ChatXmile
An LLM Assisted Closed Loop Workflow for the System Dynamics Process

Extended Abstract

Bo Hu[1,4], Bo Zhang[2], Justin Hoffmann[1,3], Michael Woywadt[4]

[1] Universität der Bundeswehr München, 85577 Neubiberg, Germany
[2] Information Center, Ministry of Ecology and Environment, Beijing, China
[3] Helmut-Schmidt-University, 22043 Hamburg, Germany
[4] Horus Advisory Group, Munich, Germany

Keywords: LLM-assisted modeling; XMILE; technical verification; model reconstruction; web-based simulators

Research purpose

ChatXmile combines large language models (LLMs) with deterministic software to support the iterative system dynamics process, from system description and formal modeling to simulation, testing, and communication. Building on the iterative approach described by Forrester (1994), it links natural-language interaction to executable models, diagnostic feedback, and model narratives. LLMs can assist with these tasks, but their stochastic outputs may contain structural omissions, equation errors, or inconsistencies between descriptions and implementations. ChatXmile addresses this uncertainty through repeated technical checks and modeler-guided revision.

We investigate to what extent this workflow can help LLMs produce XMILE models that pass deterministic conformance and technical checks, support reconstruction of equivalent models across sessions, and generate artifacts for verification and dissemination. The contribution is an auditable loop connecting narratives, formal models, and verification activities. Six use cases demonstrate its practical capabilities and limitations.

Workflow and verification

The workflow uses XMILE, a standardized interchange format for system dynamics models (Eberlein and Chichakly, 2013; OASIS, 2015). Modelers interact directly with an LLM through a standard browser-based chat interface, without requiring API integration. They supply a system description, potentially including text and figures, together with E3_ple.xsd, a deliberately simplified and more restrictive schema than the full XMILE standard. Alternatively, an existing XMILE or Vensim model provides the starting point.

The LLM generates or revises an XMILE model, which the modeler submits to ChatXmile. Deterministic software checks schema conformance, applies XSLT-based structural and equation rules, checks dimensional consistency, renders a stock-and-flow diagram, and performs an executable trial run through PySD. Compatible exports support additional testing in Vensim and Stella. The modeler feeds diagnostics and revision requests back to the LLM until the identified issues have been addressed.

A second loop develops model narratives and offline HTML/JavaScript simulators. Interactive controls, time-series plots, Monte Carlo scatterplots, lookup editors, and scenario comparisons support sensitivity analysis, extreme-condition tests, and time-step adequacy checks. Problems detected during exploration can trigger revisions to the simulator or the underlying model. Narratives document model purpose, structure, equations, and usage, support comparison with source descriptions, and enable reconstruction in another session or LLM environment. Here, equivalence requires identical variable sets and dependency structures, together with matching simulation outputs within numerical tolerance under identical time horizons, time steps, and integration methods.

Demonstration findings

We report representative dialogue trajectories from six actual use cases. Because LLM outputs vary between runs, these are qualitative demonstrations rather than a controlled benchmark of language models or workflow efficiency.

The electric-kettle example exposed a behavioral requirement that initial technical checks had not detected: the model did not represent the intended behavior after water reached 100°C. Simulator exploration prompted a return to model revision. A detailed narrative subsequently enabled equivalent reconstruction in a separate conversation. In the fishery case, 17 simulator-development iterations produced an interface with a lookup editor and Monte Carlo optimization of lookup breakpoints. Further dialogue generated a model narrative and critique of boundary choices; another LLM then reconstructed an equivalent model from the narrative.

The Island predator-prey example demonstrated numerical testing. Monte Carlo scatterplots revealed anomalous outcomes associated with parameter settings that drove the predator population to zero. Interactive exploration identified an excessively large time step as the cause; reducing it removed the anomaly. The Inclusive macroeconomic case demonstrated the importance of complete documentation. Initial reconstruction preserved the overall structure but changed several lookup functions. Explicitly documenting and correcting the lookup data was necessary to recover an equivalent model.

The evolving Ecotax lev green-economy model required 12 simulator-development iterations, including corrections to computational discrepancies identified by comparing simulator results with Vensim. The resulting interface supported sensitivity analysis, contour views, and multi-run comparisons. Narrative-based reconstruction succeeded in a subsequent workflow, but several recorded runs failed to produce a meaningfully similar model. Finally, a model derived from a news report on the gender pay gap passed the reported technical checks after two dialogue rounds. Sentence-by-sentence links between the article and model elements made the interpretation traceable. These correspondences supported conceptual inspection without establishing empirical validity.

Implications and limitations

Together, the cases illustrate three complementary forms of support: deterministic technical verification, simulator-enabled behavioral testing, and narrative-based consistency checking. Their integration helps expose errors that remain invisible when any one artifact is examined alone. Offline simulators and structured documentation also support stakeholder discussion, teaching, and continued model development across sessions. Modelers can re-enter the workflow with revised narratives, models, or requirements.

Passing schema, rule, units, and execution checks does not constitute full model validation. Confidence in a model's structure and behavior requires broader assessment appropriate to its purpose (Grösser and Schwaninger, 2012). Generated simulators and narratives also require scrutiny: technically acceptable XMILE can be translated incorrectly, and incomplete narratives can prevent faithful reconstruction. Performance depends on the underlying LLM, model complexity, and modeler intervention; diagram layout remains another limitation. The reported cases do not establish general success rates or comparative productivity gains. Further work should provide more systematic evaluation, improve narrative completeness and automated correction, and extend support for complex and modular models. ChatXmile therefore offers practical support for supervised LLM-assisted modeling while retaining human responsibility for model assumptions, validation, and interpretation.

For the full text and complete reference list, see the authors' article of the same title, in *System Dynamics Review*, Volume 42, Issue 4, e70031, **DOI:** 10.1002/sdr.70031.