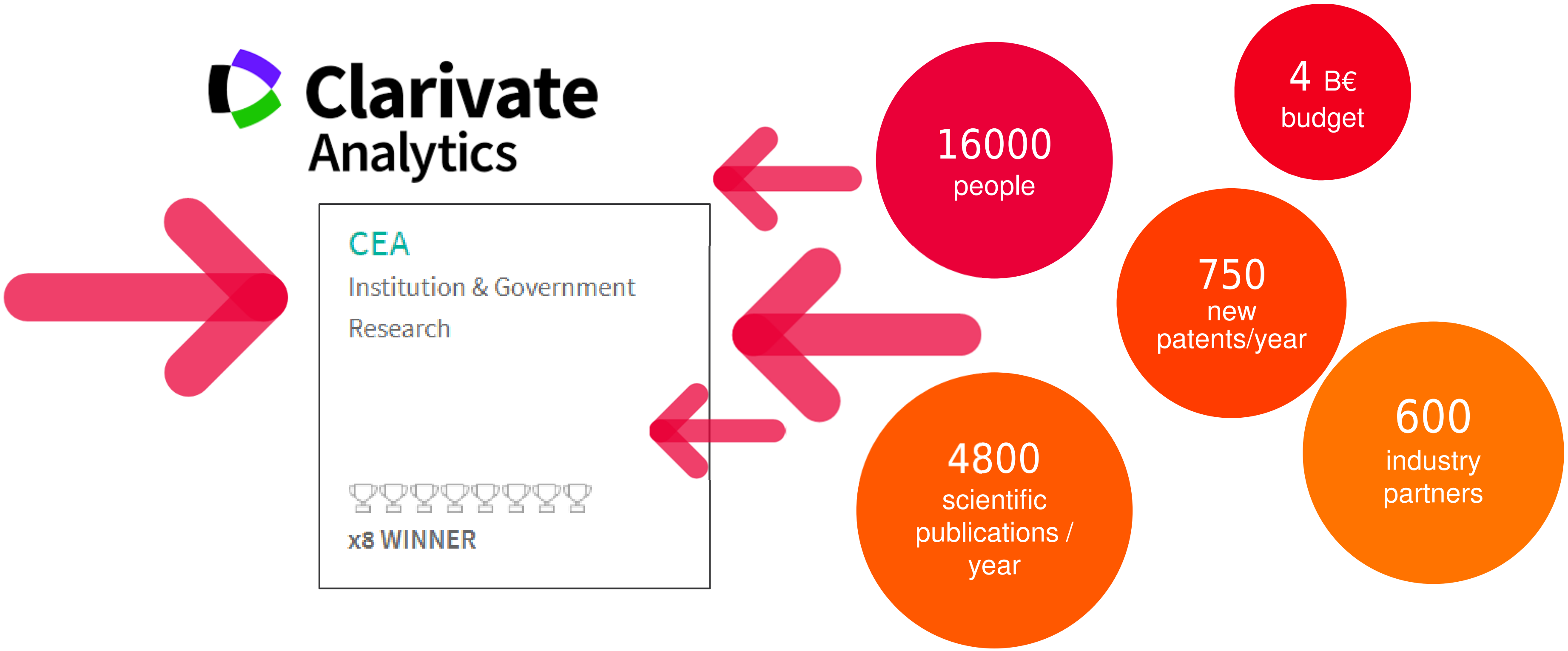
# MBE & AI: WHAT ABOUT SYNERGIES?

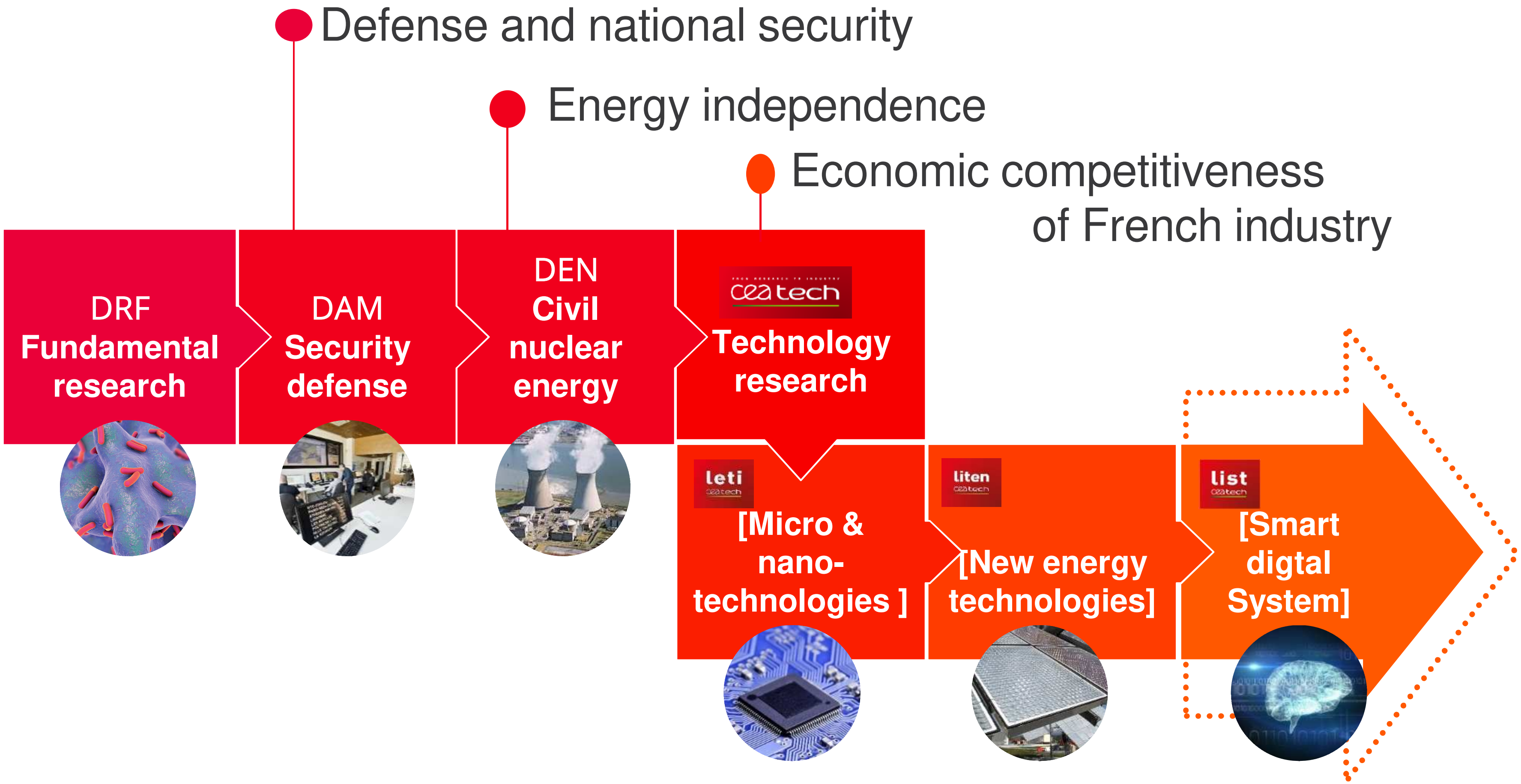
***Sébastien Gérard (Director of Research at CEA List, sebastien.gerard@cea.fr)***
**Département d'Ingénierie des Logiciels et des Systèmes**
*Software and System Engineering Department*

# CEA, a French public research organisation

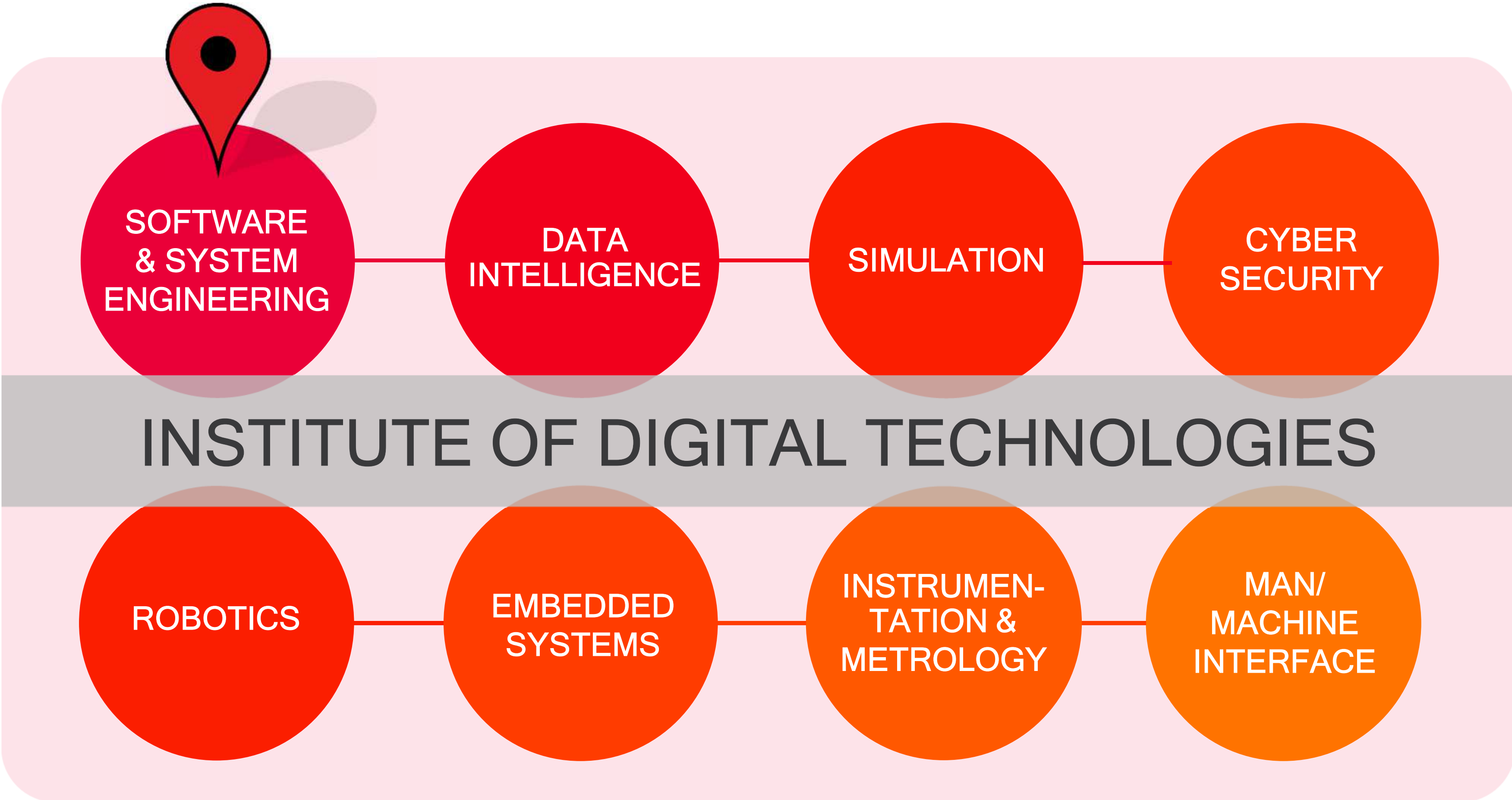
**“Top 25 Global Innovators – Government”** ranking of Reuter:
➔ in 2019, CEA is ranked at the 3rd place worldwide in terms of innovative public research organisation.



# The missions of CEA



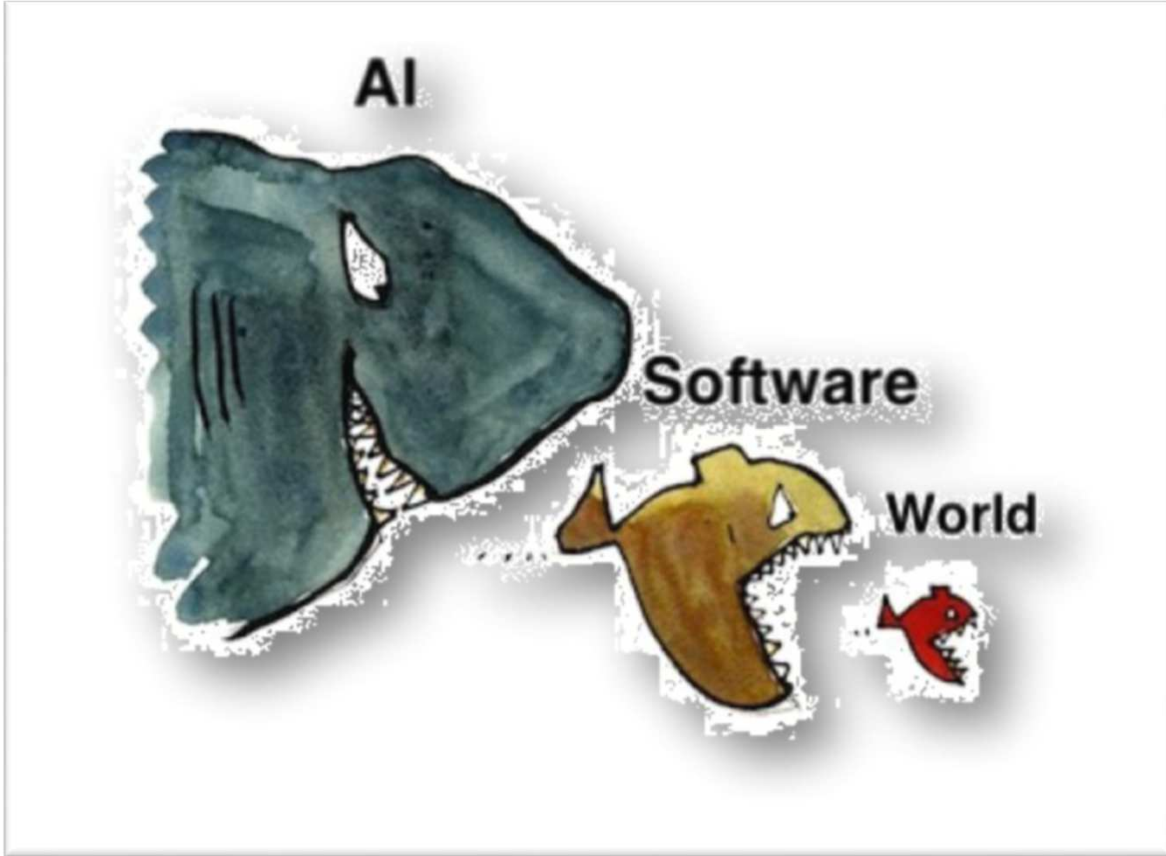
# Expertise areas of the CEA List institute



# Before going further…


This presentation will not be a tutorial on AI…

… fortunately, because I am not an expert in AI!




**"Software is eating the world, but AI is going to eat software"**
Jensen Huang, Nvidia CEO (2017)

*(Credit to Jordi Cabot for the slide)*

# More and more publications and communications around how AI for SE....

Computer Languages, Systems & Structures 50 (2017) 159–176

Computer Languages, Systems & Structures

User-story driven development of multi-ag… process fragment for agile methods

Yves Wautelet[a,*], Samedi Heng[b], Soreangsey Kiv[b], Manu…

FOR APPLICATION DEVELOPMENT & DELIVERY PROFESSIONALS

How AI Will Change Software Developm…

by Diego Lo Giudice with Christopher Mines, Amanda LeClair, Rowa…

October 13, 2016 | Updated: November 2, 201…

Universal Programmability - How A…

Walter F. Tichy, Mathias Landhäußer, Sven J…

Science of Computer Programming 191 (2020) 102416

Science of Computer Programming

Automated reasoning framework for traceability management of system of systems

Bedir Tekinerdogan[a,*], Ferhat Erata[b,c]

2018 IEEE 26th International Requirements Engineering Conference

Automated Validation of Requirement Reviews: A Machine Learning Approach

Maninder Singh
Department of Computer Science and Engineering
North Dakota State University
Fargo, USA
maninder.singh@ndsu.edu

Engineering Applications of Artificial Intelligence 26 (2013) 1631–1640

Engineering Applications of Artificial Intelligence

Linking software testing results with a machine learning approach

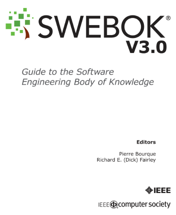
Alexandre Rafael Lenz, Aurora Pozo, Silvia Regina Vergilio*

2018 ACM/IEEE 6th International Workshop on Realizing Artificial Intelligence Synergies in Software Engineering

Semi-automatic Generation of Active Ontologies from Web Forms for Intelligent Assistants

Martin Blersch, Karlsruhe Institute of Technology
Mathias Landhäußer, thingsTHINKING GmbH
Thomas Mayer, Karlsruhe Institute of Technology

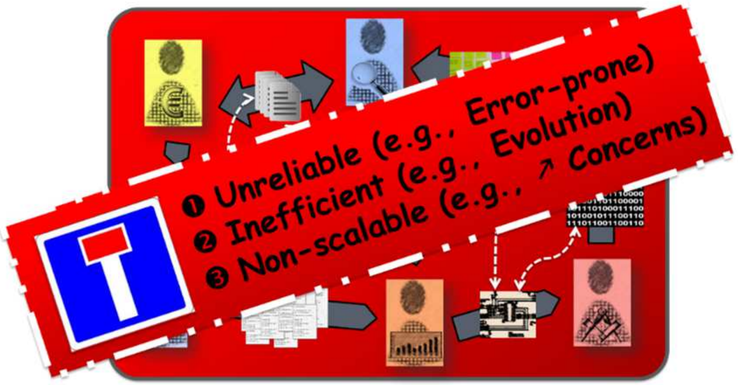
# AI can contribute all along the software development life cycle e.g.,

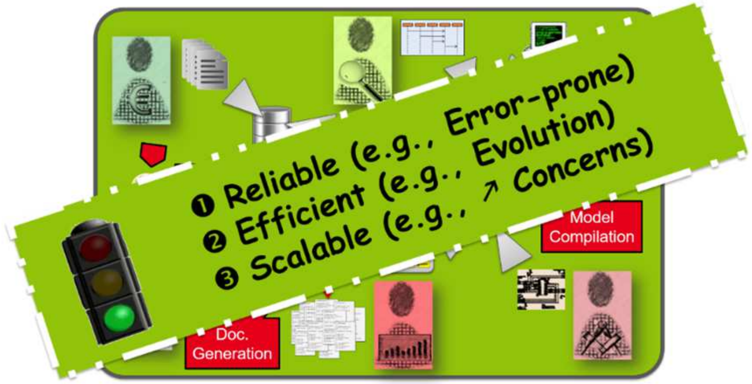


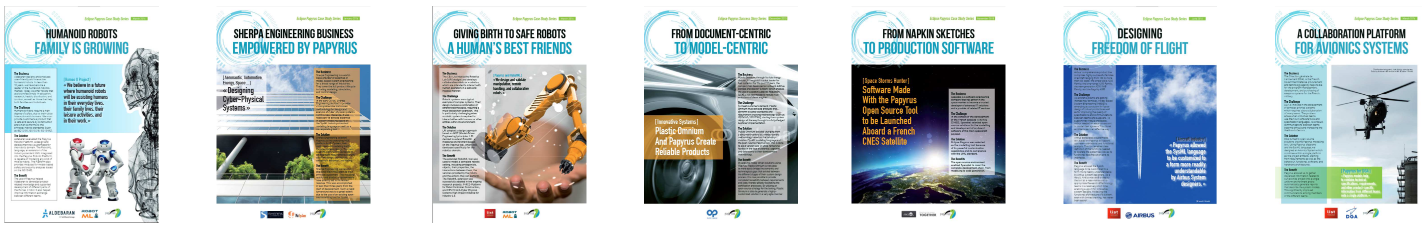

- **Software requirements**
  - NLP + Ontology used to generation of conceptual models from text-based specification:
    → e.g., class-based, use-case, statemachine, or even interaction.
  - Machine learning:
    → Requirement review, glossary extraction, requirements classifications & prioritizations, etc.
- **Software design**
  - Bots (Sw agents) to collaboratively build conceptual models,
  - Machine learning for classifying web images as UML static diagrams,
  - Recommender that suggest design solutions to the deigners.
- **Software construction**
  - Machine learning for code generation,
  - NLP for reverse engineering.
- **Etc.**

# MBE, an enabler of the digital transition for system & software engineering

**From documentware… …to modelware**

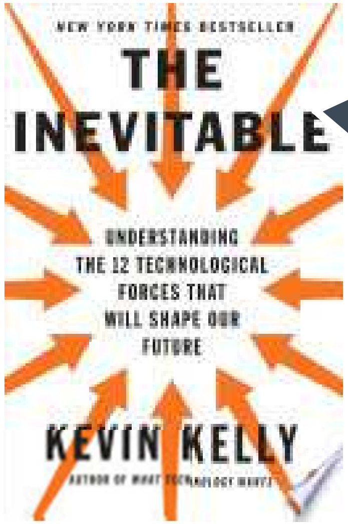



**Slow effective adoption of MBE due to a variety of social and technical factors, but the tools are often blamed!**

# What about cognifying MBE?




**Cognification:** "the process of making objects or systems smarter and smarter by connecting, integrating sensors and building software/artificial intelligence into them."

**Cognify MBE to improve drastically its benefits and hence reduce the barriers for its adoption.**

# Modelia,
# a joint R&D partnership between

"Bringing artificial intelligence to the modeling world and reversely."

**Jordi Cabot**
ICREA Research Profesor,
UOC / SOM lab

**Sébastien Gérard**
CEA Research Director,
List

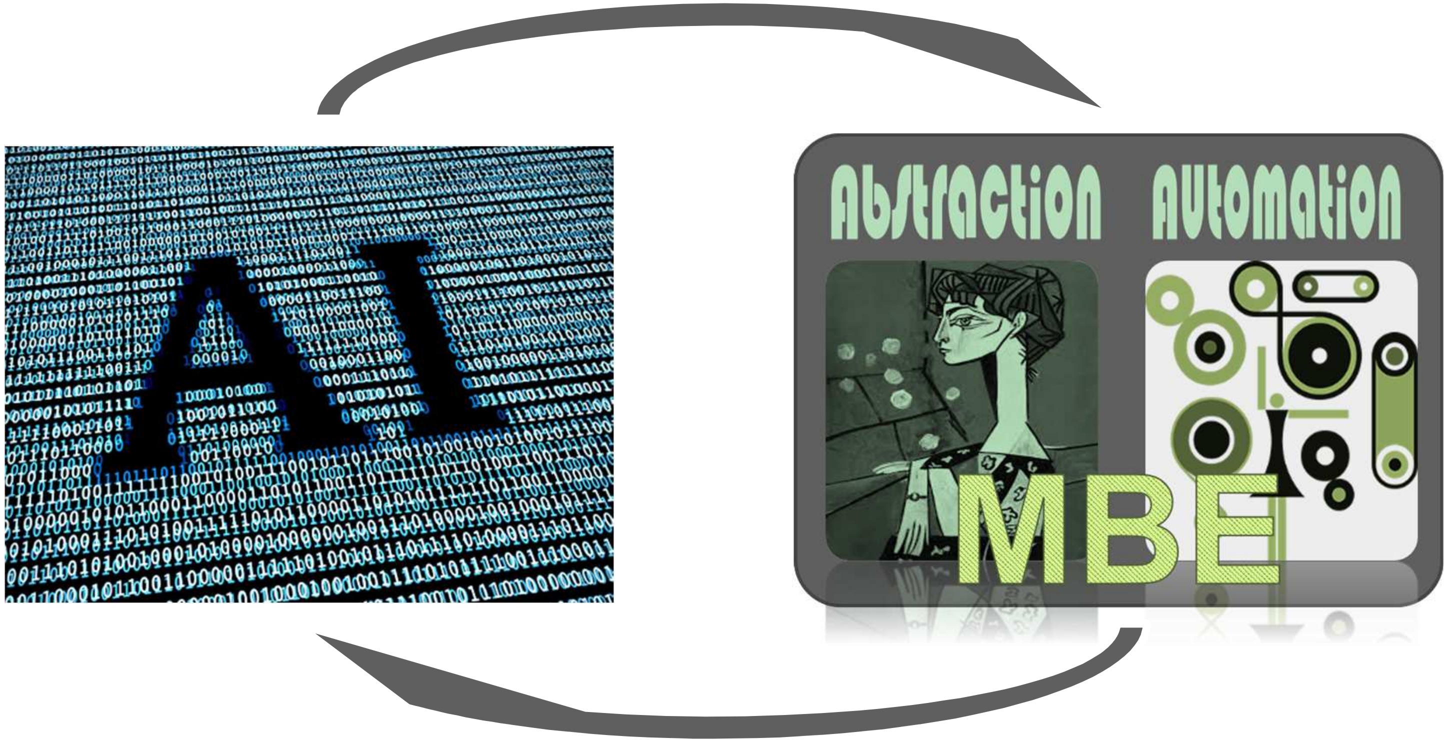
## Three ongoing projects:

**Loli Burgueño**, post-doc (2019-2020) on **"Uncertainty & AI-4-MT"**.

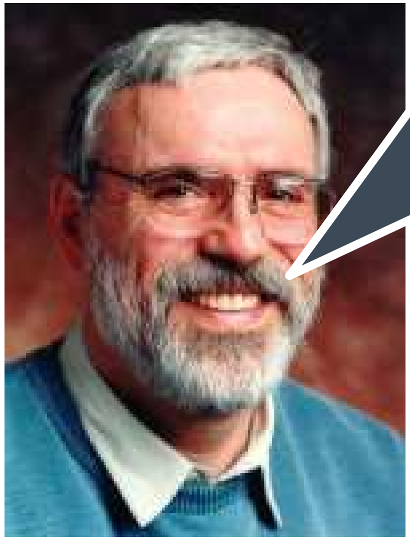
**Maxime Savary-Leblanc**, Phd student (2019-2022) on **"Modeling bots"**.

**Takwa Kochbati**, Phd student (2019-2022) on **"From Text to Conceptual Models"**.

# AI for MBE, and reversely?



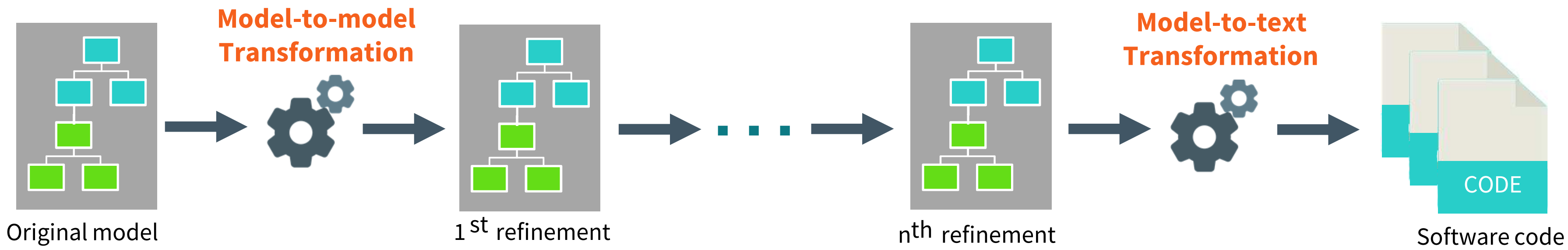
# The two main pilars of MBE

*B. Selic, "**Model-driven development: its essence and opportunities**," in Ninth IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC'06), 24-26 April 2006, Gyeongju, Korea.*

# What is model transformations (MT)

- **Model transformations define the mappings between models at a metamodel level**



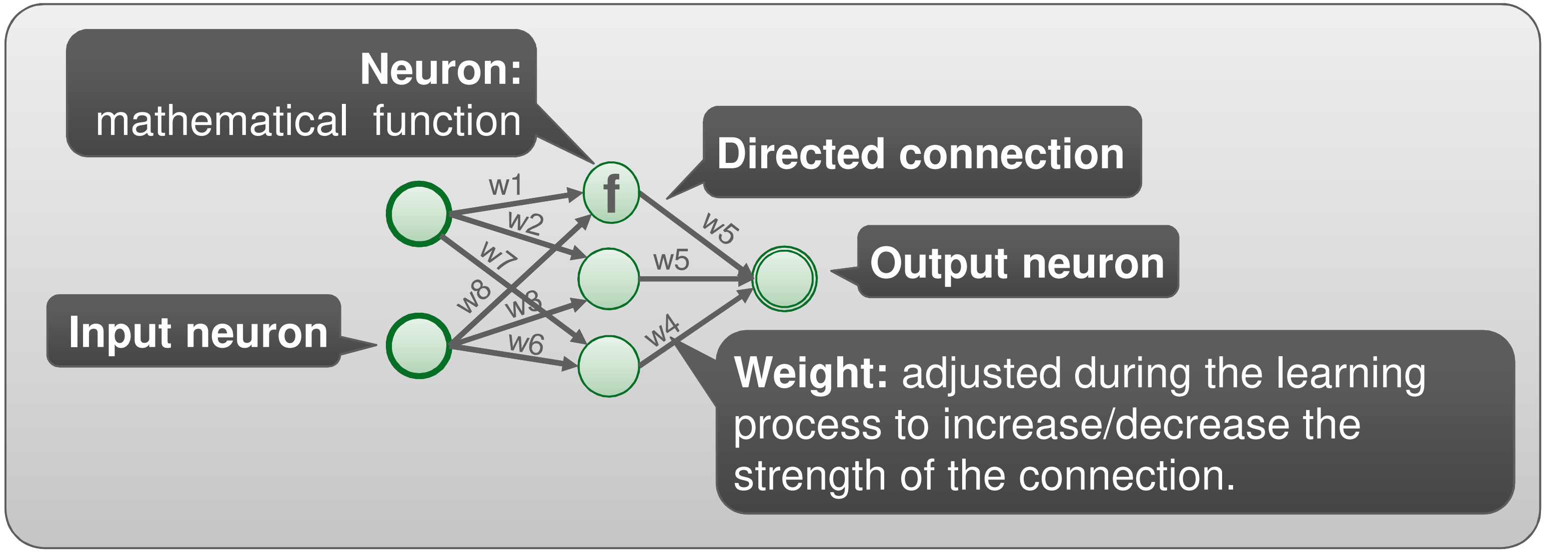

- **For people who are not familiar with the modeling paradigm, or with software development and coding in general:**
  - Requires learning a new language (the MT Language),
  - Time consuming,
  - Error prone and not easy to debug and maintain.

Model transformation ~ Language translation:
➔ From SMT / RBMT to ANN approaches.

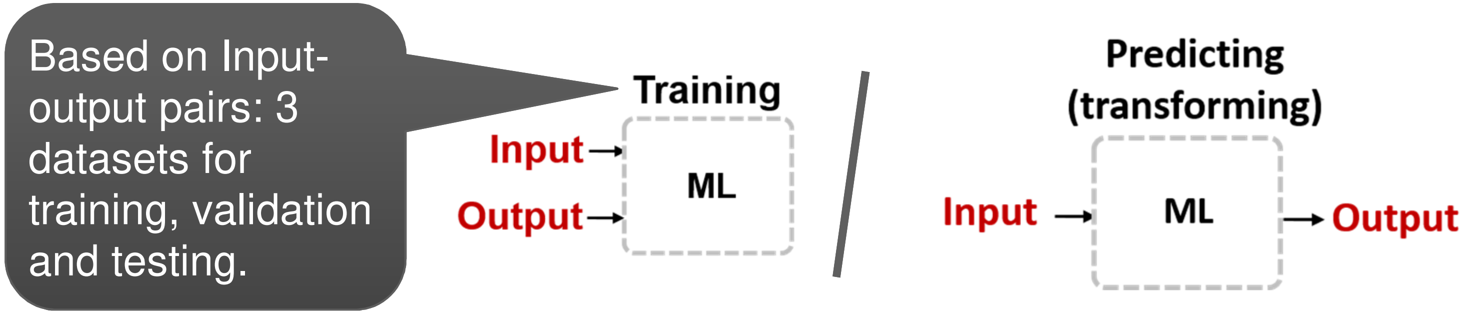
Artificial Neural Network to derive model transformations from sets of input/output models.

# Slide bar – Artificial Neural Network (ANN)

- ANN = directed graph structure of neurons



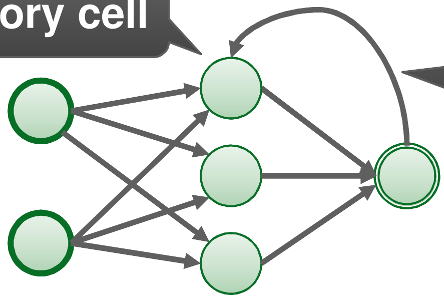

- Supervised learning process:



# Which ANN architecture to choose?

- **ANN ➔ Recurrent Neural Networks** (RNN)

**Memory cell**



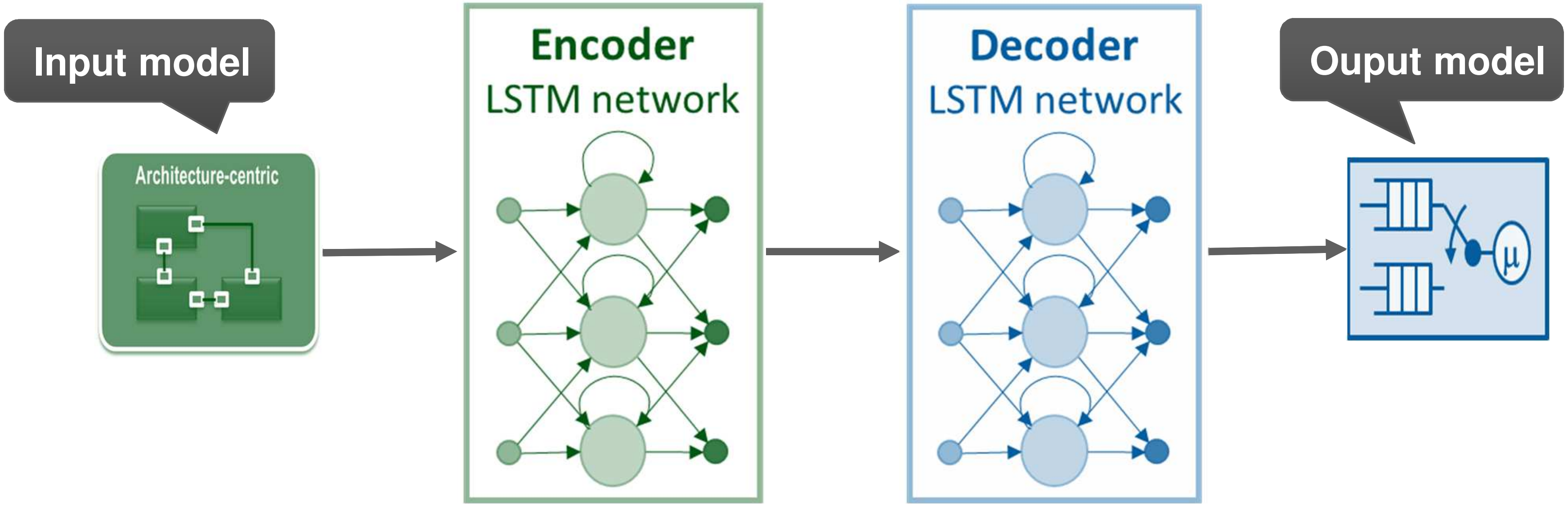
**Back propagation connection ➔ support for memory feature.**

*Note: memory cells are helpful in our case e.g., to remember the name of a variable previously declared.*

- **RNN ➔ Long Short-Term Memory** (LSTM)
  - *Longer "memory",*
  - *Can remember their context.*
- **LSTM + Encoder-Decoder architecture**
  - Proven to be the most effective architecture of ANN for dealing with translation problems.

# Outlines of the ANN architecture

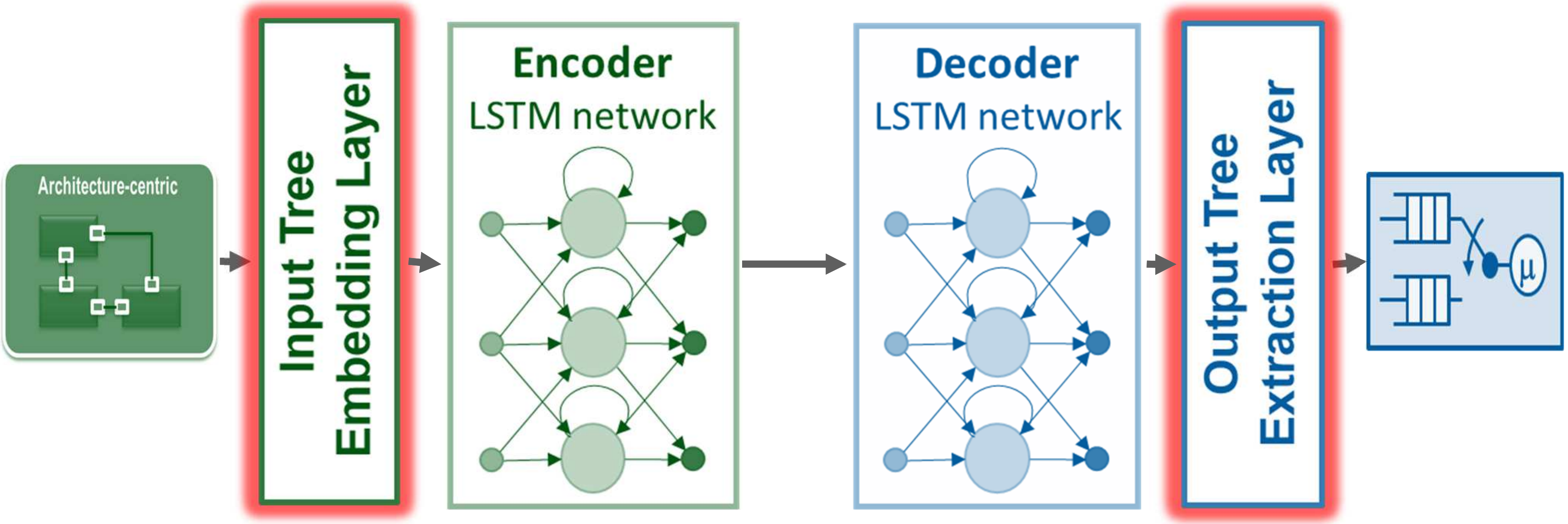
**Encoder-decoder architecture**
**+**
**Long short-term memory neural networks**



# But a model is not a sequence of words…

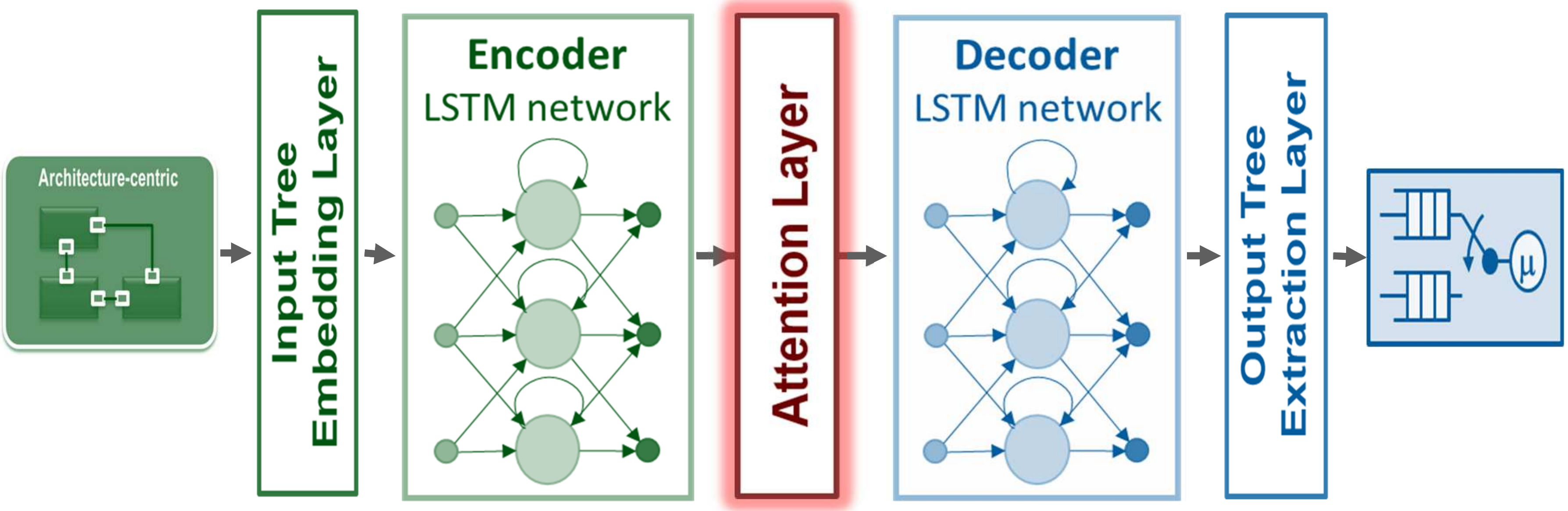
**From Sequence-to-Sequence to Tree-to-Tree!**

**→ 2 additional layers** to embed the input tree to a numeric vector and reversely.



# And with a bit of attention, it is even better…

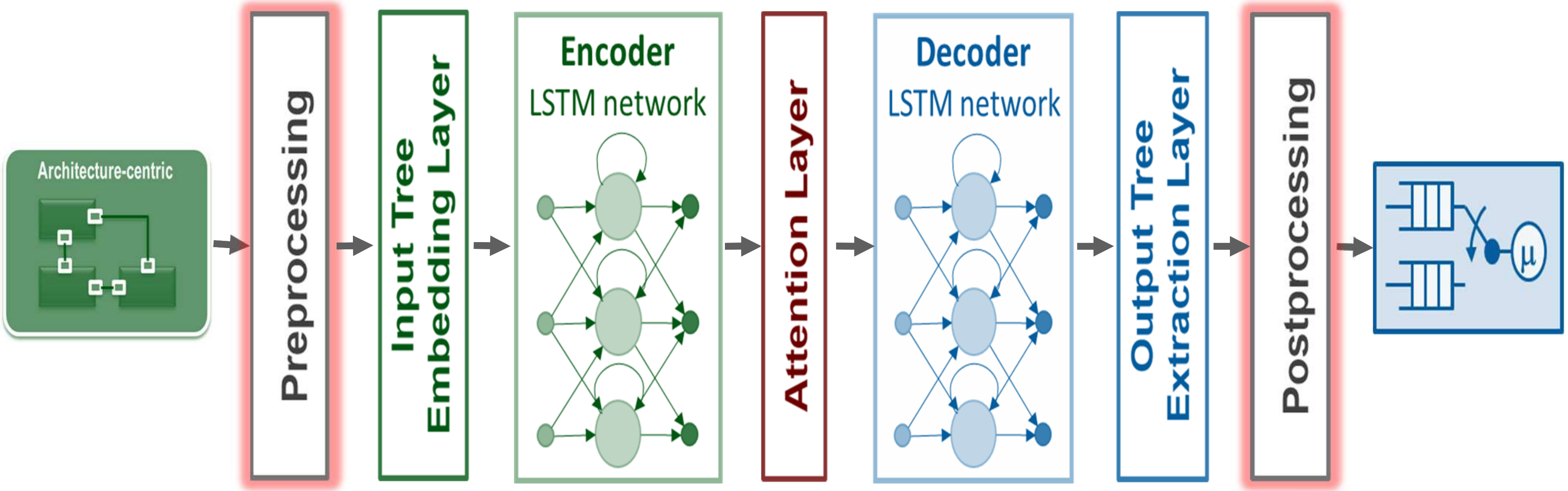
- **Attention mechanism**
  - To pay more attention (remember better) to specific parts,
  - It automatically detects to which parts are more important.



# And finally…

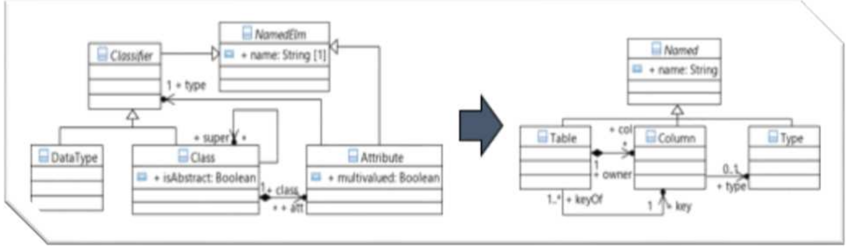
- **A bit of pre- and post-processing:**
  - To represent models as trees and conversely,
  - To eliminate symmetries in input models using a canonical form,
  - And to rename variables to avoid the "dictionary problem".
- **And some configuration of parameters:**
  - Layers#, neurons#, initial weights, learning rates, etc.

  ➔ Values are results of experiences (black-art of ML…)

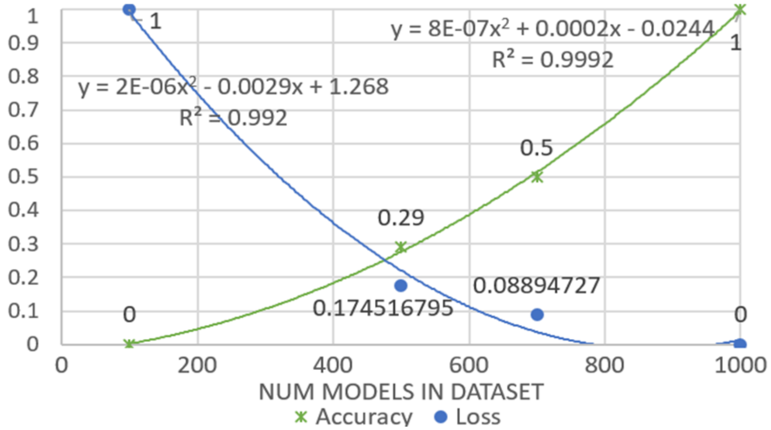


# About the results…

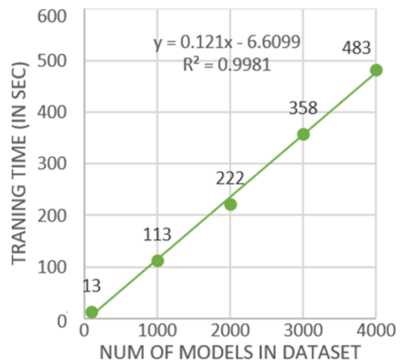
*(Use case: from Class to Relational models)*

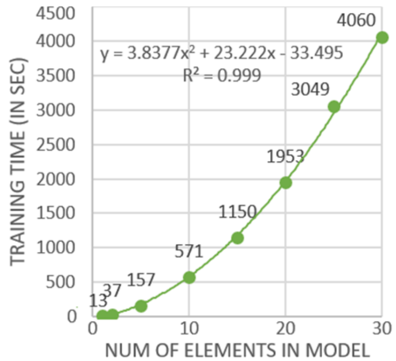



- **Correctness: measured through the *accuracy* and *loss of the ANN***

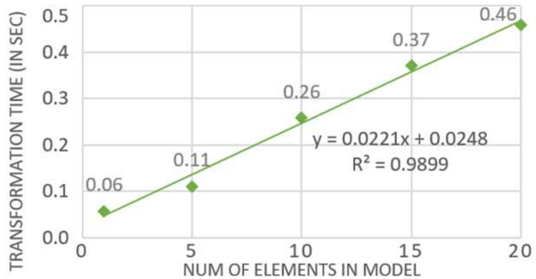



- **Performance**
  - How long is the training phase?




  - What is the performance of the ML-based MT?



# Bilan of this first experience applying ML to MT

**ML-based approach for MT is feasible but obviously there are a number of open challenges to be solved before it can actually be used in practice.**

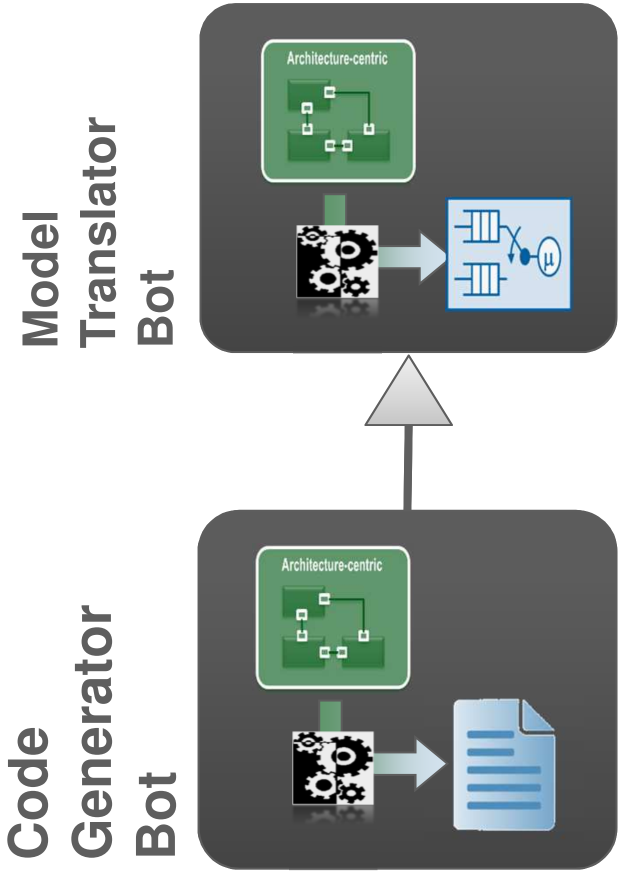
**Open Chalenges**

- **Size of the training dataset**
  - Model mutation procedures or GAN / Transfer learning / pre-trained networks for typical MT scenarios?
- **Diversity in the training set**
  - Coverage metrics for the I/O metamodels / graph kernel techniques?
- **Generalization problem**
  - Transfer learning / common sense ontologies?
- **Social acceptance & trust**
  - Explainable AI, certrified AI… ?
- **Computational limitations of ANNs**
  - Expect new results from AI domain…

# A new idea…

**Open challenges:**

- Size of the training dataset,
- Diversity in the training set.



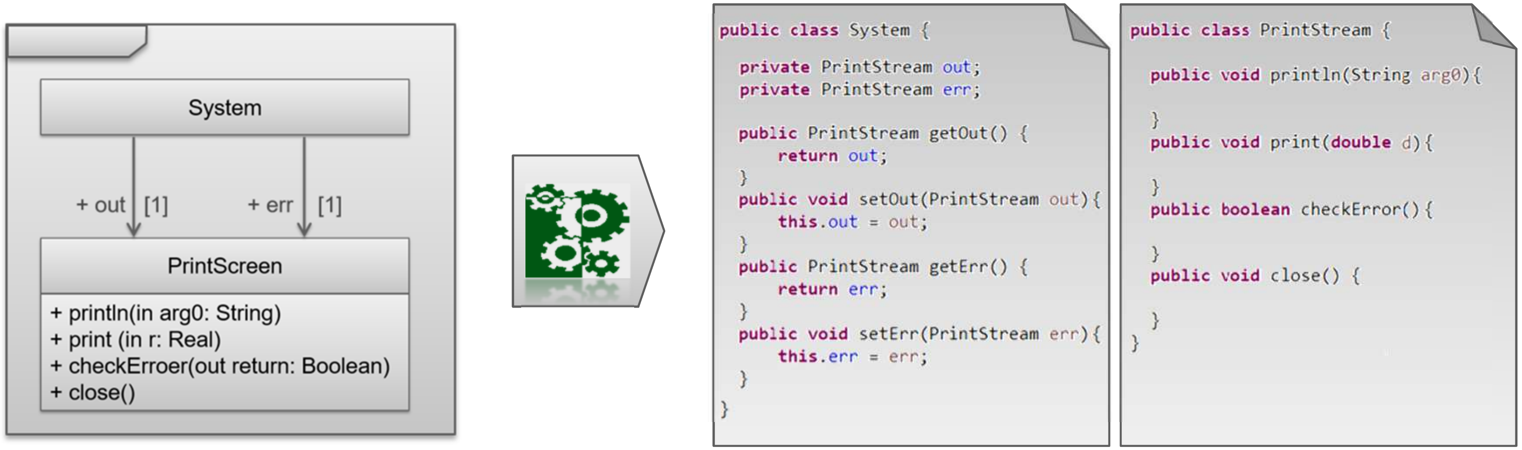

A lot of code available in open-source projects could be reversed into models.

# Outlines of the second experience

- **Scope**

UML Class-based Model

Java code




- **Main requirements for our code-generator bot**
  - Auto-inference of mappings between the structural parts of both source and target,
  - Respect coding standards (e.g., imposed by company or community) implicitly available in the dataset used to train the ANN.

# About the coding standard rules, the bot needs to learn…

## De facto standard rules:

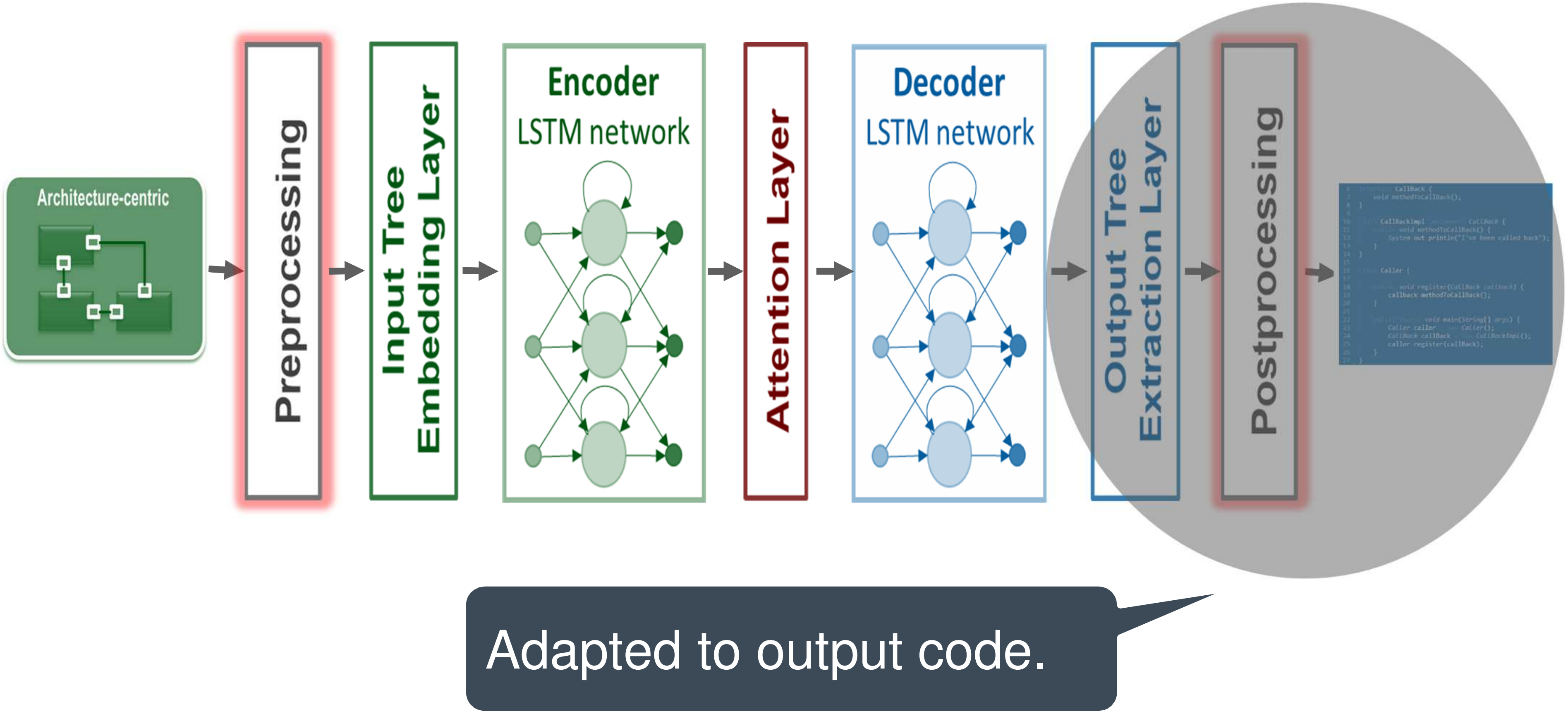
- UML classes into Java classes,
- UML attributes into Java attributes,
- UML associations into Java attributes.
- Etc.

+

## Ad hoc rules (company specific):

- Primitive data type conversions (e.g., a UML Real into a Java double or float),
- Visibility management (public UML attributes into Java public attributes or as a private attributes + getter and setter public methods).
- Etc.

# Adaptated ANN architecture for code generator

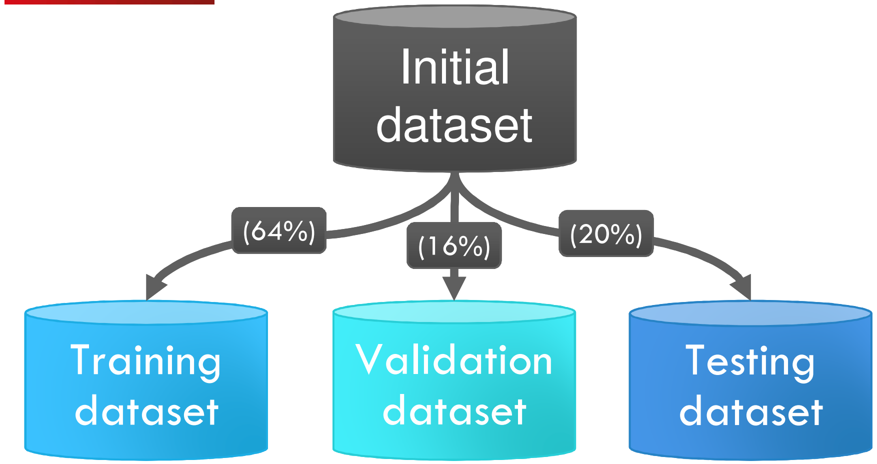


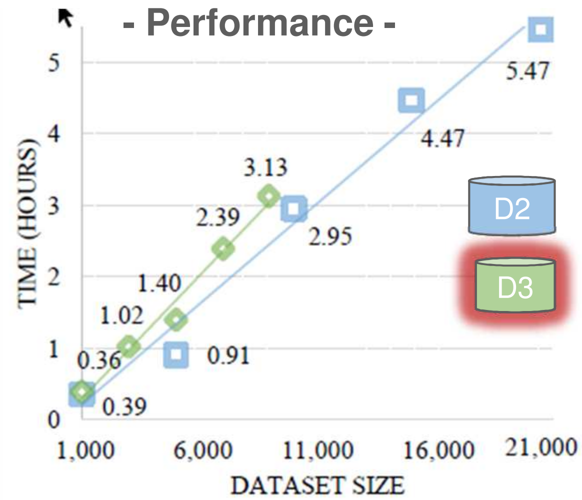
# About the training dataset of our PoC

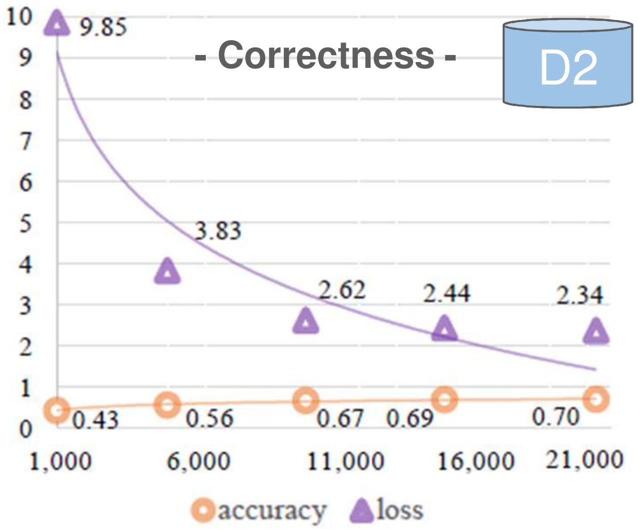
- Where the examples come from?
  - Reverse-engineered Java code of the Eclipse IDE into a Java model.
    → MoDisco (https://www.eclipse.org/MoDisco).
  - Abstract the Java model to a pure high-level UML model by removing all the "low-level" details such as method implementations.
- Pre-pocessing step:
  - Model and code converted into an AST,
  - Variable renamed (Dictionary issue).
  
  ➔ **Dataset (D1) contains 25,375 pairs of examples**.
- Cleaning step – discard pairs where the size of classes is too high:
  - Either raise RAM memory issue,
  - Or big size leds to the problem of long-term dependencies.
  
  ➔ **Curated dataset (D2) contains 20,840 pairs of examples.**
- Cleaning step-bis – coding rule variations!
  - e.g., Inheritance may involve diversity with owners of getters and setters.
  
  ➔ **New curated dataset (D3) contains 8,937 pairs of examples.**

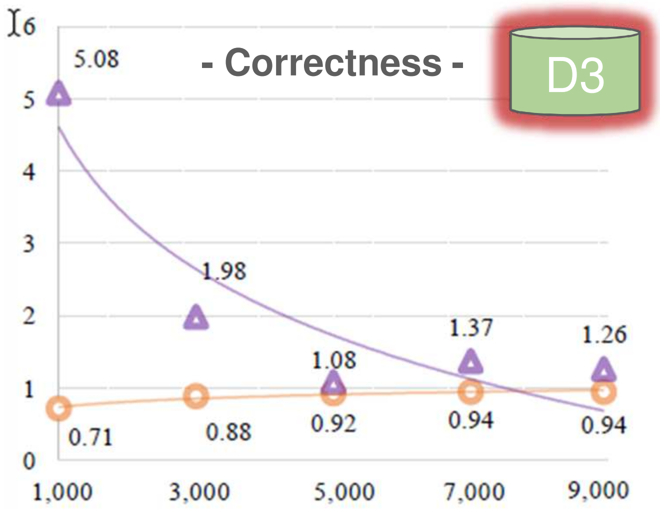
*Note: if for the same input, ANN receive different outputs (which is usually the case when writing code), they follow the "rule" which they have seen more often.*

# PoC, about correctness & performance



# Conclusions and future work related this theme

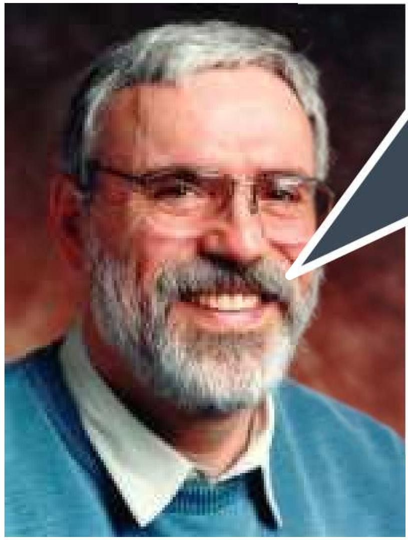
- **Demonstrate the feasibility and interest…**

  - Needs for high-quality examples… (as usual for ANN…),
  - Dictionary problem,
  - No support for operations on strings from ANN.

  - Numerous examples reversed from open-source projects,
  - Good execution performance of AI-based code generators,
  - Acceptable training performance.

- **Next steps for us:**
  - Experiment with transfer learning to enable reuse of trained networks in new projects where the styling guidelines may be slightly different,
  - Extend the network capabilities to cover the generation of basic behavioral code.
  - Continue to monitor the advance in AI and try out new features…

# The two main pilars of MBE

*B. Selic, "**Model-driven development: its essence and opportunities**," in Ninth IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC'06), 24-26 April 2006, Gyeongju, Korea.*

# What is the problem…

IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. 39, NO. 6, JUNE 2013

What Industry Needs from Architectural Languages: A Survey

Ivano Malavolta, Patricia Lago, Senior Member, IEEE, Henry Muccini, Member, IEEE

Model-driven Development of Complex Software: A Research Roadmap

Robert France, Bernhard Rumpe

Computer Languages, Systems & Structures

Model-driven engineering conceptual model

Alberto Rodrigues da Silva

Industrial Adoption of Model-Driven Engineering: Are the Tools Really the Problem?

Jon Whittle[1], John Hutchinson[1], Mark Rouncefield[1], Håkan Burden[2], and Rogardt Heldal[2]

[1] School of Computing and Communications, Lancaster University, Lancaster, UK
[2] Computer Science and Engineering, Chalmers University of Technology and University of Gothenburg, Gothenburg, Sweden

**Abstract.** An oft-cited reason for lack of adoption of model-driven engineering (MDE) is poor tool support. However, studies have shown that adoption problems are as much to do with social and organizational factors as with tooling issues. This paper discusses the impact of tools on MDE adoption and places tooling within a broader organizational context. The paper revisits previous data on MDE adoption (19 in-depth interviews with MDE practitioners) and re-analyzes the data through the specific lens of MDE tools. In addition, the paper presents new data (20 new interviews in two specific companies) and analyzes it through the same lens. The key contribution of the paper is a taxonomy of tool-related considerations, based on industry data, which can be used to reflect on the tooling landscape as well as inform future research on MDE tools.

**Keywords:** model-driven engineering, modeling tools, organizational change.

1 Introduction

When describing barriers to adoption of model-driven engineering (MDE), many authors point to inadequate MDE tools. Den Haan [1] highlights "insufficient tools" as one of the eight reasons why MDE may fail. Kuhn et al. [2] identify five points of friction in MDE that introduce complexity; all relate to MDE tools. Staron [3] found that "technology maturity [may] not provide enough support for cost efficient adoption of MDE." Tomassetti et al.'s survey reveals that 30% of respondents see MDE tools as a barrier to adoption [4].

Clearly, then, MDE tools play a major part in the adoption (or not) of MDE. On the other hand, as shown by Hutchinson et al. [5,6], barriers are as likely to be social or organizational rather than purely technical or tool-related. The question remains, then, to what extent poor tools hold back adoption of MDE and, in particular, what aspects – both organizational and technical – should be considered in the next generation of MDE tools.

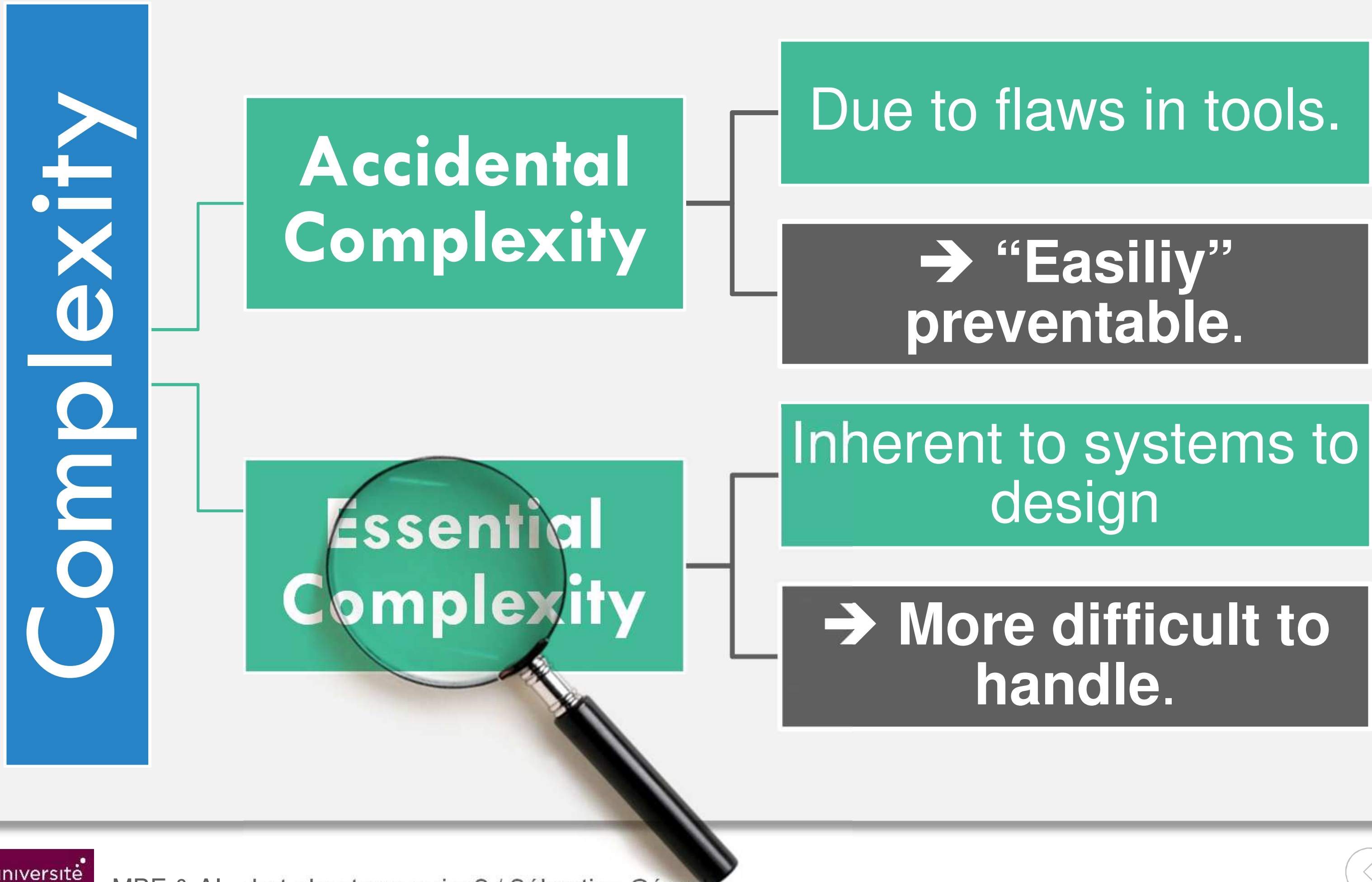
The key contribution of this paper is a taxonomy of factors which capture how MDE tools impact MDE adoption. The focus is on relating tools and their technical features to the broader social and organizational context in which they are

A. Moreira et al. (Eds.): MODELS 2013, LNCS 8107, pp. 1–17, 2013.
© Springer-Verlag Berlin Heidelberg 2013

MBSE is not an option,
but the tools are still a problem…

# Slidebar on complexity

Frederick. P. Brooks Jr., "**No Silver Bullet Essence and Accidents of Software Engineering**", Computer, vol. 20, no. 4, pp. 10–19, Apr. 1987.



# Our use case: system architecture

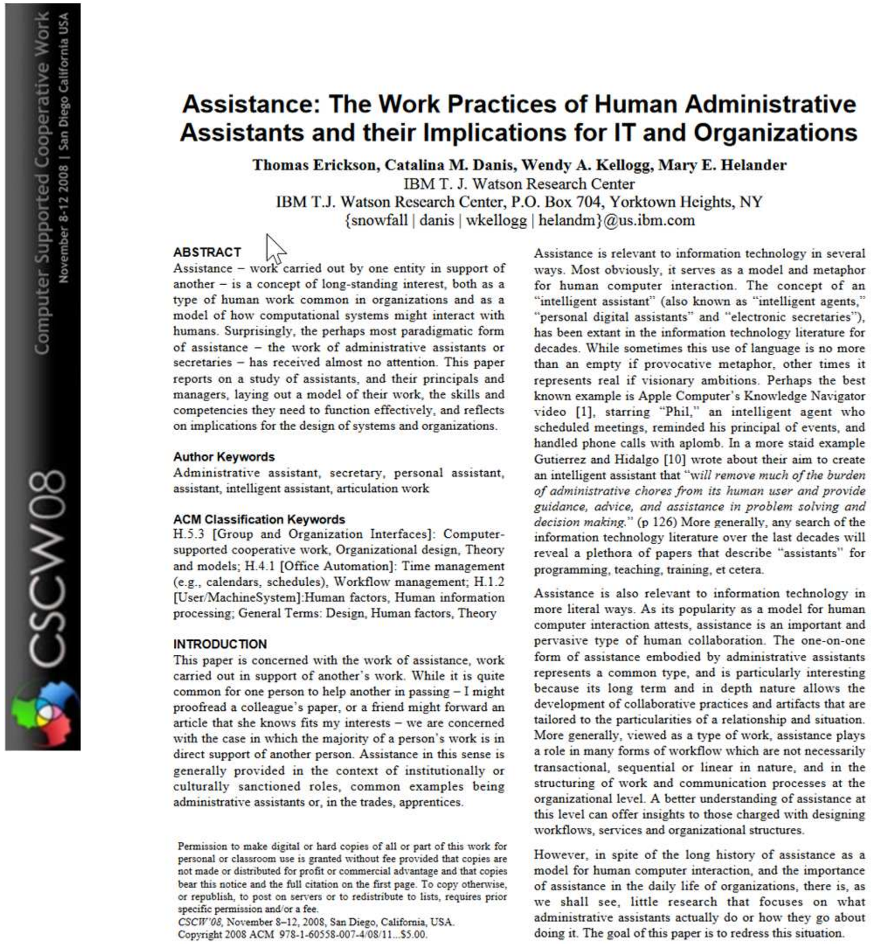
Conceptual modeling, a key step of system architecting.

Augment tools with AI-features to face essential complexity of conceptual modeling.


Empower experience if you are a debutant, or creativity if you are experienced.


**→ Empower analogy thinking by exemplification ←**

# What is an assistant?




## Assistance: The Work Practices of Human Administrative Assistants and their Implications for IT and Organizations


**Thomas Erickson, Catalina M. Danis, Wendy A. Kellogg, Mary E. Helander**
IBM T. J. Watson Research Center
IBM T.J. Watson Research Center, P.O. Box 704, Yorktown Heights, NY
{snowfall | danis | wkellogg | helandm}@us.ibm.com



**ABSTRACT**
Assistance – work carried out by one entity in support of another – is a concept of long-standing interest, both as a type of human work common in organizations and as a model of how computational systems might interact with humans. Surprisingly, the perhaps most paradigmatic form of assistance – the work of administrative assistants or secretaries – has received almost no attention. This paper reports on a study of assistants, and their principals and managers, laying out a model of their work, the skills and competencies they need to function effectively, and reflects on implications for the design of systems and organizations.



**Author Keywords**
Administrative assistant, secretary, personal assistant, assistant, intelligent assistant, articulation work

**ACM Classification Keywords**
H.5.3 [Group and Organization Interfaces]: Computer-supported cooperative work, Organizational design, Theory and models; H.4.1 [Office Automation]: Time management (e.g., calendars, schedules), Workflow management; H.1.2 [User/MachineSystem]:Human factors, Human information processing; General Terms: Design, Human factors, Theory


**INTRODUCTION**
This paper is concerned with the work of assistance, work carried out in support of another's work. While it is quite common for one person to help another in passing – I might proofread a colleague's paper, or a friend might forward an article that she knows fits my interests – we are concerned with the case in which the majority of a person's work is in direct support of another person. Assistance in this sense is generally provided in the context of institutionally or culturally sanctioned roles, common examples being administrative assistants or, in the trades, apprentices.



Assistance is relevant to information technology in several ways. Most obviously, it serves as a model and metaphor for human computer interaction. The concept of an "intelligent assistant" (also known as "intelligent agents," "personal digital assistants" and "electronic secretaries"), has been extant in the information technology literature for decades. While sometimes this use of language is no more than an empty if provocative metaphor, other times it represents real if visionary ambitions. Perhaps the best known example is Apple Computer's Knowledge Navigator video [1], starring "Phil," an intelligent agent who scheduled meetings, reminded his principal of events, and handled phone calls with aplomb. In a more staid example Gutierrez and Hidalgo [10] wrote about their aim to create an intelligent assistant that "*will remove much of the burden of administrative chores from its human user and provide guidance, advice, and assistance in problem solving and decision making.*" (p 126) More generally, any search of the information technology literature over the last decades will reveal a plethora of papers that describe "assistants" for programming, teaching, training, et cetera.

Assistance is also relevant to information technology in more literal ways. As its popularity as a model for human computer interaction attests, assistance is an important and pervasive type of human collaboration. The one-on-one form of assistance embodied by administrative assistants represents a common type, and is particularly interesting because its long term and in depth nature allows the development of collaborative practices and artifacts that are tailored to the particularities of a relationship and situation. More generally, viewed as a type of work, assistance plays a role in many forms of workflow which are not necessarily transactional, sequential or linear in nature, and in the structuring of work and communication processes at the organizational level. A better understanding of assistance at this level can offer insights to those charged with designing workflows, services and organizational structures.

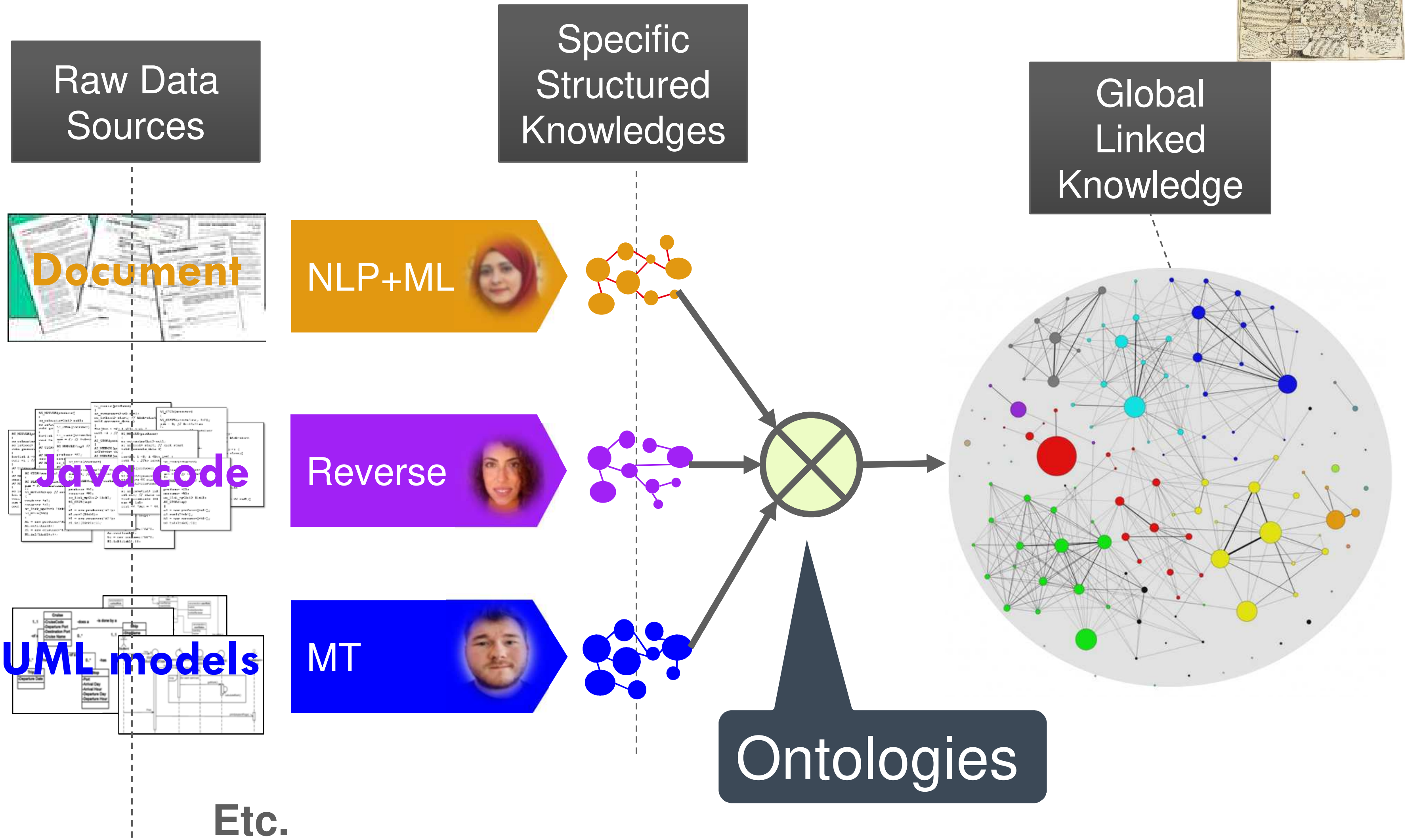
However, in spite of the long history of assistance as a model for human computer interaction, and the importance of assistance in the daily life of organizations, there is, as we shall see, little research that focuses on what administrative assistants actually do or how they go about doing it. The goal of this paper is to redress this situation.

"Assistants must **handle a stream of events** by reacting to it while **maintaining situational awareness** and **consolidating background knowledge**."

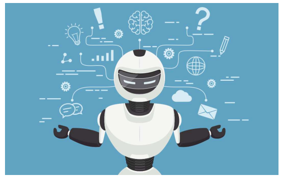
RQ1: How to get the background knowledge?

RQ2: when and how best to interact with the tool user?

# RQ1: how to get the background knowledge?

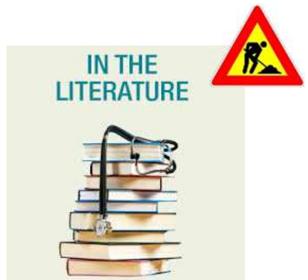


# RQ2: when and how best to interact with the tool user?



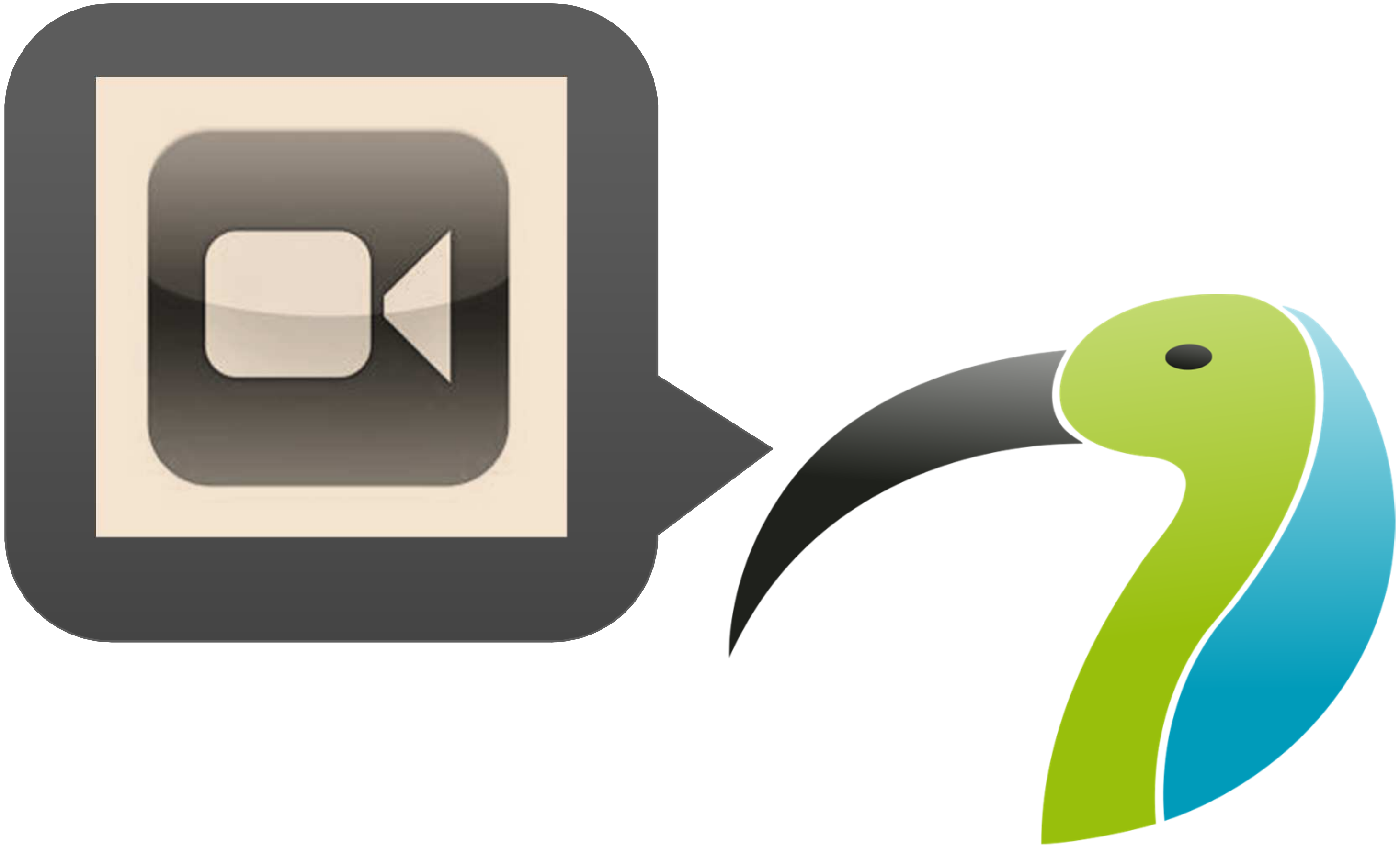
- **RQ2.1 – When the bot should interact with modellers?**
  - Rely on the bot capacities to understand the activities of the modeller
    ➔ **Bot awareness**
  - Our solution ➔ the Awareness theory of Endsley [2]:
    → **Identity**: who is assisted (its profile, experience,…)?
    → **Authorship**: who is the origin of the model currently managed?
    → **Actions**: what the modeler is doing?
    → **Artifacts**: what are the related resources?
    → **Intention**: what is the goal of the modeler?


- **RQ2.2 – How the bot should present the information to modellers?**
  - Ongoing SLR focused on:
    → Visualization information,
    → Distributed cognition (i.e., where the info should be presented),
    → And cognitive dimensions of the assistance.

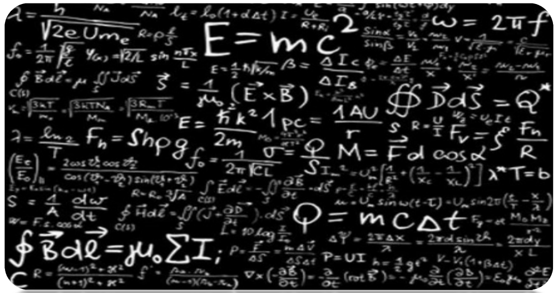



*[1] P. Dourish and V. Bellotti, "Awareness and coordination in shared workspaces", CSCW '92, Toronto, Canada, 1992.*
*[2] M. R. Endsley, "Design and evaluation for situation awareness enhancement", Human Factors Society 32nd Annual Meeting, Santa Monica, CA, 1988.*

# Cinema break on an Architecture Modeling Bot…



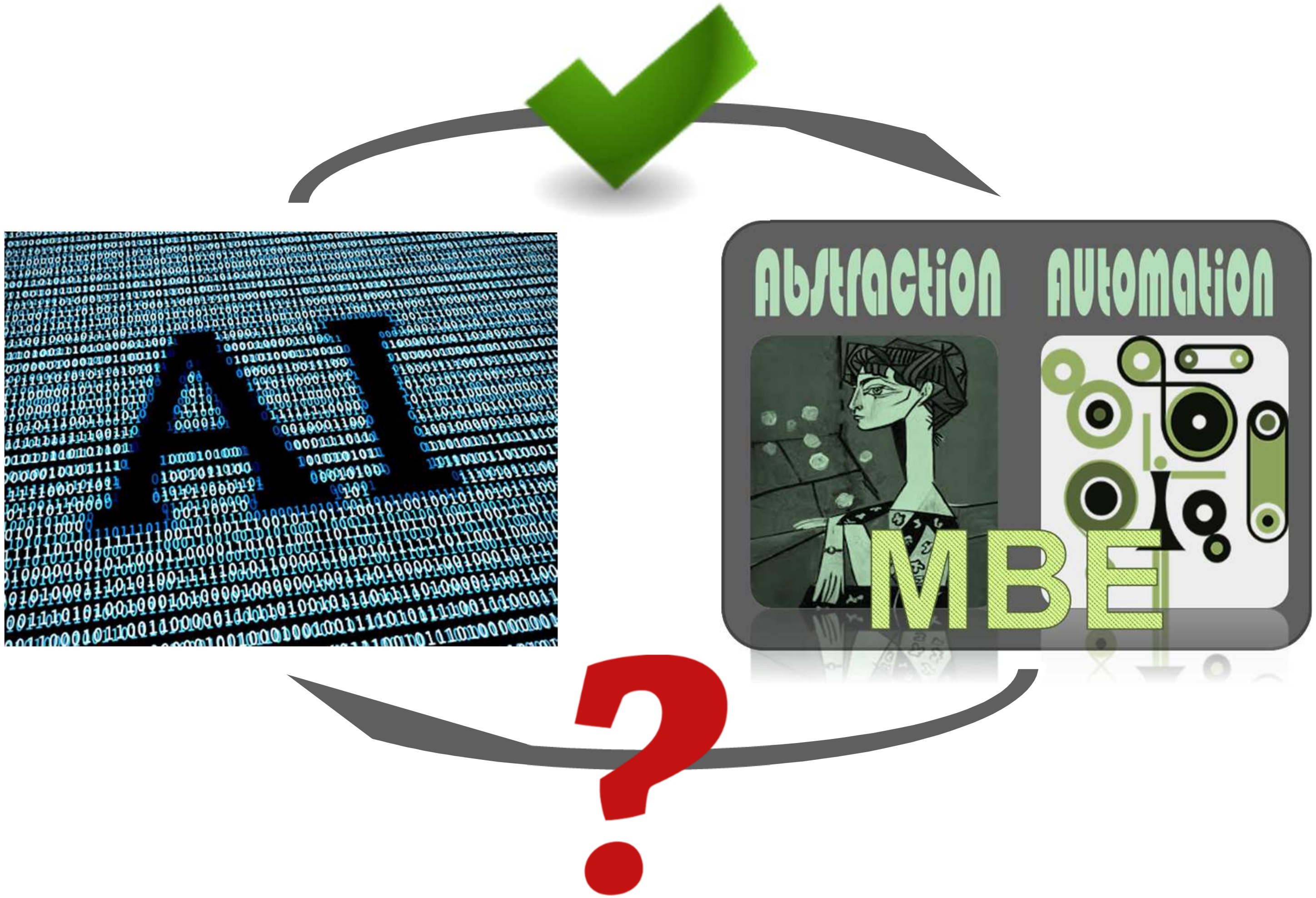
# Remaining challenges we want to focus on



- Formalization of the interactions between modelling User & Bot in the context of MBE.
- A model-based methodology to design and embed modeling bots into modeling IDE.
- How to create knowledge from data…?


- An framework that support Awareness of modeling workspace.
- A modeling bot focusses on architecture concerns in Papyrus.
- An information integration bot that can create & consolidate knowledge on its own from multiple & heterogenous data sources.

# AI for MBE, and reversely?



# AI are “black-box”!

- Modeling and analyzing AI-based systems typically requires accepting some uncertainties about their precise behavior.
- However, trust in AI is a key point for their acceptance and safe deployment.

➔ Needs to “model and operate” this uncertainty!


Premier international conferences on Uncertainty in Artificial Intelligence

# But what is "Uncertainty modeling" ?

GUM definition*: "the quality or state that involves imperfect and/or unknown information".

Bureau International des Poids et Mesures

- https://www.bipm.org -

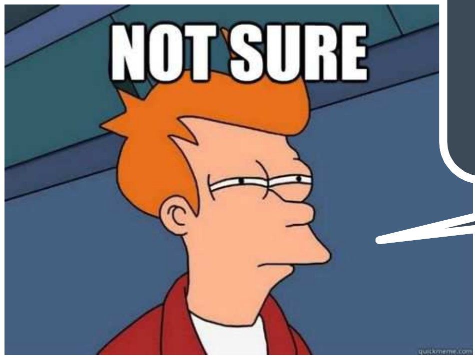
* JCGM 100:2008, Evaluation of measurement data - **Guide to the expression of uncertainty in measurement (GUM)**, Joint Com. for Guides in Metrology, 2008.

- **Three kinds of uncertainty:**
  - Measurement uncertainty
    → Inability to know with complete precision the value of a quantity (e.g., x = 3.0 ± 0.01).
  - Occurrence uncertainty
    → Likelihood that a physical entity represented in a model actually occurs in reality.
  - Belief uncertainty
    → Situation where one is uncertain about a statement made about a knowledge or a system, or even the environment of a system.


*L. **Burgueño, R. Clarisó, J. Cabot, S. Gérard, and A. Vallecillo**, "Belief Uncertainty in Software Models," in 2019 IEEE/ACM 11th International Workshop on Modelling in Software Engineering (MiSE), 2019.*

# Focus on Belief uncertainty

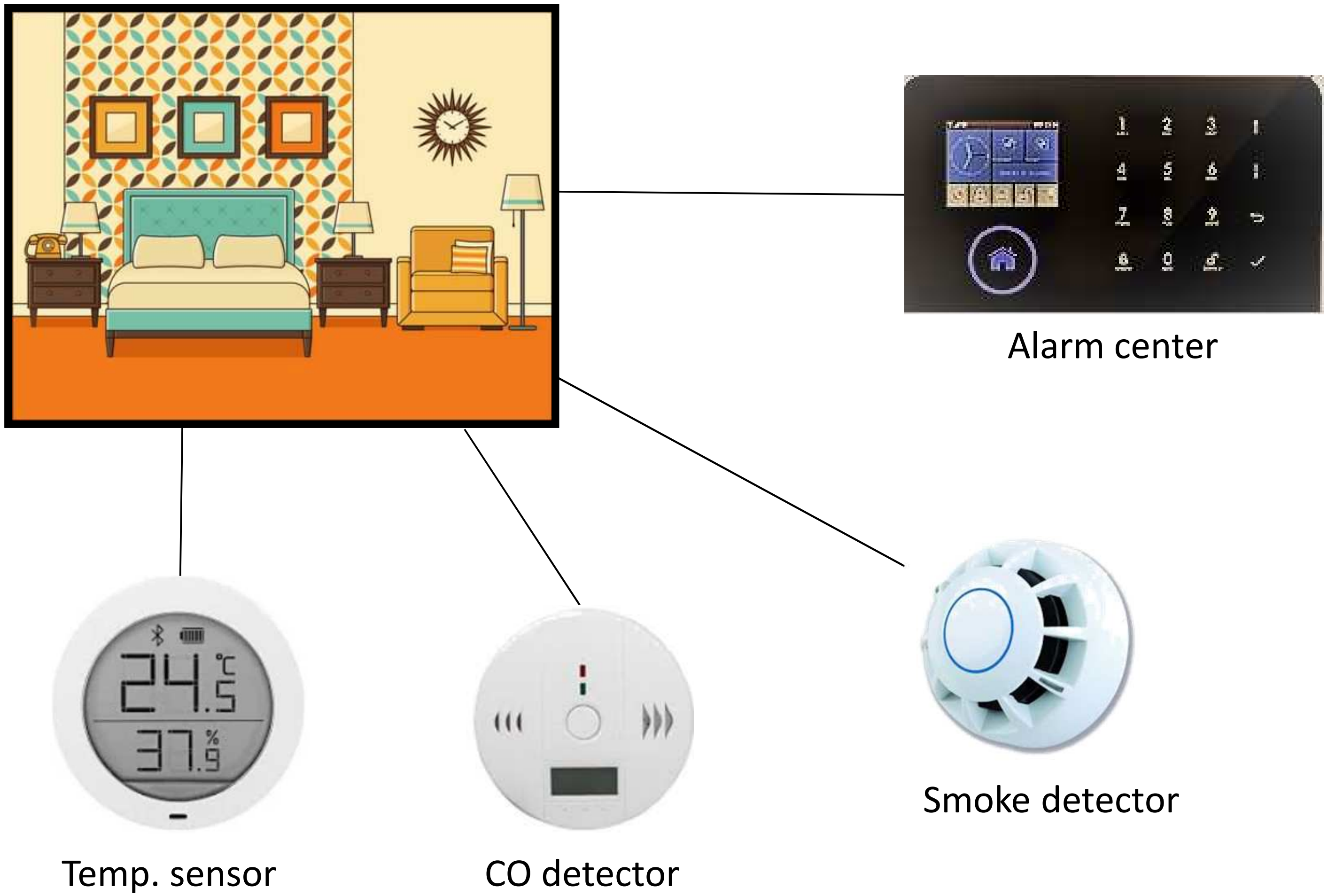
**Belief statement are usually subjective!**




Not sure my learning source is fully accurate!
Not sure how far my algo is precised!
etc.

- **This work is hence aiming at answering the following questions:**
  - **RQ1** - How to specify the belief uncertainty of data?
  - **RQ2** - How to integrate this feature in existing modeling language?
  - **RQ3** - How to use the information about the uncertainty?

# Our motivating example…



# Some belief statements on our example

- The CO and smoke detectors that we bought have a reliability of 90% (i.e., 10% of their readings are not meaningful). *→ Precision of the values*
- We can only be 98% sure that the precision of the temperature sensor is 0.5°C, as indicated in its datasheet. *→ Uncertainty of the values*
- We are 95% confident that the presence of high temperature, high CO level and smoke really means that there is a fire in the room. *→ About the behavioral rules*
- Paul only assigns a credibility of 50% to the operations that indicate if the room is hot or cold. In contrast, Marie thinks they are 99% accurate. *→ Individual belief agents*
- Room #3 is close to the kitchen and frequently emits alarms. Everybody thinks that 90% of them are false positives. *→ Individual instances*
- Paul doubts that the type of attribute "number" of class "Room" is Integer. He thinks it may contain characters different from digits. *→ About the model itself: types*
- Marie is unsure if an "AlarmCenter" has to be attached to only one single Room. She thinks they can also be attached to several. *→ About the model itself: relations*

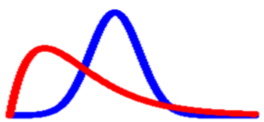
**How to represent these uncertainties?**
**How to integrate them into the system models?**

# RQ1 - How to specify the belief uncertainty?

Different stakeholders may have different estimation.

- Goal => be able to assign degree**(s)** of belief to model statements.
- Solution => Bayesian probabilities while being a classical model for quantifying subjective beliefs .

**Resp1 - Credence to measure belief uncertainty**

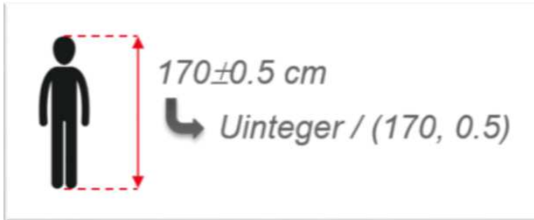


Note: **credence** is a statistical term that refers to a measure of belief strength: used here to express how much an agent believes that a proposition is true (e.g, a modeler can be 99% sure the type used to represent a given property is correct.).

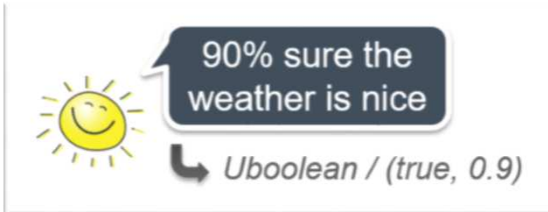
# RQ2 - How to integrate this feature in existing modeling language?

**Resp2.a - Extend OCL/SysML/UML datatypes to integrate information about the uncertainty of their values.**

- Standard datatype extensions e.g., UInteger, and UReal.
  - Represented by a pair **(x, u)**
    - expected value
    - uncertainty

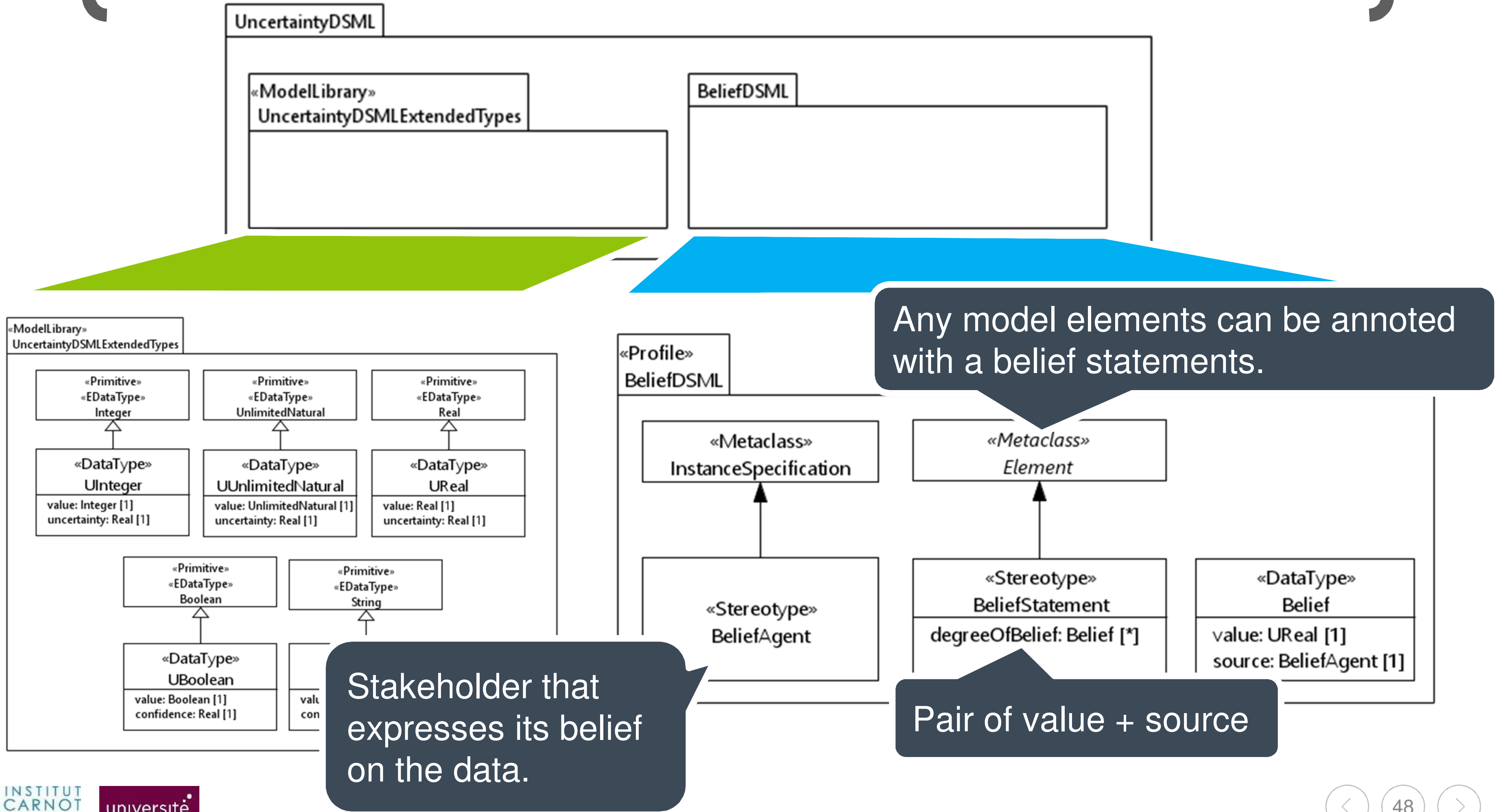



- UBoolean values uncertainty does not refer to measurement uncertainty, but to confidence.
  - Represented by a pair **(b, c)**
    - boolean value → true or false
    - confidence that b is certain → Real number in the range [0..1]



# RQ2 - How to integrate this feature in existing modeling language? (Con't)

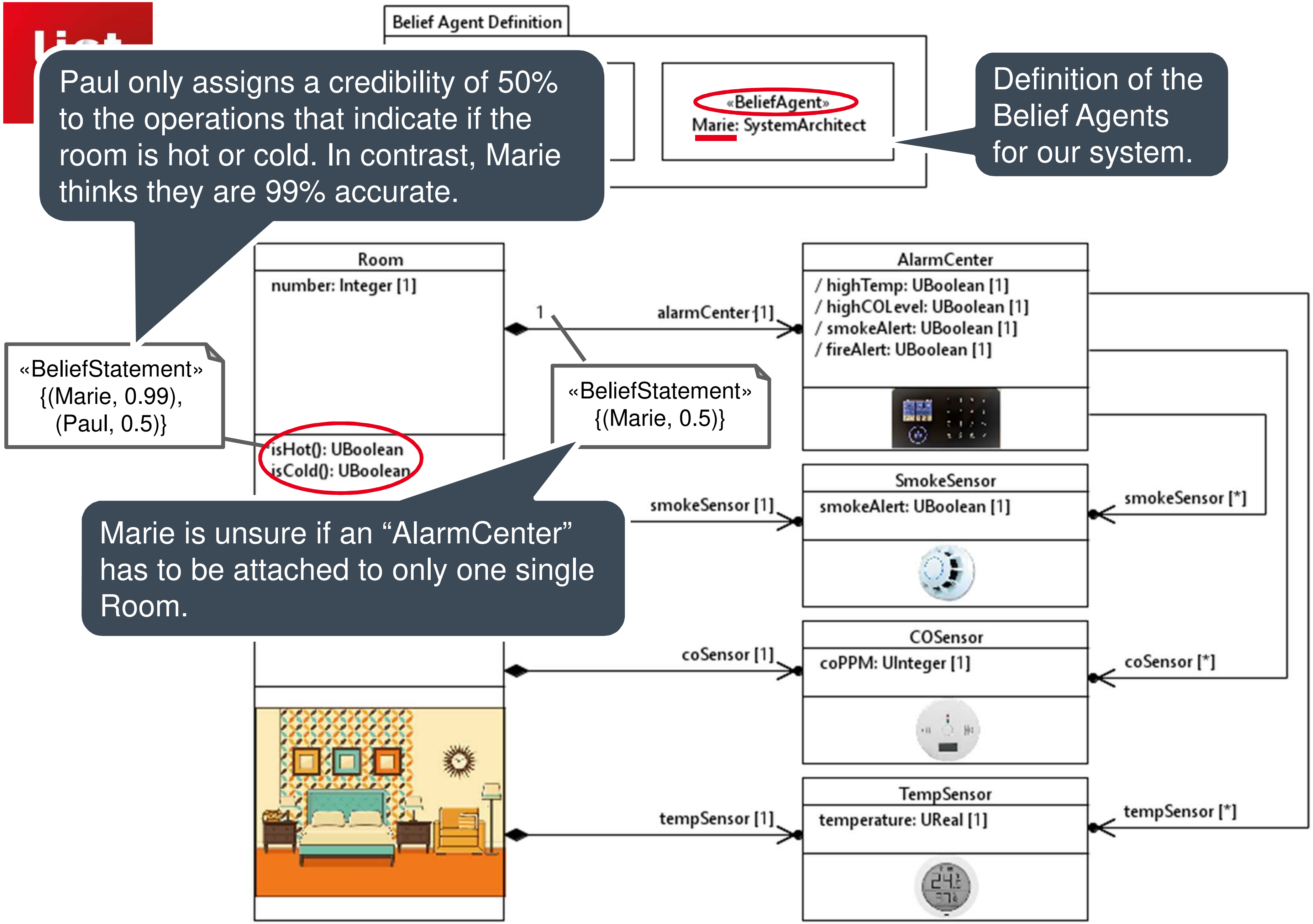
**Resp2.b – Define a UML profile for uncertainty modeling (can be applied on both UML and SysML models).**



Paul only assigns a credibility of 50% to the operations that indicate if the room is hot or cold. In contrast, Marie thinks they are 99% accurate.

Definition of the Belief Agents for our system.

Marie is unsure if an "AlarmCenter" has to be attached to only one single Room.



# Uncertainty Modeling: executive summary and future work

- **Explicit representation and management of belief uncertainty in software models…**

  **in terms of degrees of belief assigned to model elements by separate belief agents…**

  **and about the credibility of:**
  - The values of the represented elements,
  - The measurement uncertainty of these values,
  - And the way in which we have modeled the system (e.g., types of the attributes, types of relationships and their cardinalities).
- **Future work**
  - Contribute to standard:
    - → The OMG is working towards a metamodel for the Precise Specification of Uncertainty Modeling (PSUM)
  - Associating evidences to belief statements,
  - Representing degrees of beliefs in other types of models (use cases, sequence diagrams, pre- and postconditions, …),
  - And investigate further application domains (e.g., model inference, model assistant).

# And what about next?

# Conclusions and next steps…

> "Continue monitoring AI progresses and our experiences with AI."

- **Modelia projects:**

**Loli Burgueño**, post-doc (2019-2020) on **"Uncertainty & AI-4-MT"**.

**Maxime Savary-Leblanc**, Phd student (2019-2022) on **"Modeling bots"**.

**Takwa Kochbati**, Phd student (2019-2022) on **"From Text to Conceptual Models"**.

NEW

**Edouard Batot**, post-doc (2020-2022) on **"Traceable co-evolution management by ML"**.

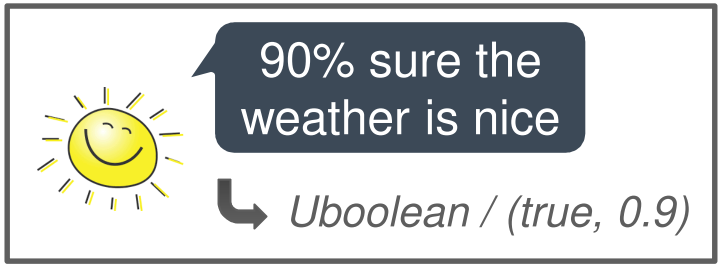
# An finally…

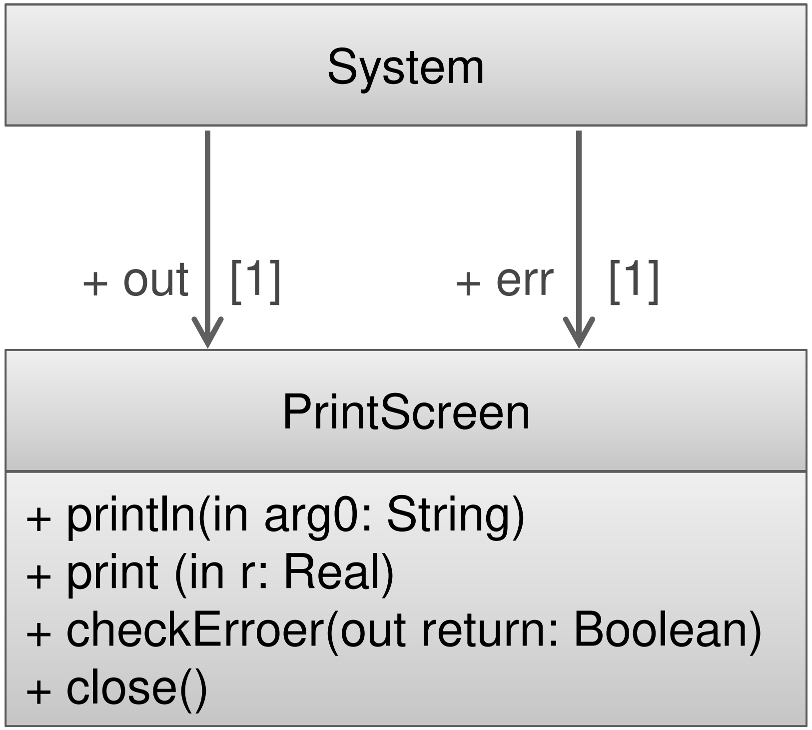
**Loli Burgueño**, post-doc (2019-2020) on **“Uncertainty & AI-4-MT”**.

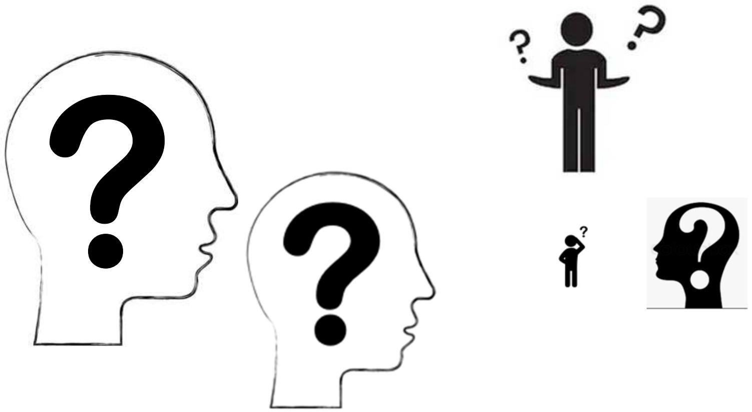
**Maxime Savary-Leblanc**, Phd student (2019-2022) on **“Modeling bots”**.

Commissariat à l’énergie atomique et aux énergies alternatives
Institut List | CEA SACLAY NANO-INNOV | BAT. 861 – PC142
91191 Gif-sur-Yvette Cedex - FRANCE
www-list.cea.fr

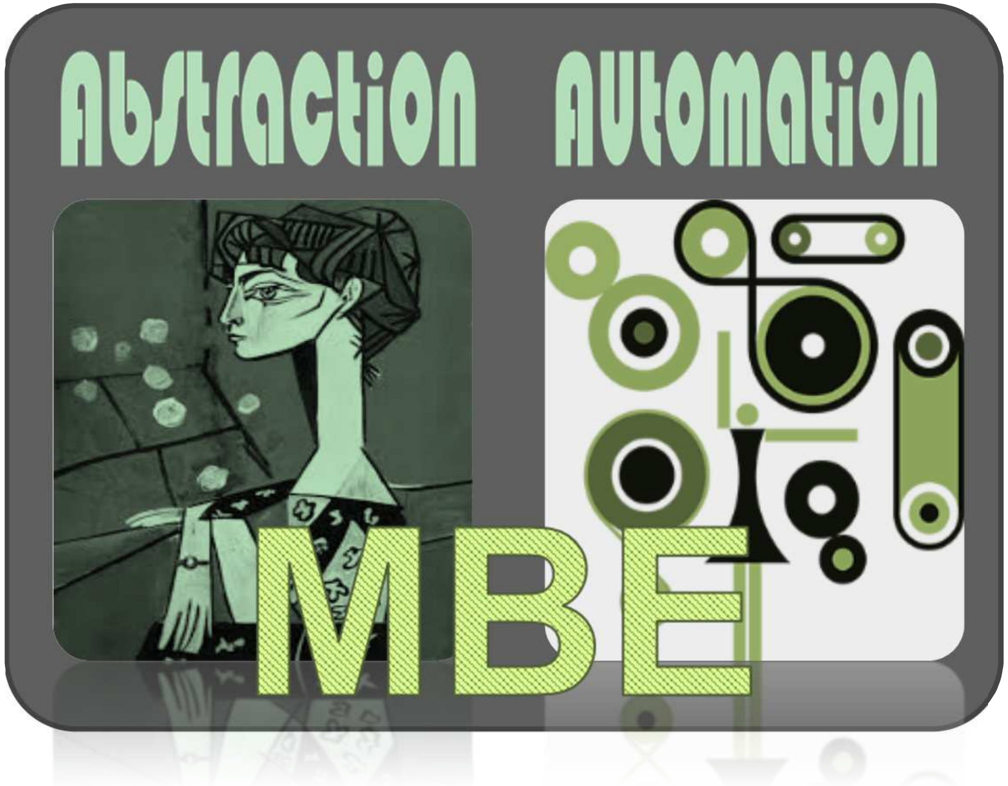
Établissement public à caractère industriel et commercial | RCS Paris B 775 685 019

usability
interoperability
workflow integration
accessibility
performance
complexity
high skill
steep learning curve


Abstraction
Automation
MBE