



ChatGPT and (Model-Driven) System Engineering

First investigations and implementations

Ludovic Apvrille

ludovic.apvrille@telecom-paris.fr

<https://perso.telecom-paris.fr/apvrille/>



Outline

Introduction

Context

Contribution overview

Application to System Engineering

Demo

Conclusion



Introduction

ChatGPT and System Engineering

- These slides give preliminary ideas on how ChatGPT (3) could be used system engineering
 - Modeling, verification
- Version 4 of ChatGPT is even better, not covered by this presentation

Outline

- Context: methodologies for designing embedded systems, ChatGPT (and similar AI)
- Contribution on S.E.: general approach, implementation aspects
- Demo, conclusion



Outline

Introduction

Context

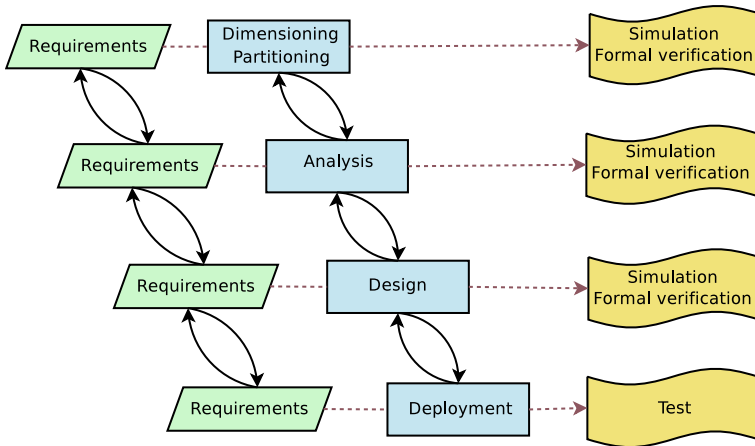
Contribution overview

Application to System Engineering

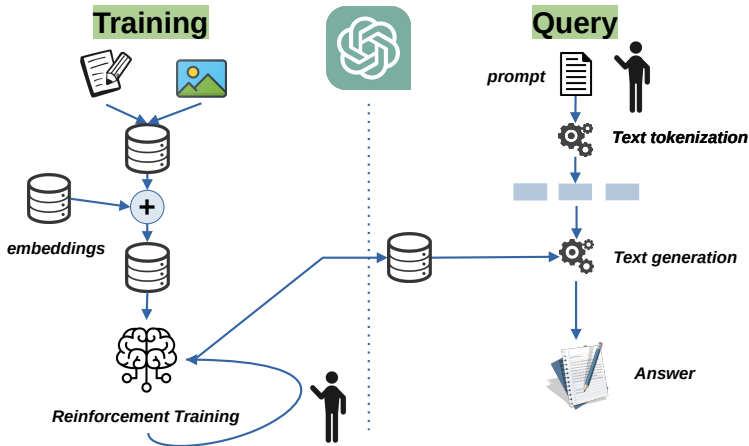
Demo

Conclusion

Designing an Embedded System



ChatGPT in a Nutshell



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

Our Idea: AI&ChatGPT to Support System Engineering

(Model-Based) System Engineering = document understanding and transformation into views/diagrams/code



Input

- Improve ChatGPT to understand inputs such as:
 - System specification
 - Objective of the request: identifying requirement, identifying system blocks, etc.
 - Constraints from system architects
 - Existing diagrams or code

Output

- Drive ChatGPT to produce answers related to system engineering
- Drive ChatGPT to produce an easy-to-parse output
- Automatically identify if produced outputs are invalid/wrong, and force ChatGPT to produce better answers



Outline

Introduction

Context

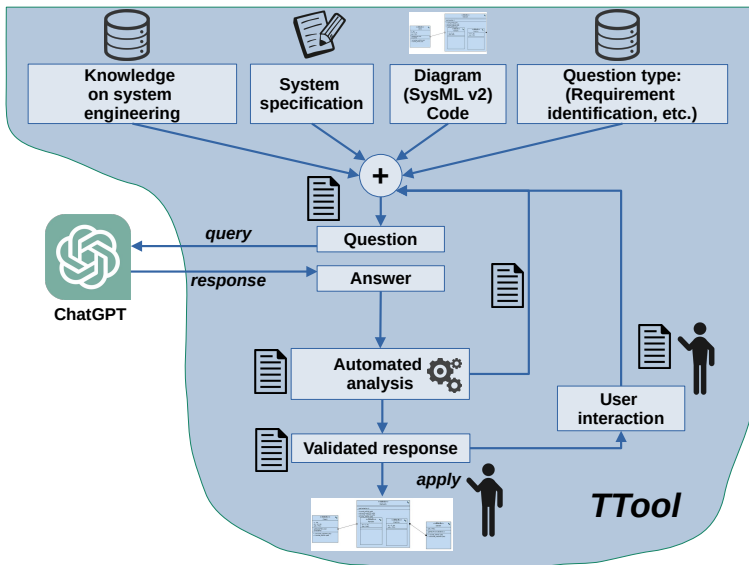
Contribution overview

Application to System Engineering

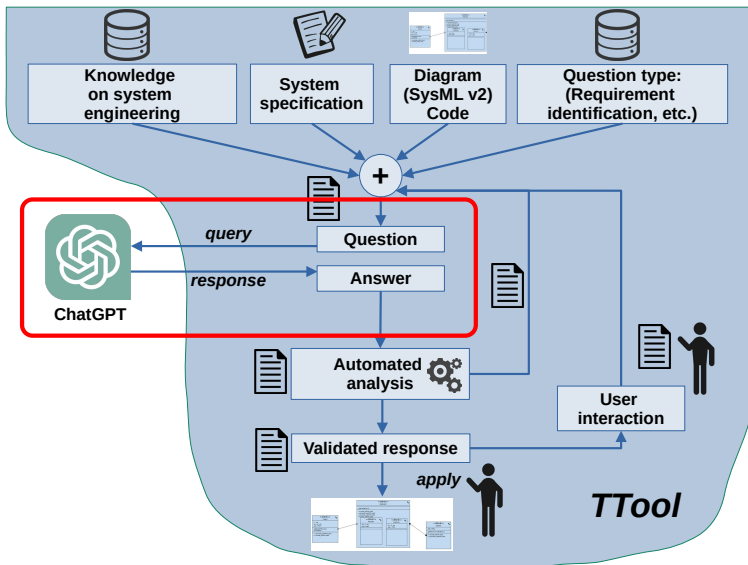
Demo

Conclusion

Contribution: Overview



Contribution: Interacting with ChatGPT



ChatGPT API: the Basic Way



```
$ curl https://api.openai.com/v1/chat/completions
-H "Content-Type: application/json"
-H "Authorization: Bearer OPENAI_API_KEY"
-d '{ "model": "gpt-3.5-turbo",
  "messages": [{"role": "user", "content": "Hello!"}]
}'

{ "id": "chatcmpl-7YzJ6Gmh0VjbWl3plUxqwzDrtDnnk",
  "object": "chat.completion", "created": 1688573172,
  "model": "gpt-3.5-turbo-16k-0613", "choices": [ { "index": 0,
  "message": { "role": "assistant",
  "content": "Hello! How can I assist you today?" },
  "finish_reason": "stop" } ], \
  "usage": { "prompt_tokens": 8, "completion_tokens": 9,
  "total_tokens": 17 } }
```

You first need to buy tokens to get a valid OPENAI_API_KEY

ChatGPT API: Still quite Basic



```
$ curl https://api.openai.com/v1/chat/completions  
-H "Content-Type: application/json"  
-H "Authorization: Bearer OPENAI_API_KEY"  
-d '{ "model": "gpt-3.5-turbo",  
      "messages": [{"role": "user", "content": "Hello!"}]  
}' | jq '.choices[]'.message.content
```

Hello! How can I assist you today?

```
$ chatgpt "Is Supaero a good engineering school?"
```

```
"Supaero, also known as the Institut Supérieur de l'Aéronautique  
et de l'Espace, is a renowned engineering school specializing  
in aerospace engineering in France. It consistently ranks among  
the top engineering schools globally and ..."
```

ChatGPT API: Pricing



Token-based

- Input / output tokens
- Limits on the number of queries per unit of time

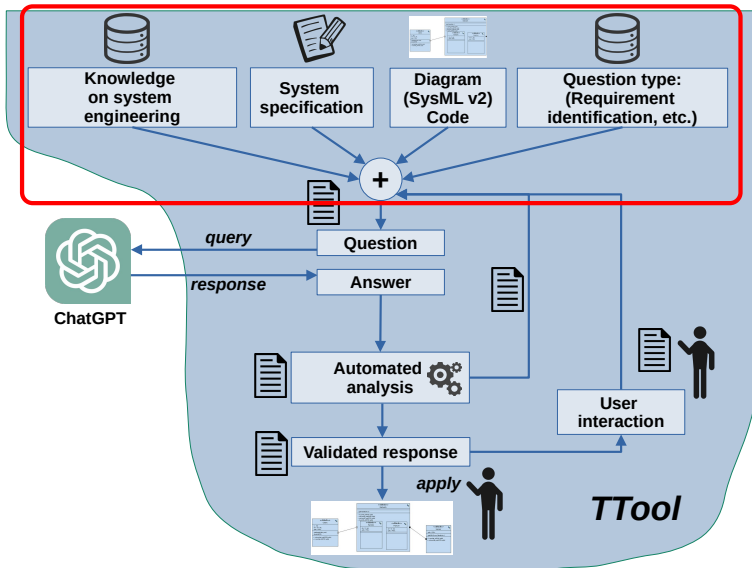
ChatGPT 3.5

Model	Input	Output
4K context	\$0.0015 / 1K tokens	\$0.002 / 1K tokens
16K context	\$0.003 / 1K tokens	\$0.004 / 1K tokens

ChatGPT 4

Model	Input	Output
8K context	\$0.03 / 1K tokens	\$0.06 / 1K tokens
32K context	\$0.06 / 1K tokens	\$0.12 / 1K tokens

Contribution: Custom Knowledge (or Context)



ChatGPT API: Custom Knowledge (or Context) (Cont.)



```
$ curl https://api.openai.com/v1/chat/completions
```

```
...
```

```
"messages": [{"role": "user", "content": "Is Sophia-Antipolis sunny today?"}]
```

```
...
```

```
"I m sorry, as an AI language model, I dont have access to current weather conditions. However, Sophia-Antipolis is located in the French Riviera, which has a generally sunny and Mediterranean climate..."
```



ChatGPT API: Custom Knowledge (or Context) (Cont.)



```
$ curl https://api.openai.com/v1/chat/completions \
```

```
...
```

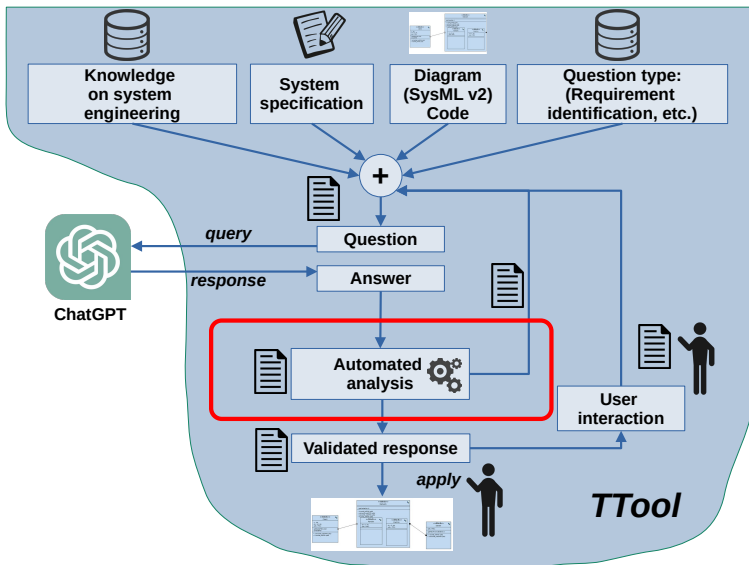
```
{"messages":[{"role":"user","content":"Today, in Sophia-Antipolis,  
the sun shines with no clouds."},{ "role":"assistant","content":"ok"},  
{"role":"user","content":"Is Sophia-Antipolis sunny today?"}]
```

```
...
```

```
"Yes, you mentioned earlier that the sun is shining with no clouds  
in Sophia-Antipolis. So, it is indeed sunny today in  
Sophia-Antipolis. Enjoy the beautiful weather!"
```



Contribution: Automated Answer Analysis



Automated Answer Analysis: Objectives

Invalid answer

Answer that does not respect the expected output:

- Invalid format (JSON, XML, SysML v2, ...)
- Constraints not respected
 - "List two blocks in json format", but the answer gives only one block

Detection and iteration

1. Detect format or constraint 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 Answer Analysis: 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 Answer Analysis: Example (Cont.)



```
{ "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 Answer Analysis: Example (Cont.)

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."}]
...
```

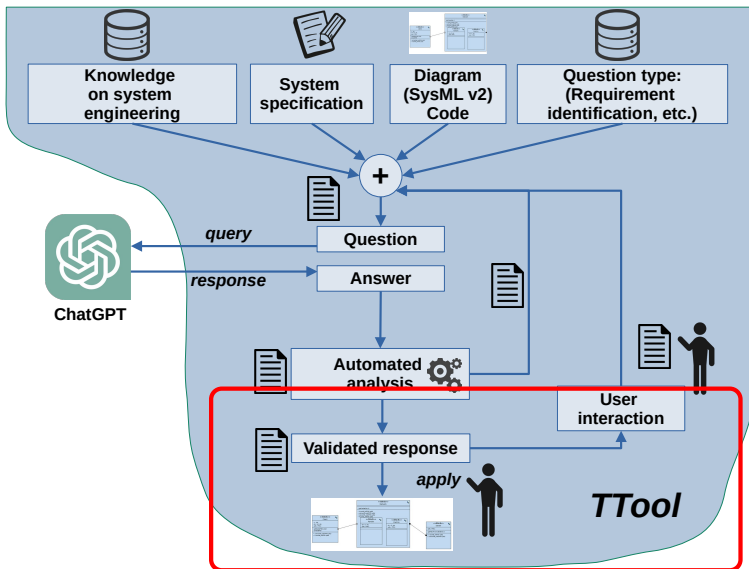


Automated Answer Analysis: Example (Cont.)



```
{ "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"}  
  ]  
}
```

Contribution: User Interaction and Result Displaying





Outline

Introduction

Context

Contribution overview

Application to System Engineering

Demo

Conclusion

General Contribution Applied to System Engineering

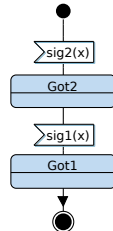
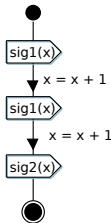
Objective: efficiently use the presented framework for automated system engineering

- Define a **context** (knowledge in system engineering, output format, expected output (a diagram, a list of elements, etc.))
- Define a **list of questions** linked with System Engineering
 - Requirement identification, requirement classification, ...
- Mandatory / optional **inputs for questions**:
 - System specification (text) - user provided
 - Diagram (in SysML V2 format)
- **Automated analysis of answers, and iterations**
- **Diagram drawing** from output

Focus on SysML Design

SysML design

= a set of interconnected blocks (structure) and one state machine for each block (behavior)



Focus on SysML Design: Blocks

Context for the identification of blocks and attributes

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.

...

Focus on SysML Design: Blocks (Cont.)

Question for the identification of blocks and attributes

From the following system specification, using the specified JSON format, identify the typical system blocks and their attributes. Do respect the JSON format, and provide only JSON (no explanation before or after).

- Question is concatenated with the user-provided specification and user-provided (extra) constraints
- Automated verification: JSON format, constraints

Similar approach for:

- Identification of connections between blocks
- Identification of state machines

Focus on SysML Design: Blocks (Cont.)

Context for connecting blocks

...

#Respect 2 signals with the same name are assumed to be connected: this is the only way to connect signals.

#Respect: Two connected signals must have the same list of attributes, even if they are defined in two different blocks. One of them must be output, the other one must be input.

#Respect: all input signals must have exactly one corresponding output signal, i.e., an output signal with the same name

...

Question for connecting blocks

From the previous JSON and system specification, update this JSON with the signals you have to identify. If necessary, you can add new blocks and new attributes.



Outline

Introduction

Context

Contribution overview

Application to System Engineering

Demo

Conclusion

Demo: a Coffee Machine

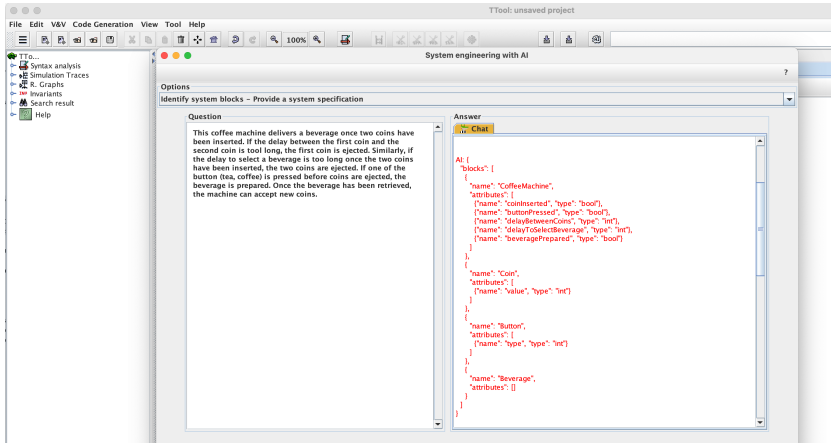
Objective: making the block and state machine diagrams of the following system:

This coffee machine delivers a beverage once two coins have been inserted. If the delay between the first coin and the second coin is too long, the first coin is ejected. Similarly, if the delay to select a beverage is too long once the two coins have been inserted, the two coins are ejected. If one of the button (tea, coffee) is pressed before coins are ejected, the beverage is prepared. Once the beverage has been retrieved, the machine can accept new coins.

TTool is used for query/answer to/from ChatGPT and diagram drawing
`ttool.telecom-paris.fr`



Demo: a Coffee Machine (Cont.)



The screenshot shows the TTool software interface. The main window is titled "System engineering with AI" and contains a chat window. The chat window has a "Question" section and an "Answer" section.

Question:

This coffee machine delivers a beverage once two coins have been inserted. If the delay between the first coin and the second coin is too long, the first coin is ejected. Similarly, if the delay to select a beverage is too long once the two coins have been inserted, the two coins are ejected. If one of the buttons (tea, coffee) is pressed before coins are ejected, the beverage is prepared. Once the beverage has been retrieved, the machine can accept new coins.

Answer:

```
AI [
  blocks: [
    {
      name: "CoffeeMachine",
      attributes: [
        {name: "coinInserted", type: "bool"},
        {name: "buttonPressed", type: "bool"},
        {name: "delayBetweenCoins", type: "int"},
        {name: "delayToSelectBeverage", type: "int"},
        {name: "beveragePrepared", type: "bool"}
      ]
    },
    {
      name: "Coin",
      attributes: [
        {name: "value", type: "int"}
      ]
    },
    {
      name: "Button",
      attributes: [
        {name: "type", type: "int"}
      ]
    },
    {
      name: "Beverage",
      attributes: []
    }
  ]
}
```


Demo: a Coffee Machine (Cont.)

Options

Identify system blocks – Provide a system specification

Question

This coffee machine delivers a beverage once two coins have been inserted. If the delay between the first coin and the second coin is too long, the first coin is ejected. Similarly, if the delay to select a beverage is too long once the two coins have been inserted, the two coins are ejected. If one of the buttons (tea, coffee) is pressed before coins are ejected, the beverage is prepared. Once the beverage has been retrieved, the machine can accept new coins.

Answer

Chat

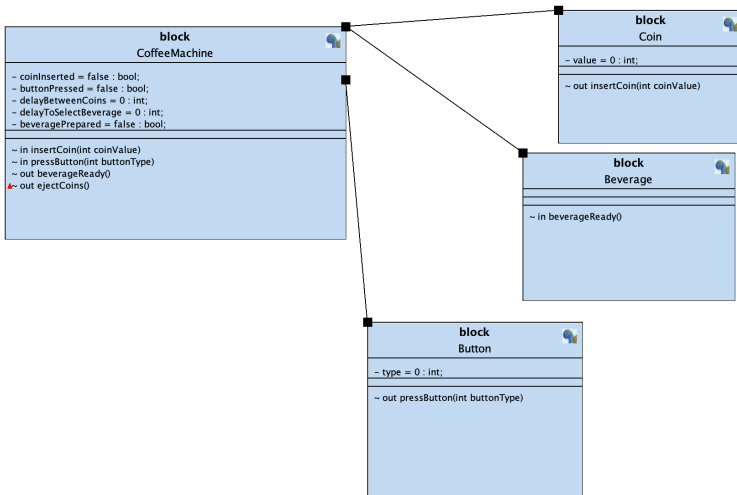
TTTool: From the previous JSON and system specification, update this JSON with the signals you have to identify. If necessary, you can add new blocks and new attributes.

```

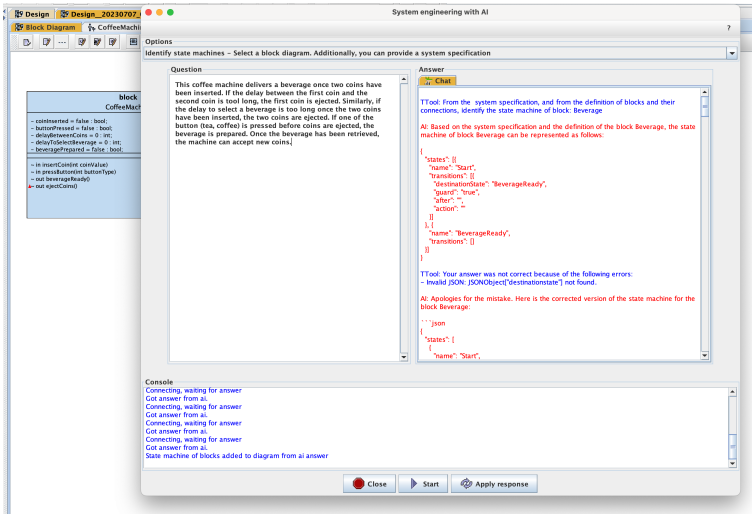
AI: {
  "blocks": [
    {
      "name": "CoffeeMachine",
      "attributes": [
        {"name": "coinInserted", "type": "bool"},
        {"name": "buttonPressed", "type": "bool"},
        {"name": "delayBetweenCoins", "type": "int"},
        {"name": "delayToSelectBeverage", "type": "int"},
        {"name": "beveragePrepared", "type": "bool"}
      ],
      "signals": [
        {"signal": "input insertCoin(int coinValue)"},
        {"signal": "input pressButton(int buttonType)"},
        {"signal": "output beverageReady()"},
        {"signal": "output ejectCoins()"}
      ]
    },
    {
      "name": "Coin",
      "attributes": [
        {"name": "value", "type": "int"}
      ],
      "signals": [
        {"signal": "output insertCoin(int coinValue)"}
      ]
    }
  ],
  {

```

Demo: a Coffee Machine (Cont.)



Demo: a Coffee Machine (Cont.)



The screenshot shows a software interface titled "System engineering with AI". On the left, a "Block Diagram" window displays a block named "CoffeeMachine" with the following properties:

```

- coinInserted = false : bool;
- buttonPressed = false : bool;
- delayBetweenCoins = 0 : int;
- delayToSelectBeverage = 0 : int;
- beveragePrepared = false : bool;

+ in insertCoin(int coinValue)
+ in pressButton(int buttonType)
+ out beverageReady()
+ out ejectCoin()
  
```

The main window contains a chat interface with the following text:

Question

This coffee machine delivers a beverage once two coins have been inserted. If the delay between the first coin and the second coin is too long, the first coin is ejected. Similarly, if the delay to select a beverage is too long once the two coins have been inserted, the two coins are ejected. If one of the buttons (tea, coffee) is pressed before coins are ejected, the beverage is prepared. Once the beverage has been retrieved, the machine can accept new coins.

Answer

TTool: From the system specification, and from the definition of blocks and their connections, identify the state machine of block: Beverage

AI: Based on the system specification and the definition of the block Beverage, the state machine of block Beverage can be represented as follows:

```

{
  "states": [
    {
      "name": "Start",
      "transitions": [
        {
          "destinationState": "BeverageReady",
          "guard": "true",
          "after": "",
          "action": ""
        }
      ]
    },
    {
      "name": "BeverageReady",
      "transitions": []
    }
  ]
}
  
```

TTool: Your answer was not correct because of the following errors:
- Invalid JSON: JSONObject["destinationState"] not found.

AI: Apologies for the mistake. Here is the corrected version of the state machine for the block Beverage:

```

{
  "states": [
    {
      "name": "Start",
  
```

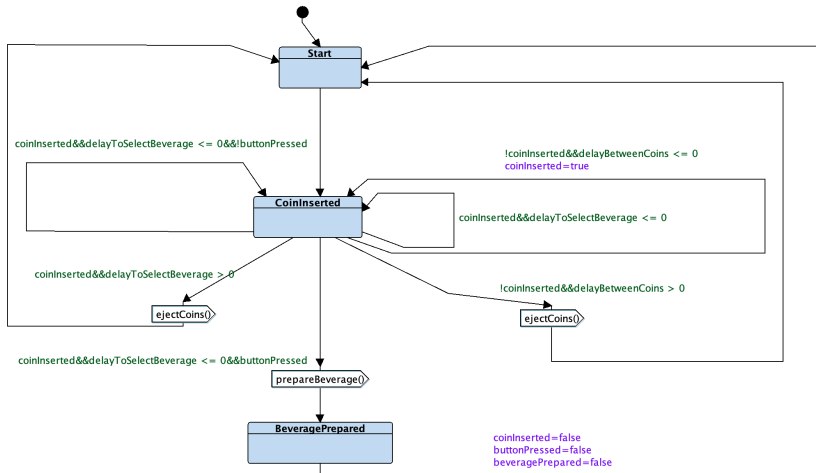
Console

```

Connecting, waiting for answer
Get answer from ai.
Connecting, waiting for answer
Get answer from ai.
Connecting, waiting for answer
Get answer from ai.
Connecting, waiting for answer
Get answer from ai.
Connecting, waiting for answer
Get answer from ai.
State machine of blocks added to diagram from ai answer
  
```

At the bottom of the chat window, there are three buttons: "Close", "Start", and "Apply response".

Demo: a Coffee Machine (Cont.)





Outline

Introduction

Context

Contribution overview

Application to System Engineering

Demo

Conclusion

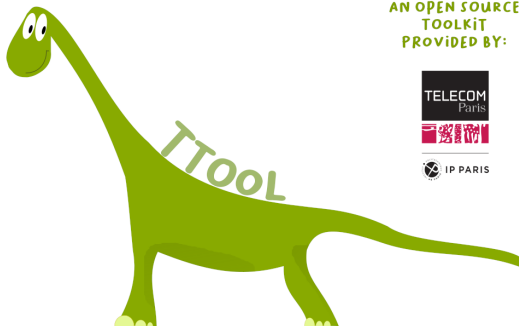


To Go Further...

ChatGPT is a chatbot: it tells stories related to your questions, elaborating a probable answer, with no guarantees that the answer is correct.

1. Analysis of system specification is rather impressive: extracting requirements, use cases, design (structure, behavior) and properties identification
2. Modification of systems (integration of patterns, general mutations) is also successful but requires very large contexts
 - Trade-off between price and time
3. Verifying that a property is true for a given system is doubtful (ChatGPT can be wrong at solving trivial mathematical exercises. . .)
 - ChatGPT refuses to prove properties for complex diagram
 - For simple cases, from which it can draw a step-by-step proof, it can succeed to propose the proof of properties from a formal system description

Questions?



Latest version of TTool includes what has been demonstrated... and much more!
ttool.telecom-paris.fr