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.

CEA
Institution & Government Research

*8 WINNER

- 16,000 people
- 750 new patents/year
- 600 industry partners
- 4800 scientific publications/year
- 4 BE€ budget
- 750 new patents/year
- 600 industry partners
- 4800 scientific publications/year
- 4 BE€ budget

MBE & AI, what about synergies? / Sébastien Gérard
The missions of CEA

- Defense and national security
- Energy independence
- Economic competitiveness of French industry
Expertise areas of the CEA List institute

- Software & System Engineering
- Data Intelligence
- Simulation
- Cyber Security
- Robotics
- Embedded Systems
- Instrumentation & Metrology
- Man/Machine Interface

Institute of Digital Technologies
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....

Universal Programmability - How /

Walter F. Tsch, Matthias Landhäusser, Soen J
http://www.cs.uni-magdeburg.de/people/tschw/index.htm

ABSTRACT
Software modeling techniques and tools are considered the cornerstones on which the future of software development will be built. The elimination of the unproductive and time-consuming tasks that are currently often performed by software engineers and developers is required to bring software intensive systems to real-time delivery. One key factor to productivity and efficiency is the programmability of the underlying tools and systems. This paper presents an approach to programmability by providing the necessary structures and mechanisms to support the construction of tools and systems that can be customized and adapted to the requirements of the different stakeholders.

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

Marta Hirsch, Katholieke Universiteit Leuven, Leuven, Belgium
marta.hirsch@eso.kuleuven.be

ABSTRACT
Intelligence systems are becoming increasingly sophisticated. A good example is the growing confusion between the terms “human” and “machine.” Management is becoming a more complex task, and as a result, there is a need for more intelligent tools to assist in the development of complex systems. This paper describes an approach to the automatic generation of active ontologies from web forms. The approach is based on the use of machine learning techniques and natural language processing.

MBE & AI, what about synergies? / Sébastien Gérard

ABSTRACT
The combination of methods and tools to improve the quality of software engineering is a crucial issue. The purpose of this paper is to explore the synergies between Model-Based Engineering (MBE) and Artificial Intelligence (AI) in the context of software engineering.

Automated Validation of Requirement Reviews: A Machine Learning Approach

Andreas Kowalski, Information Engineering, University of Technology, Graz, Austria
andreas.kowalski@technikum.ac.at

ABSTRACT
The validation of requirements is a crucial step in the software development process. The process is often time-consuming and requires a large amount of effort. In this paper, we present a machine learning approach to automate the validation of requirements. The approach is based on the use of automated test case generation and machine learning techniques.

Automated reasoning framework for traceability management of system of systems

B. D. W. Schaus, V. J. den Boer, D. van der Weide

ABSTRACT
In the context of complex systems, traceability is a crucial aspect of software engineering. This paper presents an automated reasoning framework for traceability management of system of systems. The framework is based on the use of automated reasoning techniques and the integration of different traceability management tools.
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 designers.

- **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

1. Unreliable (e.g., Error-prone)
2. Inefficient (e.g., Evolution)
3. Non-scalable (e.g., > Concerns)

1. Reliable (e.g., Error-prone)
2. Efficient (e.g., Evolution)
3. Scalable (e.g., > Concerns)

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 Jordi Cabot, ICREA Research Professor, UOC / SOM lab and Sébastien Gérard, CEA Research Director, List.

“Bringing artificial intelligence to the modeling world and reversely.”

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”.

MBE & AI, what about synergies? / Sébastien Gérard
AI for MBE, and reversely?
The two main pilars of MBE

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.
ANN = directed graph structure of neurons

- **Neuron**: mathematical function
- **Directed connection**
- **Weight**: adjusted during the learning process to increase/decrease the strength of the connection.

**Supervised learning process**:

Based on Input-output pairs: 3 datasets for training, validation and testing.
Which ANN architecture to choose?

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

Note: memory cells are helpful in our case e.g., to remember the name of a variable previously declared.
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.
• 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…)
  
And finally…
About the results...

- **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 Challenges

- 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, certified 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

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.
Adapted ANN architecture for code generator

Adapted to output code.
About the training dataset of our PoC

- Where the examples come from?
  - Reverse-engineered Java code of the Eclipse IDE into a Java model.
  - Abstract the Java model to a pure high-level UML model by removing all the “low-level” details such as method implementations.

- Pre-processing 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 leads 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

- **Training dataset** (64%)
- **Validation dataset** (16%)
- **Testing dataset** (20%)

**Performance** vs **Dataset Size**

**Correctness** vs **Dataset Size**
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 pillars of MBE

What is the problem...

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

usability
interoperability
workflow integration
performance
accessibility
complexity
steep learning curve
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?

"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?

- Raw Data Sources
- Document
- Java code
- UML models

- NLP+ML
- Reverse
- MT

Specific Structured Knowledges

Global Linked Knowledge

Ontologies

Etc.
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.

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

uai2020

Conference on Uncertainty in Artificial Intelligence
Toronto, Canada
August 3-6, 2020
But what is "Uncertainty modeling"?

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


- https://www.bipm.org -

- Three kinds of uncertainty:
  - Measurement uncertainty
    → Inability to know with complete precision the value of a quantity (e.g., \( x = 3.0 \pm 0.01 \)).
  - 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.

Focus on Belief uncertainty

Belief statement are usually subjective!

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). \(\rightarrow \text{Precision of the values}\)

• We can only be 98% sure that the precision of the temperature sensor is 0.5°C, as indicated in its datasheet. \(\rightarrow \text{Uncertainty of the values}\)

• 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. \(\rightarrow \text{About the behavioral rules}\)

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. \(\rightarrow \text{Individual belief agents}\)

• Room #3 is close to the kitchen and frequently emits alarms. Everybody thinks that 90% of them are false positives. \(\rightarrow \text{Individual instances}\)

• Paul doubts that the type of attribute “number” of class “Room” is Integer. He thinks it may contain characters different from digits. \(\rightarrow \text{About the model itself: types}\)

Marie is unsure if an “AlarmCenter” has to be attached to only one single Room. She thinks they can also be attached to several. \(\rightarrow \text{About the model itself: relations}\)

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).

Any model elements can be annotated with a belief statements.

Stakeholder that expresses its belief on the data.

Pair of value + source
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.

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”.
  - **Takwa Kochbati**, Phd student (2019-2022) on “From Text to Conceptual Models”.
  - **Edouard Batot**, post-doc (2020-2022) on “Traceable co-evolution management by ML”.

MBE & AI, what about synergies? / Sébastien Gérard
An finally…

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


Big Thanks
170±0.5 cm
\[\text{Uinteger} / (170, 0.5)\]

90% sure the weather is nice
\[\text{Uboolean} / (true, 0.9)\]

System

- + out \[1\]
- + err \[1\]

PrintScreen

- println(in arg0: String)
- print (in r: Real)
- checkError(out return: Boolean)
- close()

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