A Framework for Managing Conceptual Traceability in Simulation Experiments

Rakshit Mittal[1], Hans Vangheluwe[1], and Dominique Blouin[2]

Abstract: Repeatability and replicability are foundational principles of the scientific method, yet they remain difficult to achieve for contemporary simulation-based experiments. Existing solutions provide limited support for capturing precise semantics, design rationale, and conceptual provenance of experimental artefacts. This paper presents an ontological framework for managing conceptual and intentional traceability in simulation experiments. The framework integrates experimental concepts with various standard ontologies to provide precise, queryable semantics across experiment artefacts. It supports recording experiments at multiple levels of abstraction, linking models and executions to their underlying intent and design decisions. A proof-of-concept implementation demonstrates how the framework enables structured experiment documentation, knowledge-graph–based exploration, and export of shareable artefacts with varying levels of traceability.

Keywords: Experiment Modelling, Traceability, Replicability, Repeatability

1 Introduction

Repeatability and replicability of experiments are central notions of the scientific method [Po05]. As more and more scientific data is produced and made publicly available, experiment replicability seems to be increasingly elusive [Ba16]. Even well-established modelling and simulation tools do not offer sufficient support for replicability [BVK22]. We argue that state-of-the-art solutions such as Laboratory Information Management Systems (LIMS)[3], Electronic Lab Notebooks (ELN)[4] and Data Management Plans (DMP) [Mi19] are insufficient to achieve the FAIR (Findable, Accessible, Interoperable, Reusable) goals [Wi16]. This, as these tools have significant gaps that turn recording information needed for experiment replicability into cumbersome administrative tasks, rather than addressing the needs and supporting the workflows of scientists and engineers. The main causes are:

1. Lack of precise semantics: The state-of-the-art typically consists of natural-language documents with a few extra capabilities such as versioning, comments, persistence and accessibility, by providing unique identifiers and guaranteeing resolution of artefacts. However, natural language descriptions lack *conceptual traceability* - the ability

[1] University of Antwerp - Flanders Make, Antwerp, Belgium,
rakshit.mittal@uantwerpen.be, https://orcid.org/0000-0001-9871-800X;
hans.vangheluwe.uantwerpen.be, https://orcid.org/0000-0003-2079-6643

[2] Telecom Paris, Institut Polytechnique de Paris, Paris, France,
dominique.blouin@telecom-paris.fr, https://orcid.org/0000-0001-7606-0251

[3] Dassault Systemes' BIOVIA: https://www.3ds.com/products/biovia

[4] Protocols.io: https://protocols.io, Research Space: https://researchspace.com

to trace the meaning of concepts mentioned in the descriptions. This conceptual imprecision negatively impacts Interoperability and Reusability.

2. Lack of formal syntax (also known as machine readability): Natural language is not unambiguously machine-readable. Current approaches do enforce some structure but that is often limited to simple document metadata. Without a more structured syntax it is difficult to scale traceability management of experimental artefacts. This is a missed opportunity for mining data for emergent knowledge about (or derived from) an experimental corpus [Ce25]. This negatively impacts Findability, Interoperability, and Reusability.

3. Lack of experiment design rationale: The state-of-the-art has not yet focused on explicitly capturing the rationale and argumentation behind designing and performing an experiment. Understanding *why* an experiment was performed and why it was designed and carried out in a certain way is critical for meaningful (re-)use of experiments and their data. This, as recorded data alone cannot provide a complete picture [Zs25]. This negatively impacts Interoperability and Reusability.

Today's engineered systems rely heavily on modelling and simulation. Over the past few decades, significant advances have been made in multi-formalism, multi-abstraction, and multi-view modelling. However, these developments have not been matched by comparable progress in the foundational methods for managing all artefacts (and their links) associated with (simulation) experiments.

In this paper, we present a framework to manage the diverse relations between experiment artefacts and support conceptual traceability for systems engineers. This framework takes the form of an (extensible) ontology of experimental concepts linked with ontologies of

1. cyber-physical systems (CPS) [Te21] for a rich vocabulary of properties of systems,
2. multi-paradigm modelling (MPM) [Gi21] to enable model management, and
3. standards such as the International Vocabulary of Metrology (VIM) [07] and ISO:80000 [22] for precise conceptual definitions of measurements and quantities.

We present a proof-of-concept for our framework as well as a simple use-case in the electronics domain. The proof-of-concept exposes APIs and services to (i) enable conceptual traceability by allowing to graphically explore relations and descriptions of concepts expressed in the ontology (ii) allow recording experiments (and their design process) at varying levels of abstraction in a structured way and mapping it to a knowledge graph, and (iii) enable automatically exporting and archiving (on Zenodo[5]) share-able artefacts for recorded experiments with varying levels of traceability information.

In the next section, we discuss the related work and their gaps that we address. Following that we describe our framework and its proof-of-concept implementation. Finally, we discuss future work and conclusions.

[5] https://zenodo.org

2 Related Work

Experimental Frames [ZMK19] are the foremost conceptual methodology used by the simulation community; described as a *specification of the conditions under which the system is observed or experimented with*. An experimental frame is a combination of three components: *generator*, that generates input segments to the system; *acceptor*, that monitors an experiment to see the desired experimental conditions are met; and *transducer* that observes and analyzes the system output segments.

SESSL (Simulation Experiment Specification via a Scala Layer) [EU14] is a domain-specific language providing an abstraction layer in between users and simulation systems. It provides a declarative configuration of common experiment and simulator configuration parameters such as the input model, stop conditions, replication algorithms, and generation of results from observations.

ESS (EMF-based Simulation Specification) [MD22] is a domain-specific modelling language with a high-level, abstract, extensible meta-model of the artefacts associated with a validation experiment. It defines a workflow starting from the definition of the model instance and the scenario instance, to designing and executing experiments, and analysing the result.

SPDM (Simulation Process and Data Management) [SM15] is the set of activities undertaken to integrate and manage modelling and simulation data from various disciplines. Most commercial products such as ANSYS Minerva[6] and Hexagon SimManager[7] are an integrated suite of domain-specific finite-element simulation engines, allowing to define common workflows of data analysis, eliminating the need to manage data from different platforms.

The Model Identity Card (MIC) [20] was developed by a consortium of French and German automotive companies to facilitate quality assessment of simulation models by supporting collaboration in a gated process. It is basically a table of simulation models' essential properties. Some of the stakeholders have further developed the MIC in-house to better integrate it within their model-based simulation workflows [Ha24].

The state-of-the-art, as described above supports recording some important properties of simulation experiments in specific domains. However, there is no single general framework to model not just computer-based simulations but also real-world experiments. Approaches such as the MIC (where properties are expressed as string values), lack any formal semantics, making it very difficult to interpret the metadata. There are no provenance links between the metadata values and the experiments performed to obtain them, which is important for validity assessment [Mi23]. Approaches such as SESSL, ESS, and Experimental Frames provide reusable means to specify simulations. They do not account for managing accessibility and conceptual provenance of artefacts after an experiment was performed.

6 https://www.ansys.com/products/connect/ansys-minerva
7 https://hexagon.com/products/simmanager

3 Methodology

3.1 Conceptual Traceability in Experiments

Conceptual traceability is provided in our framework by explicit, queryable, ontological relations between the experimental concepts - down to the definition of units and measurements (provided by standards such as VIM and ISO:80000). An ontology enables 'going deeper into' the conceptual domain to answer questions such as *'what does this concept mean?'* as the meaning of entities is not just described by natural language annotations, but also given by relations with other concepts.

Description Logic (DL) semantics on the relations are based on the Ontological Modelling Language (OML) [El23], which implements an engineer-friendly syntax with patterns over a subset of the Web Ontology Language (OWL2-DL). Using DL reasoners such as Openllet[8], logical consistency of the knowledge graph can be checked and is ensured. We will describe a few important concepts and describe how they are connected to other silos of knowledge such as standards, to enable 'going deeper into' the meaning of concepts.

Following the Baconian view of experimentation as *an interrogation of nature* [BD02], we define an *experiment* as *a deliberate and intentional activity carried out by a specific agent (human or automated), on a specific instrumented system-under-study, in a specific instrumented environment, where the goals are made explicit, by measuring properties of interest*.

Experiments are organized in *phases* within an *experimental study* - *a goal-oriented aggregation of experiments*, a concept generalized from the 'simulation study' described in [Wo25]. An experiment plays a *role* in a phase of an experimental study. Typical roles played by experiments are what-if analysis, uncertainty quantification, data calibration, cross validation, verification, etc. in phases such as 'exploration', 'confirmation', 'answering research question', 'presentation'.

An *experiment specification* (ExSpec) is *a reusable, machine-readable artefact encapsulating all information relevant to an experiment*. This includes not only the final deployed artefacts and results, but also (model-based) documentation about the motivation, design rationale, and design workflows. We see this concept as complementary to related works such as SESSL and ESS, as their instances are some of the possible design models in an experiment specification. Each experiment has exactly one specification. The models contained in the experiment specification are linked to modelling concepts in the MPM ontology to provide precise semantics.

The *experiment intent* is part of the ExSpec, and it is an artefact describing the intent of an experiment such as the intended actor/s, time-base, property-of-interest, system-under-study and environment of the experiment. Note that this is complementary to the notion of intent

[8] https://github.com/Galigator/openllet

described by [Zs25]. The systems described in the intent are typically CPSs, which is how they are linked to the CPS ontology [Te21].

A *High-Level Interface* (HLI) is a conceptual abstraction representing *a possible means of measuring or stimulating (applying a controlled influence that changes the value or evolution of) a property associated with a system*. When realized, these HLIs may 'transcend' the normative system boundaries to connect (measuring or stimulating) instruments to the system. We designate this composition of the system and HLIs as a first-class entity, the *instrumented system*, as we find it important to explicitly model the measurement-induced disturbance and its implications for experimental interpretation and validity. An HLI in a real-world experiment could be realized as an instruction to a human actor, or as the reading or writing of data to/from specific ports of a computational model in a simulation experiment. HLIs measure and/or stimulate quantities associated with systems, where it relates to the concepts such as quantities, measurements, and measuring instruments in the VIM and ISO:80000 standard.

3.2 Experiment Lifecycle

In our view, the life-cycle of each experiment has six phases. Since experiments are activities, there are associated *workflows* in each stage; and also, since experiments are a composition of multiple systems, there are associated *architectures*. The workflow and architecture are usually co-designed in each stage. Artefacts from subsequent phases require more domain knowledge as they are refined based on rules (either formally specified for automated/assisted experiment design, or informally in the engineer's head).

Intent Elicitation is the first stage where the high-level intent of performing the experiment is articulated. The requester needs to identify the goals of the experiment, which includes the role, property of interest, system-under-study, high-level methodology etc. The resulting artefact, based on the choice of user-interface, could be answers recorded in a form (to be usable for non-experts), the conversation history with experiment design assistants, or in its least machine-readable form – just natural language annotations.

In the *Conceptual Design* phase, the intent specification is used to create the conceptual design models of the architecture/s and workflow/s of the experiment. These models abstract from domain-specific constraints and provide a common architecture and workflow for eventual real- or virtual-world deployment. This also serves to provide a stronger morphism between the corresponding experiments carried out in a validation experiment (meta-validity). The conceptual models can vary from pen-paper drawings to instances of a precise modelling language.

The *Refined Design* phase refines the conceptual model by taking into account domain-specific constraints, but not yet deployment constraints such as infrastructure and platform, resource scheduling, etc. In a virtual (simulation) experiment this would typically be a completely architected and instrumented model such as System Structure and Parametrization (SSP), SESSL, Finite-Element Model (FEM) with all necessary parameter values, and a domain-specific workflow for executing such a model. In the real-world these would typically be schematics with work plans. When an experiment is *replicated* the refined design models are necessarily identical.

The *Deployment Specification* phase accounts for the realization/deployment constraints, and produces a complete set of resource-mapped artefacts with a schedul-able workflow. When an experiment is *repeated* the deployment specification necessarily remains the same. The deployment specification may differ in case of replication.

Traces are the result of *Performing an Experiment* with a certain specification. They may include time-stamped records of measurements, events, state changes, logs, intermediate results, etc. Traces are deployment- and execution-specific: repeated executions of the same deployment specification may yield distinct traces as experiments may sample from a possibly stochastic process. Traces constitute the primary empirical evidence of an experiment.

In the *Reporting and Archival* phase, share-able and re-usable artefacts (including reports) are generated and archived. The experiment report is a curated and interpreted artefact derived from one or more traces. It aggregates, processes, and contextualizes trace data to answer the stated experimental intent, including the property-of interest, which may be derived quantities, statistical summaries, visualizations, etc. Reports are audience-driven; multiple reports may be generated from a single experiment.

In our framework, the ability to traverse backward through the above phases, from the report ultimately back to the goals and intentions, to answer questions such as *'why was an experiment performed (in a certain way)?'*, is a core requirement for experiment management tools. It makes the rationale behind the design of an experiment inspect-able, supporting Popperian falsifiability [Po05].

4 Proof of Concept

We encoded the ontological concepts and relations for experiments, MPM, and CPS in OML, as described earlier. The ontology also extends the classical Dublin Core metadata annotations for ontological entities, to include rich annotations such as $\LaTeX$, formulae, images, concept diagrams, and internal links (such as Wikipedia wikilinks) in textual descriptions and comments. The consistency of the resulting knowledge graph was verified using the Openllet DL reasoner. The knowledge graph was then hosted on an Apache Jena Fuseki [9] instance.

[9] https://jena.apache.org/

A backend service (coded in Rust) exposes Application Programming Interfaces (API) to manage the knowledge base, such as creating and modifying experiments and studies, creating systems, organizing the artifacts uploaded to the knowledge base, running the Openllet reasoner to ensure consistency, and synchronizing the knowledge graph in Fuseki with updated data.

A frontend service based on the React framework provides a Graphical User Interface (GUI) to facilitate user interaction with the knowledge base. The interface enables users to describe experiments in a structured manner and associate artifacts with each phase of the experiment lifecycle while ensuring consistency with existing data. It also allows users to graphically navigate the knowledge graph.

Once an experiment is recorded, the user can export a ZIP archive containing all relevant artifacts, along with an HTML (or PDF) document summarizing the ontological entities and relations associated with the experiment. The contents of this summary are obtained through traversal of the knowledge graph to a depth configurable by the user. Based on initial tests, the maximum traversal depth required for typical experiments appears to range between 12 and 14 iterations. The tool also supports directly uploading the contents of the ZIP archive to a Zenodo repository.

For the proof-of-concept, we recorded a simulation experiment involving a Modelica model of a notch filter. The conceptual architecture of the experiment was represented as a diagram, while the refined architecture was implemented as a Modelica model. The compiled Modelica code served as the deployment artifact. The conceptual workflow was specified using the FTG+PM language [PEV22], whereas the deployment workflow was implemented in a Jupyter notebook. Finally, the experiment was exported multiple times using different knowledge graph traversal depths to evaluate the completeness of the generated summaries.

The source code of the proof-of-concept, a demonstrative video, and exported artifacts for various experiments are available on Zenodo with the DOI: 10.5281/zenodo.18162277.

5 Discussion and Conclusion

We presented a high-level overview of our experiment management framework and demonstrated it with a proof-of-concept implementation, applied on the simple use-case of performing a validation experiment on a Modelica model of a notch-filter. There are open questions such as (i) the scalability of Semantic Web technologies, (ii) large-scale integration of domain-specific knowledge, (iii) versioning and co-evolution of knowledge and artefacts, and (iv) ease of use for non-experts. We plan to evaluate our framework with more complex case studies. We would also like to evaluate the User Experience (UX) provided by our proof-of-concept. We believe the proposed framework will be of interest to the NFDIxCS community as it addresses the computer-based management of experimental research artifacts with conceptual traceability.

References

[07] International vocabulary of metrology — Basic and general concepts and associated terms (VIM), tech. rep. ISO/IEC Guide 99:2007, ISO, 2007.

[20] Model Identity Card (MIC): Towards a standardization of the specification and description of simulation models, tech. rep., IRT SystemX, 2020, https://mic.irt-systemx.fr.

[22] Quantities and units - Part 1: General, tech. rep. ISO 80000-1, ISO, 2022.

[Ba16] Baker, M.: 1, 500 scientists lift the lid on reproducibility. Nature 533 (7604), pp. 452–454, 2016, http://dx.doi.org/10.1038/533452a.

[BD02] Bacon, F.; Devey, J.: Novum Organum. P. F. Collier, 1902.

[BVK22] Boll, A.; Vieregg, N.; Kehrer, T.: Replicability of experimental tool evaluations in model-based software and systems engineering with MATLAB/Simulink. Innovations in Systems and Software Engineering 20 (3), pp. 209–224, 2022.

[Ce25] Cederbladh, J. et al.: Reasonable Experiments in Model-Based Systems Engineering, ArXiv, 2025, https://arxiv.org/abs/2509.10649.

[El23] Elaasar, M. et al.: openCAESAR: Balancing Agility and Rigor in Model-Based Systems Engineering. In: MoDELS Companion Proceedings. 2023.

[EU14] Ewald, R.; Uhrmacher, A. M.: SESSL: A domain-specific language for simulation experiments. ACM Trans. Model. Comput. Simul. 24 (2), 2014.

[Gi21] Giese, H. et al.: An ontology for multi-paradigm modelling. In: Multi-Paradigm Modelling Approaches for Cyber-Physical Systems. Elsevier, pp. 67–122, 2021.

[Ha24] Hallak, Y. et al.: Model Management at Renault Virtual Simulation Team: State of Practice, Challenges and Research Directions. In: MoDELS Companion. 2024.

[MD22] Mertens, J.; Denil, J.: ESS: EMF-Based Simulation Specification, A Domain-Specific Language For Model Validation Experiments. In: Proceedings of ANNSIM. 2022.

[Mi19] Miksa, T. et al.: Ten principles for machine-actionable data management plans. PLOS Computational Biology 15 (3), ed. by Ouellette, F., e1006750, 2019.

[Mi23] Mittal, R. et al.: Towards an Ontological Framework for Validity Frames. In: MoDELS Companion. Pp. 801–805, 2023, https://doi.org/10.1109/MODELS-C59198.2023.00128.

[PEV22] Paredis, R.; Exelmans, J.; Vangheluwe, H.: Multi-Paradigm Modelling For Model Based Systems Engineering: Extending The FTG+PM. In: 2022 ANNSIM. 2022.

[Po05] Popper, K.: The Logic of Scientific Discovery. Routledge, 2005.

[SM15] Sibois, R.; Muhammad, A.: Simulation Lifecycle and Data Management, tech. rep. VTT-R-02486-15, VTT Technical Research Centre of Finland, 2015.

[Te21] Tekinerdogan, B. et al.: A feature-based ontology for cyber-physical systems. In: Multi-Paradigm Modelling Approaches for Cyber-Physical Systems. Elsevier, pp. 45–65, 2021.

[Wi16] Wilkinson, M. D. et al.: The FAIR Guiding Principles for scientific data management and stewardship. Scientific Data 3 (1), 2016, http://dx.doi.org/10.1038/sdata.2016.18.

[Wo25] Wolpers, A. et al.: Goal-Oriented Generation of Simulation Experiments. In: Proceedings of the 2025 Winter Simulation Conference. 2025.

[ZMK19] Zeigler, B. P.; Muzy, A.; Kofman, E.: Framework for Modeling and Simulation. In: Theory of Modeling and Simulation. Elsevier, pp. 27–41, 2019.

[Zs25] Zschaler, S. et al.: Dialectic Models for Documenting and Conducting Simulation Studies: Exploring Feasibility. In: Proceedings of the 2025 Winter Simulation Conference. 2025.