IEEE RCIS 2015

IEEE 9th International Conference on Research Challenges in Information Science

*May 13-15 2015, Athens, Greece*

# Meta-modeling: concepts, tools and applications

https://dl.dropboxusercontent.com/u/6656530/tutorial_rcis2015.html


**Saïd Assar** [1,2]

(1) **Associate Professor**
*Telecom Ecole de Management, Department of Information Systems*
*Evry, France*

(2) **Invited Researcher**
*Centre de Recherche en Informatique, University of Paris 1 La Sorbonne*
*France*


RCIS 2015

# 1. Introduction & background (cont'd)

- INSTITUT MINES - TELECOM: 4 leading schools in engineering and management

Telecom Bretagne

INSTITUT Mines-Télécom

Telecom Ecole de Management

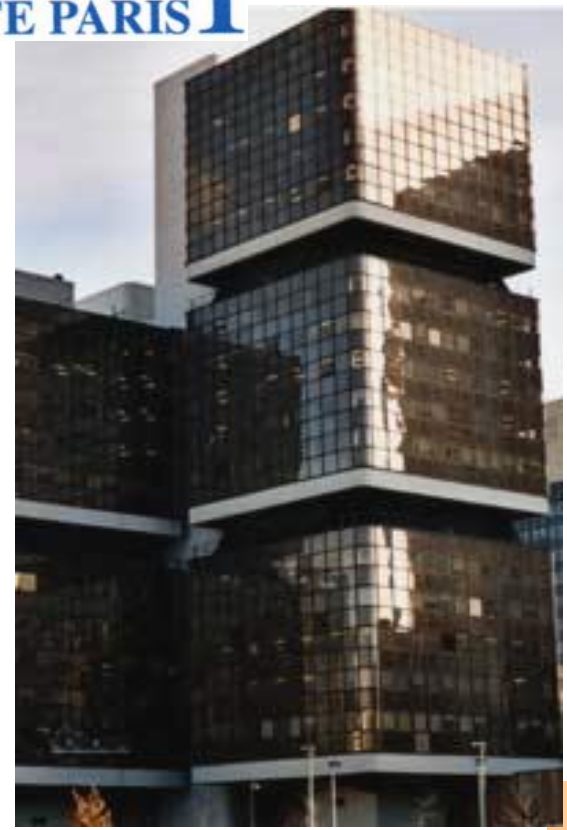
Telecom Sud Paris

Telecom Paris


# 1. Introduction & background (cont'd)

- Since 1999, long-term research collaboration with CRI at Paris 1 Sorbonne





# Learning objectives

- ✓ To understand **what is a meta-model**, its historical background, when it is needed and **how it can be defined and exploited**.

- ✓ To **witness and participate in building concrete meta-models** and exploiting them using existing meta-modeling technical infrastructures

- ✓ To be **aware of problems and challenges in meta-modeling** and the actual state-of-the-art regarding these issues.

# Agenda

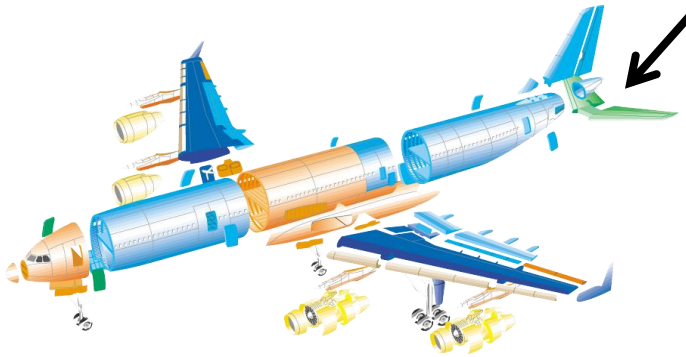
1. Introductory examples
2. Abstraction levels and the instantiation problem
3. Computerized tools for meta-modeling
4. Illustration
   a) A Domain Specific Language (DSL) for IoT v3.0 Framework
   b) A DSL to model software development processes
   c) A DSL to specify web applications
5. Research challenges
6. Conclusion

# 1. Introductory examples – what is a model ?

"A **model** (M) for a system (S) and an observer (O) is any kind of **representation** which can help O in **answering questions** and **reasoning** about S“

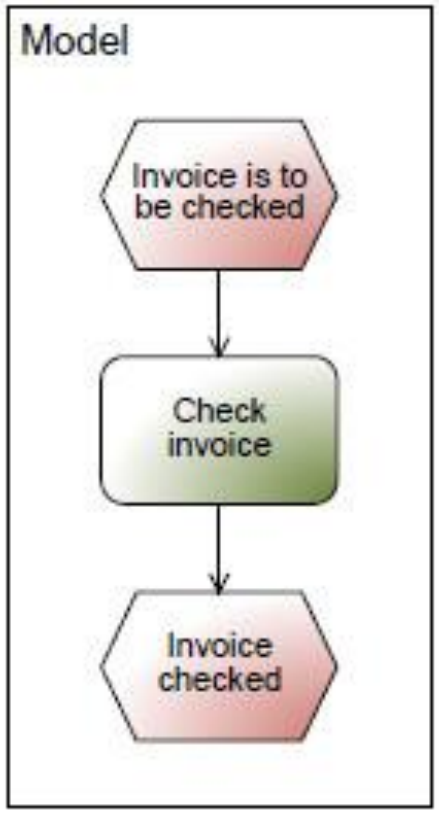
- *It is an **abstraction, a reduction** of reality to its most relevant aspects*
- *It is the result (i.e. the product) of some specific **process***
- *It is built using some kind of **notation** (formal and/or graphical)*
- ***Descriptive** vs. **prescriptive** models*
- *Models are **artifacts**, i.e. they can also be modeled*

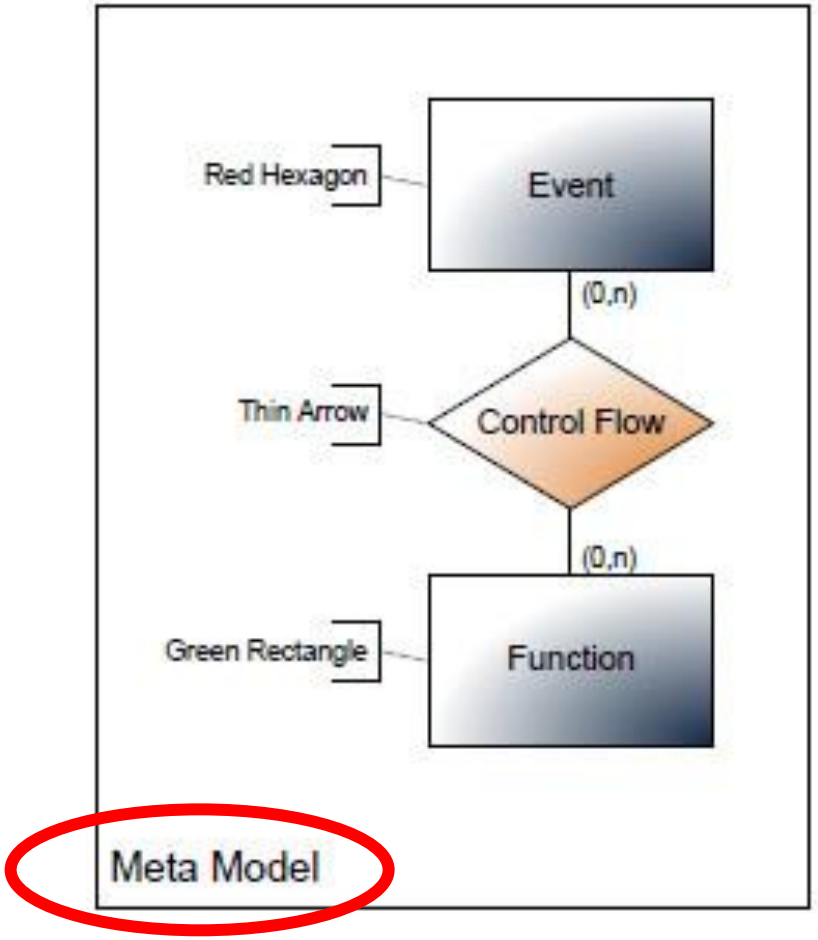
Observer



# 1. Introductory examples – what is a meta-model ?

Let's consider this very simple process model with **two concepts** (event, function) and **one link** among concepts (event – function)



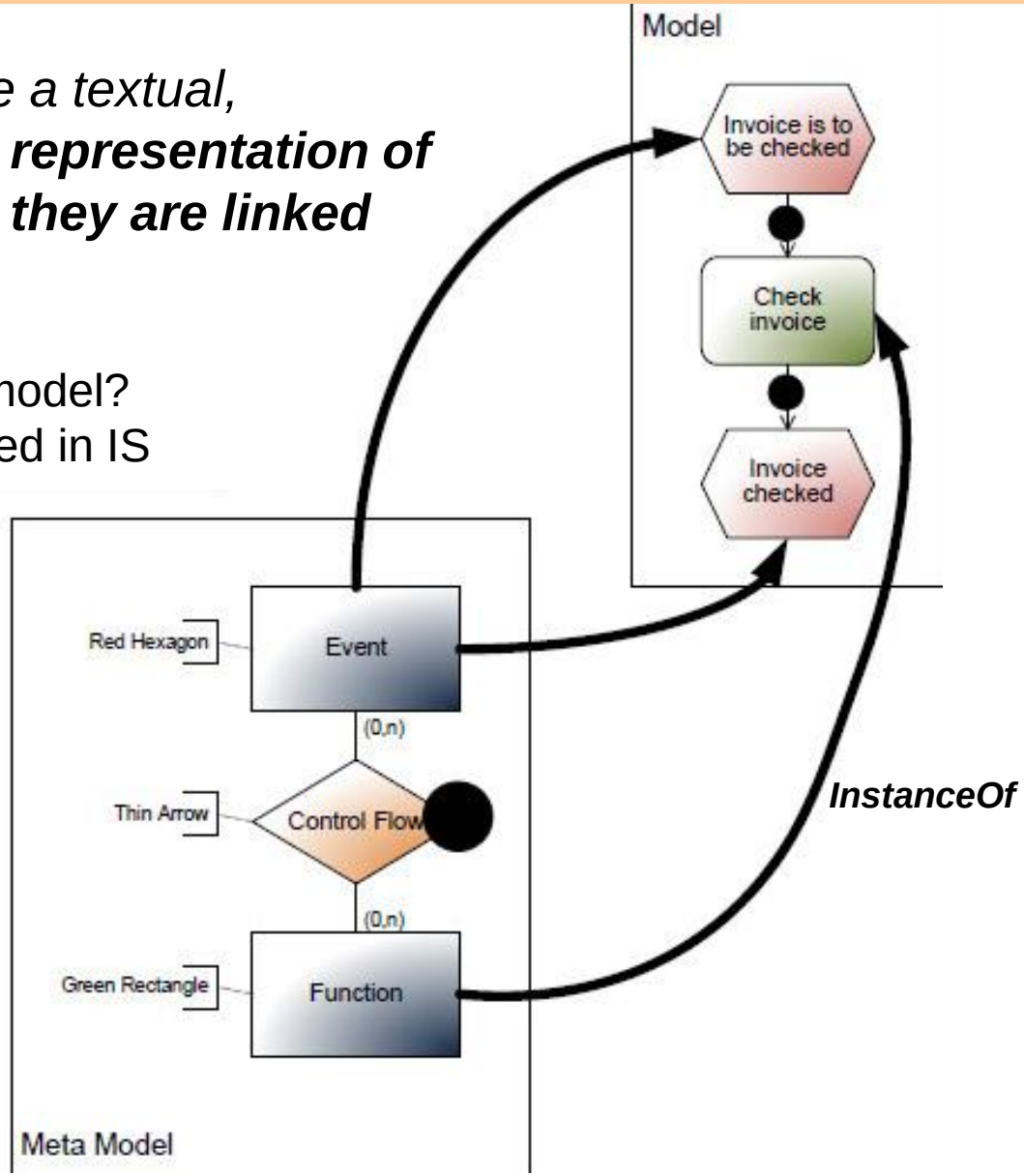

**Source**: Hanns-Alexander Dietrich, ***Exercice 1 Meta-modelling***, European Research Center for Information Systems, Information Modelling Summer Term 2012.

# 1. Introductory examples – what is a meta-model ?

- *A **meta-model** would be a textual, graphical, and/or formal **representation of the concepts and how they are linked***

Questions

- How to represent a meta-model?
- How can a MM be leveraged in IS engineering?

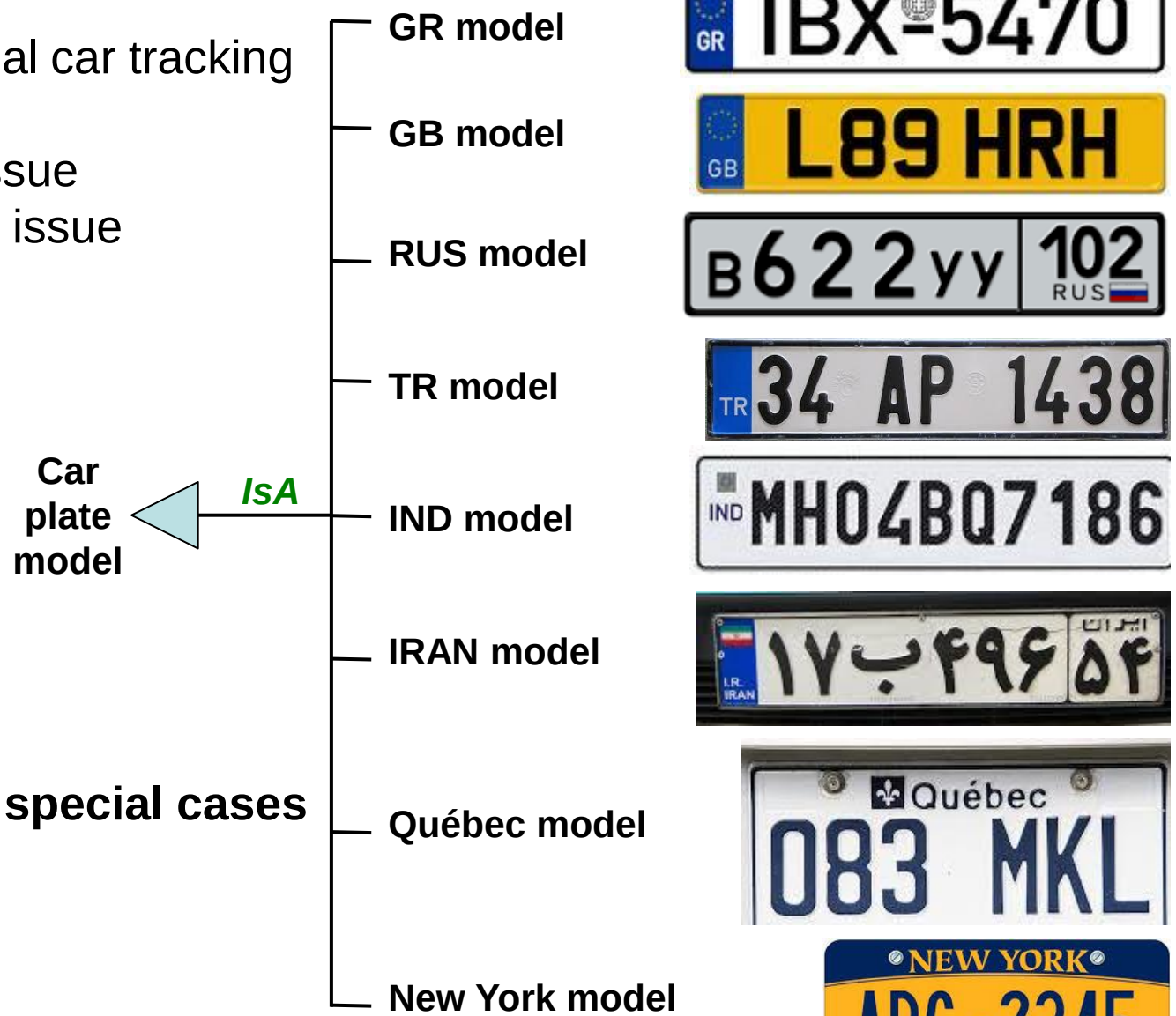



**Source**: Hanns-Alexander Dietrich, ***Exercice 1 Meta-modelling***, European Research Center for Information Systems, Information Modelling Summer Term 2012.

# 1. Introductory examples – generic models (1)

Suppose a trans-national car tracking system

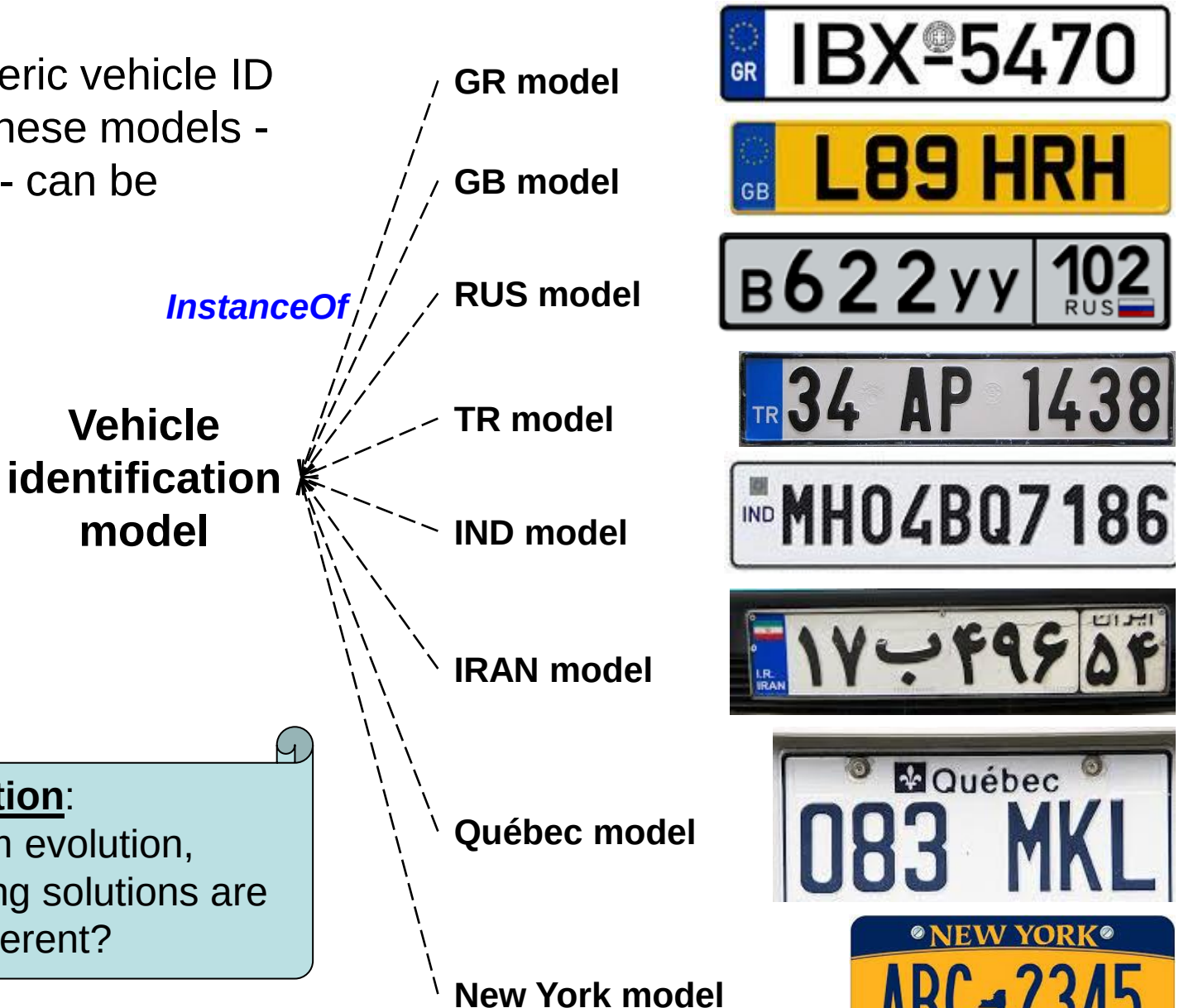
- ✓ interoperability issue
- ✓ system evolution issue




➢ **Are all these plates special cases of a car plate model?**

# 1. Introductory examples – generic models

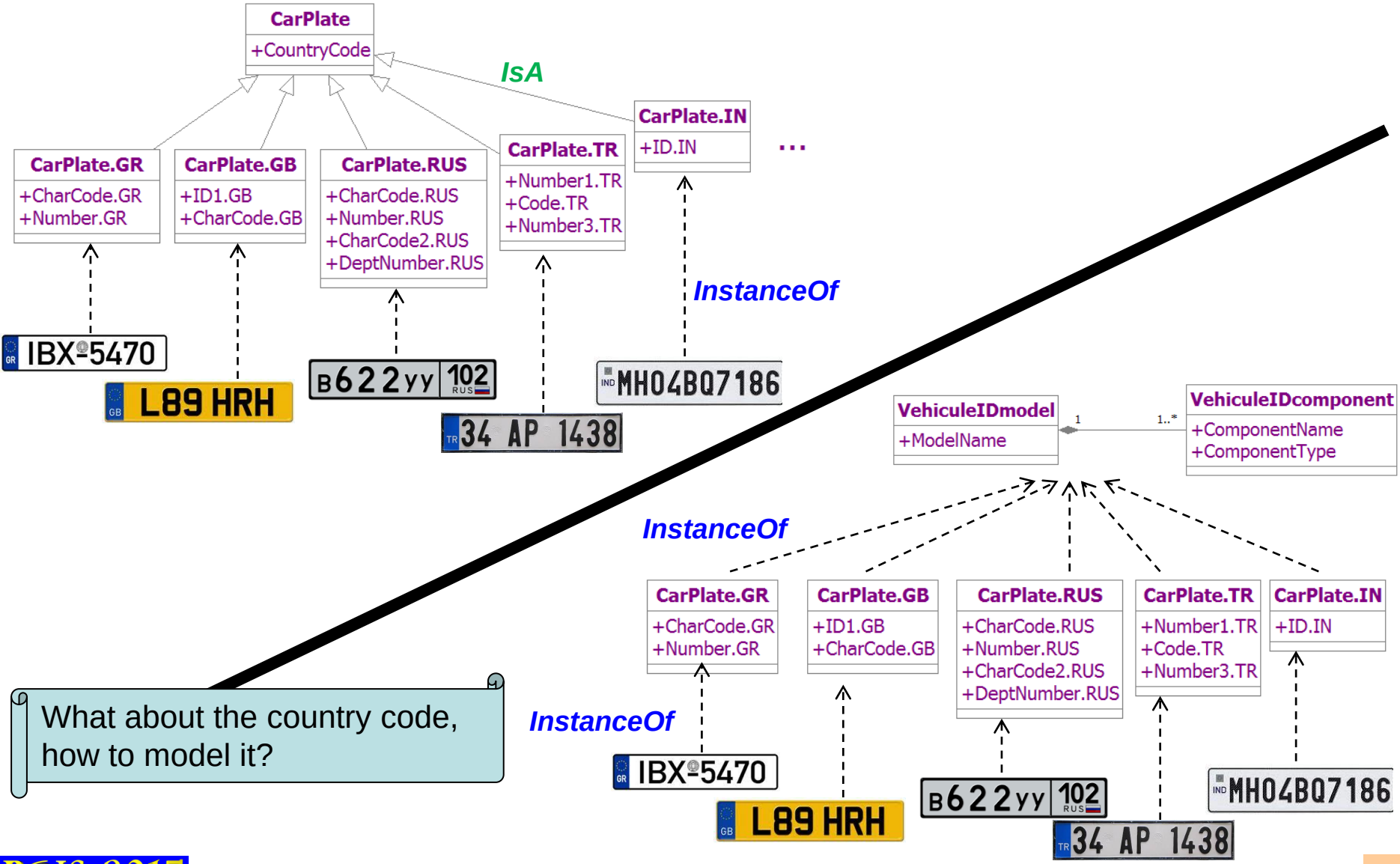
➢ Or is there a generic vehicle ID model from which these models - and may be others - can be derived?




**Interactive question**:
In terms of system evolution, these two modeling solutions are similar or very different?

# 1. Introductory examples – generic models

In terms of system evolution, these two modeling solutions are similar or very different?




What about the country code, how to model it?

# 1. Introductory examples – generic models (2)

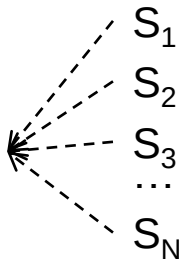
For a software development company with different methods

- ✓ evolution issue
- ✓ enactment issue

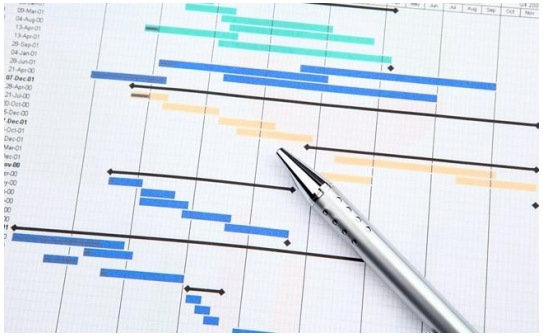
**General method** ← *IsA ?*

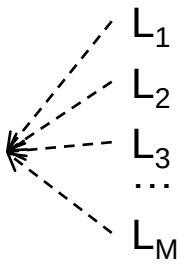
**General method structure** ← *InstanceOf ?*

**Method M1 for simple projects**

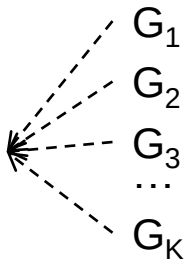




**Method M2 for large projects**





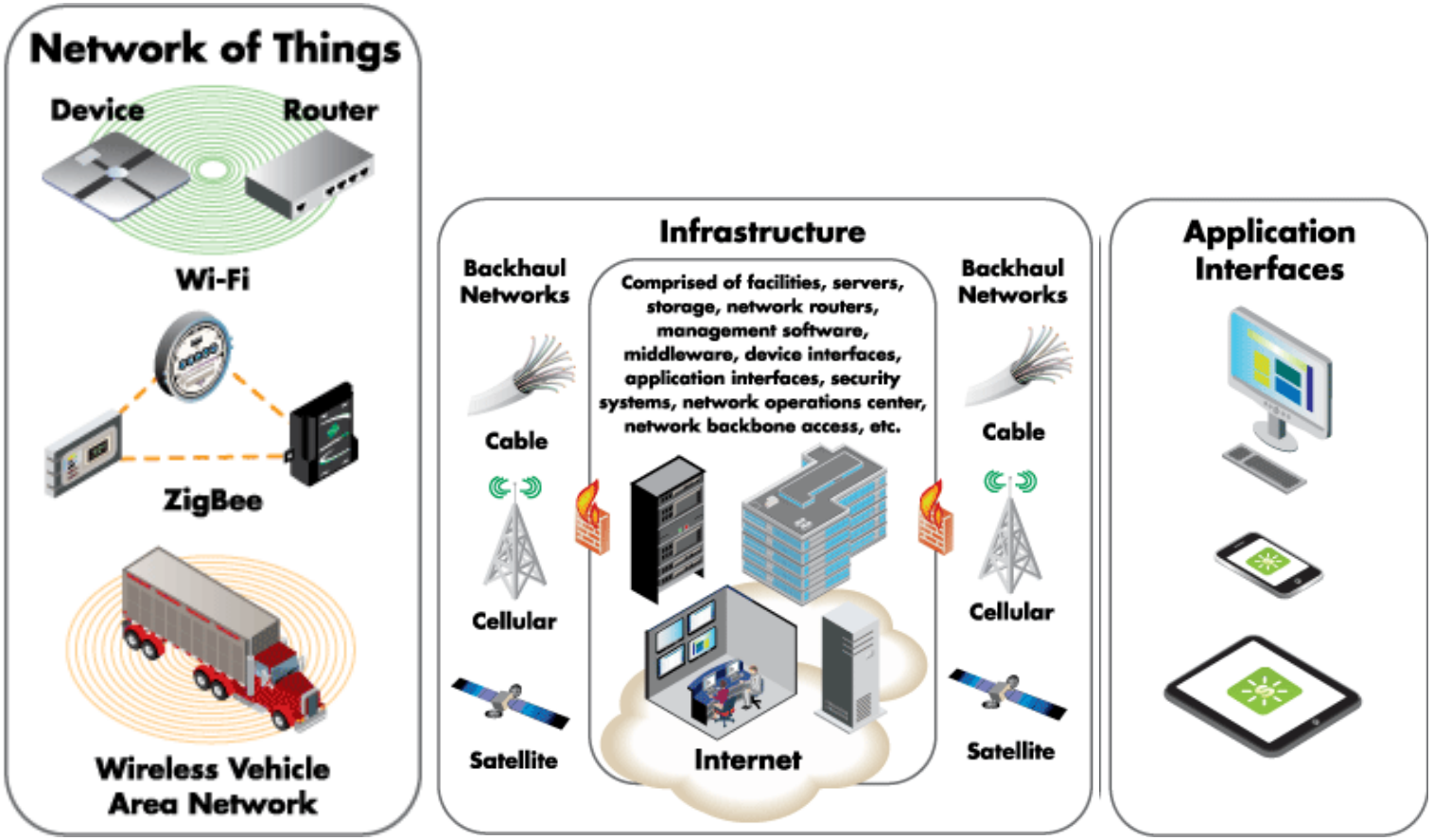

**Agile method M3 for risky projects**

# 1. Introductory examples – IoT framework

In the field of Internet of Things (IoT), a large set of heterogeneous objects are expected to interact and be part of a huge information network …

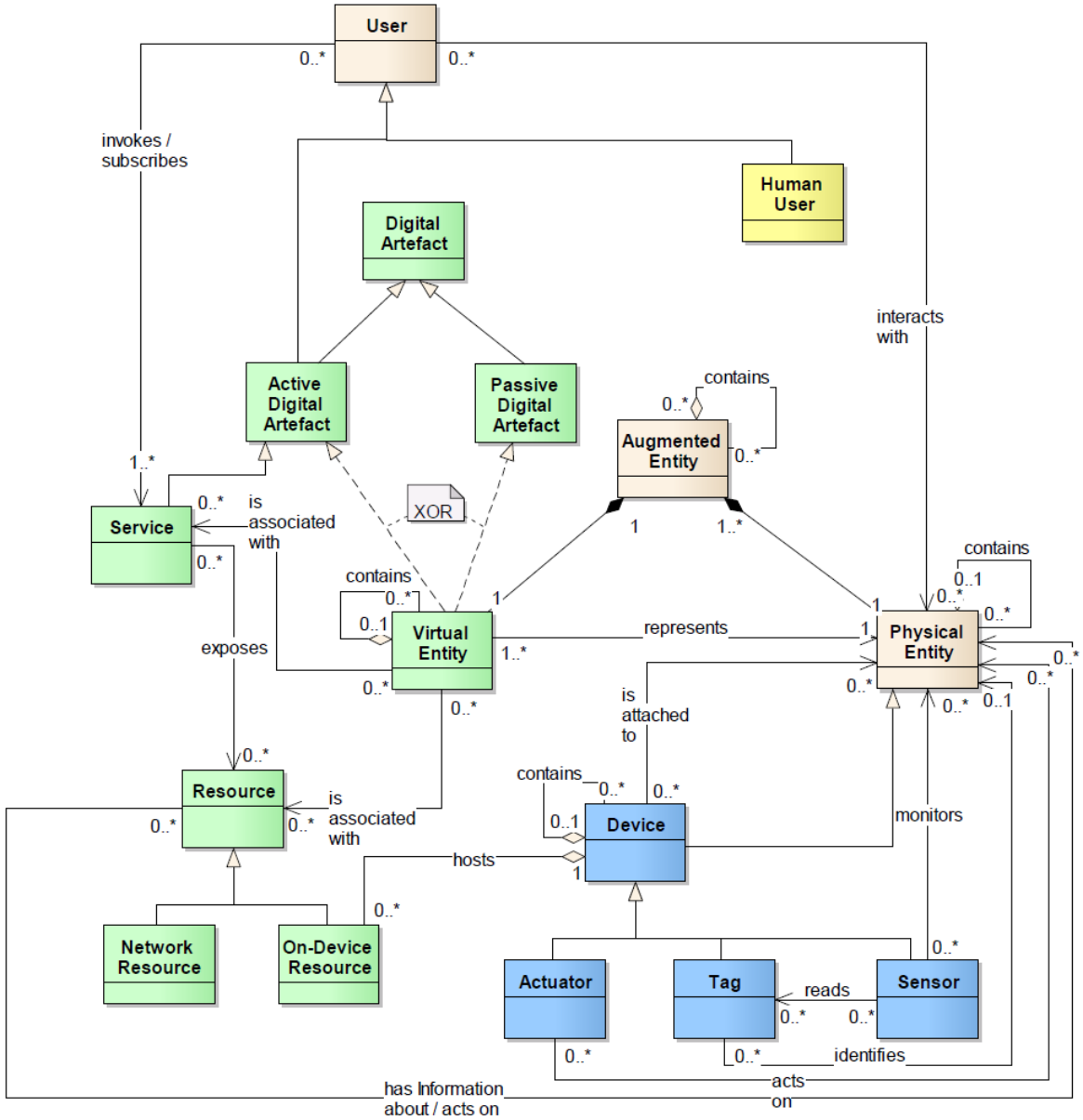
- **Is it possible to have a general model for all interacting objects?**



# 1. Introductory examples – IoT framework

A reference model of IoT is proposed by an FP7 project …

- **What this model is helpful for?**
- **Is it possible to provide computerized support for using this model?**

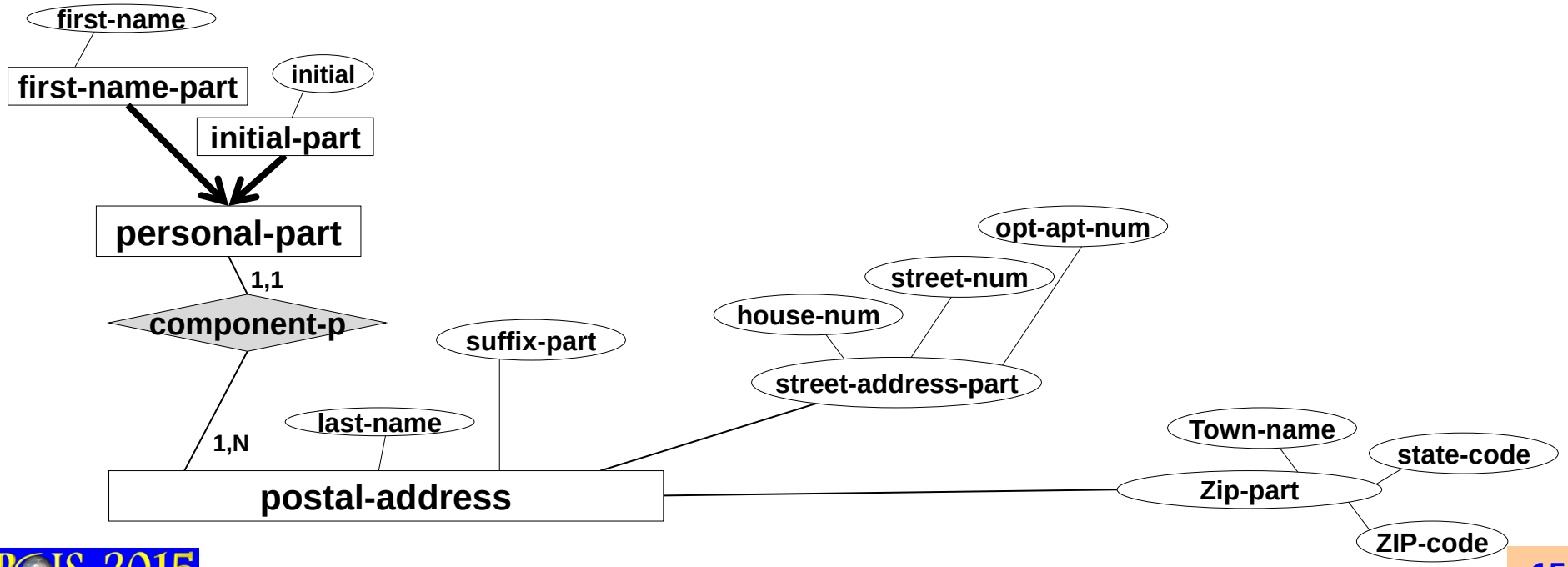



**Source**
http://www.iot-a.eu/public/public-documents/d1.5/at_download/file

# 1. Introductory examples – language definition

## Language syntax definition (a toy language)

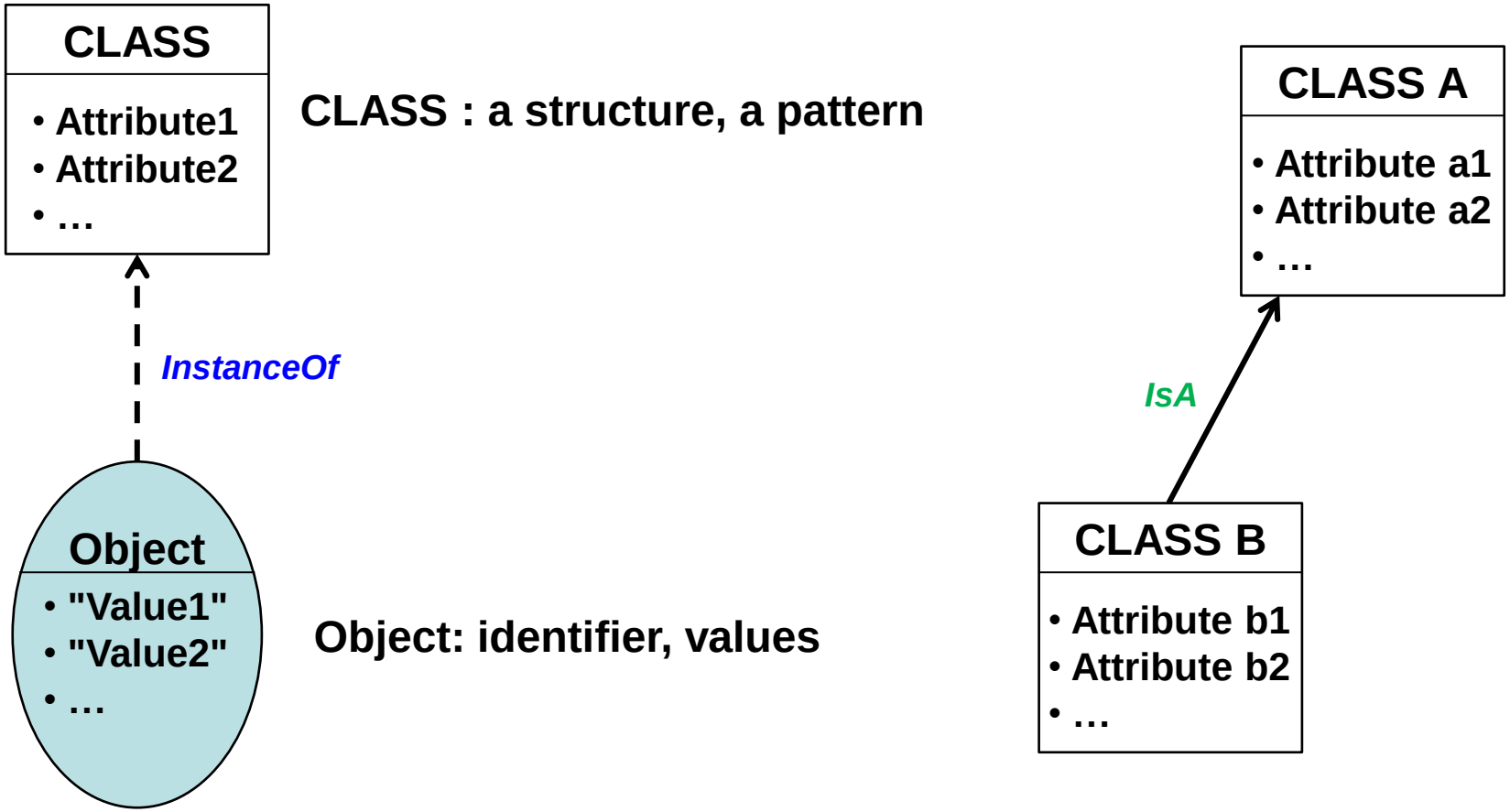
```
<postal-address> ::= <name-part> <street-address> <zip-part>
<name-part>      ::= <personal-part> <last-name> <opt-suffix-part> <EOL>
                   |<personal-part> <name-part>
<personal-part>  ::= <first-name> | <initial> "."
<street-address> ::= <house-num> <street-name> <opt-apt-num> <EOL>
<zip-part>       ::= <town-name> "," <state-code> <ZIP-code> <EOL>
<opt-suffix-part> ::= "Sr." | "Jr." | <roman-numeral> | ""
<last-name>      ::= [A-Z|a-z]
<first-name>     ::= [A-Z|a-z]
```



# Agenda

1. Introductory examples
2. Abstraction levels and the instantiation problem
3. Computerized tools for meta-modeling
4. Illustration
   a) A Domain Specific Language (DSL) for IoT v3.0 Framework
   b) A DSL to model software development processes
   c) A DSL to specify web applications
5. Research challenges
6. Conclusion

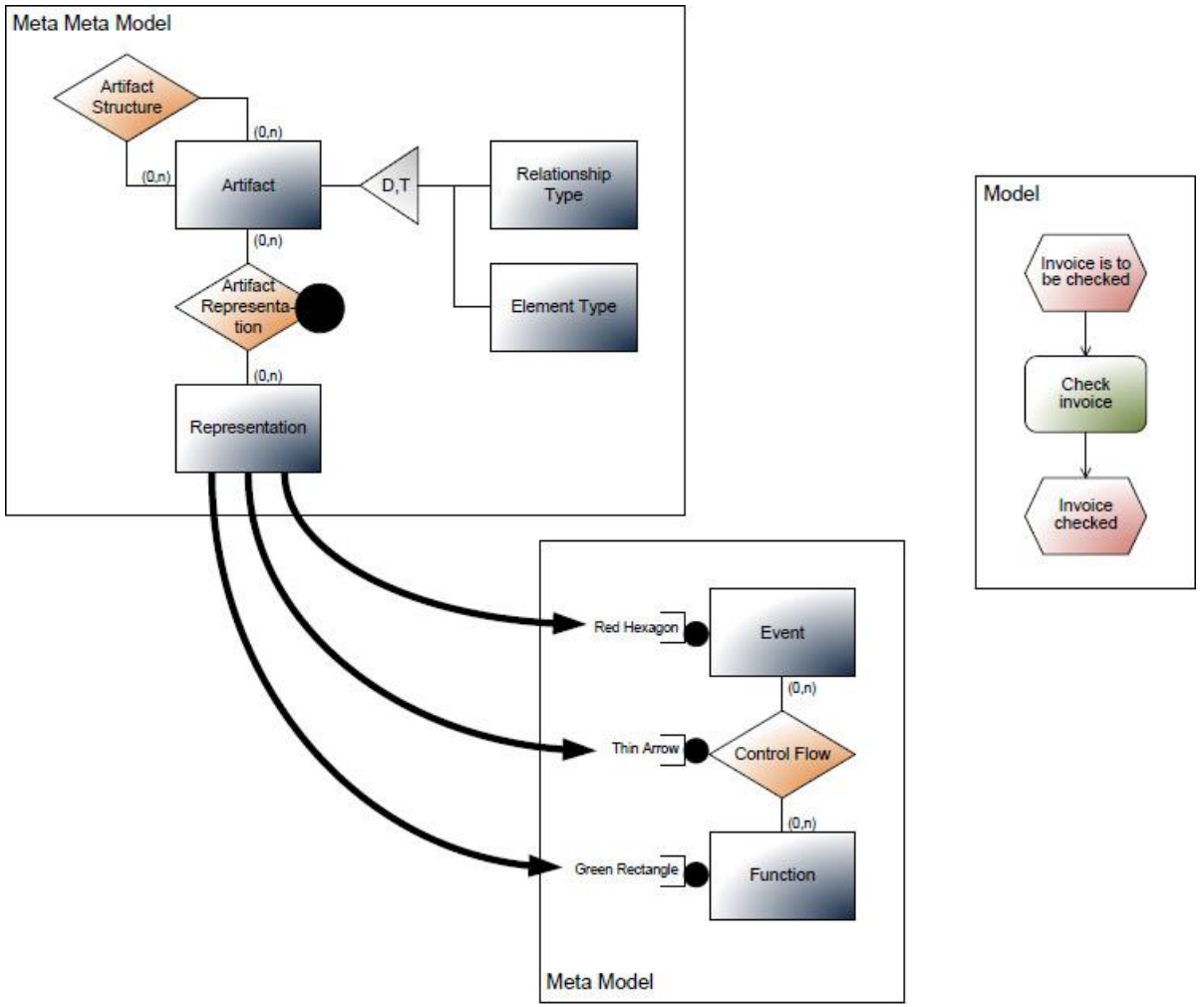
# 2. Abstraction levels – instanciation vs. generalisation



# 2. Abstraction levels – the meta-modeling language issue

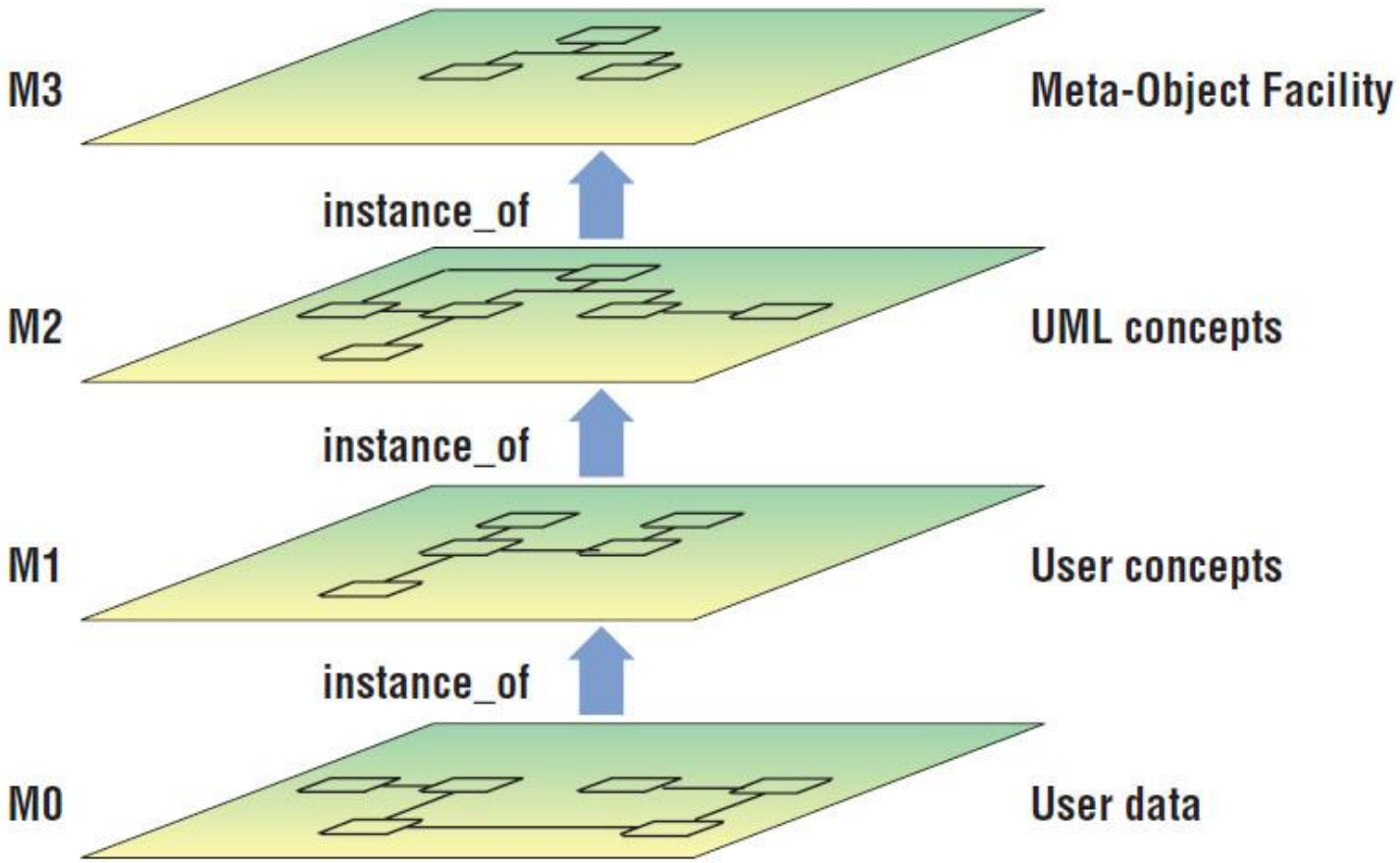
✓A meta-model is described using a meta-modeling language

➢***How to describe the meta-modeling language … a circular issue.***



# 2. Abstraction levels – instantiation levels

- Meta-modeling framework by OMG : **Meta Object Facility** (MOF)




***=> Strict instanciation***

# 2. Abstraction levels – instantiation levels

- Instantiation levels in th **Meta Object Facility** (MOF) by OMG

| Layer | Description |
|---|---|
| Meta-metamodel | The infrastructure for a metamodelling architecture. Defines the language for specifying metamodels. |
| Metamodel | An instance of a meta-metamodel. Defines the language for specifying a model. |
| Model | An instance of a metamodel. Defines a language to describe an information domain. |
| User objects (user data) | An instance of a model. Defines a specific information domain. |

| Layer | Example |
|---|---|
| Meta-metamodel | *MetaClass, MetaAttribute, MetaOperation* |
| Metamodel | *Class, Attribute, Operation, Component* |
| Model | *StockShare, askPrice, sellLimitOrder, StockQuoteServer* |
| User objects (user data) | *⟨Acme_SW_Share_98789⟩, 654.56, sell_limit_order, ⟨Stock_Quote_Svr_32123⟩* |

**Source**: OMG

# 2. Abstraction levels – the "Powertype" approach

**Where does meta-modeling come from?**

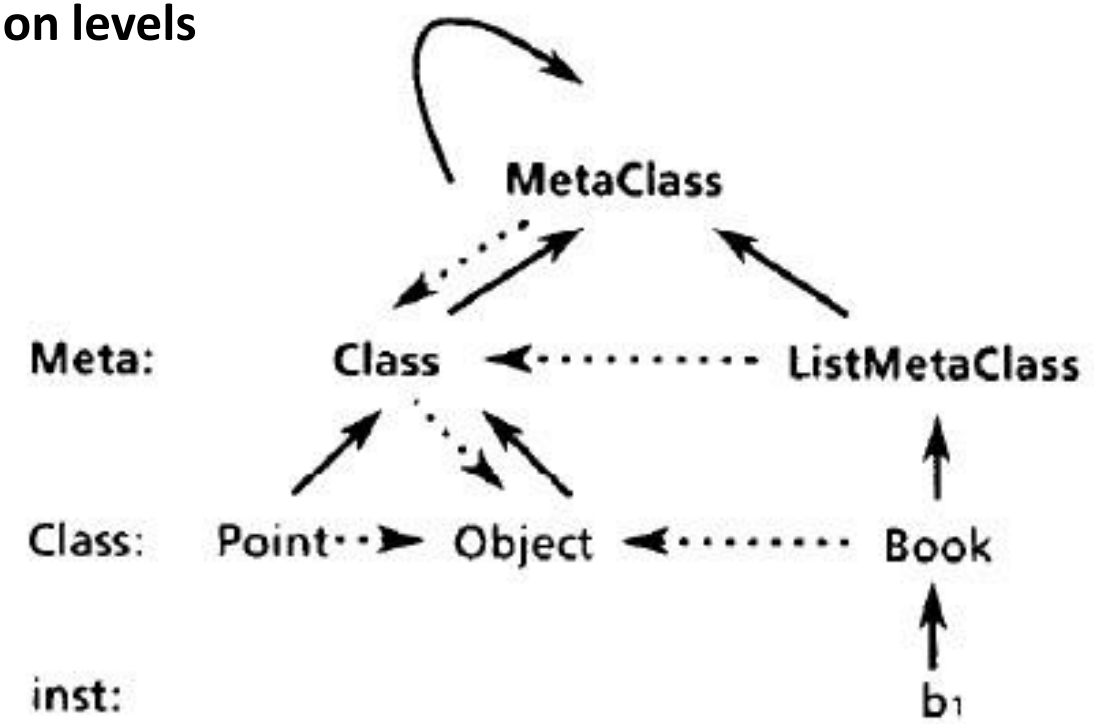
- Artificial intelligence (i.e. frame concept, Smalltalk, ObjVLisp)
- Databases (i.e. meta-database)
- Object oriented modeling (i.e. Classes are objects too => MetaClasses)

**How to simultaneously create a generalization link AND an instantiation link**

**How to a define attributes to Classes**

**How to cross the strict instantiation levels**

- *One of the earliest work on these issues is by P. Cointe in 1987*

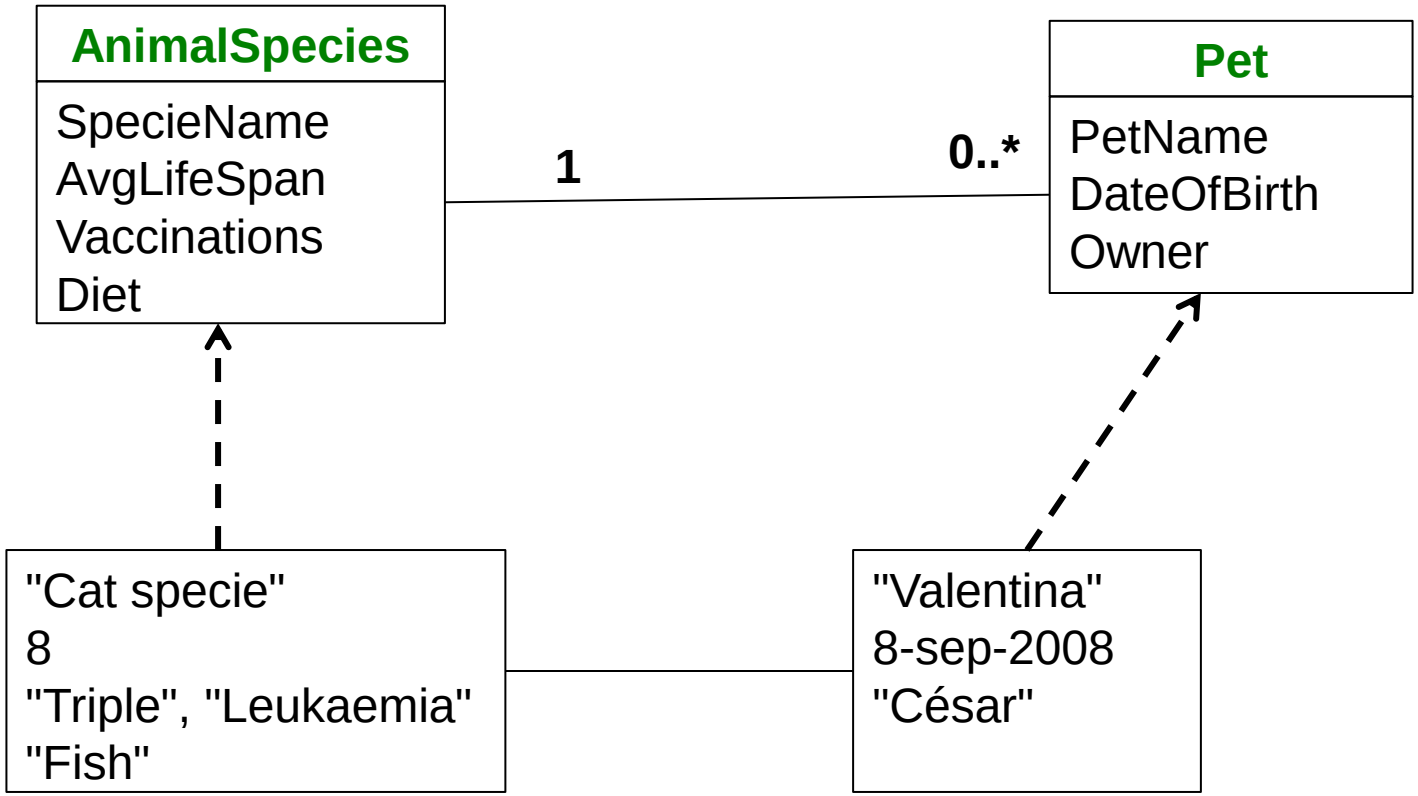



Cointe, Pierre: **Metaclasses are first class: The ObjVlisp Model**. *SIGPLAN Notices*. 22(12), 156–162 (1987).

# 2. Abstraction levels – the "Powertype" approach

- How to **simultaneously** create a generalization link AND an instantiation link
- How to a define attributes to Classes
- How to cross the strict instantiation levels

**Illustration (Source:** Gonzalez-Perez & Henderson-Sellers, ***Metamodelling for software engineering,*** p.38)

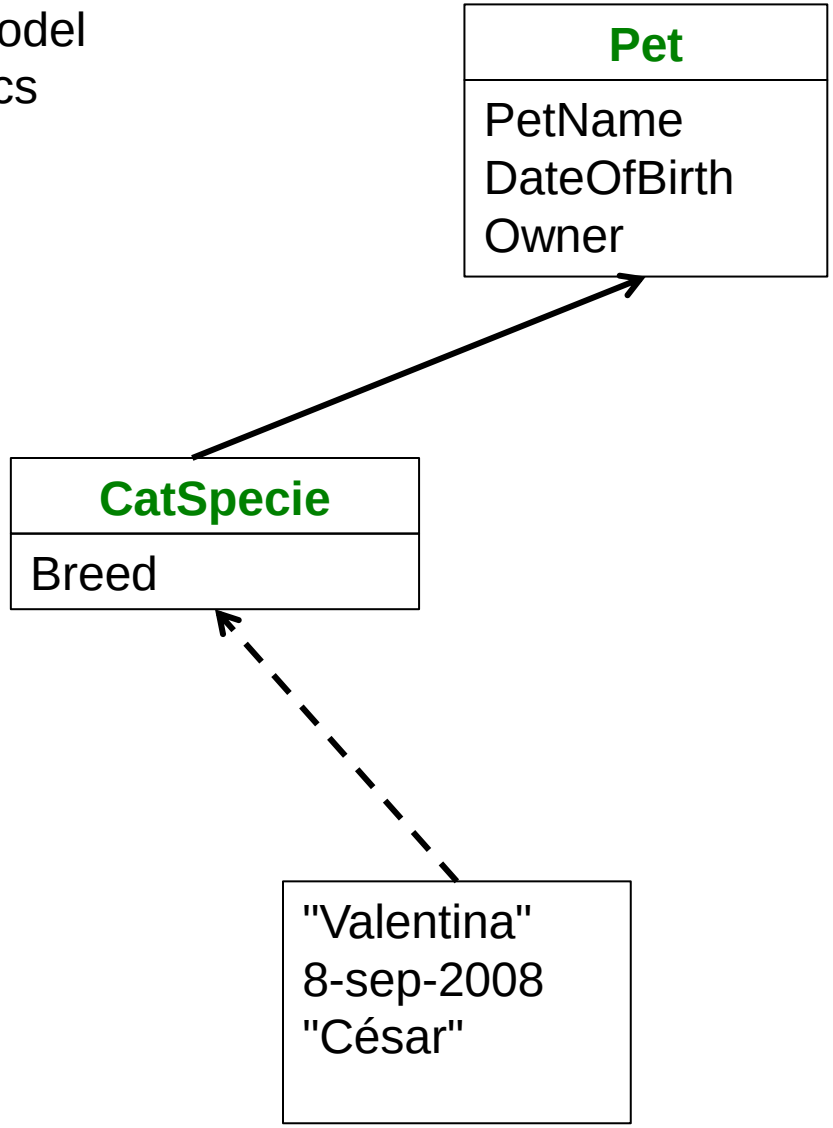



- **How to model species-specific characteristics, e.g. Breed ?**

# 2. Abstraction levels – the “Powertype” approach

**Illustration (Source:** Gonzalez-Perez & Henderson-Sellers, ***Metamodelling for software engineering,*** p.39)

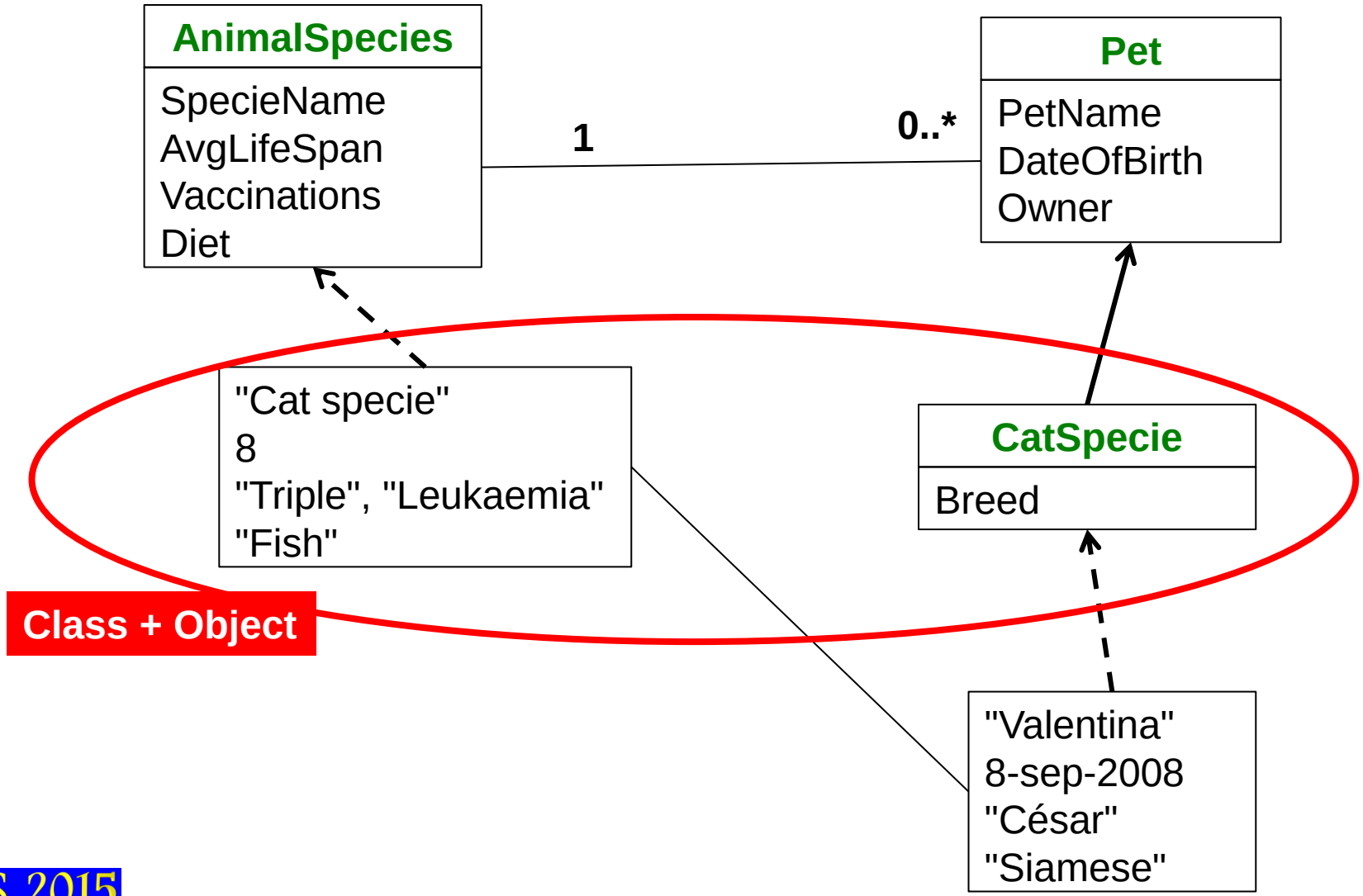
- Using specialization link to model species-specific characteristics



# 2. Abstraction levels – the “Powertype” approach

**Illustration (Source:** Gonzalez-Perez & Henderson-Sellers, ***Metamodelling for software engineering,*** p.39)

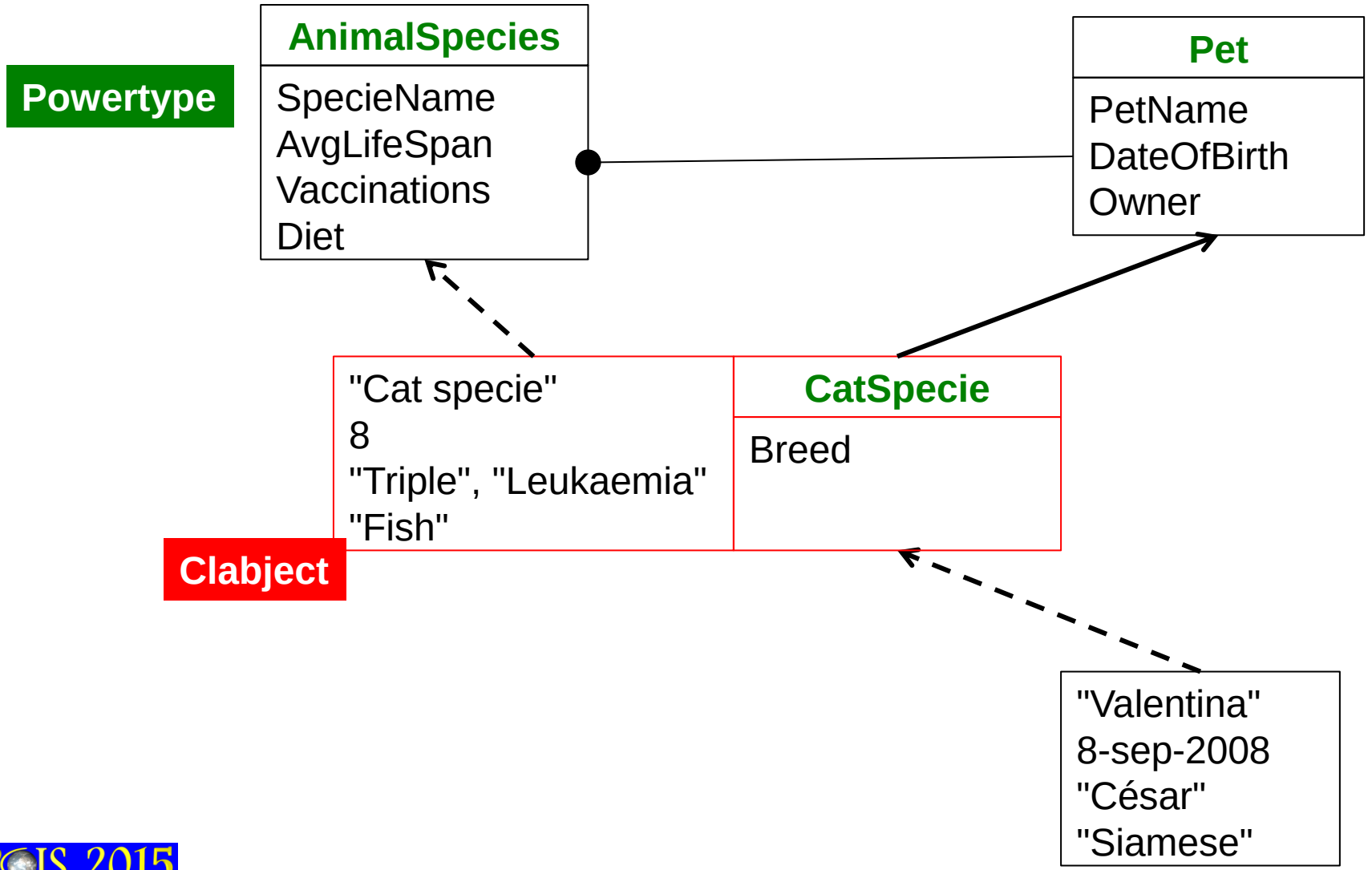
- Specific representation



# 2. Abstraction levels – the "Powertype" approach

**Illustration (Source:** Gonzalez-Perez & Henderson-Sellers, ***Metamodelling for software engineering,*** p.40)

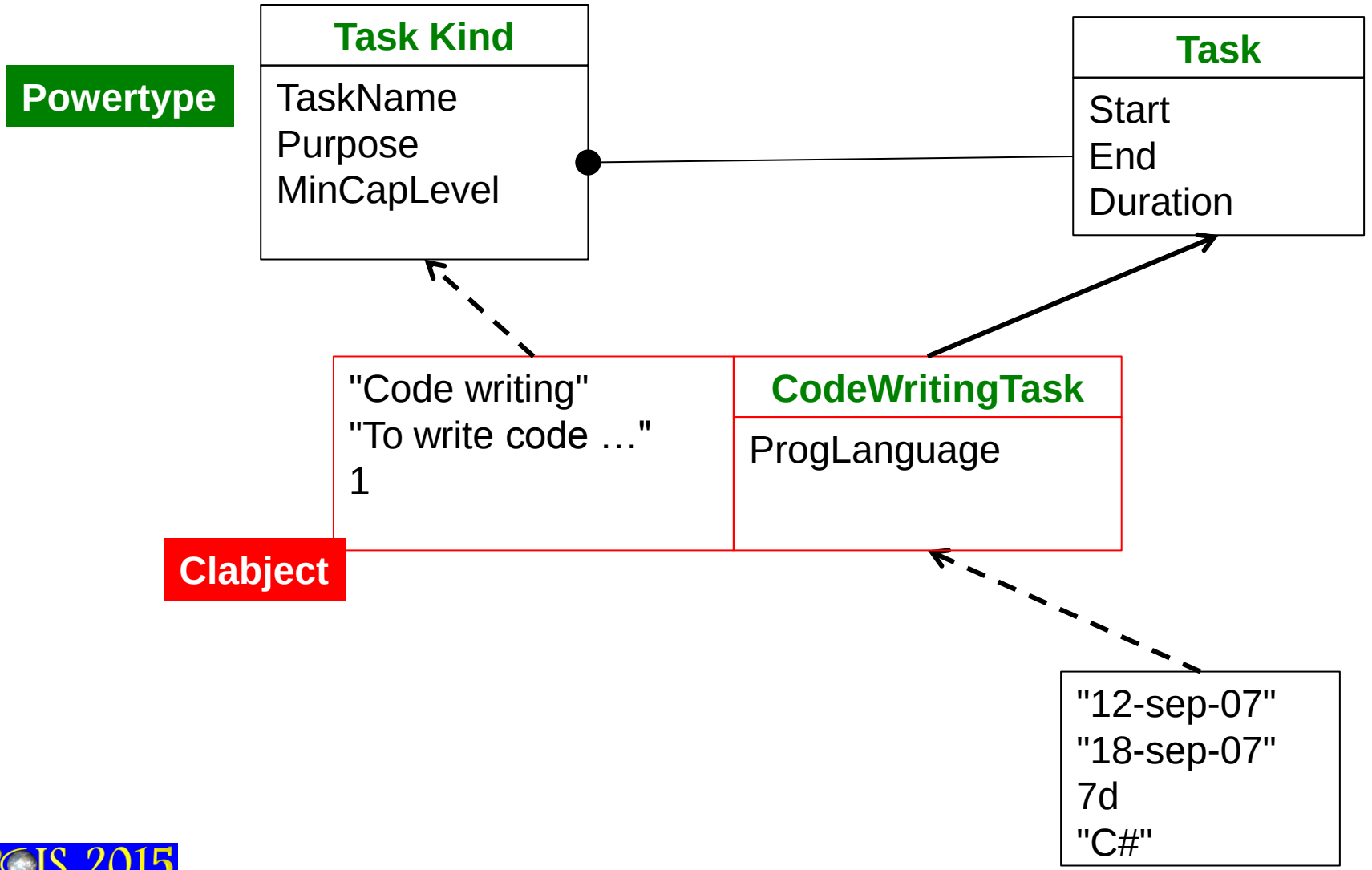
- PowerType and Clabject



# 2. Abstraction levels – the “Powertype” approach

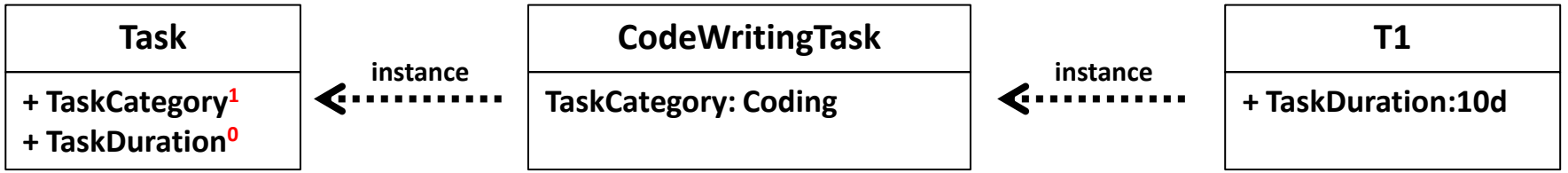
**Illustration (Source:** Gonzalez-Perez & Henderson-Sellers, ***Metamodelling for software engineering,*** p.41)

- PowerType and Clabject for method engineering



# 2. Abstraction levels – deep instantiation

- Proposal:
  - Add an indicator (*potency*) of the abstraction level at which the attribute is to be instantiated



# Agenda

1. Introductory examples
2. Abstraction levels and the instantiation problem
3. Computerized tools for meta-modeling
4. Illustration
   a) A Domain Specific Language (DSL) for IoT v3.0 Framework
   b) A DSL to model software development processes
   c) A DSL to specify web applications
5. Research challenges
6. Conclusion

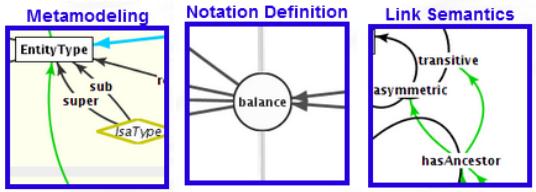
# 3. Computerized tools for meta-modeling

**Focus on 3 meta-modeling tools :**

(1) eclipse

(2) MetaCase


(3) ConceptBase.cc




ConceptBase.cc - A Database System for Metamodeling and Method Engineering

# 3. Computerized tools for meta-modeling

**Other meta-modeling tools :**

**(4)** ***metaDepth*: A framework for deep meta-modelling**

**(http://astreo.ii.uam.es/~jlara/metaDepth/)**

**(5)** 

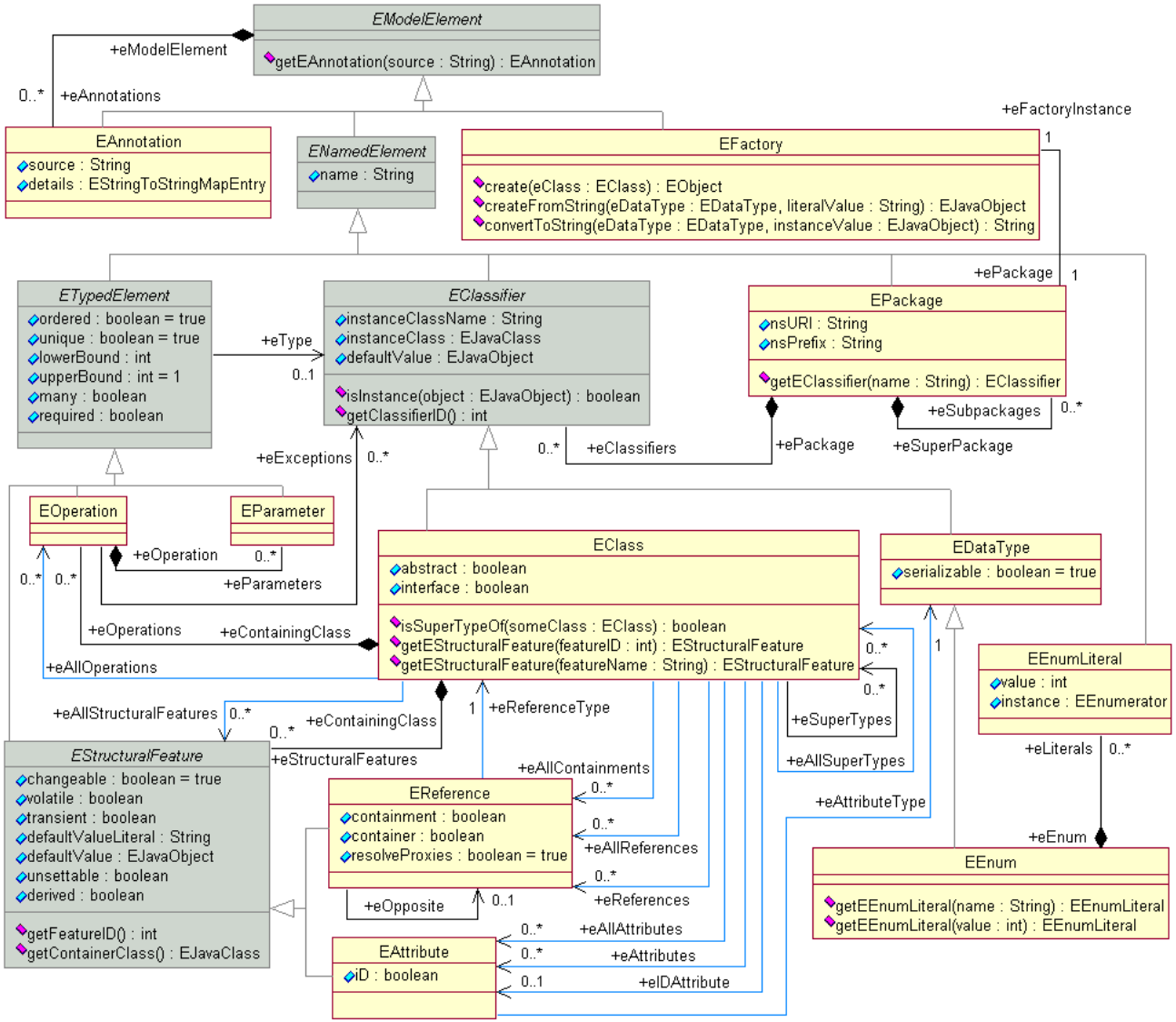
**ADO*xx* Meta Modelling Platform**
**(http://www.adoxx.org/live/home)**

**(6)** jastadd

**JastAdd (http://jastadd.org)**
**and**
**JastEMF**
**(https://code.google.com/a/eclipselabs.org/p/jastemf/)**
**for language engineering**

# 3. Computerized tools for meta-modeling

ECORE meta-modeling framework (MOF based)

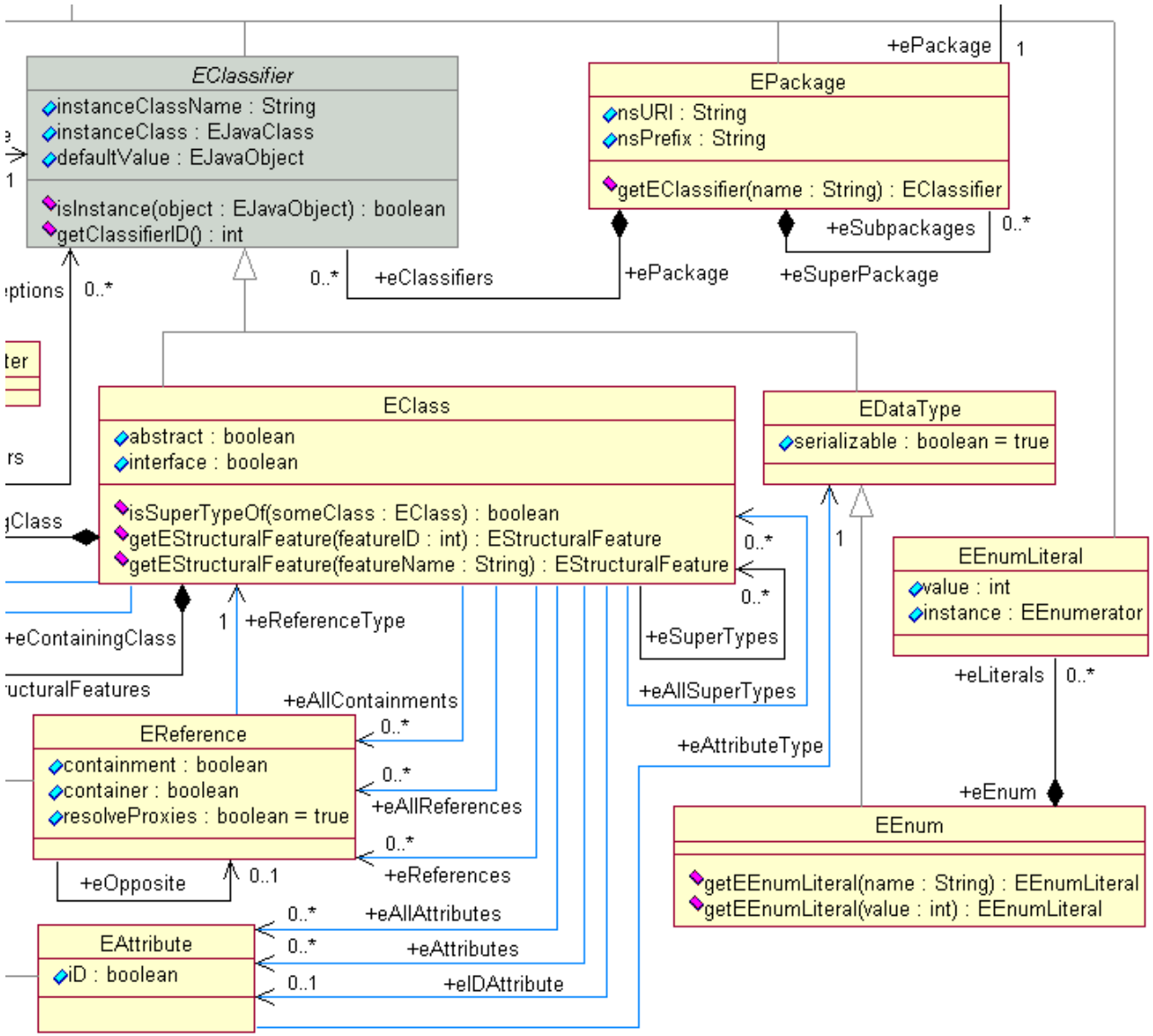



**Source** http://download.eclipse.org/modeling/emf/emf/javadoc/2.9.0/org/eclipse/emf/ecore/package-summary.html

# 3. Computerized tools for meta-modeling

eclipse

ECORE meta-modeling framework (MOF based)

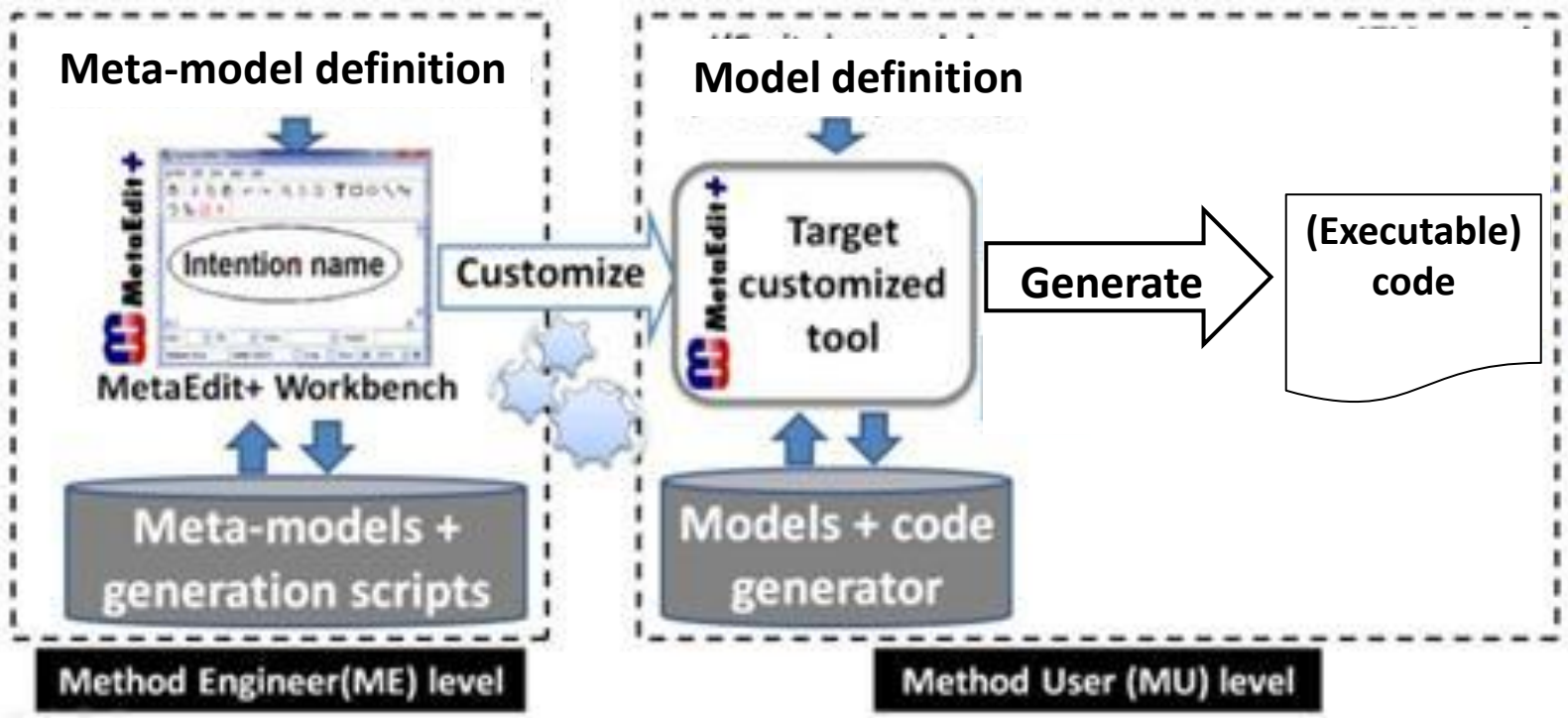



**Source** http://download.eclipse.org/modeling/emf/emf/javadoc/2.9.0/org/eclipse/emf/ecore/package-summary.html

# 3. Computerized tools for meta-modeling

## MetaEdit (1)

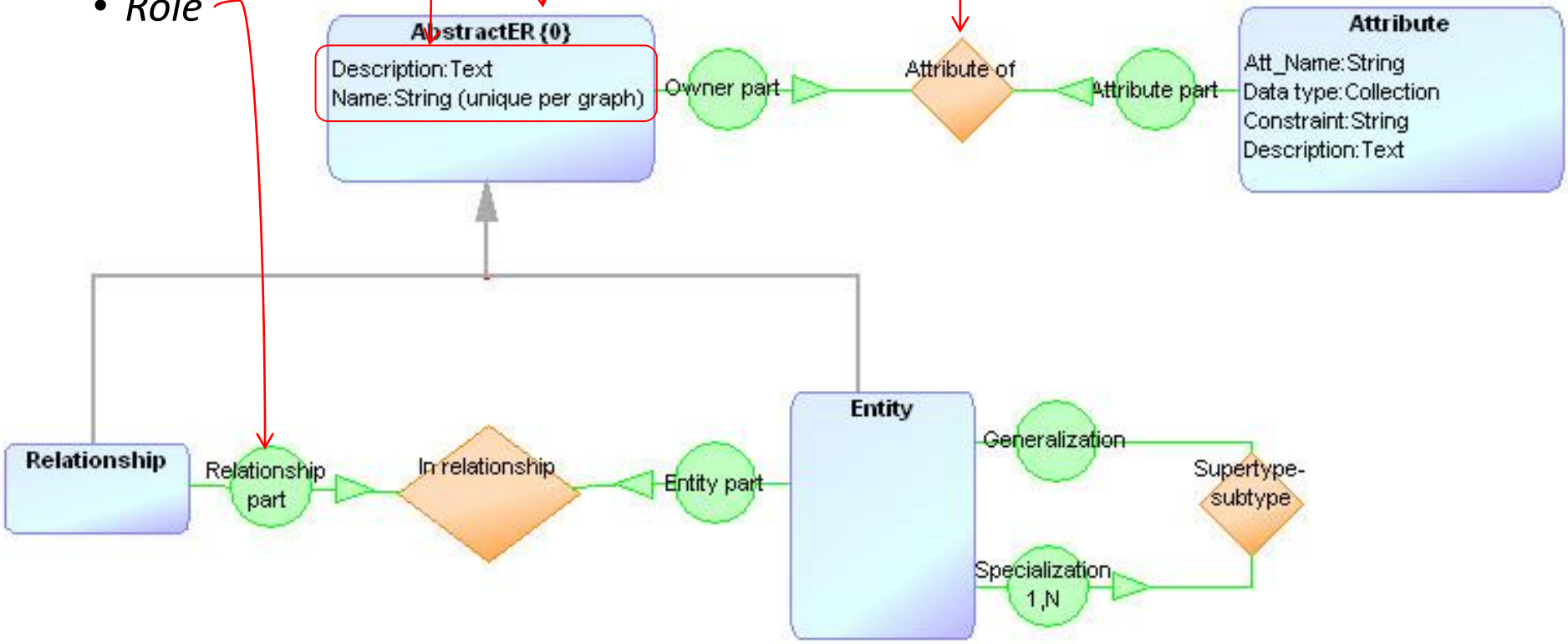
- A meta-CASE tool for automatic **customization** of CASE tools
- Result of research project at Jyväskyla (K. Lyytinen, M. Rossi, S. Kelly et al. - 1990')
- Actually commercialized by MetaCASE
- Targeted towards **Domain Specific Modeling**



# 3. Computerized tools for meta-modeling

## MetaEdit (2)

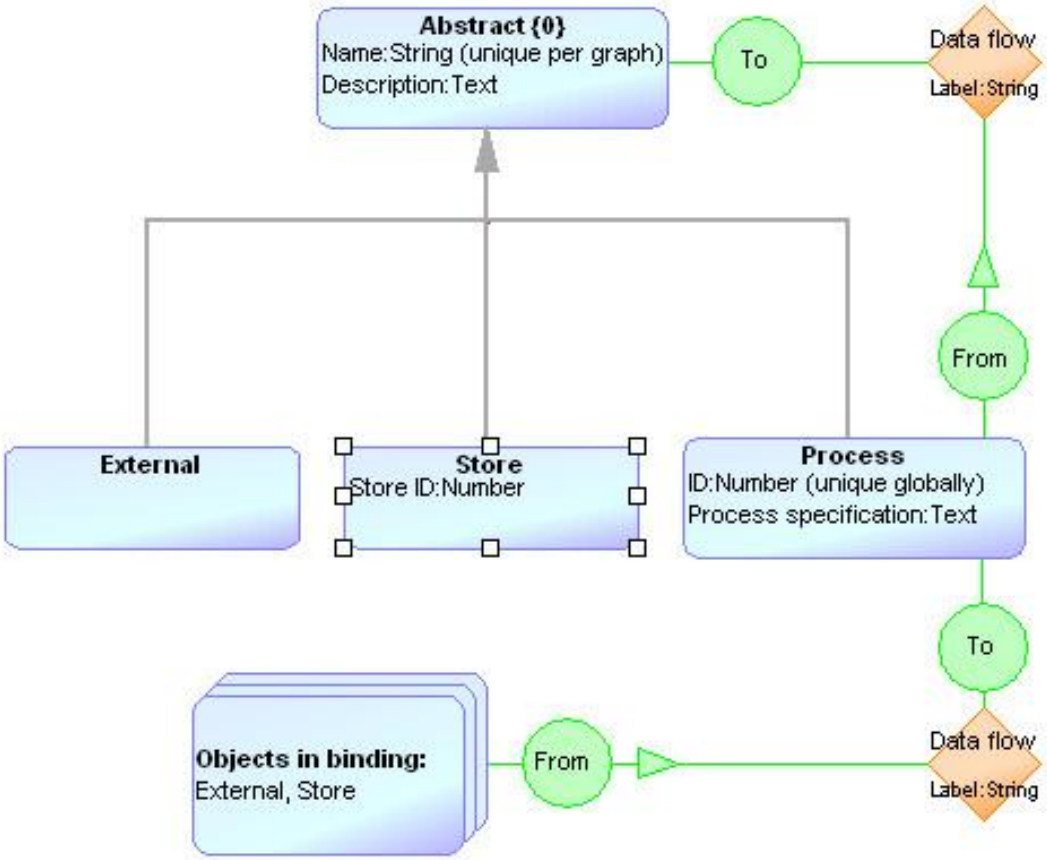
- Meta-modeling language: GOPRR
  - *Graph*
  - *Object*
  - *Property*
  - *Relationship*
  - *Role*

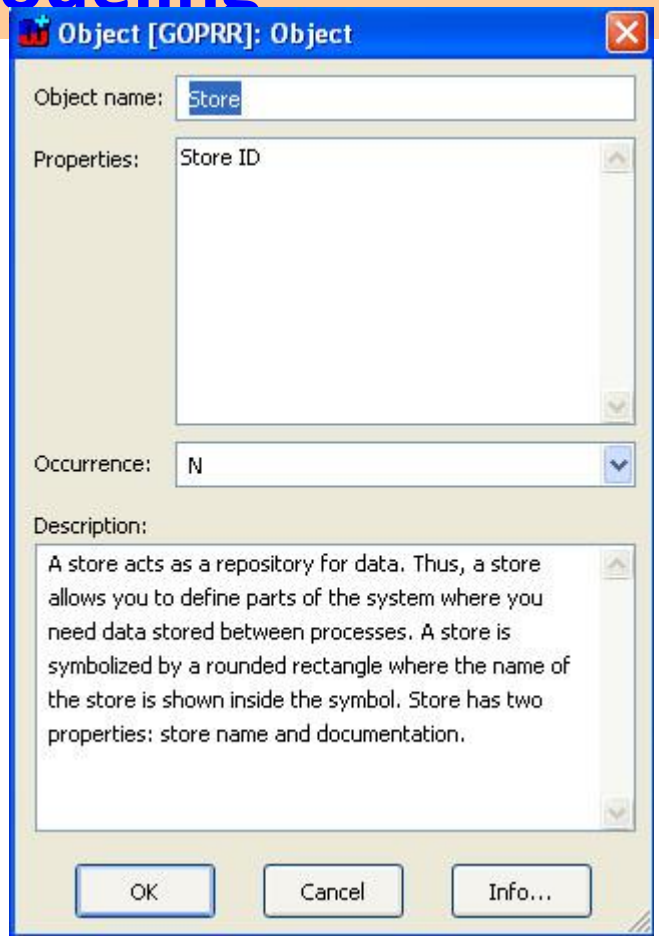


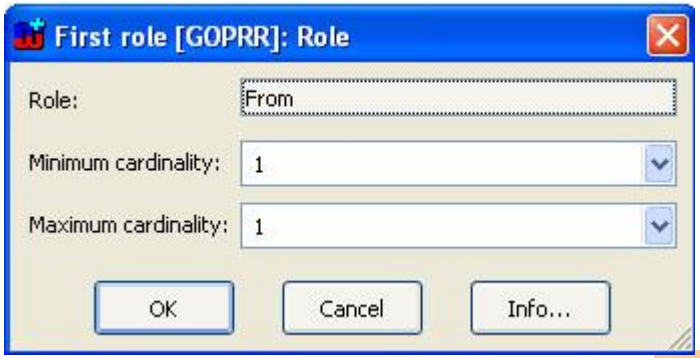
# 3. Computerized tools for meta-modeling

## MetaEdit (3)

- Meta-editing functionalities



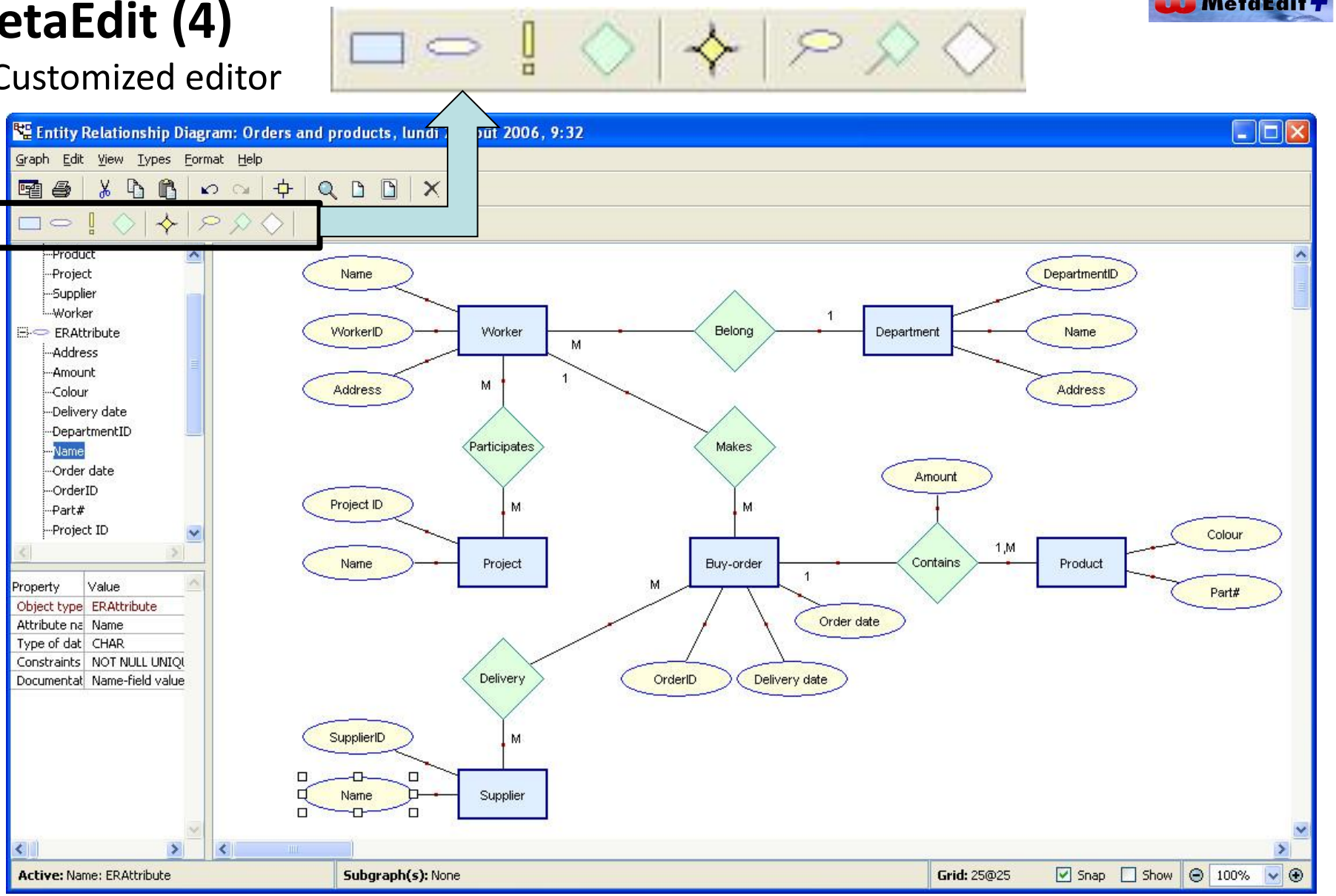



# 3. Computerized tools for meta-modeling

## MetaEdit (4)

- Customized editor



# 3. Computerized tools for meta-modeling

## MetaEdit (5)

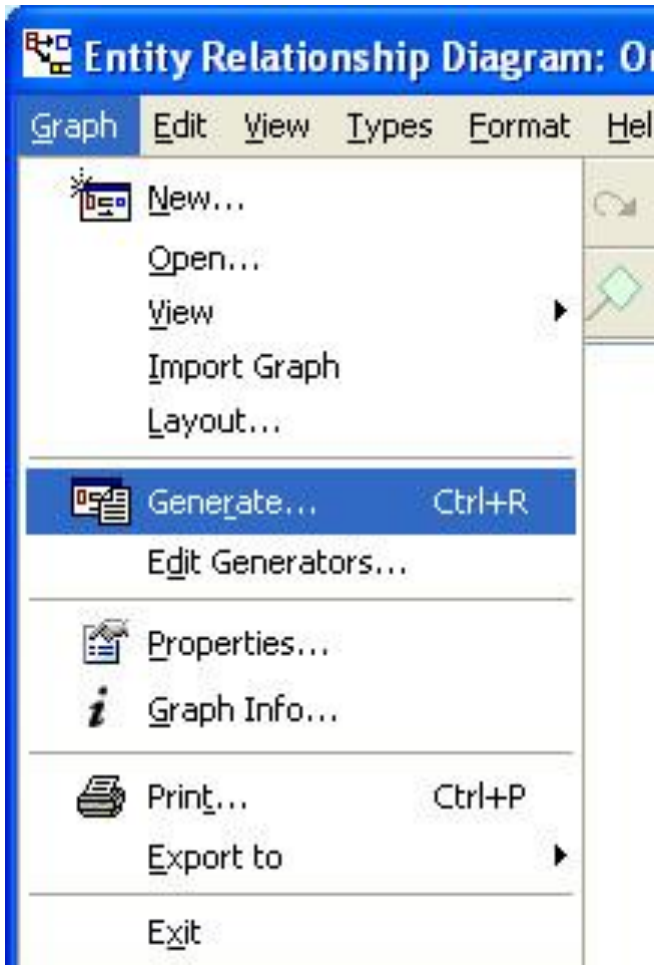
- MERL: A scripting language for generating code (i.e. flow of characters)

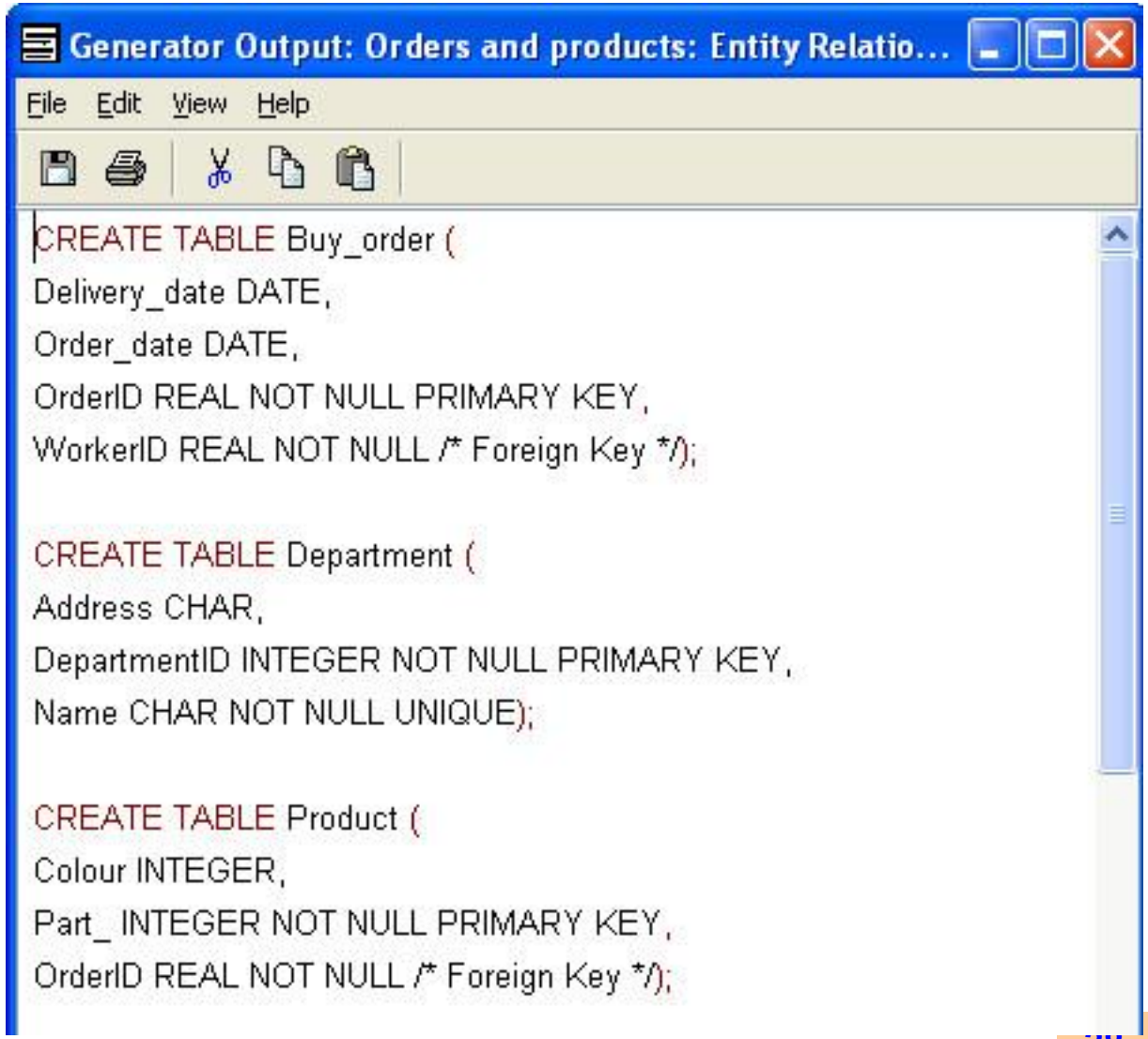
```
Report '_create related entity FKs'
/* USED FOR: ~Entity part or .Relationship */
/* INPUT: $targetTable : the for which the FK's are being created */
/* OUTPUT: foreignDeclarations (the foreign key sql declarations) */

$foreignDeclarations=''
dowhile .Entity {
   dowhile .ERAttribute; where :Constraints =~ '*PRIMARY*'{
      :Attribute name;%var; ' '
      :Type of data; ' '
      /* Assumption: FK's are never null */
      'NOT NULL /* Foreign Key */'

      variable 'foreignDeclarations' append
         'ALTER TABLE ' $targetTable; newline
         'ADD FOREIGN KEY (' :Attribute name;%var; ') '
         'REFERENCES ' :Entity name;1;%var;
                     '(' :Attribute name;%var; ');'
         newline
      close
      ',' newline
   }
   ', ' newline
}
endreport
```

# 3. Computerized tools for meta-modeling

## MetaEdit (6)

- Example in MERL for generating SQL code from E/R schema



# 3. Computerized tools for meta-modeling

## Conceptbase (1)

ConceptBase.cc

- A deductive database supporting *object-centered* TELOS modeling language
- Result from research project DAIDA (J. Mylopoulos, M. Jarke et al. - 1990')
- Continued as an open source project (M. A. Jeusfled)
- Targeted towards **Model Engineering**

**TELOS modeling language**

- Any element (class, object, attribute, link, constraint … ) is internally represented as a **predicate** in a deductive DB
- Attributes are first order objects i.e. they can have properties
- Attributes correspond to links between two elements
- Any object can be instantiated
  - Multiple instantiation ('Paul' in 'Employee' and 'Paul' in 'Student')
  - Multiple levels of instantiation
  - Multiple Generalisation/Specialisation (i.e. heritage)
  - Generalisation/Specialisation can co-exist with instantiation

ConceptBase.cc

## Conceptbase (2)

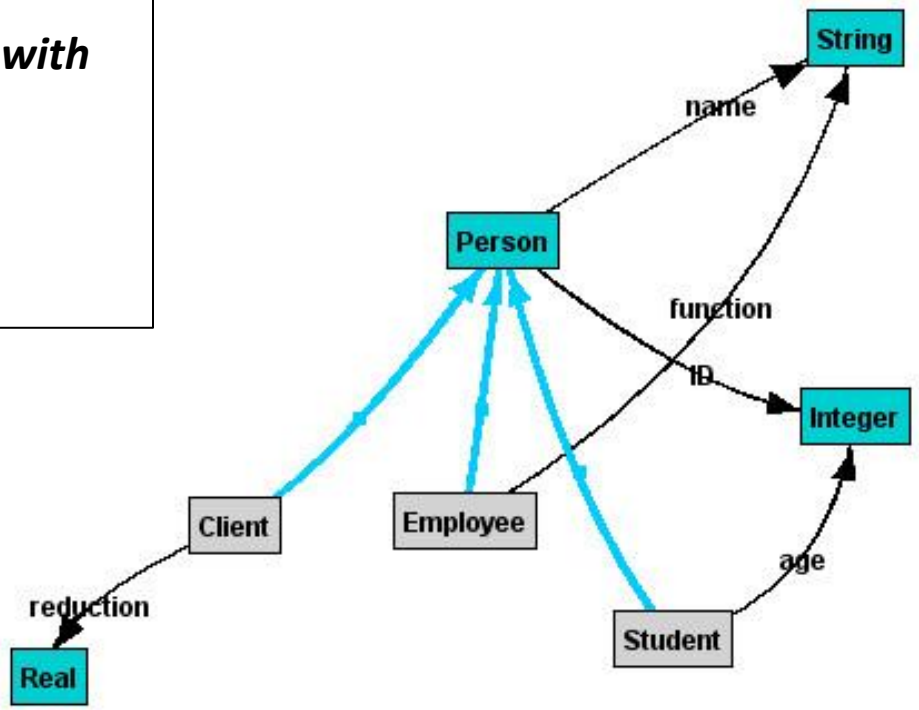
- Model llustration

*Person* ***in*** *Class* ***with***
***attribute***
*ID: Integer;*
*name:String*
***end***

*Client* ***isA*** *Person* ***with***
***attribute***
*reduction:Real*
***end***

*Employee* ***isA*** *Person* ***with***
***attribute***
*function: String*
*end*

*Student* ***isA*** *Person* ***with***
***attribute***
*age:Integer*
***end***

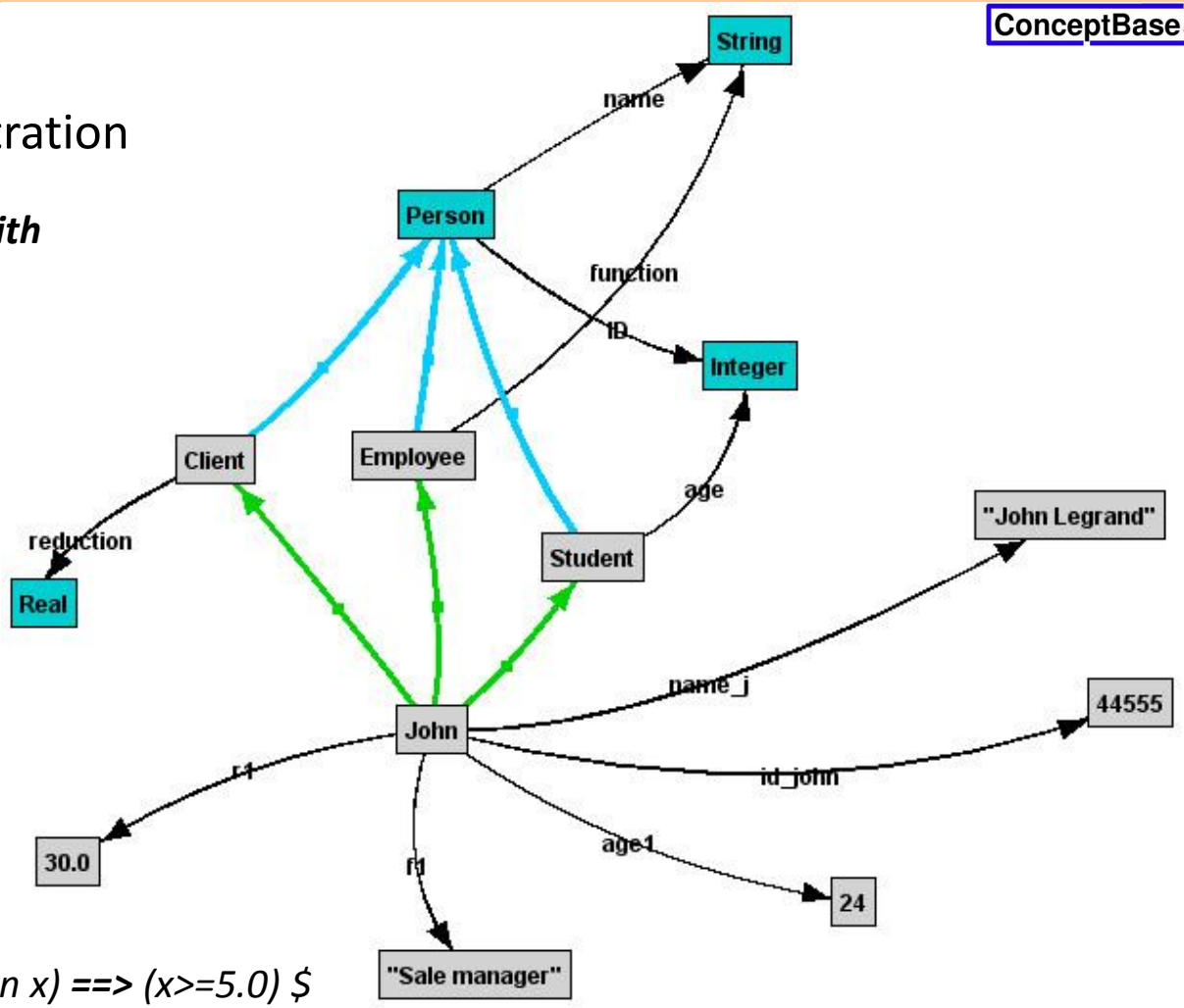


## Conceptbase (3)

- Multiple instantiation illustration

*John **in** Student, Employee, Client **with***
*ID*
*id_john: 44555*
*name*
*name_j: "John Legrand"*
*reduction*
*r1: 30.0*
*function*
*f1: "Sale manager"*
*age*
*age1: 24*
*end*

*Client **with constraint***
*reduction_OK:*
*$ **forall** c/Client x/Real (c reduction x) **==>** (x>=5.0) $*
*end*




***QueryClass** AllPeople **isA** Person **with***
***constraint***
*Allpeople_c: $ **exists** s/String (this nom s) $*
*end*

# 3. Computerized tools for meta-modeling

## Conceptbase (3)

- Implementing Class attributes and the “PowerType” concept

***AnimalSpecies*** *in Class, MetaClass with attribute*
*SpecieName: String;*
*AvgLifeSpan: Integer;*
*Vaccinations: String;*
*Diet: String*
*end*

***Pet*** *in Class, MetaClass with attribute*
*PetName: String;*
*PetDateOfBirth: String;*
*PetOwner: String*
*end*

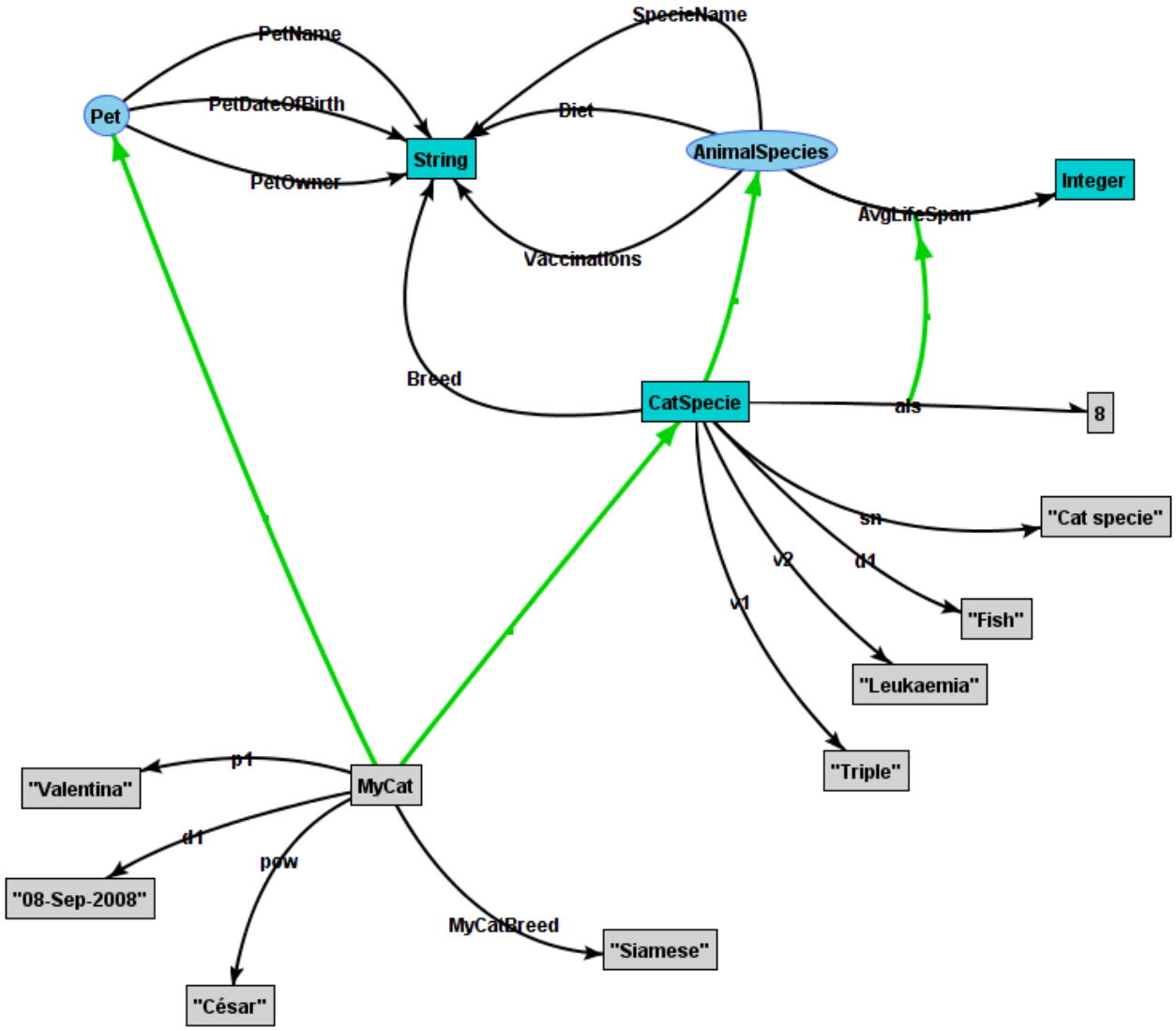
***CatSpecie*** *in* ***AnimalSpecies****, Class with attribute*
*Breed: String*
*SpecieName*
*sn: "Cat specie"*
*AvgLifeSpan*
*als: 8*
*Vaccinations*
*v1: "Triple"; v2: "Leukaemia"*
*Diet*
*d1: "Fish“ end*

***MyCat*** *in* ***CatSpecie****,* ***Pet*** *with*
*Breed*
*MyCatBreed:"Siamese"*
*PetName*
*p1: "Valentina"*
*PetDateOfBirth*
*d1: "08-Sep-2008"*
*PetOwner*
*pow: "César“ end*

# 3. Computerized tools for meta-modeling

## Conceptbase (3)

- Implementing Class attributes and the “PowerType” concept



# Agenda

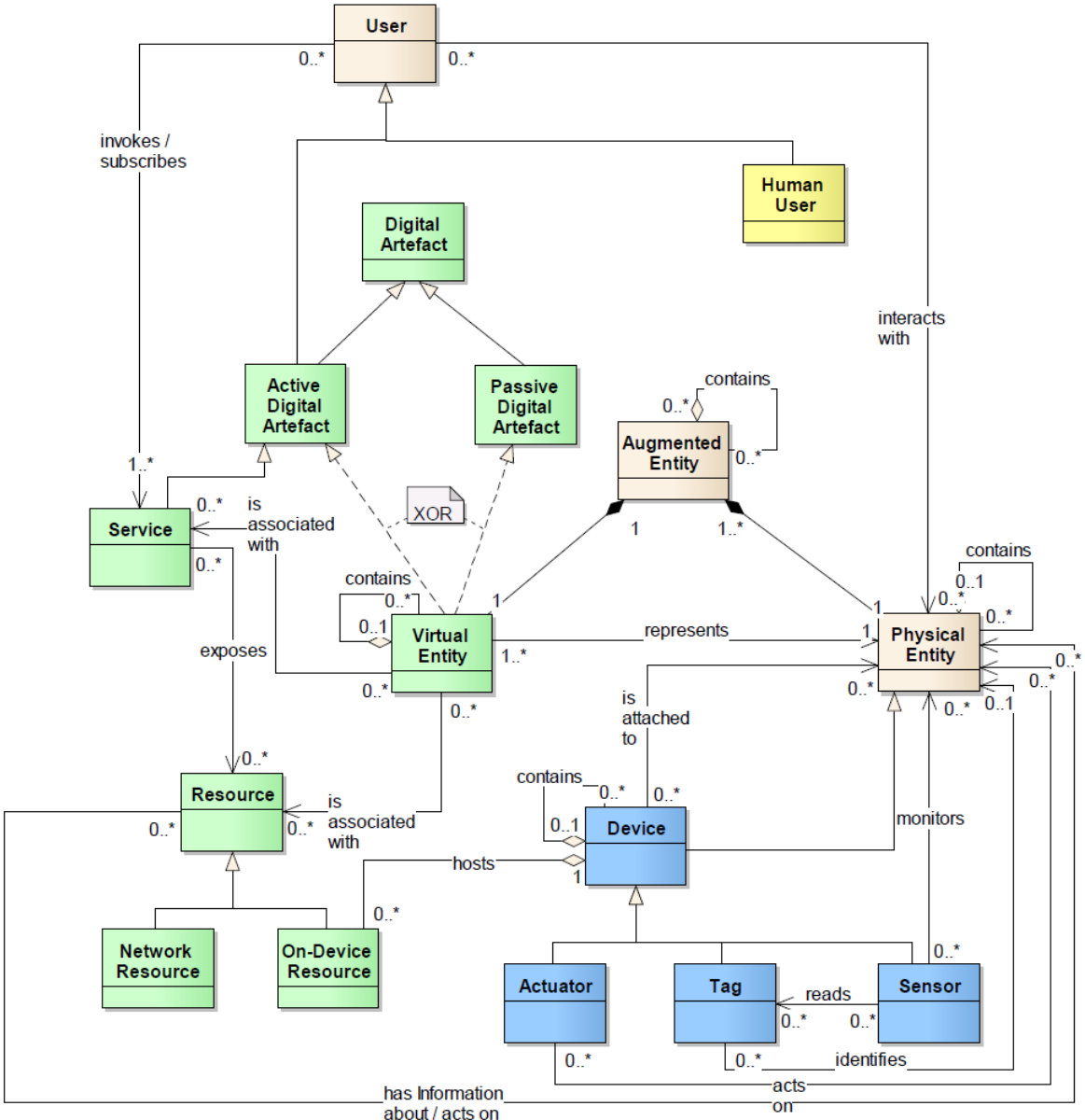
1. Introductory examples
2. Abstraction levels and the instantiation problem
3. Computerized tools for meta-modeling
4. Illustration
   a) A Domain Specific Language (DSL) for IoT v3.0 Framework
   b) A DSL to model software development processes
   c) A DSL to specify web applications
5. Research challenges
6. Conclusion

# 4. Meta-modeling illustration examples

**Source :** http://www.iot-a.eu/public/public-documents/d1.5/at_download/file

## Exercise 1 : Internet of Things framework

The IoT ARM is exemplified with a use-case scene : A load carrier is equipped with sensors and can communicate with other devices in terms of wireless radio technology. With this hardware, every load carrier continuously measures its environmental parameters and sends all measurements via the embedded event service to the mobile phone of the truck driver.



# 4. Meta-modeling illustration examples

**Source :** http://www.iot-a.eu/public/public-documents/d1.5/at_download/file

## Exercise 1 : Internet of Things framework

*"For example, Ted is a truck driver transporting highly sensitive orchids to a retail store. After loading the orchids on his truck, he attaches an array of sensors to the load carriers in order to measure the temperature. While having lunch, Ted forgot that by turning of the engine, air condition for the transported goods highly sensitive orchids - shuts off, too. The temperature inside the truck starts rising and when it reaches a predefined critical level inside, one of its sensors notices this and its node sends an emergency signal to Ted's IoT-Phone. On the IoT-Phone's display, Ted can now see that the orchids are in danger so he rushes back to the vehicle and turns the air condition on.*

*The IoT-Phone also keeps track of any alert messages it receives from the load carriers and saves this message history for future inspection in a way that cannot be altered. When the truck reaches the retail store for delivery, the sensor history is transferred to the store's enterprise system and the sensors authenticate themselves as being un-tampered."*

➢**How to model this system using the IoT framework ?**

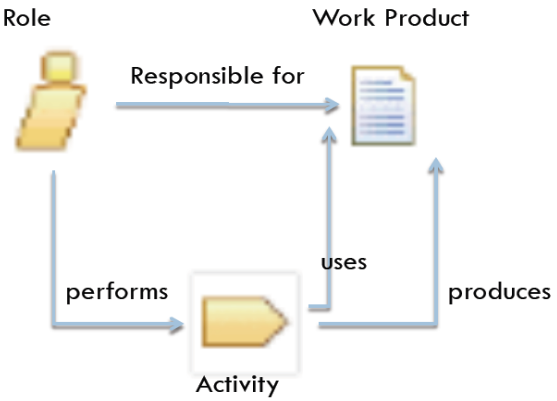
# 4. Meta-modeling illustration examples

**<u>Exercise 2</u>: Method engineering**

The design and development team in a company has defined a set of engineering methods according to the nature of projects. Among others, and in the case of small size projects (number of end-users <= 10, number of data tables <= 15, number of business process <=5), they have defined the following simple method:

*"The Requirements Engineering team (1 or 2 engineers) uses administrative documents and conduct interviews with end-users to elicit general requirements, i.e a natural language description of the required system (including NF requirements), and process definitions represented as BMPN diagrams. The design team (2 to 4 engineers) uses the process models together with the general description to sketch a system architecture, i.e. an UML class diagram, use case diagrams and sequence diagrams. The development team (3 to 5 developers) build the system according to the BPMN models and the system architecture, and deliver an operational implementation composed of Java code for web services, a relational database and a workflow script. The test team (2 to 3 persons) verify the quality of the system using a set of test cases they have designed and in accordance with the NF requirements".*

➢ **How such method can be modeled using a subset of SPEM and the *MetaEdit*+ workbench?**



# 4. Meta-modeling illustration examples

**Exercise 3: Web questionnaires [Source : LWC2014 - Language Workbenches Comparison, http://www.languageworkbenches.net/]**

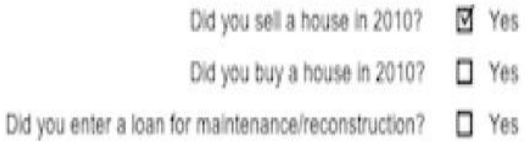
Forms-based software for data collection has found application in various areas, including scientific surveys, online course-ware and guidance material to support the auditing process. As an overall term for this kind of software applications we use the term "questionnaire". In this exercise, the goal is to create a simple DSL, called QL, for describing questionnaires. Such questionnaires are composed of sequential forms, each form is characterized by conditional entry fields and (spreadsheet-like) dependency-directed computation. The following example presents a possible textual representation of a simple questionnaire with only one form :

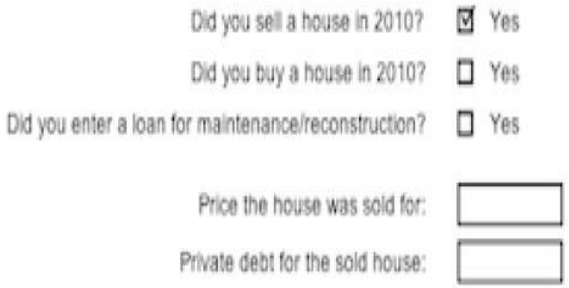
```
questionnaire HouseOwning
form Box1HouseOwning
{   hasSoldHouse: "Did you sell a house in 2010?" Boolean
    hasBoughtHouse: "Did you by a house in 2010?" Boolean
    hasMaintLoan: "Did you enter a loan for maintenance/reconstruction?" Boolean
    if (hasSoldHouse)
    {    sellingPrice: "Price the house was sold for:" money
         privateDebt: "Private debts for the sold house:" money
         valueResidue: "Value residue:" money(sellingPrice - privateDebt)
    }
}
```

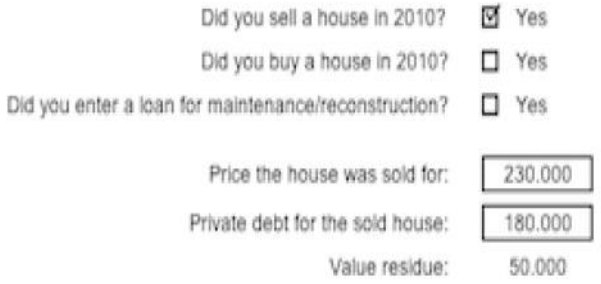
# 4. Meta-modeling illustration examples

**<u>Exercise 3</u>: Web questionnaires [Source : LWC2014 - Language Workbenches Comparison, http://www.languageworkbenches.net/]**

This simple form should generate into a GUI which allows the following user interaction:




➢**How such language can be designed and implemented ?**

# Agenda

1. Introductory examples
2. Abstraction levels and the instantiation problem
3. Computerized tools for meta-modeling
4. Illustration
   a) A Domain Specific Language (DSL) for IoT v3.0 Framework
   b) A DSL to model software development processes
   c) A DSL to specify web applications
5. Research challenges
6. Conclusion

# 6. Research challenges in meta-modeling

## Behavior meta-modeling

• *Behavior* perspectives are generally missing in software engineering meta-models

⇒ Important knowledge about modeling languages is lacking

⇒ Essential for **tools designers** and **method engineers**

⇒ For a process modeling language, "behavior" perspective inquire on its ***executable/operational semantics***

# 6. Research challenges – behavior modeling

## Research goal

- How to express the operational semantics for a modeling language?
- How to design and build enactment engines for a given modeling language?
  - ***Maintainability*** *and* ***portability*** *are central issues*

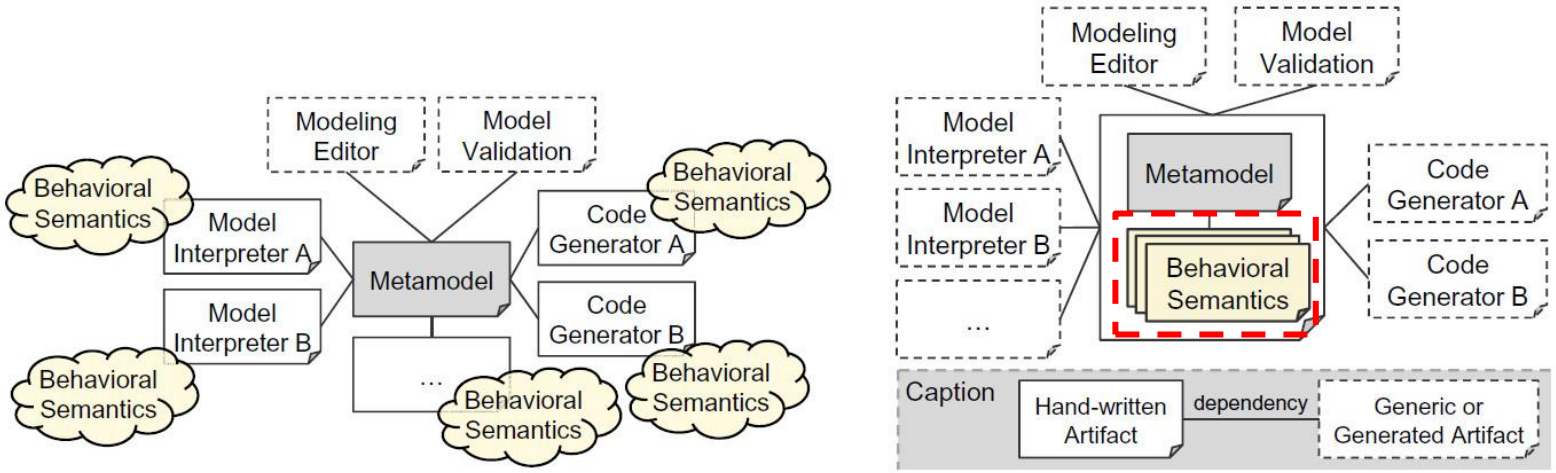



**Fig. 1.** Specification of the behavioral semantics

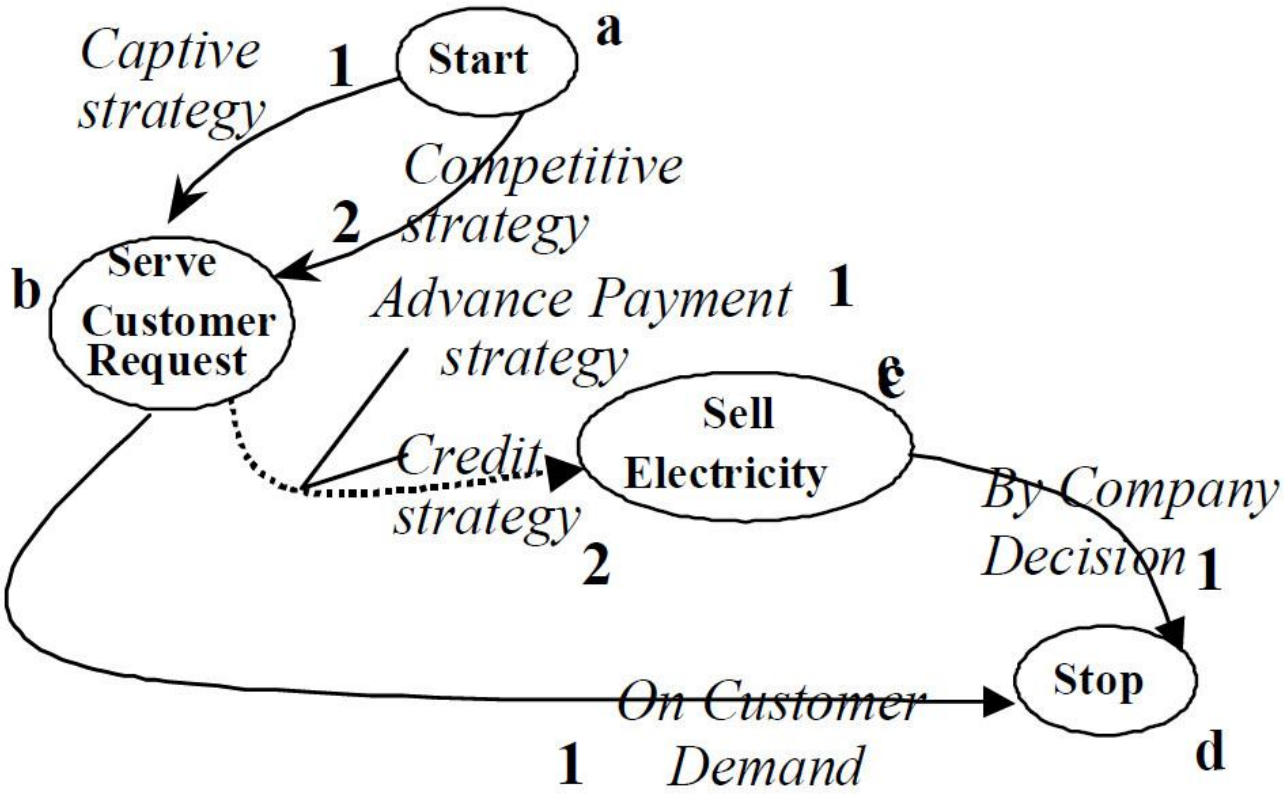
[**Source** :T. Mayerhofer et al., "xMOF: Executable DSMLs Based on fUML", SLE'2013 ]

# 6. Research challenges – behavior modeling

## Behavior meta-modeling

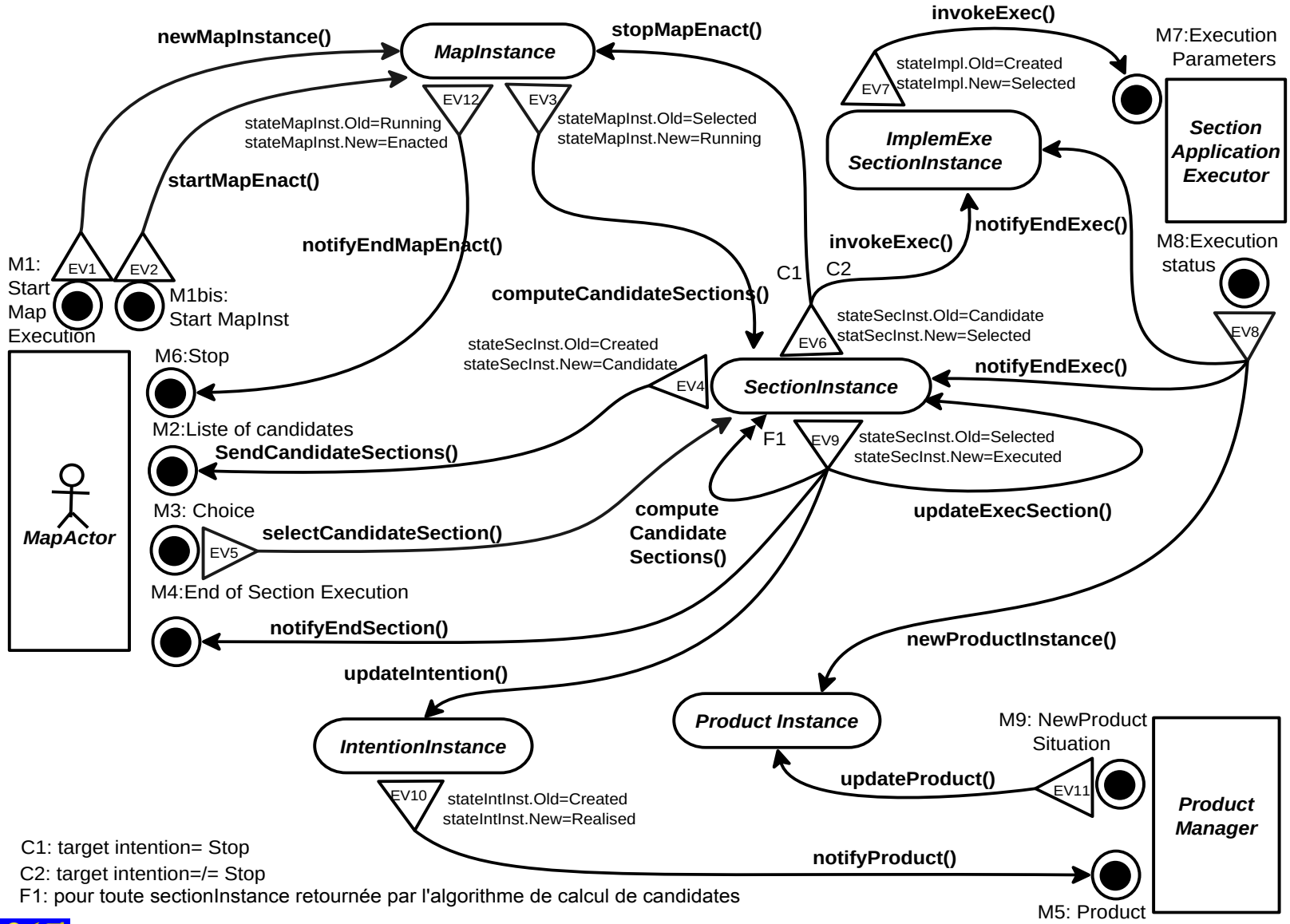
- Context : MAP process modeling notation

Example



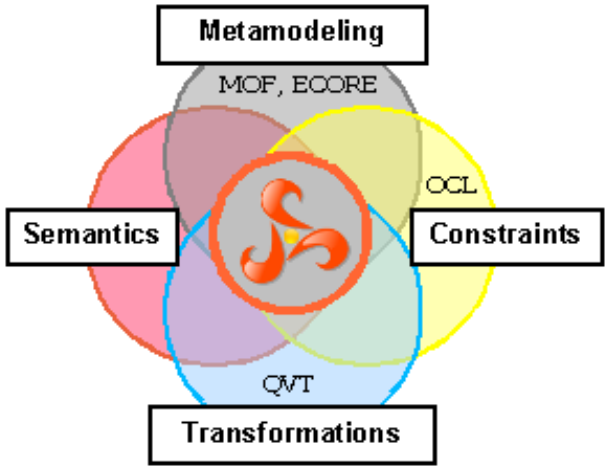
# 6. Research challenges – behavior modeling

## Event-based behavior meta-modeling



# 6. Research challenges in meta-modeling

## Behavior meta-modeling




- Other approaches

  ⇒ KERMETA


For an example, see http://en.wikipedia.org/wiki/Kermeta

# 6. Research challenges in meta-modeling

## Behavior meta-modeling

⇒ xMOF (Mayerhofer et al., 2013)

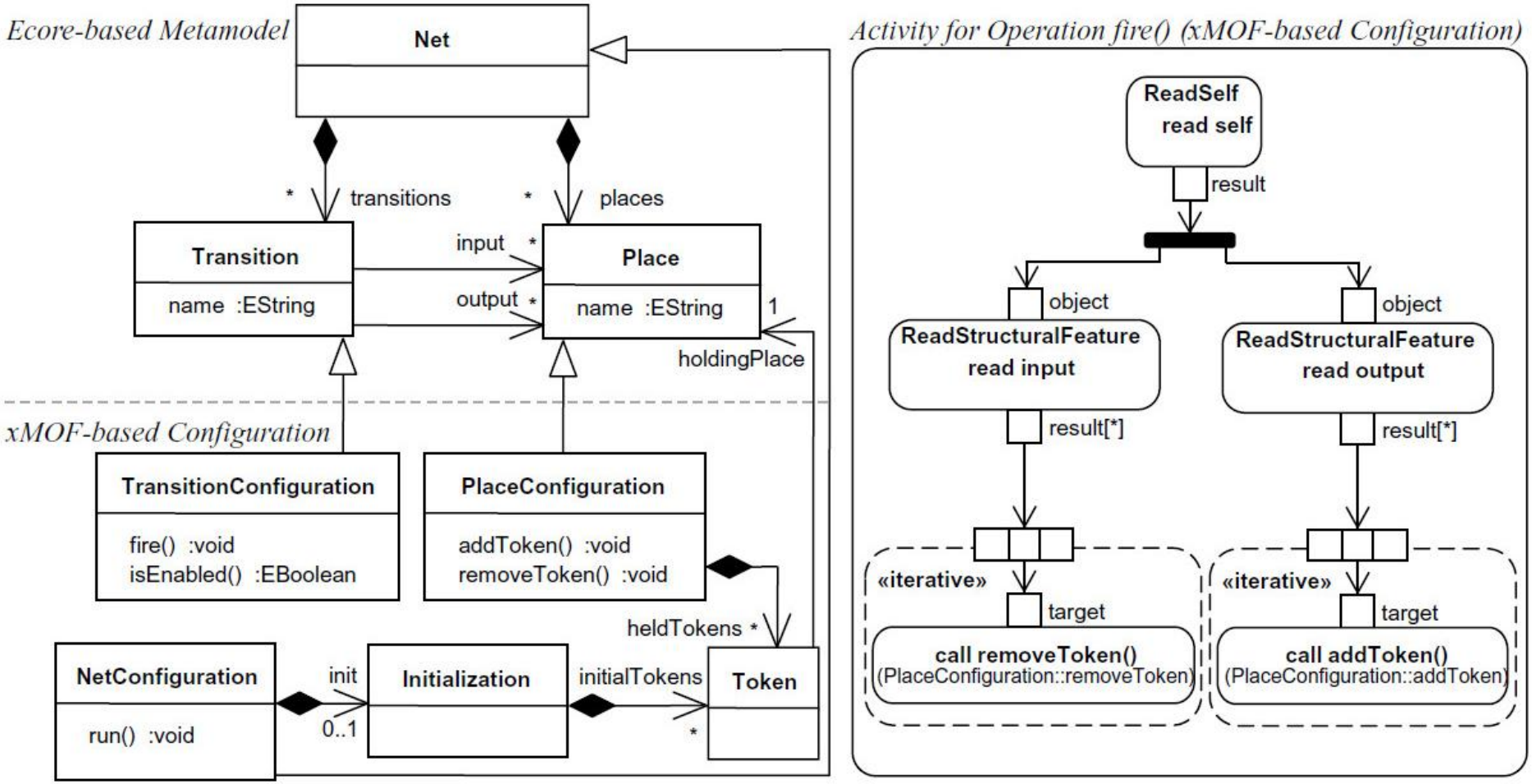



**Fig. 6.** Specification of the Petri net DSML

# Agenda

1. Introductory examples
2. Abstraction levels and the instantiation problem
3. Computerized tools for meta-modeling
4. Illustration
   a) A Domain Specific Language (DSL) for IoT v3.0 Framework
   b) A DSL to model software development processes
   c) A DSL to specify web applications
5. Research challenges
6. Conclusion

# 6. Conclusion

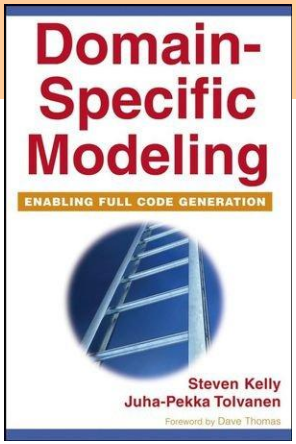
- No well established and recognized standard for meta-modeling
- Tool support is complicated
- … OMG's **MOF** and IBM **Eclipse** emerge as "market" leaders for UML & Java
- **MetaEdit**: leader for DSM engineering tools
- **Conceptbase**: most powerful and theoretically sound modeling language
- **Semantics** for process meta-models still to be defined
- Beyond DSM, few empirical studies on meta-modeling applications

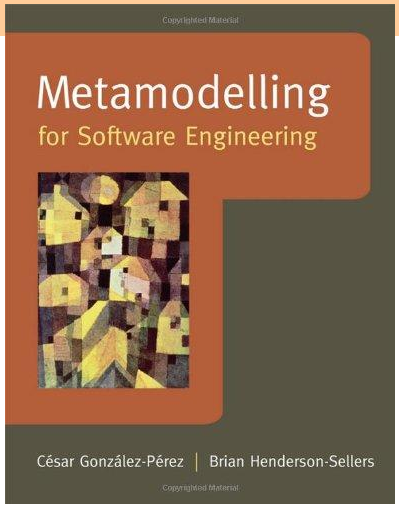
# References

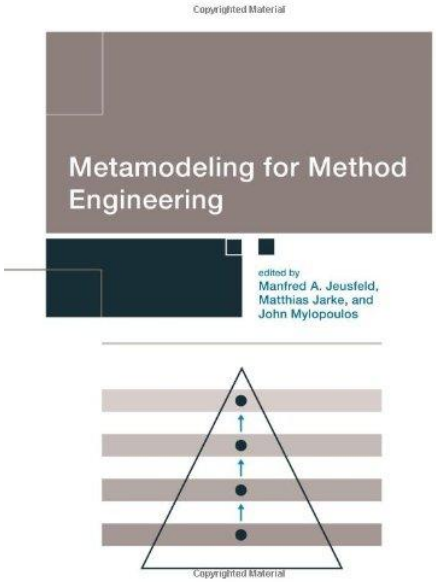

- S. Kelly et J.-P. Tolvanen, ***Domain-specific modeling: enabling full code generation***. Hoboken, N.J.: Wiley-Interscience: IEEE Computer Society, 2008.

- C. Gonzalez-Perez et B. Henderson-Sellers, ***Metamodelling for software engineering***. Wiley Publishing, 2008.

- Jeusfeld, M., Jarke, M., Mylopoulos, J.: ***Metamodeling for method engineering***. The MIT Press, (2009).


## Links

- MOF homepage: http://www.omg.org/mof/
- ISO/IEC 24744:2007 (Software Engineering -- Metamodel for Development Methodologies): http://www.iso.org/iso/catalogue_detail.htm?csnumber=38854
- OPEN (Object-oriented Process, Environment and Notation) : http://www.open.org.au/
- Metacase: http://www.metacase.com/
- Conceptbase: http://conceptbase.sourceforge.net/
- Kermeta: http://www.kermeta.org

# References


- Atkinson, C., Kühne, T., « **The Essence of Multilevel Metamodeling** », in: Gogolla, M. et Kobryn, C. (éd.) «UML» 2001 — The Unified Modeling Language. Modeling Languages, Concepts, and Tools. LNCS, vol.2185, pp. 19-33. Springer, 2001.
- C. Atkinson et T. Kühne, **« Model-Driven Development: A Metamodeling Foundation »**, IEEE Software, vol. 20, no 5, p. 36–41, 2003.
- C. Gonzalez-Perez et B. Henderson-Sellers, « **A powertype-based metamodelling framework** », Software and Systems Modeling, vol. 5, no 1, p. 72-90, 2006.
- F. Jouault, F. Allilaire, J. Bézivin, et I. Kurtev, « **ATL: A model transformation tool** », Science of Computer Programming, vol. 72, no 1-2, p. 31-39, 2008.
- A. Gargantini, E. Riccobene, et P. Scandurra, « **A semantic framework for metamodel-based languages** », Automated Software Engineering, vol. 16, no 3-4, p. 415-454, 2009.
- M. Jarke, M. Jeusfeld, H. Nissen, C. Quix, et M. Staudt, « **Metamodelling with Datalog and Classes: ConceptBase at the Age of 21** », in Proceedings 2nd Int. Conf. on Object Databases (ICOODB'09), M. Norrie et M. Grossniklaus, Éd. 2010, p. 95–112.
- J. de Lara et E. Guerra, « **Deep Meta-modelling with MetaDepth** », in Objects, Models, Components, Patterns, J. Vitek, Éd. Springer Berlin Heidelberg, 2010, p. 1-20.
- B. Bryant, J. Gray, M. Mernik, P. Clarke, R. France, et G. Karsai, « **Challenges and Directions in Formalizing the Semantics of Modeling Languages** », Computer Science and Information Systems, vol. 8, no 2, p. 225-253, 2011.

# References (cont'd)


- J. Sprinkle, B. Rumpe, H. Vangheluwe, et G. Karsai, **« Metamodelling »**, in Model-Based Engineering of Embedded Real-Time Systems, vol. 6100, H. Giese, G. Karsai, E. Lee, B. Rumpe, et B. Schätz, Éd. Berlin/Heidelberg: Springer, 2011, p. 57-76.
- A. El Kouhen, C. Dumoulin, S. Gerard, et P. Boulet, « **Evaluation of Modeling Tools Adaptation** », 2012. Available at http://hal.archives-ouvertes.fr/hal-00706701
- B. Henderson-Sellers, O. Eriksson, C. Gonzalez-Perez, et P. J. Ågerfalk, **« Ptolemaic Metamodelling? The Need for a Paradigm Shift »**, in Progressions and Innovations in Model-Driven Software Engineering:, V. G. Díaz, J. M. C. Lovelle, B. C. P. García-Bustelo, et O. S. Martínez, Éd. IGI Global, 2013, p. 90-146.
- O. Eriksson, B. Henderson-Sellers, et P. J. Ågerfalk, « **Ontological and linguistic metamodelling revisited: A language use approach** », Information and Software Technology, vol. 55, no 12, p. 2099-2124, 2013.
- C. Bürger, S. Karol, C. Wende, et U. Aßmann, « **Reference Attribute Grammars for Metamodel Semantics** », in Software Language Engineering, vol. 6563, B. Malloy, S. Staab, et M. van den Brand, Éd. Berlin/Heidelberg: Springer, 2011, p. 22-41.
- T. Mayerhofer, P. Langer, M. Wimmer, et G. Kappel, « **xMOF: Executable DSMLs Based on fUML** », in Software Language Engineering (SLE'13), M. Erwig, R. F. Paige, et E. V. Wyk, Éd. Springer International Publishing, 2013, p. 56-75.
- B. Neumayr, M. A. Jeusfeld, M. Schrefl, et C. Schütz, « **Dual Deep Instantiation and Its ConceptBase Implementation** », in Advanced Information Systems Engineering, M. Jarke, J. Mylopoulos, C. Quix, C. Rolland, Y. Manolopoulos, H. Mouratidis, et J. Horkoff, Éd. Springer International Publishing, 2014, p. 503-517.
- R. F. Paige, D. S. Kolovos, et F. A. C. Polack, **« A tutorial on metamodelling for grammar researchers »**, Science of Computer Programming, vol. 96, Part 4, p. 396-416, 2014.

# THANK YOU FOR YOUR ATTENTION !

Questions? Comments? Insights?

Homepage http://www-public.it-sudparis.eu/~assar/

Linked in http://fr.linkedin.com/pub/sa%C3%AFd-assar/4/68a/66a

slideshare http://fr.slideshare.net/SaidAssar/

twitter https://twitter.com/Said_Assar/