CimClipse

Bringing MDE to CIM

CIMug meeting

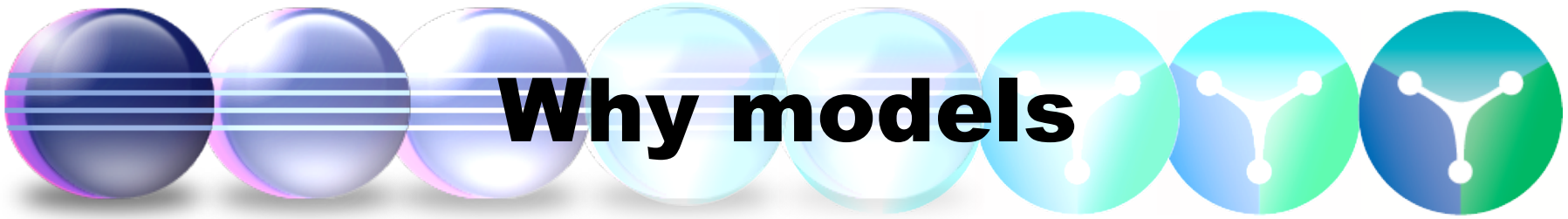
Dominique.Marcadet@supelec.fr

Background

- Professor and researcher in computer science at Supélec
- Collaboration with EDF on CIM since 2002
  - C++ framework for working with CIM objects
  - Used for converters between CIMXML files and other formats
  - Also, CimViewer, a tool written in Java to visualize CIMXML files

Why CimClipse

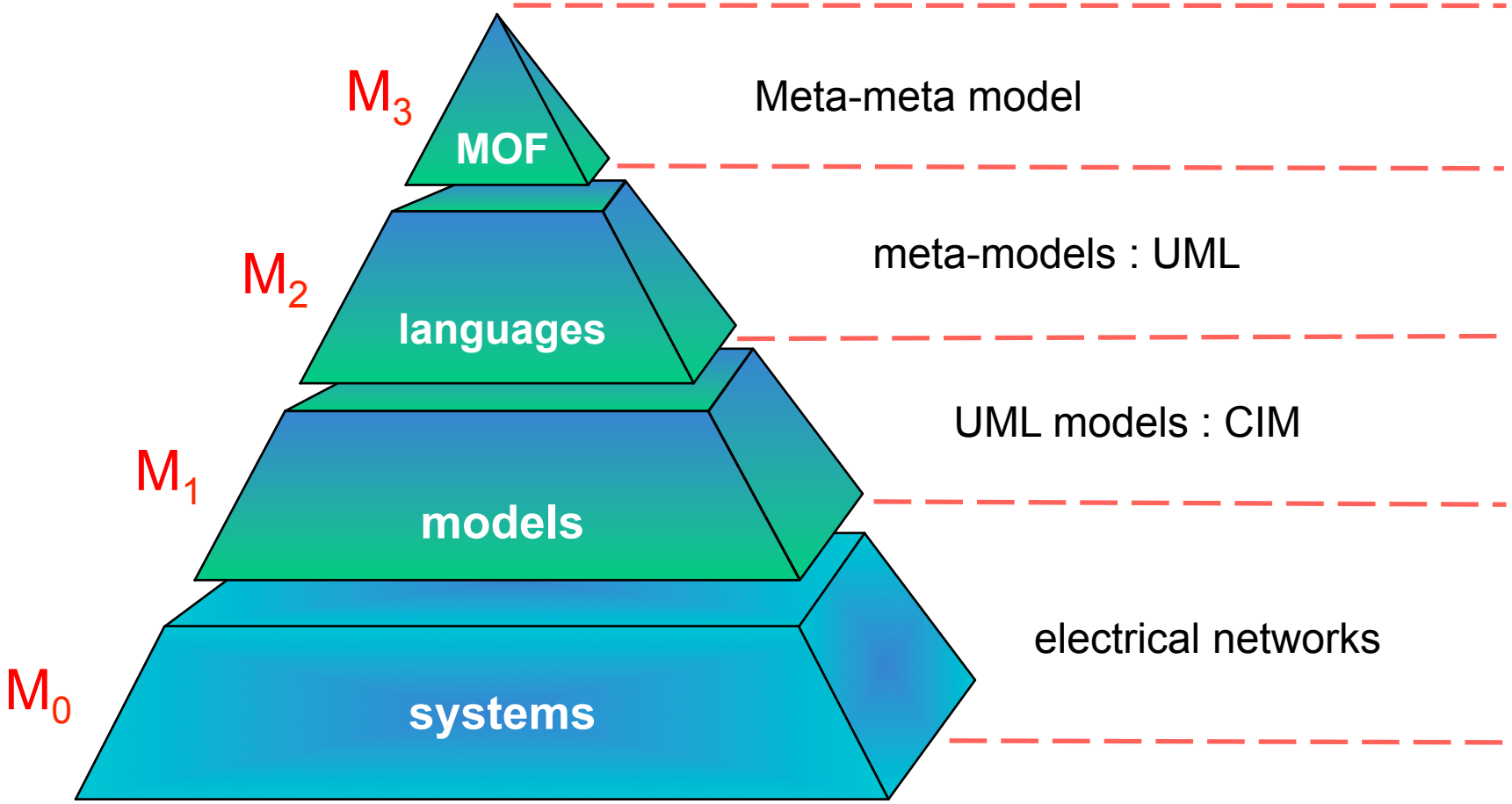
- The Model Driven Engineering (MDE) is a (not so) new way to build software focusing on models instead of algorithms and code
- The CIM is a model
- Eclipse is an open source environment offering the broader set of tools for MDE



Why models

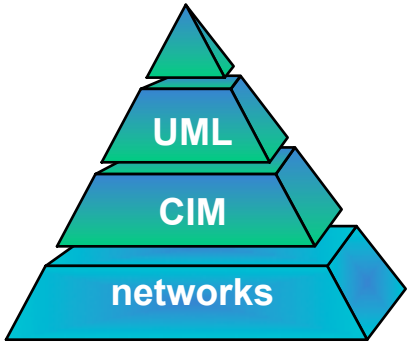
- Abstraction
  - Offer higher level concepts
  - Hide implementation details
  - Unify concepts before going to code, data, file format…
- Durability
  - Are more stable than programs
  - Can be more easily adapted to new technologies

The OMG modeling stack

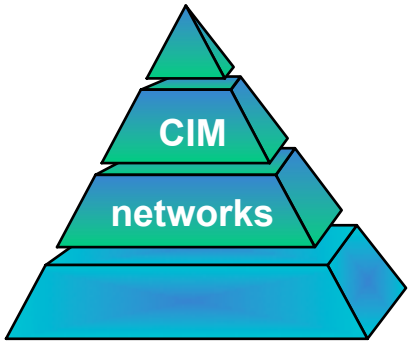


CIM in the modeling stack

- CIM as a UML model
  - Generate models/code/data
  - Compare CIM models

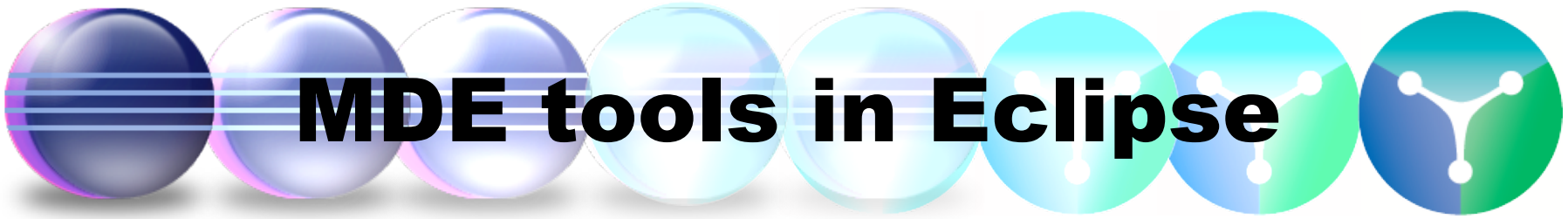


- CIM as a language
  - CIM got a promotion, it is now an Ecore model (a DSML)
  - Networks are models
  - Networks can be manipulated with Eclipse/MDE tools



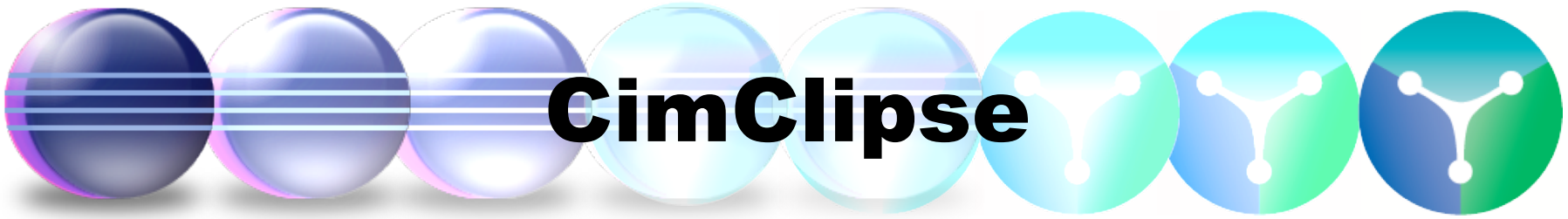
MDE with Eclipse

- The meta-meta-model is Ecore (M3 layer)
- UML is available as an M2 layer
- Ecore Modeling Framework (EMF) is the base framework for building MDE tools, it unifies Java, UML and XML



MDE tools in Eclipse

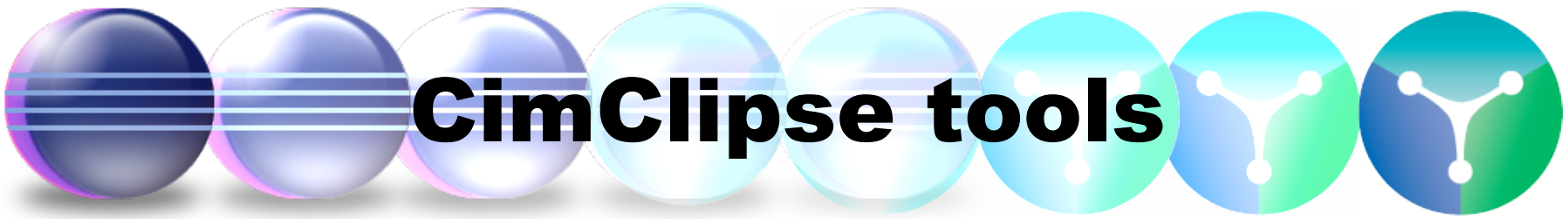
- EMF
  - Java API for model manipulation
  - XML (de)serialisation
  - Comparison
  - Query and transaction
  - Validation
  - Storage, sharing
- Other projects
  - Model-to-Model transformation (M2M)
  - Model-to-Text transformation (M2T)
  - Graphical model edition
  - …



CimClipse

- CimClipse is an umbrella name for tools:
  - used within or based on Eclipse or its plugins
  - used for CIM related tasks
  - released as Open Source

- http://wwwdi.supelec.fr/software/cimclipse/



CimClipse tools

- CIM models as Eclipse/UML2 models
  - manual modifications on EA XMI exports
- CIM models as Ecore models
  - M2M (with ATL) transformation
- OCL Validator to validate a CIMXML file against a set of constraints written in OCL

OCL

- Object Constraint Language is a textual language used to formally specify properties (invariants)
- It can be used at the M1 level to be applied on M0 artifacts
- It can also be used at the M2 level to be applied on M1 models

Why OCL

- OCL is easy to read/write:
  - OWL:

```
<rdf:Description rdf:nodeID= "…" >
  <owl:minCardinality rdf:datatype="http://www.w3.org/…#int">1</owl:minCardinality>
  <owl:onProperty rdf:resource="http://iec.ch/…cim14#TapChanger.highStep"/>
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Restriction"/>
</rdf:Description>
```

  - OCL:

```
context TapChanger inv:
 self.highStep->size() >= 1
```

- Values of attributes can be checked

```
context IdentifiedObject inv name_rule:
   self.name.size() <= 32
```

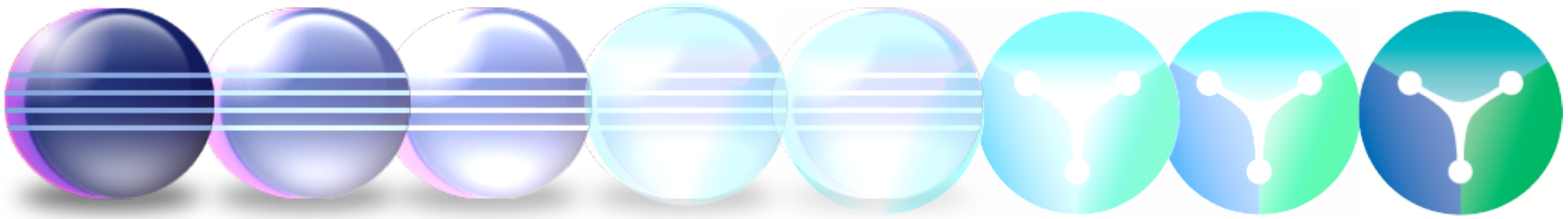
Why OCL

- OCL works at the model level:
  - Associations are bidirectional

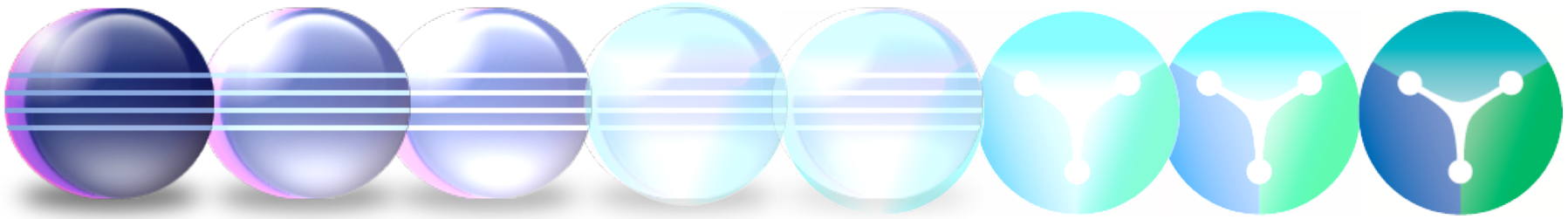
```
context Terminal inv connected_terminal:
   self.ConnectivityNode->size() = 1

context ConnectivityNode inv isolated_node:
  self.Terminals->size() >= 2
```

  - Inheritance is known

```
context ACLineSegment inv:
   self.EquipmentContainer.oclIsTypeOf(Line)
```



- The profile is not the model
- The model:
  - Every ConductingEquipment has 0 or more Terminals
- The profile:
  - An ACLineSegment has 2 Terminals



- A file must be valid for the model, may be valid for the profile
- The tool to check the validity against a model must be distinct from the one to check for the validity against the profile
- CimClipse:
  - Standard EMF load engine to check against the model
  - OCL constraints to check against the profile