International Council on Systems Engineering
A better world through a systems approach

Accelerating Model-Based Systems Engineering with Large Language Models

Khushnood Adil Rafique, Sanan Shah,
Šandor Dalecke, Christoph Grimm

INCOSE International Symposium 2025 | Ottawa, Canada

Today's Agenda

- SysML v2 & MBSE
- Methodology
- Results (Syntax, Semantic, Structure)
- Output Samples
- Conclusion

Why?

- MBSE adoption is slow
- Document based legacy documents exist
- Steep learning curve

MBSE

- Early Analysis and Simulation
- Traceability and Consistency
- Automation and Integration
- Life-cycle Management

SysML v2

- Based on KerML
- General-purpose modeling language
- Textual & graphical model representation
- Textual especially suited for AI

LLM selection

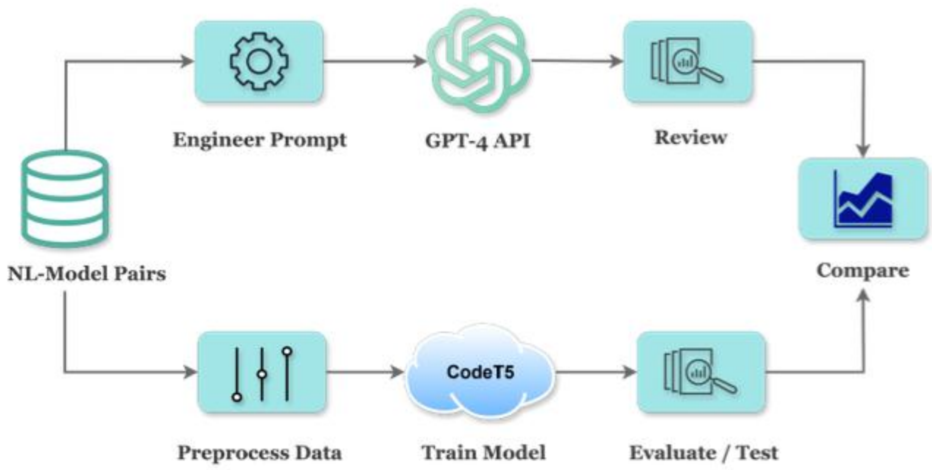
GPT-4

- Large-scale
- General-purpose
- In-context learning
- Massive parameter count

CodeT5

- Small & efficient
- Domain-specific
- Fine-tuned for programming tasks
- Pre-trained on open-source dataset

Methodology



Natural Language to SysML v2 Skeleton

GPT-4

- Engineer Prompt
- Connect to API
- Review model skeleton

CodeT5

- Preprocess Data
- Train Model
- Evaluate/Test model skeletons

Methodology

Dataset CodeT5

- 2.000 curated NL + SysML v2 model samples
- 80% training data, 10% validation data, 10% testing subsets

Prompting GPT-4

- In-context learning
- Prompt structure: 3 templates from training set
- One previously unknown NL description

GPT-4 Prompt Structure & Example

- Instruction prompt
- 3 Description + model examples
- New Model Description

GPT-4 Prompt Structure
Instruction Prompt: Based on the examples of SysML V2 models, and their respective natural language descriptions, shown below, analyze them the best you can, and with the knowledge you acquire, create a SysML V2 Model from the 'New NL description'
Example 1: Description. [NL Example 1] SysML V2 Model: [SysML V2 Model Example 1]
Example 2: Description. [NL Example 2] SysML V2 Model: [SysML V2 Model Example 2]
Example 3: Description. [NL Example 3] SysML V2 Model: [SysML V2 Model Example 3] '**New NL Description**': [NL Test Sample]

GPT-4 Prompt Structure

GPT-4 Prompt Structure & Example

Description. "The *State Actions* package models the operational behavior of a vehicle using a state machine driven by external signals. It defines three signal attributes—`VehicleStartSignal`, `VehicleOnSignal`, and `VehicleOffSignal`—alongside a part definition for a `Vehicle`. The core behavior is captured in the `VehicleStates` state machine, which operates on an input vehicle instance. Upon entry, the state machine transitions to the `off` state. When the `VehicleStartSignal` is received, the system moves from `off` to `starting`, and upon receiving the `VehicleOnSignal`, it transitions to the on state. In the on state, a `performSelfTest` action is executed on entry using the vehicle instance, followed by a continuous `providePower` action during the state, and an `applyParkingBrake` action upon exit. At any point, receiving the `VehicleOffSignal` causes the system to return to the `off` state. This structure ensures that the vehicle transitions through defined operational phases while responding to external control signals."

- Signals: Start; On; Off
- Off → Starting → On
- On: Perform Selftest → continous: provide power → exit: apply parking break

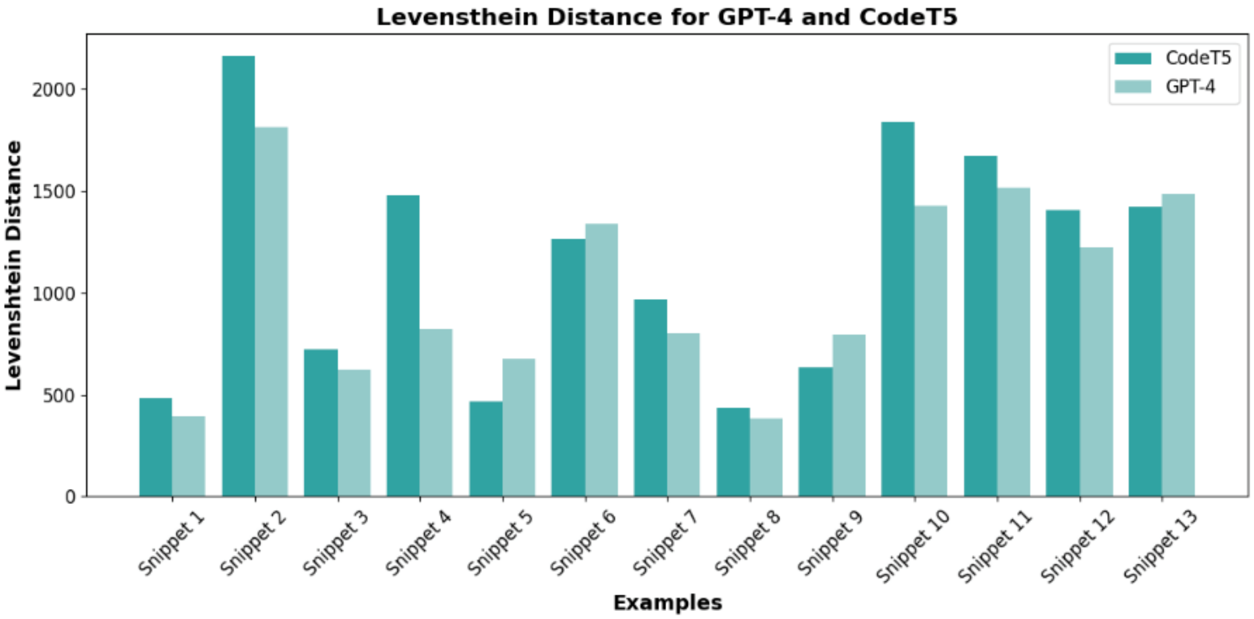
GPT-4 Prompt Structure & Example

```
SysML V2 Model:

package 'State Actions' {
  attribute def VehicleStartSignal;
  attribute def VehicleOnSignal;
  attribute def VehicleOffSignal;

  part def Vehicle;

  action performSelfTest { in vehicle : Vehicle; }

  state def VehicleStates { in operatingVehicle : Vehicle; }

  state vehicleStates : VehicleStates {
    in operatingVehicle : Vehicle;

    entry; then off;

    state off;
    accept VehicleStartSignal
      then starting;

    state starting;
    accept VehicleOnSignal
      then on;

    state on {
      entry performSelfTest { in vehicle = operatingVehicle; }
      do action providePower { /... / }
      exit action applyParkingBrake { /... / }
    }

    accept VehicleOffSignal
      then off;
  }
}
```

SysML v2 model created by GPT-4 with previous NL description

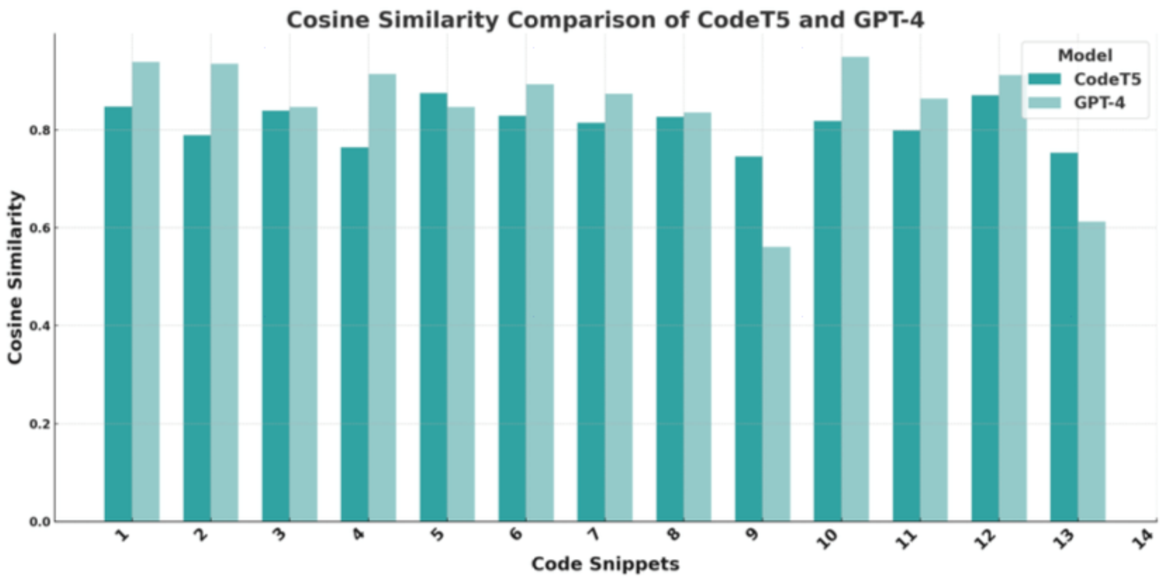
Inference Metrics – Syntax Accuracy

- Levenshtein (1966) Distance
- GPT-4 usually closer to reference models
- Snippet 5&9 have more precise NL-description → CodeT5 outperforms GPT-4



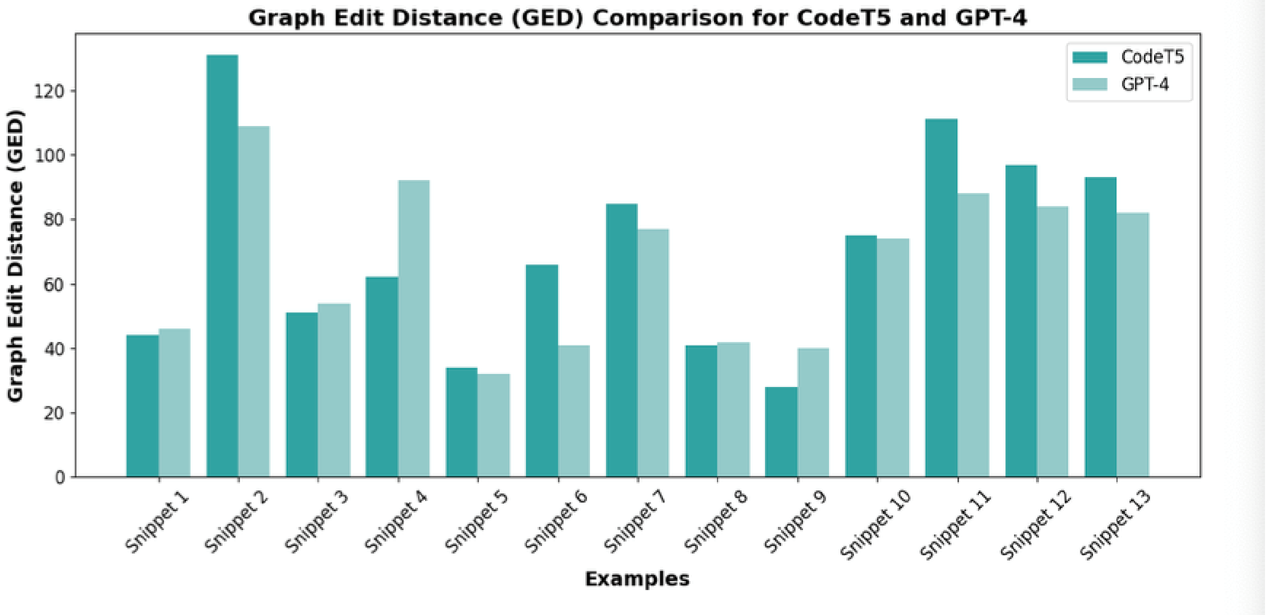
Inference Metrics – Semantic Alignment

- Cosine Similarity (Manning, 2008) and Jaccard (1901) Similarity
- GPT-4 shows more variance
- CodeT5 still high results



Inference Metrics – Structural Correctness

- Graph Edit Distance (Bunke, 2000)
- SysML v2 models converted to directed graphs
- Variability reflects complexity of samples
- GPT-4 more consistent



Inference Metrics Example 1

NL Description

Test 1:

A SysML V2 model defines a Traffic Light. In the package, model a traffic light. The traffic light shall have three states: red, yellow, and green.The behavior is modeled by a state machine. The state machine is first in the state red.Then, it goes into the state yellow and then green.From green it goes into red.

CodeT5 output

Output Test 1:

```
package 'Traffic Light' {
        import ISQ::*;
        package TrafficLight {
                part def Light {
                        state red;
                        then yellow;
                then green;
                state green;
                }
        }
}
```

GPT-4 output

Output Test 1:

```
package 'Traffic Light' {

    part def TrafficLight {
        state machine LightBehavior {
            state Red;
            state Yellow;
            state Green;

            transition RedToYellow from Red to Yellow;
            transition YellowToGreen from Yellow to
Green;
            transition GreenToRed from Green to Red;

            initial state Red;
        }
    }
}
```

Example 1 with NL iinput and CodeT5 and GPT-4 output

Inference Metrics Example 2

NL Description

Test 2:

This SysML model defines the composition of vehicles. Define a vehicle as a part that has between two and six wheels, one or no engine, and a body. Special vehicles are electric vehicles and bicycles.Bicycles have only two wheels. Electric vehicles also have a battery.

CodeT-5 output

Output Test 2:

```
package 'Vehicle Composition' {
        import ScalarValues::*;
        part vehicle : Vehicle {
                part wheels;
                part Engine;
                part body;
        }
           part ElectricVehicle {
        part Battery;
    }
    part Bicycle {
        define Wheels;
    }
}
```

GPT-4 output

Output Test 2:

```
package 'Vehicle Composition' {
    part def Vehicle {
        attribute Wheels : Integer [2..6];
        attribute Engine : Boolean [0..1];
        part def Body;
    }
    part def ElectricVehicle specializes Vehicle {
        part def Battery; // Electric vehicles have a
battery
    }
    part def Bicycle specializes Vehicle {
        redefine Wheels : Integer [2..2];
        redefine Engine : Boolean [0..0];
    }
}
```

Example 2 with NL input and CodeT5 and GPT-4 output

Performance across metrics

index	Aspect	GPT-4	CodeT5
0	Model Type	General-purpose	Task-specific
1	Parameter Count (Estimated)	Hundreds of billions	220M (base model)
2	Training Dataset Size	Diverse corpus, including natural language and code	CodeSearchNet dataset (9 programming languages)
3	Pre-training Domain	General text and code	Programming code
4	Fine-tuning Capability	No fine-tuning needed	Requires fine-tuning
5	Training Time (SysML Task)	N/A (uses in-context learning)	Approx. 2 hours on NVIDIA A100
6	Evaluation Metrics	Semantic similarity, syntax accuracy, structural correctness	Semantic similarity, syntax accuracy, structural correctness
7	Average Levenshtein Distance	1034.2	1151.5
8	Average Cosine Similarity	0.85	0.81
9	Average Jaccard Similarity	0.42	0.40
10	Average Graph Edit Distance (GED)	68.3	74.7

T-Test

index	Metric	P-Value (p)	Interpretation	Key Insights
0	Levenshtein Distance	0.045	Statistically significant differences in syntax accuracy.	CodeT5 demonstrates superior performance in generating structured syntax.
1	Cosine Similarity	0.019	Statistically significant differences in semantic alignment.	GPT-4 excels in tasks requiring contextual understanding of natural language inputs.
2	Jaccard Similarity	0.032	Statistically significant differences in semantic alignment.	GPT-4 further confirms its strength in maintaining semantic relationships.
3	Graph Edit Distance (GED)	0.081	Differences are not statistically significant at the 0.05 level.	Smaller models like CodeT5 can be used when structural correctness is prioritized over semantic nuances.

Correction Effort Estimate

- Post-hoc effort estimation on 5 models each
- Minor: naming/small structure fixes
- Moderate: missing relationships/components
- Major: restructuring key elements

Model	Minor Fixes	Moderate Fixes	Major Fixes	Total Estimated Effort (minutes)
GPT-4 Output 1	2	1	0	8
GPT-4 Output 2	1	0	0	2
GPT-4 Output 3	3	1	0	10
GPT-4 Output 4	1	1	0	6
GPT-4 Output 5	2	0	0	4
CodeT5 Output 1	2	2	1	18
CodeT5 Output 2	3	1	0	10
CodeT5 Output 3	1	2	1	16
CodeT5 Output 4	2	1	1	14
CodeT5 Output 5	1	2	0	10

Conclusion

- Evaluation of two LLM classes for SysML v2 model skeleton creation from NL
- GPT-4 performs well at understanding semantic relationships
- CodeT5 demonstrates aptitude for syntax accuracy
- Integration of AI powered model generation can streamline workflows
- GPT-4 outperforms CodeT5 less than expected

References

- Bunke, H. (2000). Graph Matching: Theoretical Foundations, Algorithms, and Applications. *Proceedings of Vision Interface*, 82–88.
- Jaccard, P. (1901). Étude Comparative de la Distribution Florale dans une Portion des Alpes et des Jura. *Bulletin de la Société Vaudoise des Sciences Naturelles, 37*, 547–579.
- Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady, 10*(8), 707–710.
- Manning, Christopher D.; Raghavan, Prabhakar; Schütze, Hinrich. (2008). *Introduction to Information Retrieval.* Cambridge: Cambridge University Press.
- Student, F. (1908). The probable error of a mean. *Biometrika*, 1-25.

Limitations

- Implicit reliance on style and structure of SysML v2 training examples
- Organization specific style guides can pose a challenge
- Intellectual property and data confidentiality constrain use of open source models