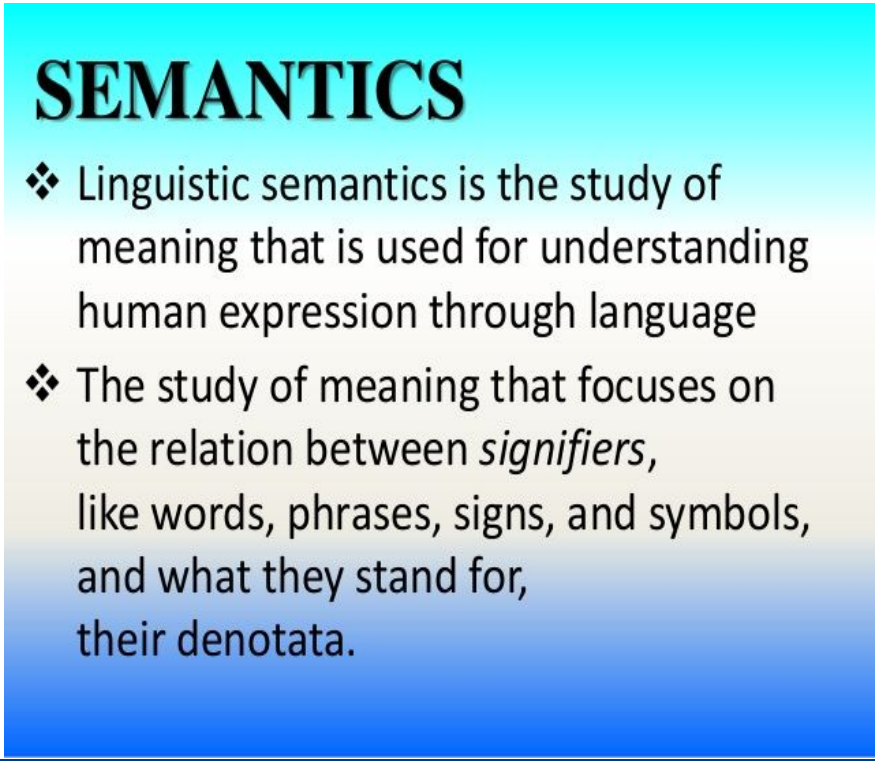


Definition of Terms

- **syntax -** physical shape and form of a language textual and/or graphical
- semantics
- human-readable parsable textual concrete syntax
- grammar
- abstract syntax tree (AST)
- metamodel

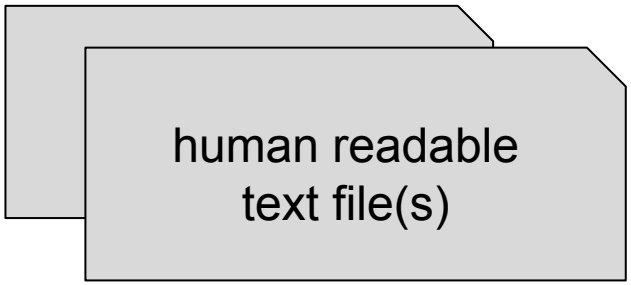
Definition of Terms

- syntax
- **semantics** - *meaning* encoded, stored and communicated in a language
- human-readable parsable textual concrete syntax
- grammar
- abstract syntax tree (AST)
- metamodel



Definition of Terms

- syntax
- semantics
- **human-readable parsable textual concrete syntax** - text-based, regularly formed language that can be viewed, read, modified and stored as text by humans and by machines. Examples include C, Java, Python and Action Language (e.g. Alf, OAL, MASL)
- grammar
- abstract syntax tree (AST)
- metamodel



Definition of Terms

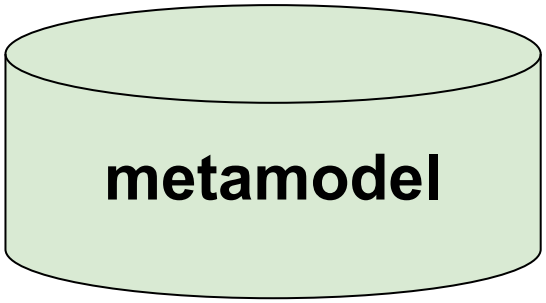
- syntax
- semantics
- human-readable parsable textual concrete syntax
- **grammar** - rules for parsing a textual language. A grammar mathematically encodes the rules that govern the syntax of a language. Tools that operate on grammars include antlr, Xtext, lex, yacc and bison.
- abstract syntax tree (AST)
- metamodel



Definition of Terms

- syntax
- semantics
- human-readable parsable textual concrete syntax
- grammar
- **AST (Abstract Syntax Tree)** - hierarchical structure capturing an intermediate form of a parsed language. Parsers parse text and store the interesting bits into a tree. The AST is more abstract (closer to the semantics) than the textual syntax but still largely influenced by the physical shape of the language.
- metamodel

Definition of Terms



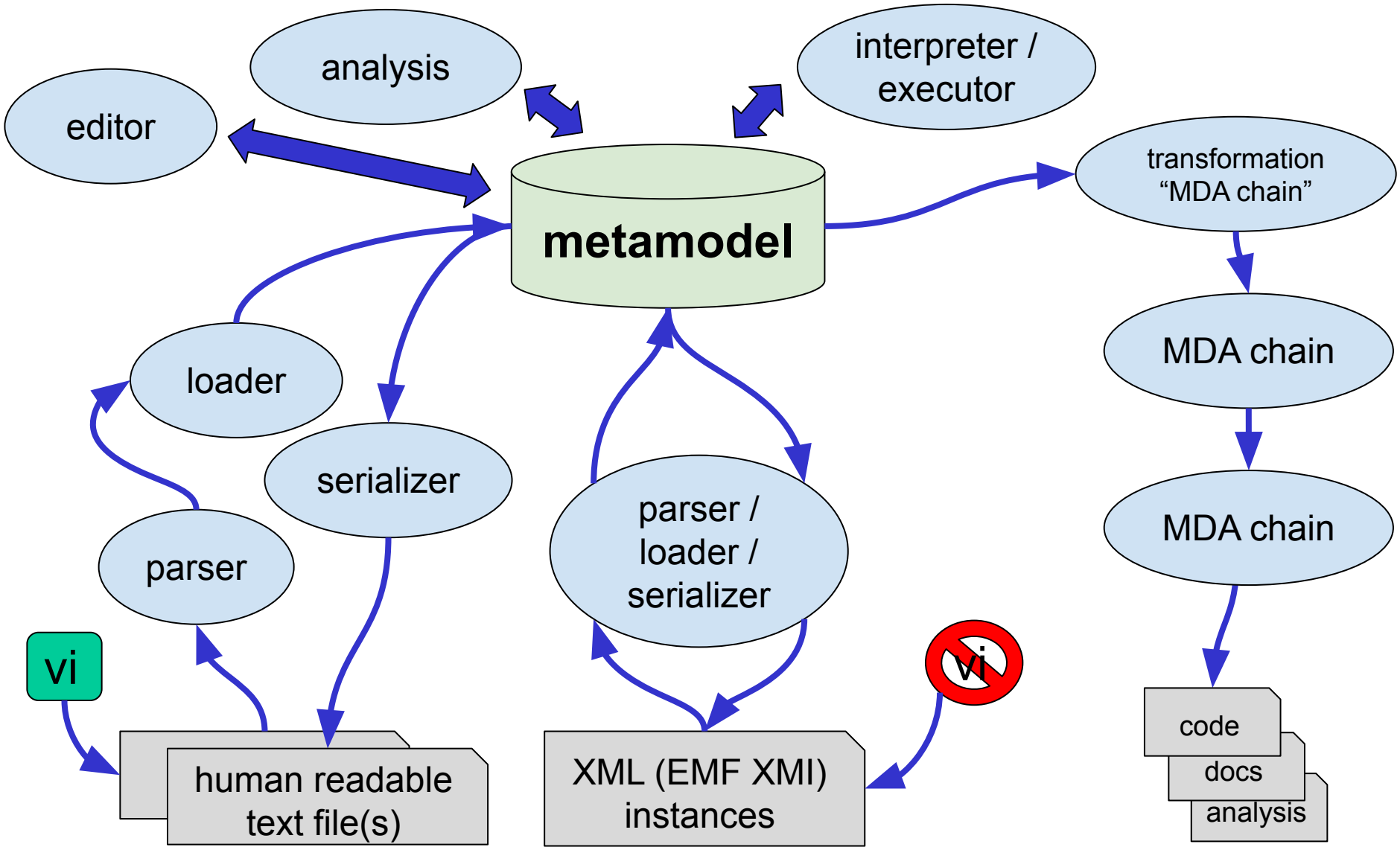
- syntax
- semantics
- human-readable parsable textual concrete syntax
- grammar
- abstract syntax tree (AST)
- **metamodel** - Model of the language loaded into memory and inter-related often as Java or Ecore objects. Examples include UML2, xtUML and UML-RT.

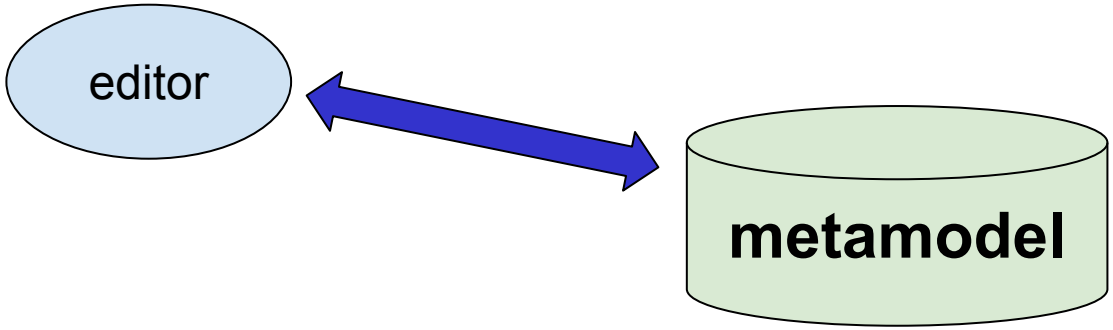
  AST ≠ metamodel

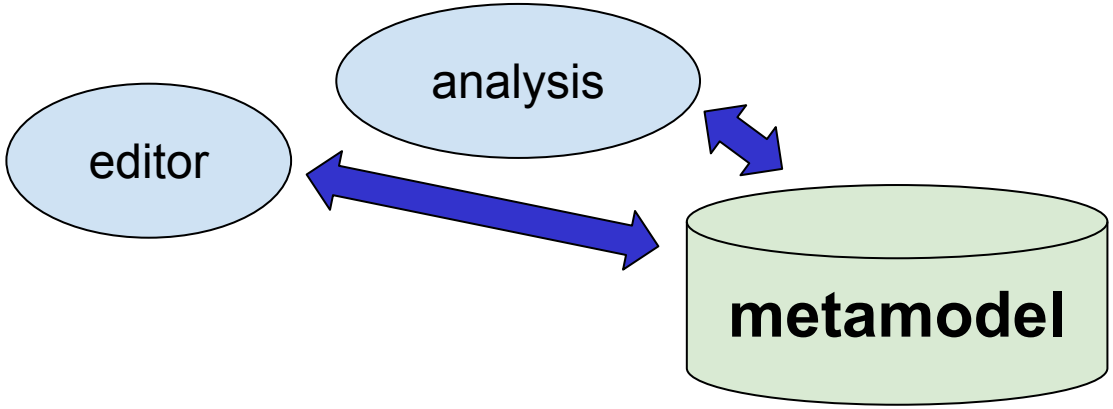


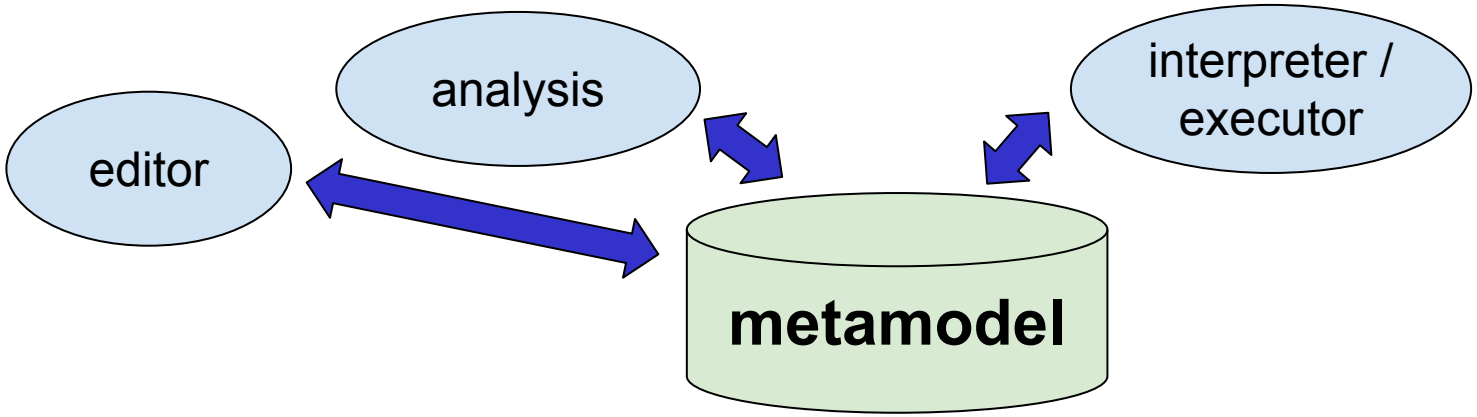


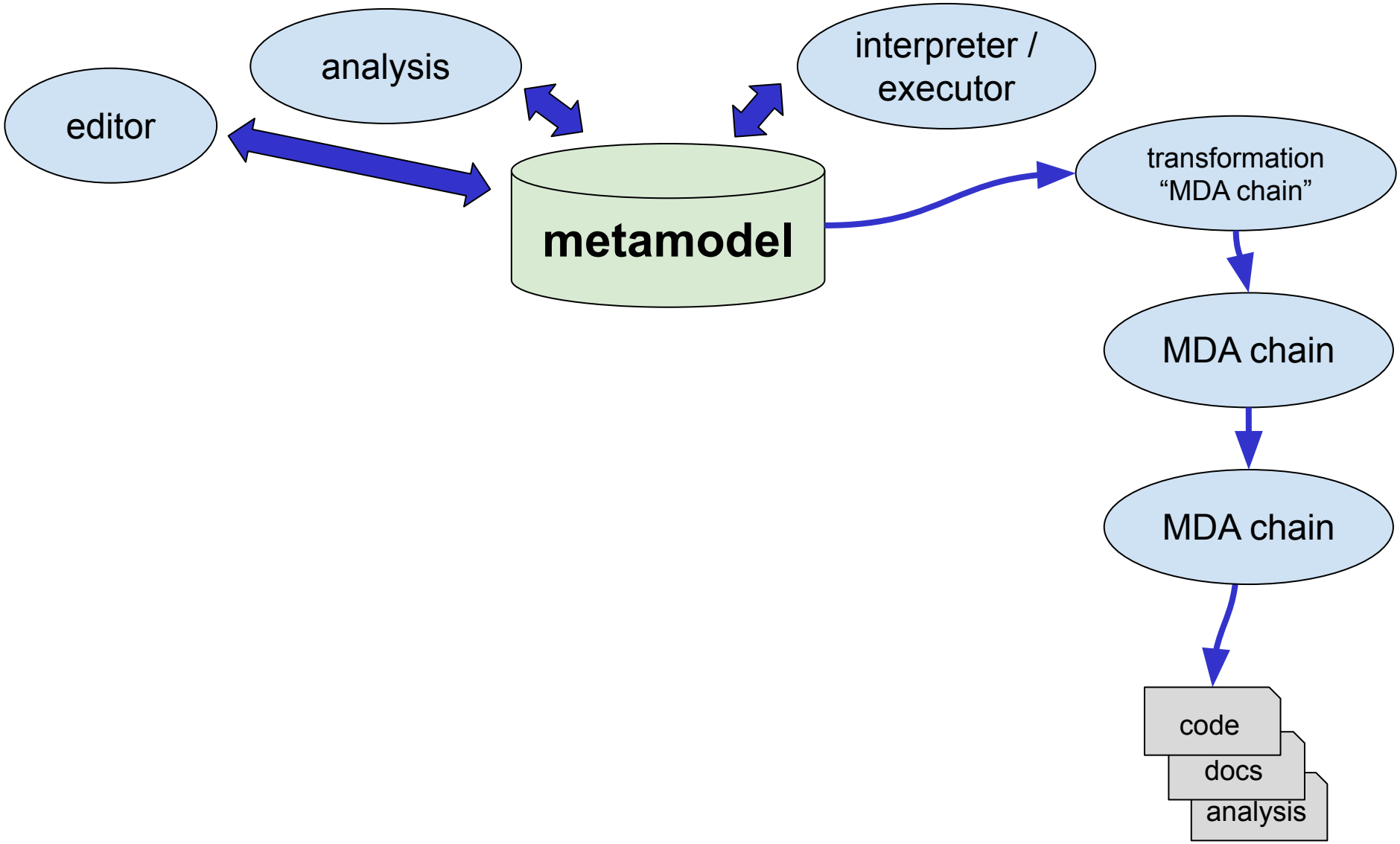


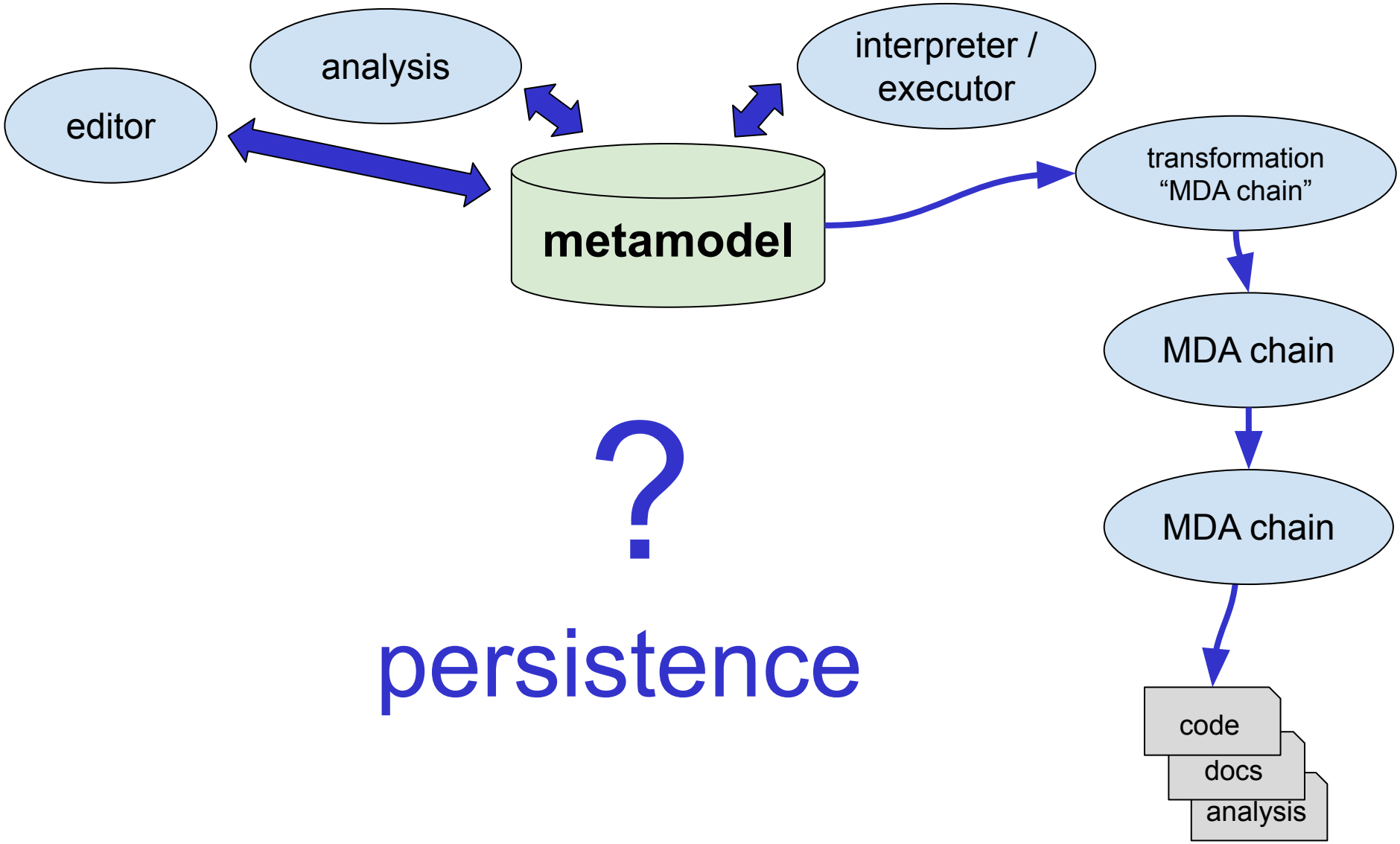

analysis
editor
interpreter / executor
metamodel
transformation "MDA chain"
loader
serializer
parser
parser / loader / serializer
MDA chain
MDA chain
vi
vi
human readable text file(s)
XML (EMF XMI) instances
code
docs
analysis

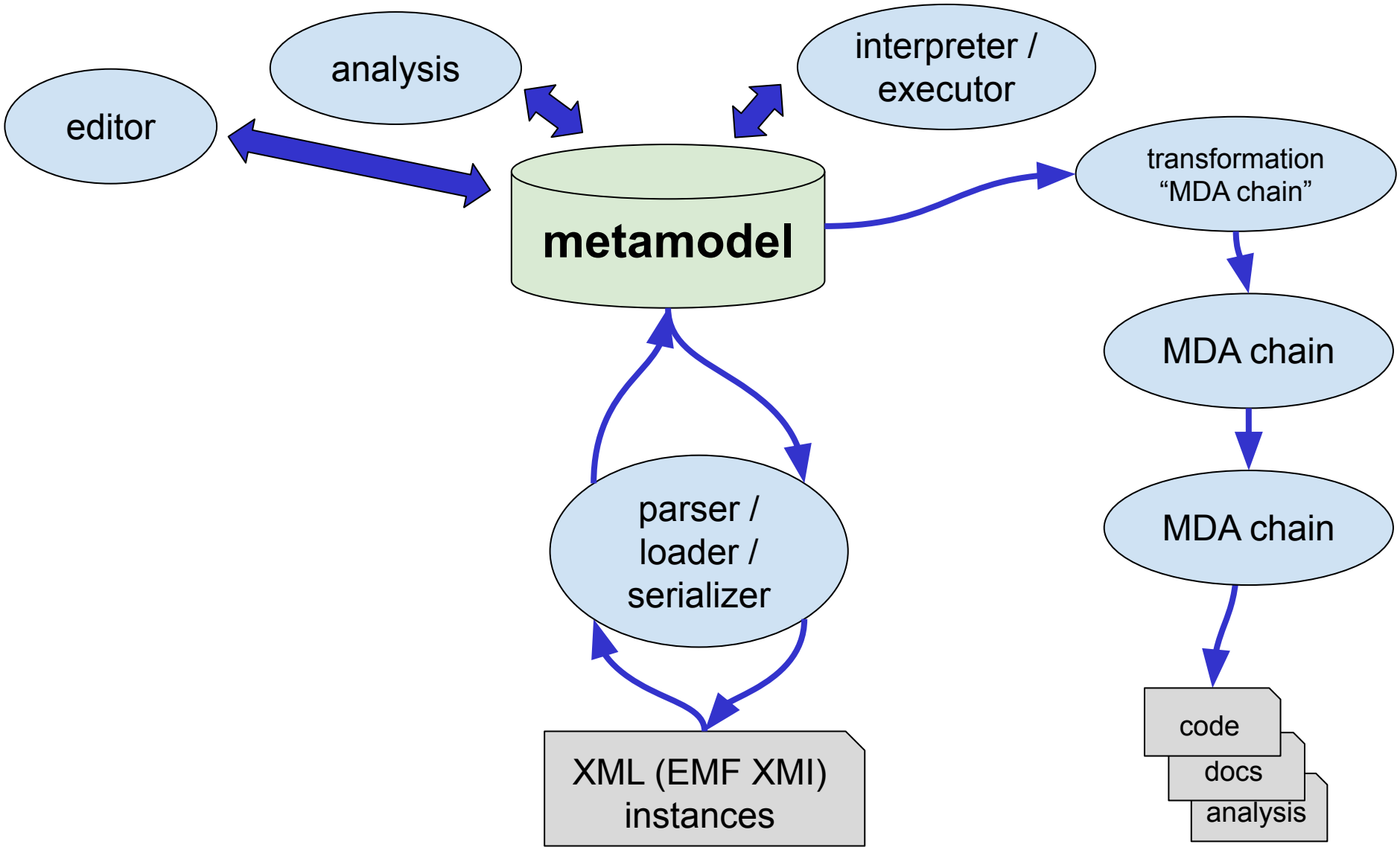

editor
metamodel


analysis
editor
metamodel


analysis
interpreter /
executor
editor
metamodel

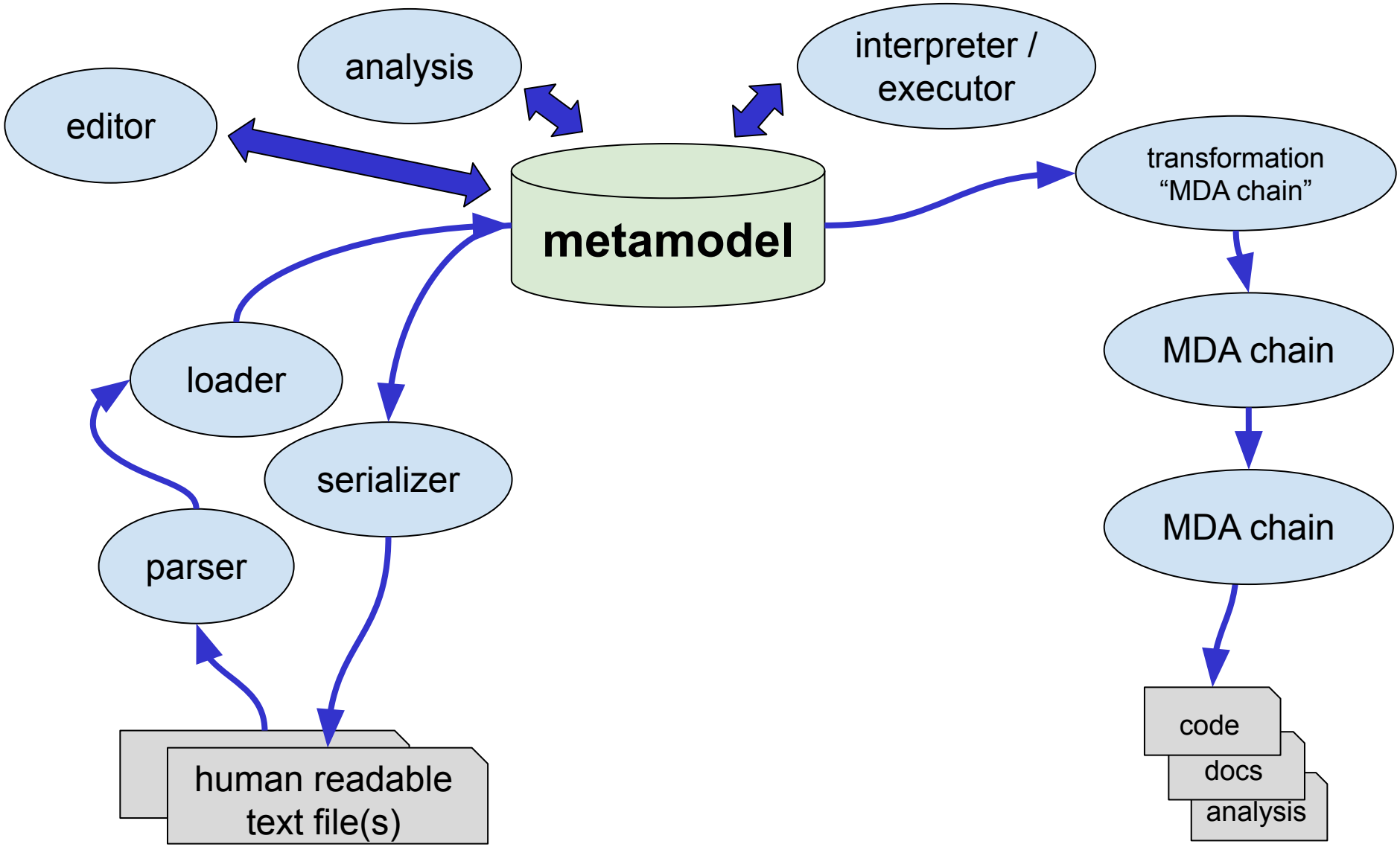

analysis
interpreter / executor
editor
metamodel
transformation "MDA chain"
MDA chain
MDA chain
code
docs
analysis


analysis
interpreter / executor
editor
metamodel
transformation "MDA chain"
MDA chain
MDA chain
?
persistence
code
docs
analysis


analysis
interpreter / executor
editor
metamodel
transformation "MDA chain"
MDA chain
MDA chain
parser / loader / serializer
XML (EMF XMI) instances
code
docs
analysis

Example: Persisting UML2 Instances in Papyrus-RT

```
<xmi:XMI xmi:version="20131001" xmlns:xmi="http://www.omg.org/spec/XMI/20131001"
  <uml:Model xmi:id="_R_b1UD1vEeaoHtWsKsKbKg" name="pingpong">
    <packagedElement xmi:type="uml:Package" xmi:id="_6VghMD1vEeaoHtWsKsKbKg" na
      <packagedElement xmi:type="uml:Collaboration" xmi:id="_6QznYD1vEeaoHtWsKs
        <interfaceRealization xmi:type="uml:InterfaceRealization" xmi:id="_6Vq5
          <name xsi:nil="true"/>
        </interfaceRealization>
        <interfaceRealization xmi:type="uml:InterfaceRealization" xmi:id="_6V5i
          <name xsi:nil="true"/>
        </interfaceRealization>
      </packagedElement>
      <packagedElement xmi:type="uml:Interface" xmi:id="_6VlZsD1vEeaoHtWsKsKbKg
      <packagedElement xmi:type="uml:Interface" xmi:id="_6VujoD1vEeaoHtWsKsKbKg
      <packagedElement xmi:type="uml:Usage" xmi:id="_6VzcID1vEeaoHtWsKsKbKg" cl
        <name xsi:nil="true"/>
      </packagedElement>
      <packagedElement xmi:type="uml:AnyReceiveEvent" xmi:id="_6V0DMD1vEeaoHtWs
      <packagedElement xmi:type="uml:Interface" xmi:id="_6V1RUD1vEeaoHtWsKsKbKg
      <packagedElement xmi:type="uml:Usage" xmi:id="_6V6w4D1vEeaoHtWsKsKbKg" cl
        <name xsi:nil="true"/>
      </packagedElement>
    </packagedElement>
```


analysis
interpreter / executor
editor
metamodel
transformation "MDA chain"
loader
serializer
MDA chain
parser
MDA chain
code
docs
analysis
human readable text file(s)

Example: Persisting Alf in Papyrus

```
class Order {

  /*Attributes*/
  public datePlaced: Date;
  public lineItems: OrderLineItem[0..*];
  public totalAmount : Money = (Money)0;

  /*Operations*/
  public addProduct(in product : Product, in quantity : Integer) {
    lineItem = new OrderLineItem(product, quantity);
    this.lineItems->add(lineItem);
    this.totalAmount = this.totalAmount + lineItem.getAmount();
  }

}
```

Persisting UML-RT in Papyrus-RT [experimental]

```
capsule Top
{
        part pinger : Pinger [ 1 ] ;
        part ponger : Ponger [ 1 ] ;
        connect pinger : pingPort to ponger : pongPort ;
}
capsule Pinger
{
        port pingPort : PingPongProtocol [ 1 ] ;
        statemachine
        {
                initial i0 ;
                state Playing
                {
                        entry action {|#
                            log.log(LCAPINST, "(enter Playing) sending ping");
                            pingPort.ping().send();
                            int n = 1;
                            int z = 2;
                        #|}
                }
```

Persisting MASL in Papyrus-xtUML (BridgePoint) [experimental]

```
relationship R4 is Session_Specification conditionally defines_rules_for many Session,
                   Session unconditionally rules_are_defined_by one Session_Specification;

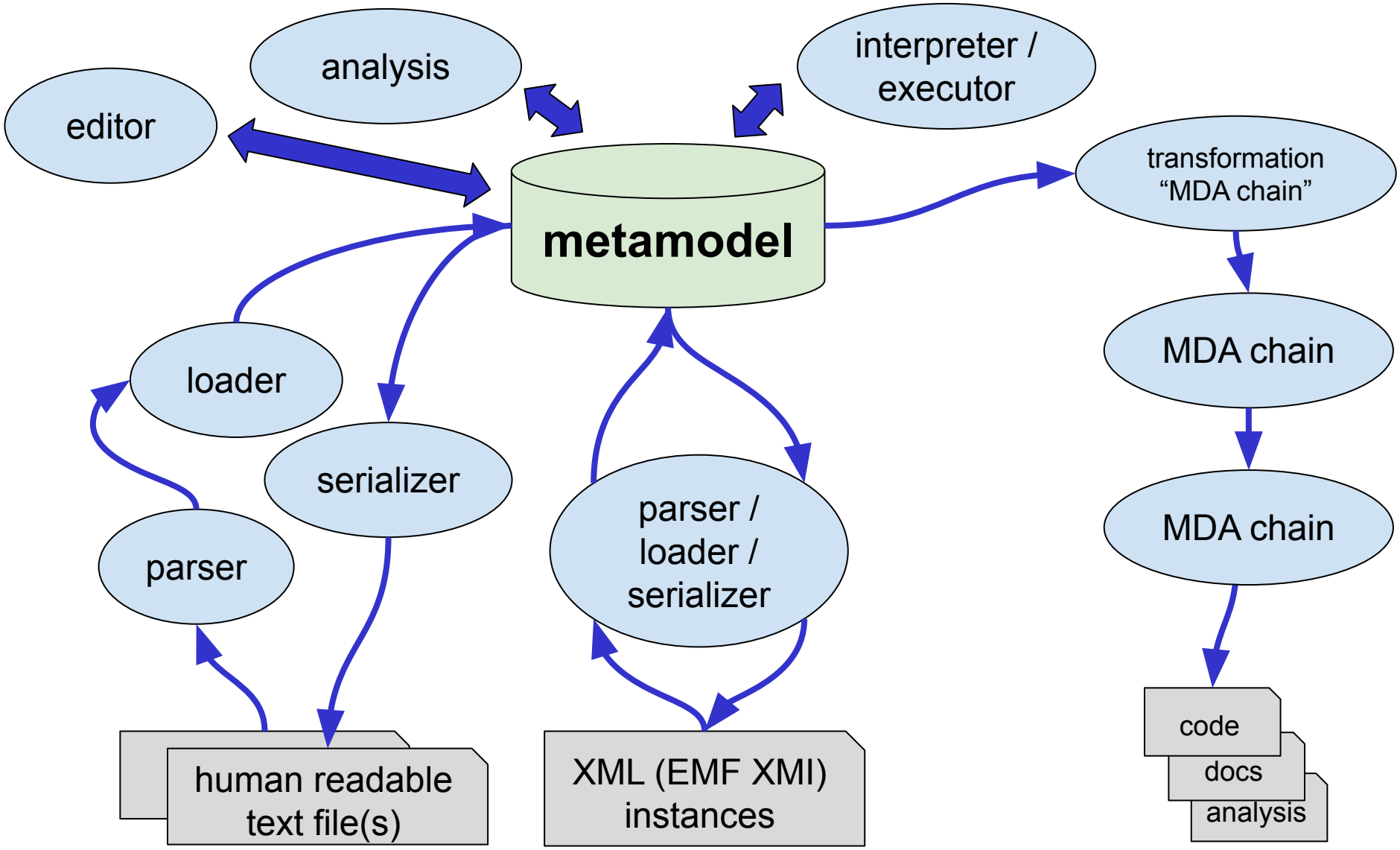
relationship R5 is Session unconditionally has_allowed many Operation,
                   Operation unconditionally has_been_allowed_for many Session
                   using Session_Operation;

object Group is
  group_id   : preferred unique integer;
  group_name : string;

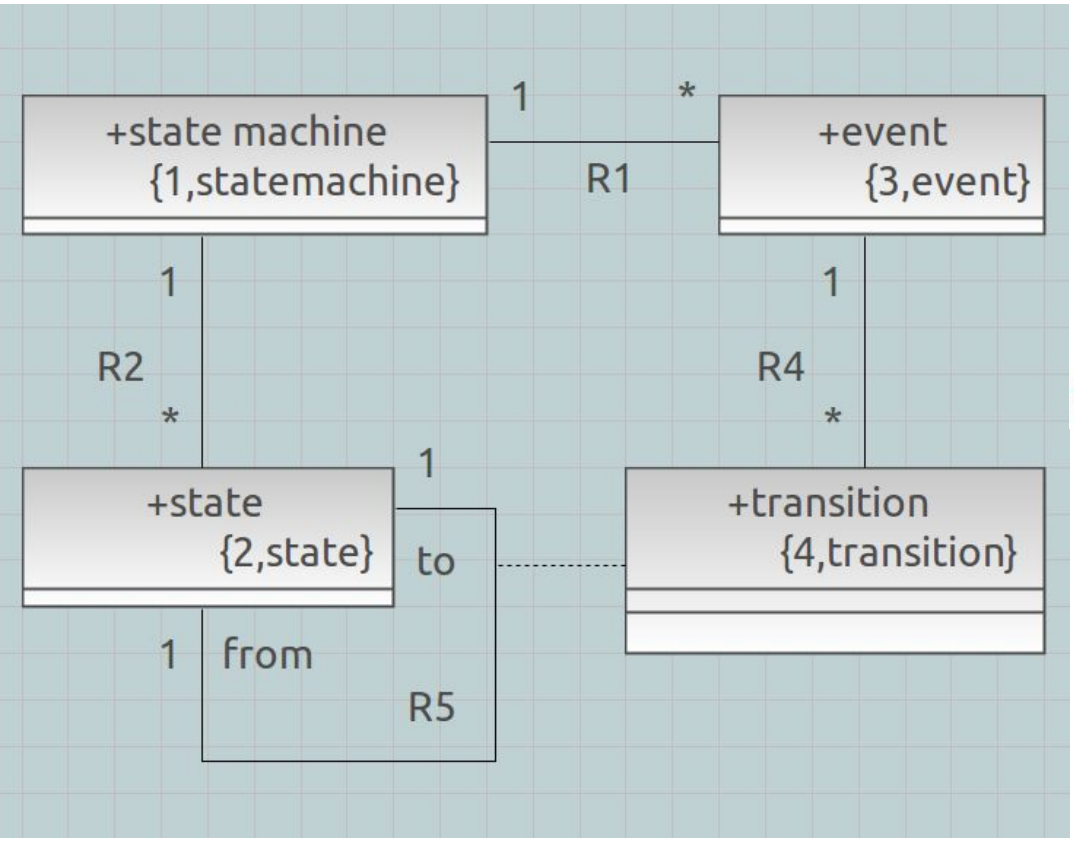
end object;
pragma id (3);
pragma key_letter ("G");

object Group_For_Session is
  group_id   : preferred referential (R2.is_member_of.Group.group_id) integer;
  session_id : preferred referential (R2.current_members_are.Session.session_id) integer;

end object;
```


analysis
editor
interpreter / executor
metamodel
transformation "MDA chain"
loader
serializer
parser
parser / loader / serializer
MDA chain
MDA chain
code
docs
analysis
human readable text file(s)
XML (EMF XMI) instances

Metamodel Fragment



Input Model

```
object PUMP is
  matrix is
    Ready_For_Use (
      Gun_Removed        => Pumping,
      Trigger_Depressed  => Ignore,
      Trigger_Released   => Ignore,
      Customer_Finished  => Ignore);
    Pumping (
      Gun_Removed        => Ignore,
      Trigger_Depressed  => Ignore,
      Trigger_Released   => Pumping_Paused,
      Customer_Finished  => Ignore);
    Fuel_Delivery_Complete (
      Gun_Removed        => Ignore,
      Trigger_Depressed  => Ignore,
      Trigger_Released   => Ignore,
      Customer_Finished  => Ready_For_Use);
  end transition;
end object;
```

Grammar and **AST** have:

- Matrix
- TransitionTable
- TransitionRow

AST ≠ metamodel

Advantages of XMI (Instance-Based Persistence)

UML2/EMF/XMI and xtUML/SQL

- Already works
- Simple parsing at the XML level
- Same for all EMF-based models and all UML2-models
- Does not have to care for semantics
- Graphics (Layout) can be persisted as instances
- Separation of concerns: Persistence layer vs. Editing
- ID-based referencing (more robust against dangling refs)
- Not editable by any editor, avoids risk of destruction
- Can still be mixed with partial textual representations

Advantages of Human-Readable Parsable Textual Syntax

MASL, textual UML-RT, Alf

- Human readable!
- Editable (by any text editor)
- Better Searchable without deserializing (and sed/grep/awk-able)
- Textual Cut/Copy/Paste
- Textual Diff/Merge is possible
- Natural scoping
- Model corruption may be more easily addressed
- Compact persistence
- Consistency between structural and activity persistence

Conclusion

Human-readable parsable textual concrete syntax offers benefits but brings costs.

XMI is mature, flexible and more straight-forward.

Backup Slides

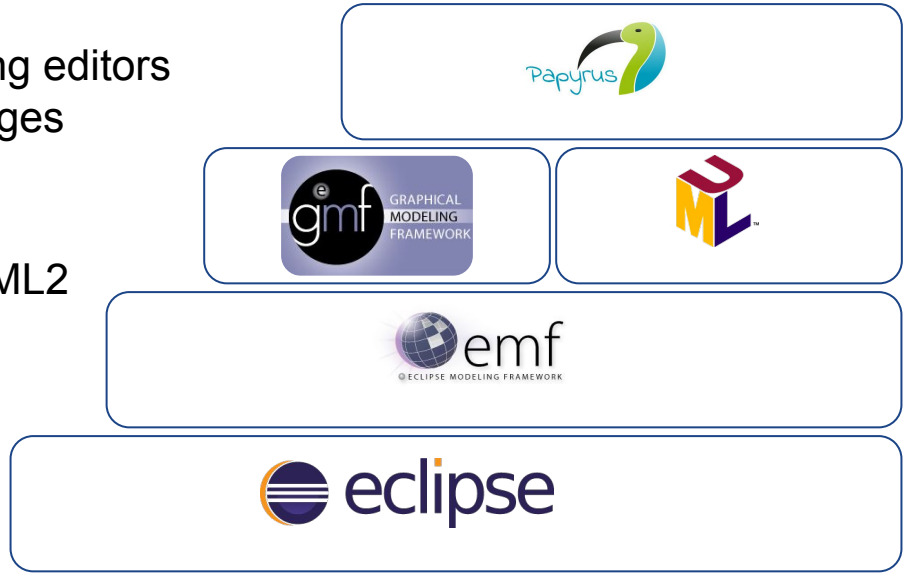
Persisting xtUML Instances in Papyrus-xtUML (BridgePoint)

```
-- BP 7.1.6 content: ModelClass syschar: 3 persistence-version: 7.1.6

INSERT INTO O_OBJ
        VALUES ("66920350-b77d-4876-9b05-0c545b5437ae",
        'TrackPoint',
        3,
        'TrackPoint',
        '',
        "00000000-0000-0000-0000-000000000000");
INSERT INTO O_NBATTR
        VALUES ("7080b4f6-ebf6-47fe-be82-a1c2cf39143c",
        "66920350-b77d-4876-9b05-0c545b5437ae");
INSERT INTO O_BATTR
        VALUES ("7080b4f6-ebf6-47fe-be82-a1c2cf39143c",
        "66920350-b77d-4876-9b05-0c545b5437ae");
INSERT INTO O_ATTR
        VALUES ("7080b4f6-ebf6-47fe-be82-a1c2cf39143c",
        "66920350-b77d-4876-9b05-0c545b5437ae",
        "00000000-0000-0000-0000-000000000000",
        'time',
```

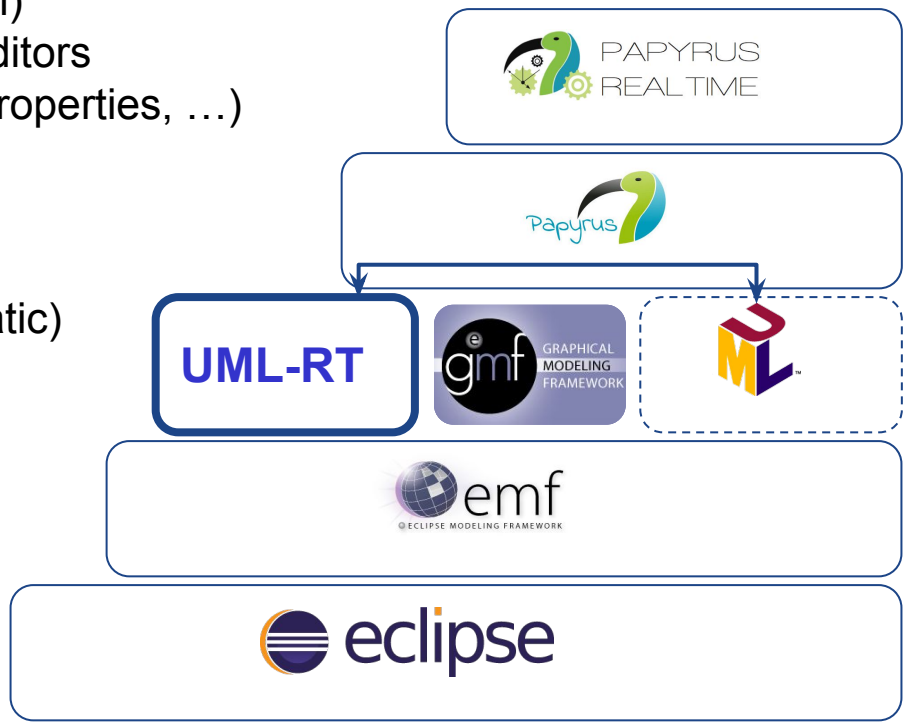
Example: Papyrus

- Papyrus
  - Built on UML2 and GMF
  - GMF-based diagrams for UML2
- GMF
  - FW for building graphical modeling editors
  - For EMF-based modeling languages
- UML2
  - EMF-based implementation of UML2
- UML2/GMF models
  - Plain EMF models
  - Any EMF technology works ~



Example: Papyrus-RT

- Papyrus-RT
  - UML2-based profile (~metamodel)
  - Extension of Papyrus' diagram editors
  - Customization of tools (palette, properties, …)
  - Code generators, ...
- Papyrus/UML2-based profiles
  - Ecore metamodel generation (static)
  - EMF metamodel API generation



Example: Papyrus-xtUML (BridgePoint)

- Bridgepoint
  - GEF-based Modeling editors
  - Model compilers (rely on xtUML metamodel)
  - Model interpreters (rely on xtUML metamodel)
- xtUML
  - Metamodel of the language
  - API to interact with model
  - Implementation of the xtUML API
- SQL
  - Format to persist models
- GEF
  - Technology to build modeling editors

