# A Lightweight Approach to the Concurrent Use and Integration of SysML and Formal Methods in Systems Design

Robert Thorburn, Asieh Salehi Fathabadi, Leonardo Aniello, Michael Butler, Dana Dghaym, Thai Son Hoang, Vladimiro Sassone
University of Southampton, UK {robert.thorburn, a.salehi-fathabadi, l.aniello, m.j.butler, d.dghaym, t.s.hoang, vsassone}@soton.ac.uk}

## THE STANDARD APPROACH

- Increased systems complexity and the rise of ubiquitous computing make the case for model-based systems engineering.
- SysML provides a means to capture requirements, design systems, and attend to lifecycle management but not to address correctness-by-construction.
- Some systems and subsystems require formal model checking.
- SysML models can be translated to formal models via triple graph grammars or other means.[1]

## THE CHALLENGE

- Larger systems not requiring formal proof for their operations may contain subsystems which do.
- Full model translation is an expensive and time-consuming undertaking.
- Model translation is often only conducted on full SysML models as a "final" step, which is incongruent with a lifecycle-based approach.
- One way translation does not allow for synchronisation between models and accordingly decreases the value of both.[2]

## THE SOLUTION

- We propose a new system for linking SysML and formal models.
- Our system:
  1. Is presented as a SysML model library, making it reusable across projects.
  2. Is a lightweight alternative to full translation.
  3. Allows for iterative interaction between models over the project lifecycle.
  4. Allows for any appropriate formal modeling method to be selected.
  5. Works with existing SysML tooling.

## CASE STUDY

The new requirement interchange system (RIS) presented here is demonstrated by way of a case study. This case study is currently in active development and consists of an ARM Morello board with CheriBSD as operating system, which forms the heart of a larger system. This larger system is a smart ballot box which contains a number of off-the-shelf components, structural components, and various interfaces. Microsoft STRIDE is used for threat modelling, Event-B for formal modelling and SysML for overall systems modelling. All system requirements, including those derived from threat modelling, are captured in the SysML model and passed to the formal model if appropriate.[3]

## INTERCHANGE SEQUENCE: See Figure 1

1. Determine the formal modelling method to use and document this decision.
2. Derive requirements to pass to the formal model.
3. Develop rationale for feedback from the formal model, including how to act on said feedback.
4. Ensure synchronisation between models, including passing new requirements and returning to step two.
5. Construct or update the formal model.
6. Assure that the formal model is fit for purpose.
7. Conduct model checking assessing not only newly passed requirements but also the model as a whole.
8. Determine if source code should be generated.
9. Pass model checking results and source code (if generated) back to the SysML model.
10. Implement revised requirements and/or source code.
11. Check formal model returns against the expected return.

## MODEL INTERCHANGE REQUIREMENTS: See Figure 2

1. MIR01: The SysML model shall include documentation describing and justifying the formal modelling approach.
2. MIR02: The SysML model shall include a dedicated package for collecting, deriving and managing all requirements to be passed to a formal model.
3. MIR03: The SysML model shall, by way of either a <<Rationale>> note or preferably a linked document, described the expected revised requirements and source code returned from the formal model and how this is to be implemented into the SysML model.
4. MIR04: Any relevant model changes shall be reflected in the MIR02 requirements, reinitiating the formal modelling process and updating the MIR03 process.
5. MIR05: The return from the formal model shall be checked against the expected results of MIR03, with any variance triggering MIR04.
6. MIR06: Updating the formal model shall proceed once requirements are passed to it via MIR04 and shall cease once revised requirements or source code is returned.
7. MIR07: The formal model shall be assessed for consistency with the SysML model with any variance addressed immediately and thereafter triggering MIR04.

1. Lars Fritsche, Jens Kosiol, Adrian Möller, Andy Schürr, and Gabriele Taentzer. 2020. A Precedence-driven Approach for Concurrent Model Synchronization Scenarios Using Triple Graph Grammars. In Proceedings of the 13th ACM SIGPLAN International Conference on Software Language Engineering. 39–55.
2. Holger Giese, Stephan Hildebrandt, and Stefan Neumann. 2010. Model Synchronization at Work: Keeping SysML and AUTOSAR Models Consistent. In Graph transformations and model-driven engineering. Springer, 555–57.
3. For further information please see
   - Morello: https://www.arm.com/architecture/cpu/morello
   - CheriBSD: https://www.cheribsd.org/
   - STRIDE: https://docs.microsoft.com/en-us/previous-versions/commerce-server/ee823878(v=cs.20)

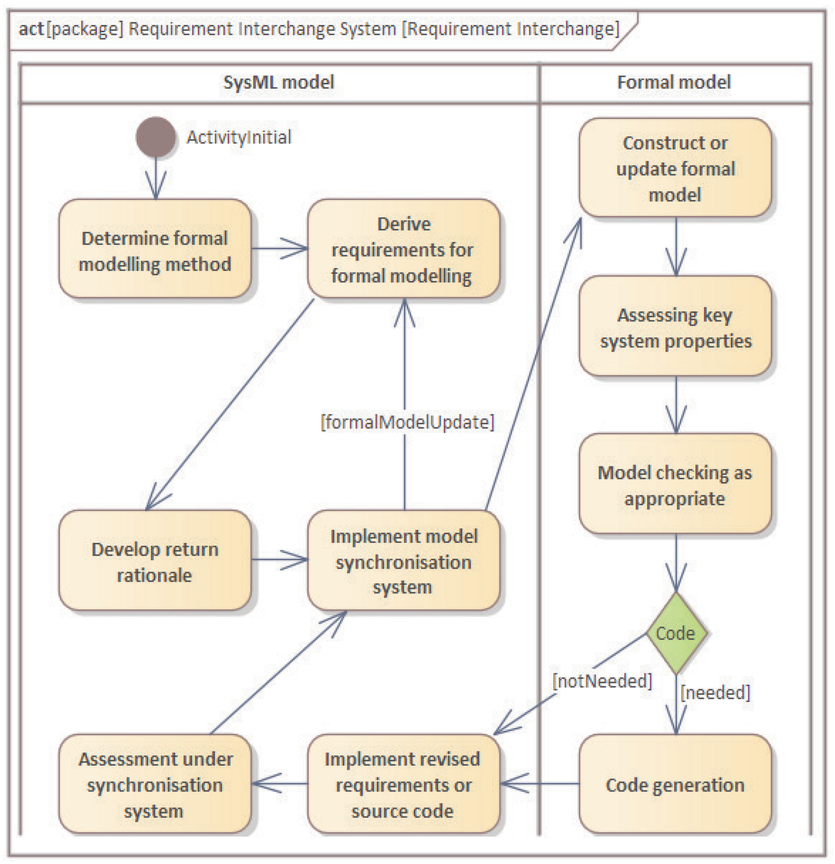


**Figure 1:** Activity diagram for the RIS.

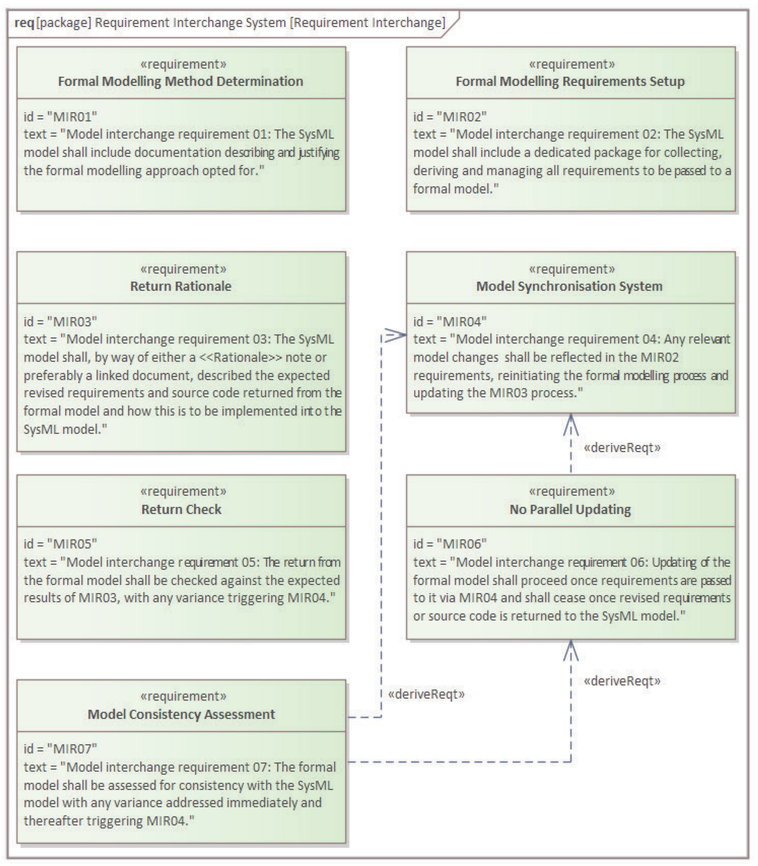


**Figure 2:** The Requirement Interchange System.

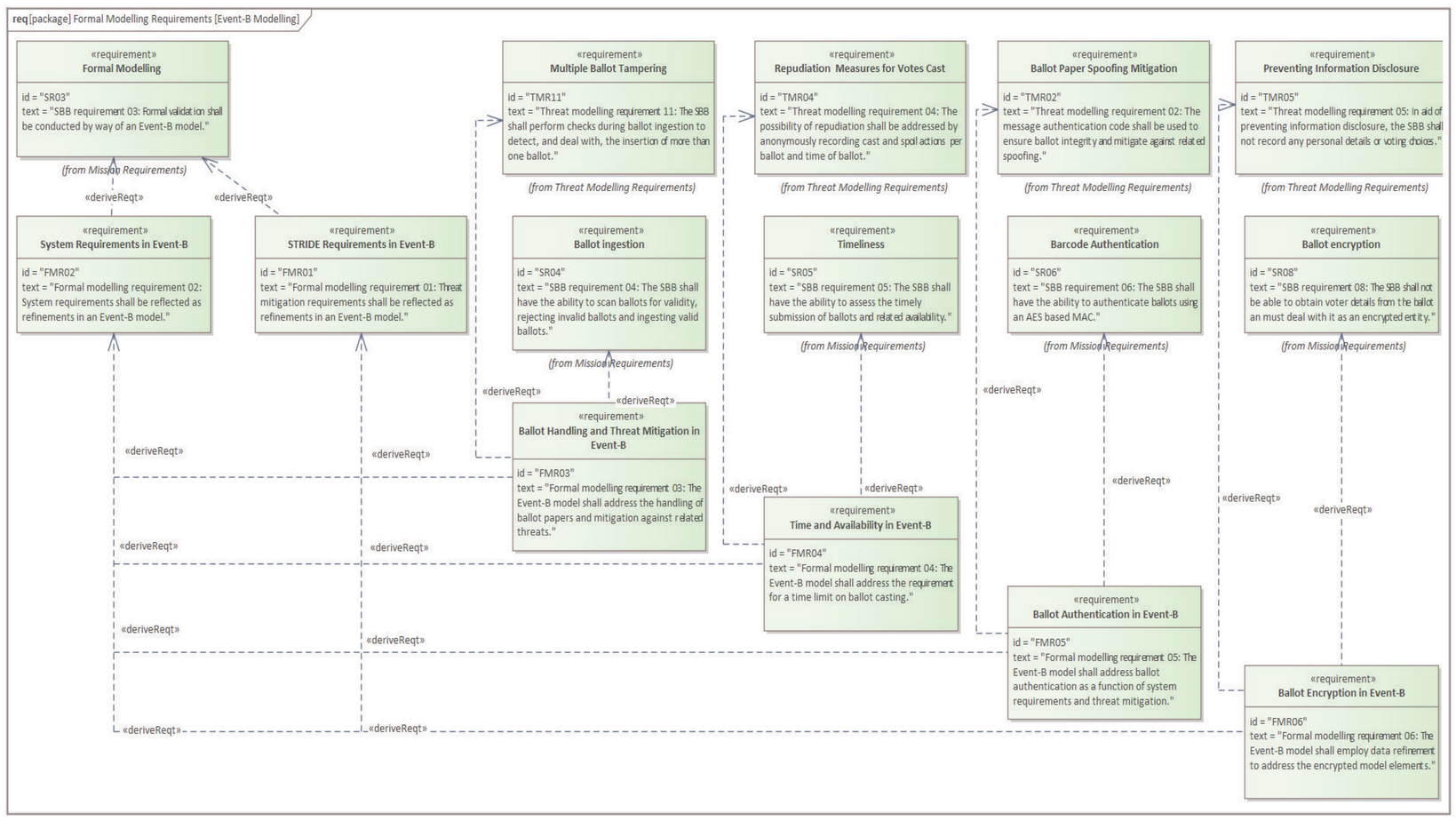


**Figure 3:** Deriving requirements to pass to the formal model.

**Notes**

The Requirements Interchange System has two primary components: an activity sequence (Fig 1) and the top-level requirements that must be met (Fig 2).
Applying these to the case study yielded a set of primary requirements for formal modelling (Fig 3).
Figure 3 shows the process of deriving the first set of requirements to pass to the formal model from the SysML model. On Figure 1, this would be at step 2.
The activity diagram, RIS requirements package, and a document capture package are included in the current version of the model library.
The latest version of the model library is available at: https://hd-sec.github.io/publications/

DSbD
Digital Security by Design
UKRI Engineering and Physical Sciences Research Council
UNIVERSITY OF Southampton