



Automatic System Modeling with AI

Ludovic Apvrille, Bastien Sultan
{ludovic.apvrille, bastien.sultan}@telecom-paris.fr

ModelsWard 2024, Rome



Outline

Introduction

Technical context

TTool-AI: rationale

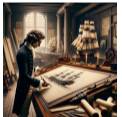
TTool-AI: architecture

A student-vs-*TTool-AI* case-study

Conclusion and Future Works



Introduction



Introduction

Generative AI and System Engineering

- Generative AI is now extensively used in various contexts (including generation of images for ModelsWard presentations)
- LLMs (large language-models) are particularly useful for software engineers (e.g., for test-case generation, code refactoring, reverse engineering)
- **MBSE is another area where the use of LLMs is undergoing extensive exploration**
 - A growing number of papers on the topic over the past year
 - A dedicated session and a workshop at ModelsWard 2024!

Rationale

MBSE involves

- Translating a specification, written in natural language, into architecture
- Often using pre-existing architectural patterns
- Proceeding through incremental modeling/simulation/proof
- Exploring different architectures to decide on the optimal one

Rationale

MBSE involves

- Translating a specification, written in natural language, into architecture
 - And LLMs (very) well understand and analyze natural language
- Often using pre-existing architectural patterns
- Proceeding through incremental modeling/simulation/proof

- Exploring different architectures to decide on the optimal one

Rationale

MBSE involves

- Translating a specification, written in natural language, into architecture
 - And LLMs (very) well understand and analyze natural language
- Often using pre-existing architectural patterns
- Proceeding through incremental modeling/simulation/proof
 - And it is possible to embed the syntax of SysML mutation languages within LLMs. LLMs can also easily parse textual SysML models and determine optimal insertion points for patterns
- Exploring different architectures to decide on the optimal one

Rationale

MBSE involves

- Translating a specification, written in natural language, into architecture
 - And LLMs (very) well understand and analyze natural language
- Often using pre-existing architectural patterns
- Proceeding through incremental modeling/simulation/proof
 - And it is possible to embed the syntax of SysML mutation languages within LLMs. LLMs can also easily parse textual SysML models and determine optimal insertion points for patterns
- Exploring different architectures to decide on the optimal one
 - And LLMs exhibit variability among several good responses

Introduction

Outline

- A few words about TTool and OpenAI's GPT
- *TTool-AI*: rationale
- *TTool-AI*: architecture
- A student-vs-*TTool-AI* case-study
- Future works



Outline

Introduction

Technical context

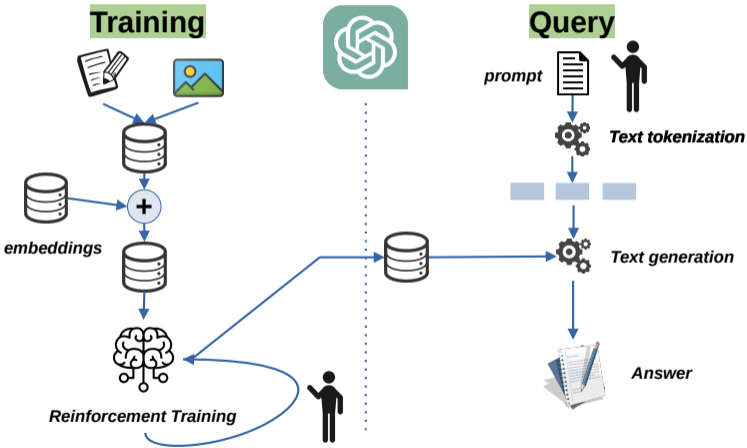
TTool-AI: rationale

TTool-AI: architecture

A student-vs-*TTool-AI* case-study

Conclusion and Future Works

ChatGPT in a Nutshell



<https://writings.stephenwolfram.com/2023/02/what-is-chatgpt-doing-and-why-does-it-work/>

TTool in a Nutshell



An Open-Source Toolkit

- Formal system modeling in SysML
 - Views: fonctional, architecture, deployment, system, etc.
 - Diagrams: requirement, use case, block, state machines, ...
- Support for all system development cycle (from requirements to code)
- Safety / security / performance evaluation at the push of a button
- Code generation
- Industrial and academic contribution and use
- `ttool.telecom-paris.fr`



Outline

Introduction

Technical context

TTool-AI: rationale

TTool-AI: architecture

A student-vs-TTool-AI case-study

Conclusion and Future Works

TTool-AI: rationale

Main limitations

1. LLMs often generate responses that may not precisely align with the engineers needs initially: engineers need to incrementally refine their request: this process is time-consuming, diminishing the benefits of utilizing LLMs
2. LLMs' training base don't include precise knowledge on some specific tools or formalisms
3. LLMs' answers need to be incorporated into models

TTool-AI: rationale

Main limitations

1. LLMs often generate responses that may not precisely align with the engineers needs initially: engineers need to incrementally refine their request: this process is time-consuming, diminishing the benefits of utilizing LLMs
2. LLMs' training base don't include precise knowledge on some specific tools or formalisms
3. LLMs' answers need to be incorporated into models

TTool-AI solutions

1. Automated feedback loop
2. Automated knowledge embedding
3. Automated parsing and model generation



Outline

Introduction

Technical context

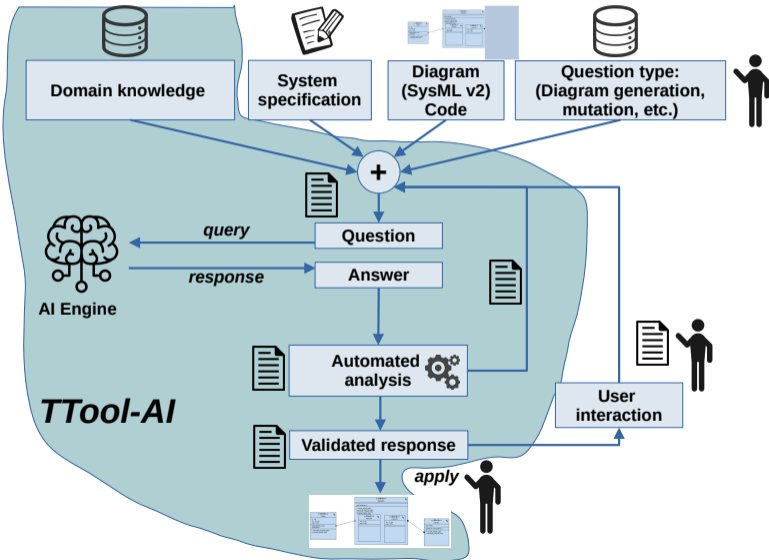
TTool-AI: rationale

***TTool-AI*: architecture**

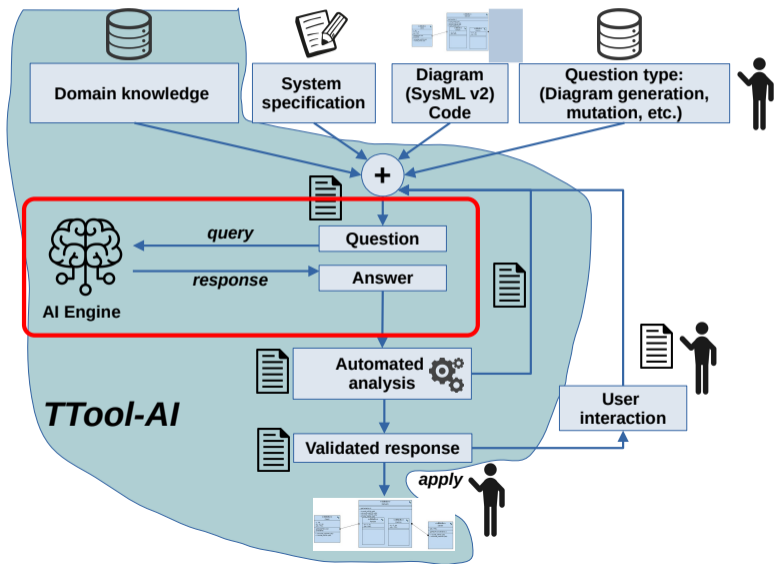
A student-vs-*TTool-AI* case-study

Conclusion and Future Works

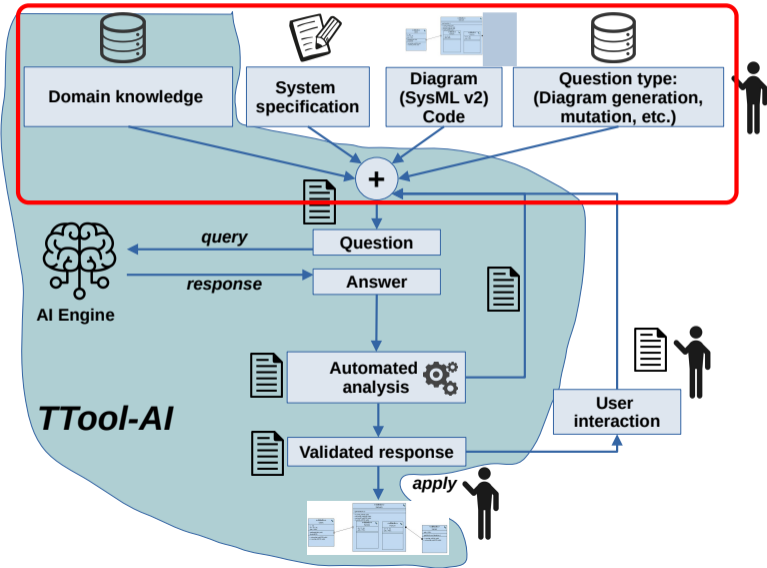
TTool-AI: architecture



TTool-AI: Interacting with ChatGPT



TTool-AI: Knowledge (or Context) Embedding



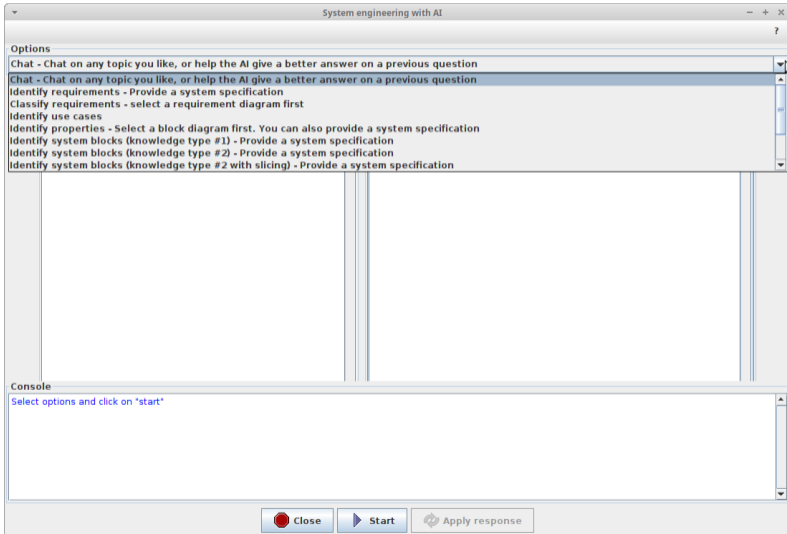
TTool-AI: Knowledge (or Context) Embedding

Dynamic knowledge embedding

- Constraints on response format
 - Constraints on SysML syntax
 - Structure of the expected JSON file to ease response parsing
- SysML mutation language syntax and semantics
- Depends on the question type selected by the user in the UI



TTool-AI: Knowledge (or Context) Embedding



TTool-AI: Knowledge (or Context) Embedding

Constraints on response format

When you are asked to identify SysML blocks, return them as a JSON specification formatted as follows:

```
{blocks: [{ \"name\": \"Name of block\", \"attributes\": [\"name\":  
\"name of attribute\", \"type\": \"int or bool\" ...} ...]}
```

#Respect: each attribute must be of type `\"int\"` or `\"bool\"` only
#Respect: Any identifier (block, attribute, etc.) must no contain any space. Use `\"_\"` instead.

...

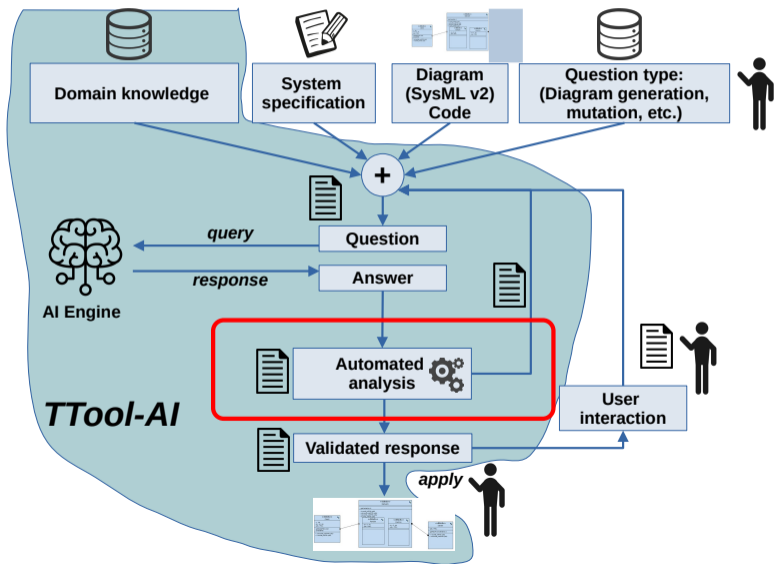
TTool-AI: Knowledge (or Context) Embedding

SysML mutation language syntax

AMULET is a SysML mutation language. With AMULET, adding a block `b` in a block diagram is written `\add block b\`. Removing a block `b` from a block diagram is written `\remove block b\`.

...

TTool-AI: automated feedback loop



TTool-AI: automated feedback loop

Invalid answer

Answer that does not respect the expected output:

- Invalid format (JSON, SysML v2, ...) or mutation syntax/semantics
- Constraints not respected
 - "List two blocks in json format", but the answer gives only one block
 - Unexisting objects (attributes, blocks, ...) used in a mutation command

Detection and iteration

1. Detect format, constraint or mutation errors in answer
2. If at least one error is detected and $cpt < maxIteration$ not reached
 - 2.1 The previous message and its answer are added to the context
 - 2.2 A description d_i is generated for each error e_i
 - 2.3 A new message is formed based on the list of d_i
 - 2.4 The message is fed into ChatGPT
 - 2.5 $cpt++$; Goto 1

Automated feedback loop: example



```
$ curl https://api.openai.com/v1/chat/completions
...
"messages": [
{"role": "system",
"content": "You are a helpful assistant for
system engineering."},

{"role": "user",
"content": "Give me the example of a SysML block, in json
format, with attributes. No attribute of type String. The block should
have one more attribute than ports, and one less port than operations.
The number of ports must be equal to 1."}]
...
```

Automated feedback loop: example



```
{ "name": "Block1", "attributes": [  
  {"name": "attribute1", "type": "Integer"},  
  {"name": "attribute2", "type": "Boolean"},  
  {"name": "attribute3", "type": "Float"},  
  {"name": "attribute4", "type": "Double"}],  
  "ports": [  
    {"name": "port1", "type": "Port1"}],  
  "operations": [  
    {"name": "operation1", "parameters": [], "returnType": "ReturnType1"},  
    {"name": "operation2", "parameters": [], "returnType": "ReturnType2"}  
  ]  
}
```

Automated feedback loop: example

We compute an updated question and we send it:

```
$ curl https://api.openai.com/v1/chat/completions \
...
"messages": [
{"role": "system",
"content": "You are a helpful assistant for
system engineering."},

{"role": "user",
"content": "Give me the example of a SysML block, in json
format, with attributes. No attribute of type String....
The number of ports must be equal to 1."},

{"role": "assistant",
"content": { "name": "Block1", "attributes": [...]}},

{"role": "user",
"content": "Your answer does not respect the following constraint:
The block should have one more attribute than ports.
Update your answer accordingly."}]
...
```



Automated feedback loop: example

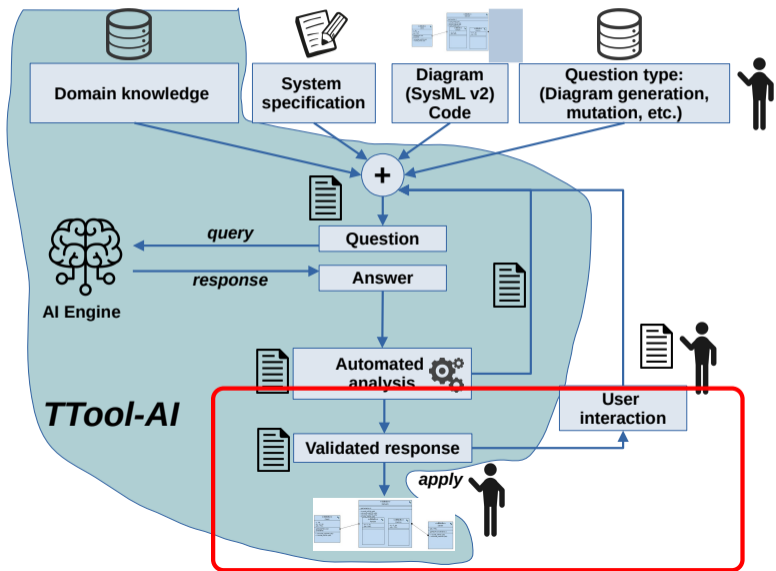


```

{ "name": "Block1", "attributes": [
  {"name": "attribute1", "type": "Integer"},
  {"name": "attribute2", "type": "Boolean"},
  "ports": [
    {"name": "port1", "type": "Port1"}],
  "operations": [
    {"name": "operation1", "parameters": [], "returnType": "ReturnType1"},
    {"name": "operation2", "parameters": [], "returnType": "ReturnType2"}
  ]
}

```

TTool-AI: user is still in the decision loop!





Outline

Introduction

Technical context

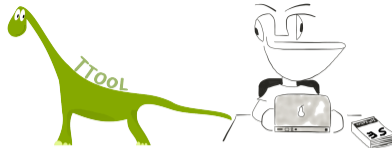
TTool-AI: rationale

TTool-AI: architecture

A student-vs-*TTool-AI* case-study

Conclusion and Future Works

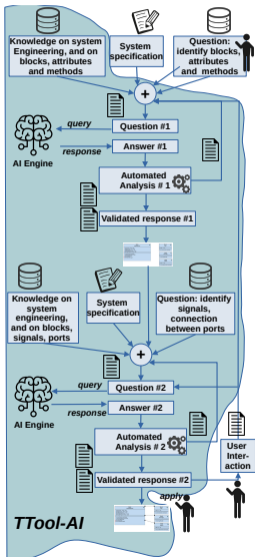
A student-vs-TTool-AI case-study



Experimentation setup

- Submission of the same exam topics to:
 - 15 final-year engineering school students, at the end of a 21-hour course on MBSE with TTool
 - *TTool-AI*
- The exam consisted in:
 - Analyzing a textual specification
 - Designing a block diagram
 - Designing state-machine diagrams for each of these blocks
- Grading of the models by applying the same criteria (syntactic correctness, consistency w.r.t. the initial specification, legibility, ...)

A student-vs-TTool-AI case-study



Focus on the TTool-AI model design feature

- A two-step approach:
 1. Designing the block-diagram
 2. Once validated, designing the state-machine diagrams
- Knowledge is embedded following this two-step approach (constraints on block diagrams, then constraints of state-machine diagrams)
- ⇒ Lowers the response error rate



A student-vs-TTool-AI case-study



	Time BD (s)	Grade BD (/100)	Time SMD (s)	Grade SMD (/100)
Average	40	81	178	63
Std dev.	10	16	97	15

(a) TTool-AI

	Time BD (s)	Grade BD (/100)	Time SMD (s)	Grade SMD (/100)
Average	2700	70	2700	58
Std dev.	—	26	—	32

(b) Students

- Better average for TTool-AI on grades, and it is much faster
- Best students always do better than TTool-AI, but with a much higher time
 - And it is hard to predict which students will be the best for a given specification

A student-vs-TTool-AI case-study



Evaluation limitations

1. Relies on final-year master students, not experienced professionals
 - Experienced engineers may produce better quality diagrams
 - However, the time gain with AI should still be substantial
2. A single evaluator
 - Yet evaluation criteria are well-defined: adequacy of the diagrams to the specification, diagrams legibility, number of blocks and states and consistency of their naming, absence of attributes declared in blocks but not utilized in state-machine diagrams, number of errors and warnings detected by TTool's syntax checker.



Outline

Introduction

Technical context

TTool-AI: rationale

TTool-AI: architecture

A student-vs-*TTool-AI* case-study

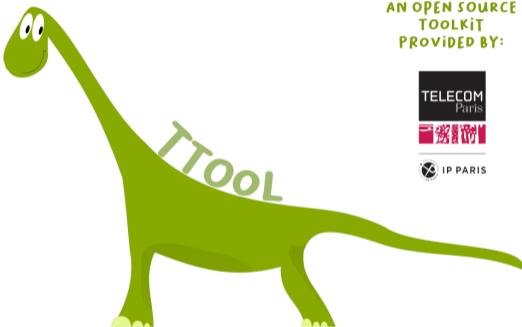
Conclusion and Future Works



To Go Further with TTool-AI

- **Main strengths of the tool:** dynamic knowledge embedding, automated feedback loop, automated integration of the LLM responses in the models
- **Analysis of system specification is rather impressive:** extracting requirements, use cases, design (structure, behavior), properties identification and model mutation
- **Integration of GPT-4:** our current tests suggest that knowledge embedding shall be more efficient as GPT-4 can efficiently analyze technical documentation
- **Extension of TTool-AI to our other SysML profile** targeting software/hardware partitioning
- **Embedding knowledge on security countermeasures** to automatically integrate security patterns in models

Questions?



Latest version of TTool includes *TTool-AI*... and much more!
ttool.telecom-paris.fr