



INSTITUT
POLYTECHNIQUE
DE PARIS

Well-Formed Executable Suggestions for Continuous MDE

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

MODELS '26

Outline

Some context

Well-formed executable suggestions

Deriving Well-formed Executable Suggestions

SysML Instantiation and Some Results

Takeaways

Some context

Problem addressed

A multi-purpose modeling recommender framework (model repair, completion, and creation from scratch) producing recommendations that are *executable* and preserve model *well-formedness*

Related work

- 50+ approaches, including recent contributions from the community (MORGAN, MemoRec, MAGDA, NEMO, ...)
- Our goal: combine multi-purpose recommendation, executability, and well-formedness preservation within a single framework

Some context

In the last episode

- ✓ An LLM-based modeling recommender system integrated into TTool, targeting SysML requirement, use-case, block, and state-machine diagrams
- ✗ Suggestions remained textual: execution required a separate translation step, with no guarantee of applicability or of well-formedness preservation

Our approach

- A two-stage LLM-based generation approach
 1. Generates a modeling suggestion
 2. Turns it into an executable model mutation
- LLMs propose, MDE verification mechanisms establish applicability and preservation of model well-formedness

Outline

Some context

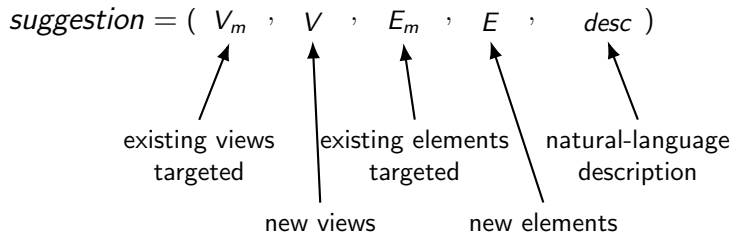
Well-formed executable suggestions

Deriving Well-formed Executable Suggestions

SysML Instantiation and Some Results

Takeaways

Suggestions and executable suggestions



$executable\ suggestion = (V_m, V, E_m, E, desc, scr)$

mutation script

Well-formedness criteria (with respect to a model)

At suggestion level

The model elements identified to be modified actually exist in the model
(→ *ensures that no nonexistent element is passed to the executable suggestion generation process*)

At executable suggestion level

1. The mutation script is syntactically correct and the identifiers it contains refer to existing model elements
(→ *ensures that the script is applicable to the model*)
2. Applying this script to the model yields a syntactically well-formed model
(→ *ensures that the resulting model can be simulated, verified, etc.*)

⇒ An *executable suggestion* satisfying these two criteria is a *well-formed executable suggestion*.

Outline

Some context

Well-formed executable suggestions

Deriving Well-formed Executable Suggestions

SysML Instantiation and Some Results

Takeaways

Verification mechanisms

⊆ Suggestion inclusion check

Checks that the model elements identified by the suggestion as targets actually exist in the current model

→ invalid references are reported back to the LLM

⊨ Well-formedness checks

- **Mutation script**

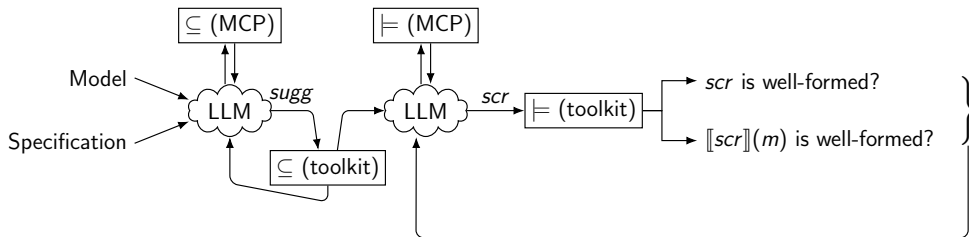
Checks whether the script satisfies the well-formedness rules of the target *mutation language*

- **Resulting model**

Applies the mutation and checks whether the resulting model satisfies the well-formedness rules of the target *modeling language*

→ Exposed to the LLM through MCP to provide feedback during generation; run again externally to ensure the well-formedness guarantees.

Deriving well-formed executable suggestions



Outline

Some context

Well-formed executable suggestions

Deriving Well-formed Executable Suggestions

SysML Instantiation and Some Results

Takeaways

Instantiation



Modeling language: a SysML subset

- Design models composed of a **block diagram** and associated **state-machine diagrams**
- Model transformations provide interoperability with **SysML v2** representations

Mutation language: AMULET

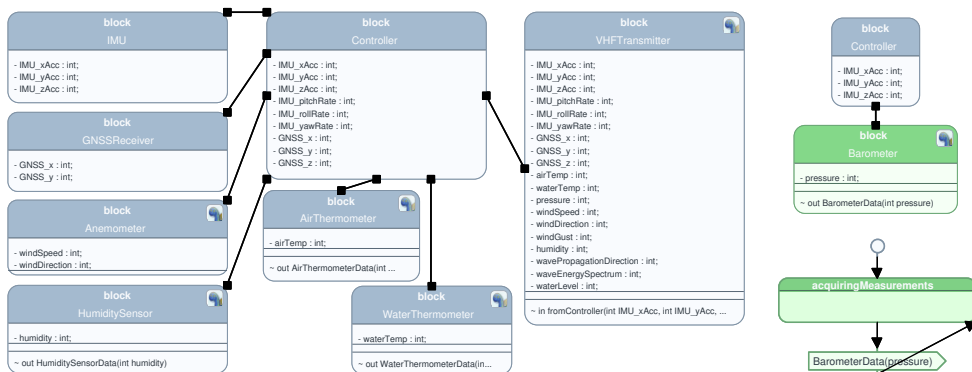
- A mutation language specifically targeting this SysML subset
- Built on **deterministic mutation operators** for adding, removing, and modifying model elements and views
- Expressive enough to construct any model **from scratch**

NB: The implementation displays both well-formed and non-well-formed executable suggestions (yet only suggestions marked as well-formed can be directly applied to the model).

Evaluation setup

- **Two case studies, seven model variants**
 - Meteorological oceanographic buoy: incomplete and overspecified models, each with an additional syntactic variant
 - Coffee-machine controller: incomplete, overspecified, and timing-inconsistent models
- **35 executable-suggestion generation runs**
 - 5 runs for each of the 7 model variants, using GPT-5.4 Codex
- **Metrics**
 - Generation time and well-formedness
 - Suggestion relevance
 - Mutation-script faithfulness to the natural-language suggestion
 - Model improvement after application of the suggestion

Evaluation



Overall results

Well-formedness rate

27 / 35 generated suggestions were well-formed (77%)

For well-formed executable suggestions

- **98/100** average suggestion relevance
- **97/100** average mutation-script faithfulness
- **93/100** average model improvement after application
- **194 s** average generation time

Across suggestion purposes

Completion, deletion, and modification suggestions were all generated → well-formedness rates were respectively **73%**, **100%**, and **56%** in our sample.

Outline

Some context

Well-formed executable suggestions

Deriving Well-formed Executable Suggestions

SysML Instantiation and Some Results

Takeaways

Takeaways

- Executable suggestions bridge natural-language recommendations and actual model transformations
- Mutation scripts make suggestions directly applicable and externally verifiable
- Rule-based verification is used twice: during generation via MCP, and after generation in the MDE toolkit
- Initial results show that the approach can support completion, deletion, and modification suggestions

Next step

Move from syntactic well-formedness preservation toward semantic validation, e.g., through model checking.