Automatic Reuse, Adaption, and Execution of Simulation Experiments via Provenance Patterns

PIA WILSDORF and ANJA WOLPERS, University of Rostock, Germany
JASON HILTON, University of Southampton, UK
FIETE HAACK and ADELINDE M. UHRMACHER, University of Rostock, Germany

Simulation experiments are typically conducted repeatedly during the model development process, for example, to re-validate if a behavioral property still holds after several model changes. Approaches for automatically reusing and generating simulation experiments can support modelers in conducting simulation studies in a more systematic and effective manner. They rely on explicit experiment specifications and, so far, on user interaction for initiating the reuse. Thereby, they are constrained to support the reuse of simulation experiments in a specific setting. Our approach now goes one step further by automatically identifying and adapting the experiments to be reused for a variety of scenarios. To achieve this, we exploit provenance graphs of simulation studies, which provide valuable information about the previous modeling and experimenting activities, and contain meta-information about the different entities that were used or produced during the simulation study. We define provenance patterns and associate them with a semantics, which allows us to interpret the different activities, and construct transformation rules for provenance graphs. Our approach is implemented in a Reuse and Adapt framework for Simulation Experiments (RASE) which can interface with various modeling and simulation tools. In the case studies, we demonstrate the utility of our framework for a) the repeated sensitivity analysis of an agent-based model of migration routes, and b) the cross-validation of two models of a cell signaling pathway.

CCS Concepts: • **General and reference → Experimentation**; • **Computing methodologies → Simulation environments**; **Model verification and validation**.

Additional Key Words and Phrases: simulation experiment, simulation study, reuse, provenance, graph patterns

1 INTRODUCTION

Simulation experiments reveal important information about the behavior of a model. Therefore, a wide variety of simulation experiments are conducted during a simulation study [5, 61]. Automatically generating and executing simulation experiments allows simulation studies to be conducted in an easier and more systematic manner. One option presents the goal-directed reuse of simulation experiments. In [51], statistical model checking experiments [1] were reused to check whether the composition of simulation models still exhibits certain behavioral properties. The tested properties can be interpreted as requirements, specifying the expected behavior of the simulation model [58], or as hypotheses to be tested in the development of a simulation model [39]. In the area of cardiac cellular electrophysiology, simulation experiments have been automatically reused to compare different model variants specified in CellML to assess their underlying hypotheses and their validity [13]. Other approaches focus on the reuse of a simulation experiment's results (outputs) for settings in which experiments

Authors' addresses: Pia Wilsdorf, pia.wilsdorf@uni-rostock.de; Anja Wolpers, anja.wolpers@uni-rostock.de, University of Rostock, Albert-Einstein-Str. 22, Rostock, Germany, 18059; Jason Hilton, J.D.Hilton@soton.ac.uk, University of Southampton, University Road, Southampton, UK, SO171BJ; Fiete Haack, fiete.haack@uni-rostock.de; Adelinde M. Uhrmacher, adelinde.uhrmacher@uni-rostock.de, University of Rostock, Albert-Einstein-Str. 22, Rostock, Germany, 18059.

1049-3301/2022/9-ART $15.00
https://doi.org/10.1145/3564928

are performed repeatedly with the same simulation model, e.g., with various parameter configurations, and aim to increase computational efficiency by avoiding the execution of simulation experiments [19].

In the above approaches, the type of simulation experiment, and/or kind of simulation model (including the modeling formalisms) have been constrained to support an automatic reuse of simulation experiments in a specific setting. However, various simulation experiments tend to be conducted repeatably with different variants of the simulation model during its development, and thus the repetition of simulation experiments forms a salient feature of the modeling and simulation life cycle.

To approach the question of how to support the automatic reuse of simulation experiments more generally while conducting simulation studies, necessary ingredients and accessible information sources need to be identified. A prerequisite for the reuse of simulation experiments is a clear separation of concerns between model, simulator, and simulation experiment. In addition, simulation experiments need to be explicitly specified to be accessible and reusable. Over the last two decades, various approaches have been developed that allow an explicit specification of simulation experiments. To those belong model-based approaches such as [66], domain-specific languages such as SESSL [17], or standardized formats such as SED-ML [72]. Only if simulation experiments are explicitly specified, they can be automatically interpreted. Their interpretation is facilitated by schemas [74] for the different types of experiments, possibly complemented by ontologies about their various roles [58] and designs [60].

Also the past contains valuable information that can be exploited in a variation of Santayana's phrase "Those who cannot remember the past are condemned to repeat it.": Those who can remember and interpret the past can effectively plan the steps ahead which might include, in the case of simulation studies, a deliberate repetition of steps. Provenance information about simulation models reveals crucial information about a simulation model's past in terms of how a model has been developed. This includes information sources as well as activities, such as the conduction of simulation experiments that contributed to its development [57]. Provenance information may be used to relate information sources, activities, and generated entities, within and beyond individual simulation models thus forming entire families of models [10]. Here, we will pursue the question of how to automatically detect new experiments to be reused based on what has been done before. The reuse of simulation experiments refers to the reuse of a simulation experiment specification, which is then adapted and executed.

As the central building block of our approach, we define typical patterns that can be observed in the provenance graph of simulation studies, and associate them with semantics. Based on the patterns we specify rules that automatically identify experiments to reuse, and then adapt, generate, and execute a new experiment. Updates of the provenance graph function as triggers to this process. The approach is implemented as the open-source Reuse and Adapt framework for Simulation Experiments (RASE).

We demonstrate the utility of our framework in two simulation studies, from demography and cell biology. We show that simulation experiments as well as other provenance entities can be effectively reused and exploited for automatically generating a simulation experiment.

The outline of this paper is as follows. First, in Section 2 we introduce the prerequisites for our approach, including provenance and means for explicitly specifying simulation experiments. Then, we go deeper into when and which simulation experiments are typically reused (Section 3). In Section 4, we present our reuse-and-adapt framework for simulation experiments. This is then followed by implementation details in Section 5. In Section 6, we apply our framework to two simulation studies from demography and cell biology, respectively: one aimed at developing a simulation model to study the impact of information flow on migration, the other aimed at revealing crucial mechanisms of a central signaling pathway. We finish the paper with related work in Section 7, and conclusions and future work in Section 8.

2 BACKGROUND

Our approach is based on two main ingredients: 1) the concept of provenance for simulation studies and 2) explicitly specified simulation experiments. This section provides important background knowledge on these topics.

2.1 Provenance of Simulation Studies

As stated by the W3C Provenance Working Group, provenance provides "information about entities, activities, and people involved in producing a piece of data or thing, which can be used to form assessments about its quality, reliability, or trustworthiness" [45]. To apply provenance to products of modeling and simulation studies requires identifying the central activities and products of modeling and simulation, and putting those into relation in a directed acyclic graph [57]. Based on an earlier specialization of the PROV Data Model (PROV-DM) [45] for telling the tale behind a simulation model or a simulation study [10], we identified the important entities to be the research questions (RQ), simulation models (SM), simulation experiments (SE), simulation data (SD), other data (D), e.g., from the wet-lab or surveys, requirements (R), qualitative model (QM), assumptions (A), theories (T), and other information (O). Entities contain meta-information about a product, e.g., the simulation experiment entities will include the experiment specifications either as text or as a reference to a file. Entities are related to activities by the dependencies *wasGeneratedBy* (activity ← entity) and *used* (entity ← activity). The activities will represent the typical activities of the modeling and simulation life cycle such as building a simulation model, its calibration, or validation [5, 61].

To record and view provenance information about simulation studies different possibilities exist. While conducting a simulation study, usually fine-grained provenance information about the various activities, information sources, and products is collected, e.g., in artifact-based workflows [58]. Recording the information within a graph database then enables zooming in and out of simulation studies and displaying provenance information on different levels of aggregation [2, 59]. These can be created from every lower-level provenance graph by aggregating smaller (similar) activities and then using only the most recent entities as inputs and outputs to the aggregated activity. For example, instead of representing multiple conceptual modeling and model building steps that consider the research question, qualitative model, and assumption one after the other, and producing various intermediate versions of the simulation model, the aggregated view would show simply a model building activity that takes as inputs the research question `RQ`, the qualitative model `QM`, and the requirement `R1`, and produces a simulation model (see activity a1 in Figure 1). More coarse-grained provenance information still reveals important information about the development process of the individual simulation model and its potential for reuse, or of entire sets of simulation models [10]. In the latter case, the provenance information elucidates a family of simulation models with their specific relations, partly shared data and information sources, and close ties realized by cross-validations. Coarse provenance is also often recorded manually after a simulation study has been conducted (ideally by those who conducted the simulation study) [10].

2.2 Explicit Simulation Experiments

The execution of simulation experiments plays a central role in the modeling and simulation life cycle. Treating simulation experiments as first-class objects and making their specifications explicit facilitates their generation, reuse, repetition as well as their reproducibility, which has been identified as one of the main challenges in computational sciences [29]. Not surprisingly, reporting guidelines for simulation studies increasingly demand information about simulation experiments, e.g., the "Strengthening the reporting of empirical simulation studies" (STRESS) in the field of operational research and management sciences [44], the "Overview, Design concepts and Details" (ODD) protocol for agent-based models in ecology [23], TRACE (TRAnsparent and Comprehensive Ecological modelling documentation) [22], which aims at documenting entire simulation studies, including

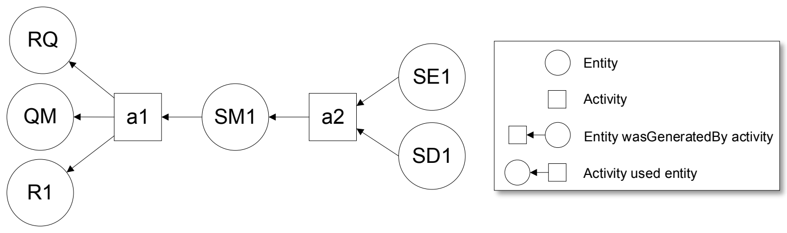


Fig. 1. Example of a provenance graph in the PROV-DM notation [6].

calibration, analysis, and validation experiments, or MIASE (Minimum Information About a Simulation Experiment) [71].

Another strand of research focuses on specifying simulation experiments unambiguously in a machine-accessible and executable manner. Numerous domain-specific languages and formats have been developed that all target different application domains. These include, e.g., the Simulation Experiment Description Markup Language for experiments in systems biology (SED-ML) [72], a description language for the network simulator ns-3 (NEDL) [38], or Xperimenter for the design of experiments (DoE) [16]. Furthermore, model-based approaches allow for the generation of factorial experiment design [66] and many other experiment types [74]. These approaches also allow one to conquer a myriad of tools based on a single, tool-agnostic specification format if software bindings are implemented [17, 74].

The development of the mentioned approaches for explicit experiment specification is also the product of increasing awareness and application of the FAIR principles (Findable, Accessible, Interoperable, Reusable) [73]. These do not only apply to simulation experiments but to all artifacts of the scientific process. Also, these principles do not only refer to how simulation experiments are specified, but also how they are made available. E.g., scripts can be shared via Jupyter notebooks [35], and the simulation models, experiments, and data can be bundled in archives such as COMBINE [7] or reproducible research objects such as SciUnits [69]. With regards to the Findable and Accessible principles, these bundles can be shared via open model databases, e.g., BioModels [41] and OpenABM [32].

3 TYPICAL REUSE SCENARIOS IN SIMULATION STUDIES

First, we want to describe some example scenarios to illustrate when our framework could be of assistance. These are reuse scenarios that occur in simulation studies where the development of a valid simulation model is the focus. In these kinds of studies, the reuse of simulation experiments is typically evoked by the creation of a new simulation model (version). Many scenarios refer to the reuse of simulation experiments within the same simulation study. However, simulation experiments may also be reused across studies, i.e., the reused entities are taken from another related simulation study, presuming that the provenance graphs of the two studies are connected.

3.1 Repeated Model Validation after Model Extension and Composition

Validation is an important task in the modeling and simulation life cycle. It is the process of substantiating whether the model behaves consistently with our expectations, e.g., regarding data that has been measured in the real system [54]. However, the model development is usually not completed after the first validation. Further model features are added, and the modeler moves again through the phases of the modeling and simulation life cycle. Making changes to the simulation model then requires re-validating it [58]. In software engineering, successive validation is known as regression testing: "Regression testing is performed between two different versions of software in order to provide confidence that the newly introduced features of the [system under test] do not interfere with the existing features" [76]. For the scientific community, also test-driven model validation procedures have been proposed [49].

A special case of this regression testing occurs in model composition. Frequently, models are not built from scratch but are created by composition of existing models. The composed models may have been developed as separate modules during the same simulation study or may originate from different, previously conducted simulations studies. Once two or more models have been merged, it should be evaluated if everything still works as expected [51]. Thus, validation experiments that were conducted with the individual models are typically repeated with the composed model. This works independently of the used tools and formats, and for true composition as well as model fusion [53]. In multi-disciplinary studies, the composition might also refer to a co-simulation [21], where after orchestration of the simulation units, requirement checks are repeated on the global level before the partial solutions are combined.

3.2 Repeated Model Calibration

Calibration, also called model fitting, is the process of finding a parameterization of the model which can reproduce observed behavior of the real system [54]. While calibration and validation both typically relate to real data, the conclusions drawn from them are essentially different. Whereas calibration refers to adjusting the input parameters such that the resulting agreement of the model output with a chosen set of experimental data is maximized, the goal of validation is to establish confidence in the model predictions [70]. Therefore, if previous calibration experiments exist in a simulation study, they should be repeated before the validation experiments. Consequently, when a new model version is produced, calibration experiments are repeated. Only if the calibration was successful, can the validation be attempted. If the model cannot be calibrated such that it reproduces the data with sufficient accuracy, further model revisions are necessary.

3.3 Repeated Model Analysis

In simulation studies, many experiments are conducted that do not serve as validation or calibration. Nevertheless, they still reveal important information about the model, e.g., via parameter scans, optimization, sensitivity analysis, perturbation analysis, or time course analysis [10]. It could be argued that all experiments contribute to the validation of the simulation model as they increase our trust that the right model has been built. In the context of this paper we assume validation experiments to be experiments distinguished as such by the modelers, and whose outcome can be evaluated as a *success* or *failure*, see also [58]. Especially sensitivity analysis is increasingly becoming an integral part of simulation studies as it allows quantifying how the parameters contribute to the uncertainty in the model output [15]. It is becoming good practice, to attach each model version with uncertainty and sensitivity information. Therefore, automatically reusing and repeating sensitivity analysis experiments after each major model version is instrumental in enhancing the quality of simulation models.

3.4 Cross-Validation with Related Simulation Studies

Cross-validation, or model alignment, is the process of comparing a simulation model with another, independently developed model [3]. In particular, models that deal with a similar research question should be able to reproduce each other's results. By comparing (and i.e., validating) simulation models with other, already calibrated and validated models, a "domain validity" can be achieved and overall confidence in the models can be established. To facilitate cross-validation, simulation experiments conducted with related, validated models should be reused and adapted in the validation activities of the current simulation study.

3.5 Comparison of Alternative Implementations

Usually, there is not just one way of modeling a system. From the same conceptual model different computerized models can be built. For example, the same model can be simulated using discrete event simulation (based on Continuous-time Markov Chains) or using System Dynamics (based on Ordinary Differential Equations ODEs), and with or without spatial features. Comparing alternative modeling and simulation approaches is crucial, e.g., to uncover discrepancies in simulation results, or to find a more efficient implementation in terms of simulation runtime.

The comparison is done by reusing and reapplying simulation experiments that were conducted with the other model implementation. Sometimes this comparison is done as part of the same simulation study, e.g., when switching to a different platform to improve performance. However, the re-implementation of a previous model could also be the primary goal of a simulation study to gain a better understanding and confidence in the results [25]. In this case, simulation experiments have to be reused across simulation studies.

3.6 Synchronization of Concurrently Developed Models

In large simulation studies, often numerous models, e.g., candidate models or submodels, are developed concurrently. This can be either the work of a single modeler or multiple modelers in a collaborative simulation study. In collaborative workflows, the sharing and reusing of data and other products between peers is crucial to keep the work on the different branches synchronized [43]. For instance, when one modeler completes a calibration in one branch, this could trigger new experiments for the other submodels based on the calibration results.

4 REUSE AND ADAPT FRAMEWORK FOR SIMULATION EXPERIMENTS

To automate the reuse of simulation experiments as in the scenarios described above, we develop a Reuse and Adapt framework for Simulation Experiments (RASE). The framework presents a provenance-based mechanism for reusing simulation experiments either within or across simulation studies. It exploits the observation that certain activities of the modeling and simulation life cycle produce characteristic patterns in the provenance graphs. Based on such patterns, the production rules of a graph transformation system can be constructed. In the following, we first introduce the architecture of the framework. Next, we define the provenance graph transformation system for producing new experiment activities based on patterns of the last modeling activity and previously conducted simulation experiments. Finally, we describe how to adapt simulation experiment specifications to the context information from the new simulation model, which is also given by provenance.

4.1 Framework Architecture

Figure 2 provides an overview of the framework's architecture. The entry point is the application programming interface (API) that allows submitting provenance from a variety of applications, ideally on the fly while the modeler works on the simulation study. The provenance recording applications may be stand-alone GUIs or workflow systems that capture provenance while the users run through a number of workflow stages. The recorded provenance is stored in a graph database. Each new activity added to that database initiates the evaluation of the provenance graph transformation rules. When a rule can be applied, and thus an experiment shall be reused for a new simulation model, the graph transformation system coordinates with the adaption and generation component. This component is required to transform the old experiment specification to the context of the new simulation model. The adaption is based on a model-driven engineering (MDE) approach that uses metamodels for automatically identifying and accessing the parts of experiment specifications that have to be adapted based on new provenance information. In some cases, however, it might be necessary to interact with the user during this step as they may carry relevant yet implicit knowledge about the simulation study. For example, via the API the modeler may be asked to specify a requirement in a formal manner or to review the

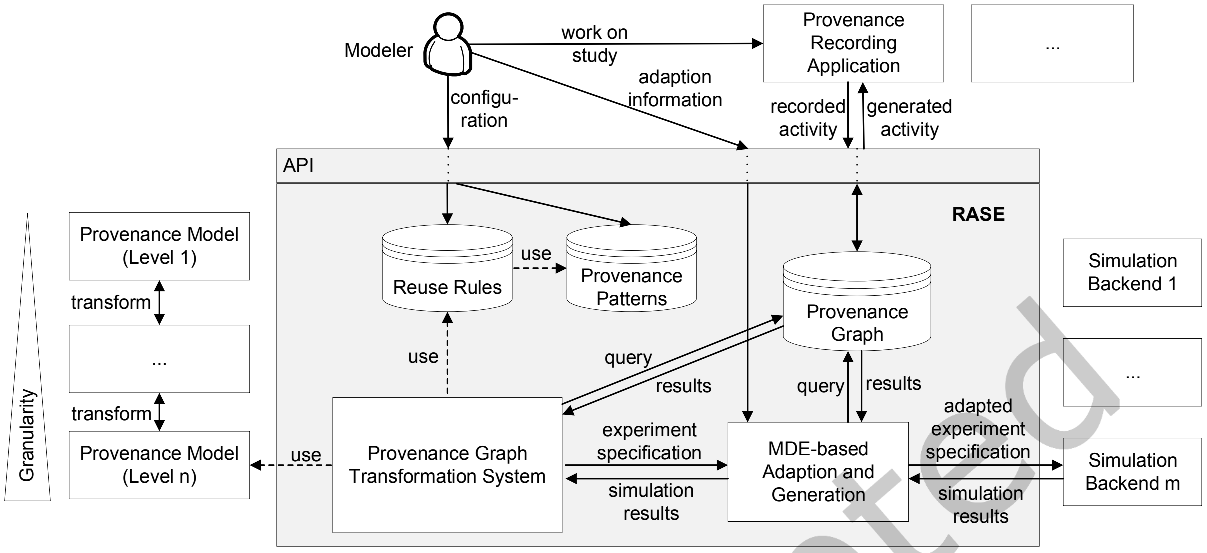


Fig. 2. Overview of the reuse and adapt framework for simulation experiments (RASE).

adapted experiment design. After the adaptations, the MDE component generates experiment code for a specific modeling and simulation tool called backend. The code can be automatically executed using a backend binding. When the experiment terminates, the simulation results are returned to the graph transformation component and added to the provenance graph, as they contain valuable new information about the simulation study that needs to be documented. The new provenance created by the rule evaluation can be displayed to the user via the API.

Note that for now, we consider a provenance model that provides a macro-level view on the provenance (see also discussion in Section 2). This allows us to efficiently interpret the intention behind an activity (e.g., calibration vs. validation). If provenance is recorded on a more fine-grained level, aggregation methods are used to obtain the required provenance view. However, in the future, this framework could work on arbitrary provenance views if adequate patterns and rules were specified. The configuration of the rules and patterns can be done by any user of the framework via the API.

4.2 Experiment Reuse by Provenance Graph Transformation Rules

The framework will be driven by rules made up of provenance patterns, which form a graph transformation system that, based on a given provenance graph, extends this graph with new simulation experiments.

4.2.1 Provenance Patterns in Simulation Studies. Provenance patterns are the central building block for our approach. Associating provenance patterns with semantics enables us to interpret what happened during the simulation study. In particular, patterns are used in four ways by our approach.

(1) *Trigger patterns* initiate the reuse, adaptation, and execution of certain experiments. For us, trigger patterns always denote activities that produce a simulation model. Note that, especially if more fine-grained provenance information is available, other triggers that do not produce a simulation model might be possible.

(2) *Experiment patterns* are used to identify and retrieve previous simulation experiments. They describe either an experimentation activity with a specific role, e.g., experiments used for calibration, validation, or analysis [10], or experiments of a specific type, such as sensitivity analysis or statistical model checking.
(3) Eventually, from the provenance information a new activity including the adapted simulation experiment will be generated. Therefore, experiment patterns are used as *blueprints* for creating a new activity and connecting all entities correctly.
(4) *Condition patterns* represent the relationship between the trigger patterns, experiment patterns, and the rest of the provenance graph. They are particularly important when expressing rules for complex use cases that need to incorporate the context of the activities.

In the following, we predefine a number of provenance patterns based on our experience with simulation studies (from cell biology) where the development of a valid model is in focus and thus multiple model iterations are produced [10]. These patterns, illustrated in Figure 3, allow us to cover the scenarios described above in Section 3. However, this list is not exhaustive and there may be special use cases and patterns from other domains. The modular architecture of our framework allows users to add custom patterns and to use them in defining new rules.

"Refining Simulation Model" Pattern: Model refinement or extension is the typical model-building step during a simulation study. It involves a single simulation model and produces a new one. Usually, a variety of other entities are used during a model building step such as research questions, qualitative models, theories, assumptions, requirements, and data. If the used model entity belongs to a different simulation study than the generated model, the pattern describes the extension or refinement of an existing typically already validated model.

"Creating Simulation Model" Pattern: When a simulation model is created from scratch no simulation model is used. However, analogously to the refining pattern, various entities can be used: research questions, qualitative models, theories, assumptions, requirements, data, or other information sources. The output of the "creating simulation model" activity is an initial simulation model that is either entirely new or just has not been linked to another study yet.

"Re-Implementing Simulation Model" Pattern: Sometimes models are not refined or extended but re-implemented using other tools or languages. When re-implementing a simulation model, the provenance graph reflects this as a model-building activity, i.e., a simulation model is used, and another simulation model is generated by the activity. But in contrast to extending or refining, no new conceptual materials (such as a qualitative model, or input data) are used.

"Composing Simulation Model" Pattern: The composition pattern involves two simulation models as input to an activity (used-dependency). These are fused into a composed model, i.e., the output of this activity. Similar to the refining or creating simulation model pattern, further entities can be used that deliver context information about why and how to combine the models. Moreover, the used models and the composed model may belong to different simulation studies.

"Calibrating Simulation Model" Pattern: In a calibration experiment, the model parameters are fitted with the help of additional context data. Thus, the input of the calibrating simulation model pattern consists of a simulation model as well as data or alternatively a requirement entity (if the requirement, e.g, expresses properties of the target trajectory and a distance measure). The output of a calibration activity is a simulation experiment containing the experiment specification, a data entity containing the simulation output and the calibration status (success/failure), as well as a modified (fitted) simulation model. Of course, further entities are allowed as input to the calibration, e.g., assumptions or theories or input data, but these are not required. Also, simulation

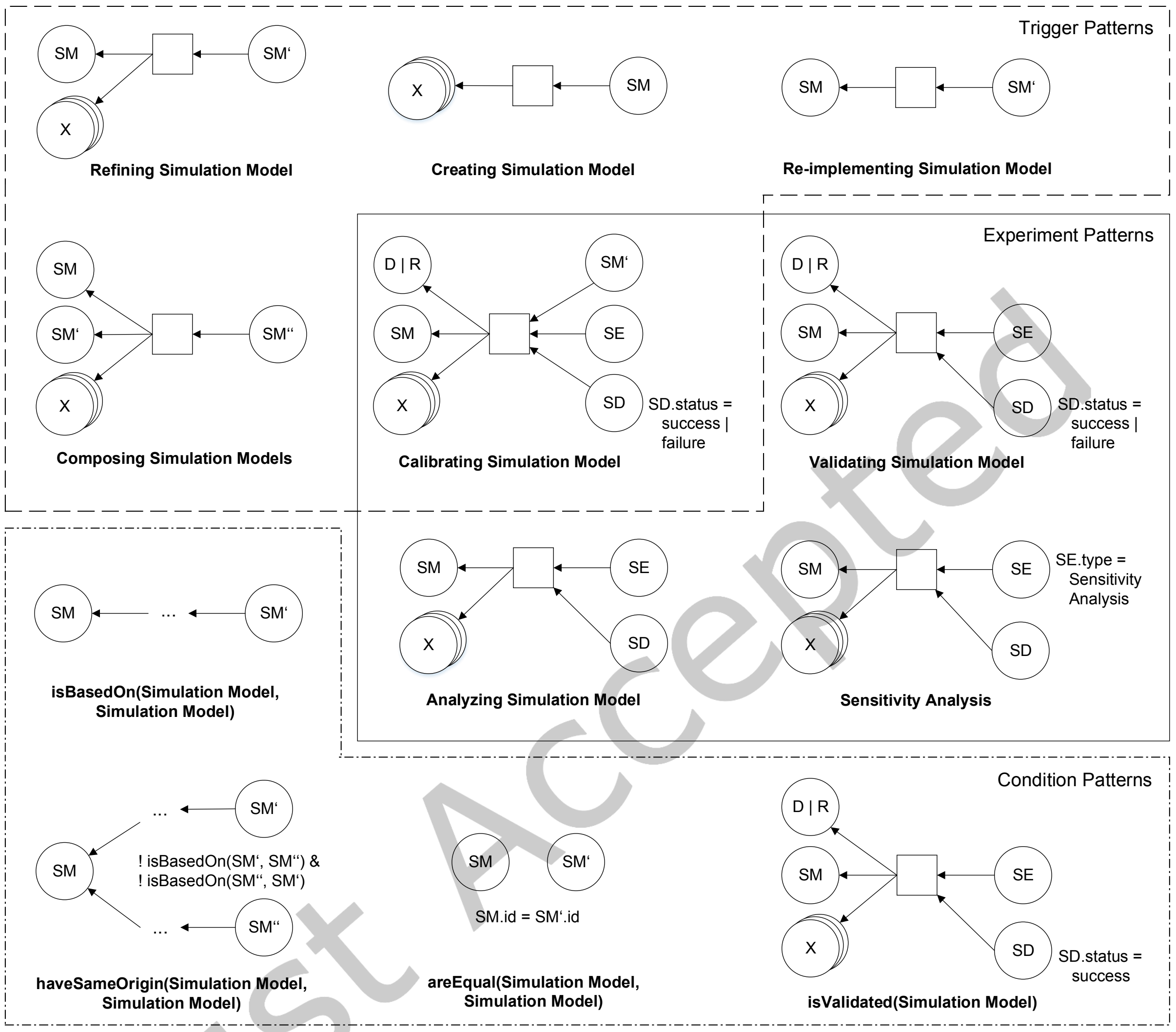

Trigger Patterns
Refining Simulation Model
Creating Simulation Model
Re-implementing Simulation Model
Composing Simulation Models
Experiment Patterns
Calibrating Simulation Model
SD.status = success | failure
Validating Simulation Model
SD.status = success | failure
Analyzing Simulation Model
Sensitivity Analysis
SE.type = Sensitivity Analysis
Condition Patterns
isBasedOn(Simulation Model, Simulation Model)
! isBasedOn(SM', SM'') & ! isBasedOn(SM'', SM')
haveSameOrigin(Simulation Model, Simulation Model)
SM.id = SM'.id
areEqual(Simulation Model, Simulation Model)
SD.status = success
isValidated(Simulation Model)
SM
SM'
SM''
X
D | R
SE
SD

Fig. 3. Provenance patterns for interpreting changes in the simulation models (trigger patterns), finding suitable simulation experiments to reuse (experiment patterns), and expressing the relationship between trigger pattern, experiment pattern, and the rest of the provenance graph (condition patterns). The patterns show what entities are required as inputs and outputs for a certain type of activity. The multi-entities (X) are used to capture all remaining inputs, however, they must not contain entities of the type simulation model (SM).

experiments may be used as input to a calibration. This indicates that the new experiment was created by reusing a previous experiment.

"Validating Simulation Model" Pattern: The validating simulation model pattern looks in large parts similar to the calibration patterns. Here, also either a data or requirement entity is required. As in the case of calibration, other entities can be used, e.g., to input data containing the initial configuration, or reused simulation experiment.

Products of the validation activity are a simulation experiment and simulation data indicating the success or failure of the validation. In contrast to calibration, no new simulation model is generated.

"Analyzing Simulation Model" Pattern: An analyzing simulation model activity uses a single simulation model as input, produces a simulation experiment and simulation data. In contrast to calibration and validation activities, they do not require specific input entities and their results are not interpreted as success or failure.

"Sensitivity Analysis" Pattern: Especially with analysis experiments, we could also ask more specifically about an experiment type that was conducted during the simulation study. Therefore, the experiment patterns can be refined by including meta-data from the information model of the experiment entity. For example the analysis pattern can be refined to a "sensitivity analysis" pattern. This, of course, assumes that the experiment type was made explicit as an attribute in the information model of the experiment entity.

"isBasedOn" Pattern: This pattern relates two simulation models. A simulation model SM' is said to be based on another simulation model SM if a directed path exists in the provenance graph from SM' to SM.

"haveSameOrigin" Pattern: This pattern is used to express that two simulation models (SM' and SM") have a common predecessor model from which their development started. This is the case if there exists a third simulation model SM to which from both SM' and SM" exists a path, and SM' is not based on SM", and SM" is not based on SM'.

"areEqual" Pattern: This pattern takes two simulation models. The two models are considered to be equal if their IDs, contained in the information models, are equal.

"isValidated" Pattern: This pattern is used to describe that a simulation model is validated. A simulation model is considered to be validated if a simulation experiment exists that was executed with this model, and the corresponding data entity shows the validation status "successful" in its information model.

4.2.2 Construction of the Rules. The reuse of simulation experiments will be based on a graph transformation system. The graph transformation system is given by a set of production rules R, which we will also call the reuse rules. A reuse rule is given in the form $(T, E, f) \rightarrow E_{gen}$, where the left-hand side of the rule consists of a trigger pattern T, an experiment pattern E, and a condition function f relating the two. A trigger pattern is a provenance graph that is considered to contain exactly one activity node and one simulation model entity that was generated by that activity. Similarly, an experiment pattern is considered to contain exactly one activity, however, it must contain one simulation experiment entity that was generated by the activity. The condition is a Boolean function that evaluates on the trigger and the experiment pattern. It is composed of predicates and logic operator symbols, where a predicate refers to a condition pattern C (e.g., *isBasedOn(Simulation Model, Simulation Model)* or *isValidated(Simulation Model)*). Condition patterns can be arbitrary provenance patterns, with the restriction that they have to share some parts with the trigger pattern or the experiment pattern, $C \cap (T \cup E) \neq \emptyset$.

The right-hand side of a rule takes the entities from the left-hand side and describes how they have to be extended, i.e., $E_{gen} = (T, E, C) \cup E'$, where E' is also an experiment pattern either of the same kind or different kind as E. This way the new experiment activity can be recognized and reused in future model development cycles. Moreover, the entities e used by the activity a of E' have to be reused parts of the trigger or experiment pattern, i.e., $\forall e, a \in E' : used(a, e) \implies e \in E \cup T$. Note that thereby we do not allow rules that modify or delete existing nodes.

For our framework, we predefine a set of rules that map to the scenarios from Section 3 using the patterns from Section 4.2.1. However, there might be domains where these patterns and rules do not apply. Therefore, we allow the rule set to be customized by enabling or disabling rules depending on the user's level of expertise.

Furthermore, custom rules can be created for specific settings by combining trigger patterns, experiment patterns, and condition patterns as needed. Also, new patterns can be defined.

4.3 Rule Matching Algorithm

The rule matching procedure is given in Algorithm 1. It is started each time a new activity a including dependencies and newly generated entities was completed and added to the given provenance graph G. For each rule, first, the trigger pattern is matched in G. Here the most recent activity a serves as an anchor point, i.e., the trigger pattern can only match at that activity (identified by its id). This is used to limit the search space and to only retrieve subgraphs relevant to the current activity and thus the current simulation model. If, for example, already a cascade of generated simulation experiments exists, not all of these need to be reused, but only the latest version conducted with the predecessor model is used. Note that according to our definitions, a trigger pattern only contains one activity.

If the trigger of the rule matches the current activity, all occurrences of E that fulfill the conditions in relation to the matched trigger $f(t, e)$ are collected. We will often find multiple reusable experiments for the same trigger, for example, various validation experiments where each checks a different behavioral property of the model given by a requirement entity. These experiments can be reused, adapted, and executed in parallel, as the results of experiments conducted with the same trigger model are independent of one another. For the same reason, the rules do not have to be evaluated in a specific order.

For each of the matches $e \in M$, first the old experiment specification is retrieved from e, and an adapted version s' is generated for the new simulation model. Subsection 4.4 describes this in detail. Then, the experiment specification is executed with the new model, and the simulation result data d and possibly a new fitted model m are obtained. The generation pattern E_{gen} from the right-hand side of the rule provides the blueprint for creating new entities, activities, and dependencies. By instantiating this blueprint, a new subgraph E_{new} is created. To complete the new provenance, the simulation results have to be stored in the information model of the new simulation data entity, the generated experiment specification has to be stored in the new simulation experiment entity, and if available, the new model specification has to be stored in the new model entity. Finally, the E_{new} is temporarily added to the collection N.

When all rules have been evaluated, the collection of new subgraphs can be added at once to the provenance graph, i.e., $G = G + N$. Before that, however, the algorithm has to wait until all experiment executions are completed. This is necessary since a rule might produce a new simulation model and thus would immediately trigger a new round of rule matching. In case the next experiments to be generated depend on the results of the previous experiments, the simulation results should be available before the algorithm proceeds.

The complexity of the matching procedure depends on the number of rules evaluate and the size of the provenance graph and the provenance patterns to be matched. To support, e.g., the scenarios described in Section 3, only seven rules are required (where the validation scenario accounts for two rules: model extension and model composition). Further, our approach works on aggregated provenance where only the macroscopic steps are viewed, i.e., model building, model calibration, model validation, and model analysis. These high-level provenance graphs and query patterns are easily manageable for state-of-the-art graph databases. Overall, in light of the runtimes required to execute the experiments themselves, which can easily add up to several hours for a single simulation run, the time required for rule matching is negligible.

4.4 Adaptation of the Experiment Specifications

The old experiment specifications extracted via the experiment patterns may not be executable with the current model (version). To evaluate whether changes have to be made to the experiment specification, the contexts of the old and new simulation experiments need to be taken into account. Under the term context, we subsume all

Algorithm 1: Rule matching at activity a

```
Input: a provenance graph $G$, a set of reuse rules $R$, the current activity $a$
$N \leftarrow \emptyset$
for each $(T, E, f) \rightarrow E_{gen}$ in $R$ do
    $t \leftarrow$ find match of $T$ in $G$ where activity id of $t$ equals id of $a$
    if $t \neq$ NULL then
        $M \leftarrow$ find all matches $e$ of $E$ in $G$ where $f(t, e) = true$
        for each $e$ in $M$ do in parallel
            $s \leftarrow$ get experiment specification from $e$
            $s' \leftarrow$ adapt $s$ based on $e$
            $d, m \leftarrow$ execute $s'$
            $E_{new} \leftarrow$ instantiate $E_{gen}$
            add experiment specification $s'$ to $E_{new}$
            add result data $d$ to $E_{new}$
            add simulation model $m$ to $E_{new}$
            $N \leftarrow N \cup E_{new}$
wait for all parallel tasks
$G \leftarrow G + N$
```

entities that participated in the model building process. These may be, e.g., the qualitative model, assumptions, data and information sources, research question, assumptions, or theories, which are also known as the conceptual model [56]. We trace these entities in the provenance graph and check newer versions (i.e., since the last experiment execution) for relevant information.

But before any adaptations can be made, we have to make all the parts of the experiment specifications easily accessible. Therefore, they are translated to an intermediate representation, which we will call the canonical form. In the canonical form, each part of the experiment specification is assigned a quasi-standardized identifier. Based on these identifiers, adaptions can be defined just once for a variety of different modeling and simulation tools and approaches. For the translation, we use a metamodel of simulation experiments, see Figure 4. The metamodel defines the vocabularies for various simulation approaches (e.g., discrete-event simulation) as well as experiment-type-specific vocabulary (e.g., for sensitivity analysis or statistical model checking).

Which of these parts needs to be adapted in the selected experiment specifications can be derived from the information models of the provenance entities. The better these are structured and substantiated with formal expressions, the better they can be exploited automatically. Plenty of work on conceptual modeling [56], workflows [58], and provenance ontologies [10] focuses on the contents of the various entities. In the following we list some adaptations that are currently supported in our framework:

- The current simulation model may be located in a different folder than the old model, or it may have a different name. To update the experiment specification accordingly, meta-information from the simulation model entity can be used.
- Parameter names and initial values belong to every experiment specification. If the name of a parameter changed or if the initial values of the model were revised, they also need to be updated in the reused experiment specification. The qualitative model is the entity that comprises valuable information about model inputs. Also, the simulation model itself might be annotated with useful information.

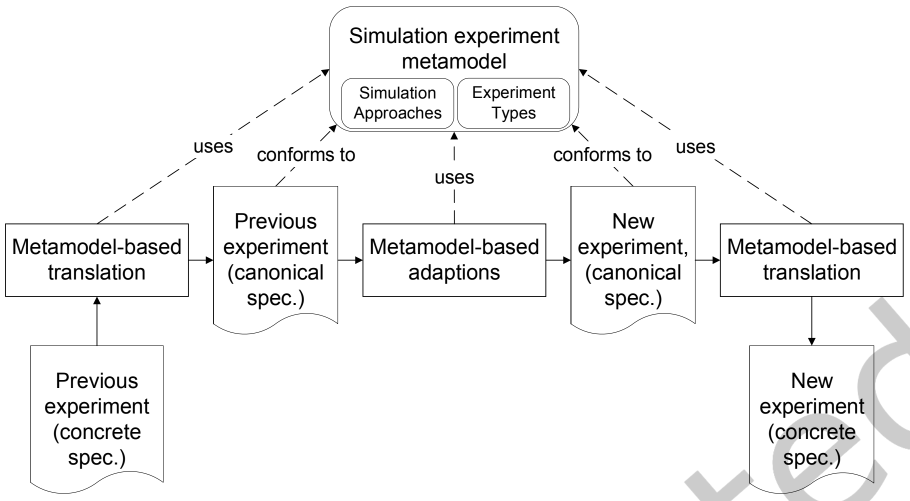

Simulation experiment metamodel
Simulation Approaches
Experiment Types
uses
conforms to
uses
conforms to
uses
Metamodel-based translation
Previous experiment (canonical spec.)
Metamodel-based adaptions
New experiment, (canonical spec.)
Metamodel-based translation
Previous experiment (concrete spec.)
New experiment (concrete spec.)

Fig. 4. Adapting and generating simulation experiments based on metamodels. The pipeline and the metamodels were defined in [74].

- Similarly to the set of parameters, the specification of the observed species might have to be adapted. Again the qualitative model or the simulation model can provide information about the model outputs.
- For simulation experiments such as parameter scans or sensitivity analyses, a central aspect of the specification is the experiment design. It includes specifying the minimum and maximum value as well as the probability distribution of each factor (i.e., parameter varied during the experiment). Information about the values of a factor may be available in the form of assumption entities.
- Other experiment types, such as statistical model checking, rely on model properties expressed as temporal logic formulas. Occasionally, the expression may have changed since the last execution of the simulation experiment, which will be indicated by an update on a requirement entity in the provenance graph. If the formula is included explicitly in the requirement entity, it can automatically be inserted into the new experiment specification.

While some changes are easy to detect, others require special mechanisms for comparing provenance entities. One option is annotating the information models with tags from various ontologies. These can be ontologies to identify a parameter or compartment (e.g., the Gene Ontology (GO) [67]) or ontologies to identify an algorithm or methodology (e.g., the Kinetic Simulation Algorithm Ontology (KiSAO) [14], or an Ontology for Discrete-event MOdeling and simulation (DeMO) [65]). Sometimes, if not enough meta-data is available in other entities, the simulation model specifications need to be compared. In that case, an approach for characterizing changes in model files could be used [62].

After adapting the experiment specification in the canonical form, it translated back to a specification in a concrete language of a backend using the metamodel. Which backend (and thus experiment language) is used, depends on the models and experiments at hand. If experiments are reused within the same simulation study, typically the same language and backend can be used as with the original experiment. On the contrary, when an experiment is reused across simulation studies, it is likely that these studies use different tools, and thus the new simulation experiment will be generated in a different language. Information about which backend the current simulation study uses can be determined by looking at other simulation experiments from that study. While it is favorable to use tools consistently during a simulation study, the metamodel-based approach allows us to use any backend, as long as the kind of simulation model and the type of experiment are supported. For example, a parameter scan of a model formulated in SBML [31] could be conducted in both COPASI [30] and a general-purpose programming language like Python or Java using the library LibSBML [9].

5 IMPLEMENTATION

The implementation of RASE is openly available in a Git repository[1]. All software components were implemented using Java 8. The API allows users to connect the framework to any application that produces provenance. This can be done by creating different kinds of provenance nodes and relationships, and sending them to the Prov Model Controller via Provenance Commits (see Figure 5). Currently, the high-level provenance model introduced in Section 2.1 is assumed, which is implemented in a separate module, and provides the classes Assumption, Data, Experiment, etc. as entity types. Note that the provenance model shown in Figure 5 merely reflects the current state of the implementation and may be extended in future work to accommodate new use cases. The recorded provenance is stored in a Neo4j graph database [42]. Accordingly, the queries for the various provenance patterns are expressed in the Cypher query language [20].

We facilitate a previously developed MDE-based toolchain [74] for adapting, generating, and executing simulation experiment specifications. This allows us to handle a variety of specification formats and simulation backends. As intermediate representation for the experiment adaption, it uses JSON documents [33] based on metamodels defined in JSON Schema [34]. For the automatic execution of simulation experiments, it already provides a number of bindings to backends, e.g., for SESSL [17] and ML-Rules [26], which are used in the Wnt signaling case study. For the migration case study of this paper, we implemented an additional binding for simulation experiments conducted with Julia [8] and R [52].

Note, that in this paper we focus on fully automated cases, i.e., everything from recording provenance to executing the generated experiments is done automatically. However, often user inputs are required to correctly adapt the experiment specifications. For this purpose, the experiment generator includes a graphical user interface (GUI) based on JavaFX for presenting the generated experiment specifications to the user and to allow them to make manual changes [74]. The graphical support feature can be enabled via the API. Furthermore, the API allows users to customize the set of provenance patterns and reuse rules via abstract classes, analogously to the implementation of our case studies, to receive the necessary level of support. For users with little programming experience, the implementation of new rule can still be difficult. Therefore, we are planning to evolve the API's usability and documentation as part of our future work.

6 CASE STUDIES

We demonstrate the usefulness of our approach in two different simulation studies. In the first case study, we will show a repeated sensitivity analysis of a model of migration routes. In the second case study, we will reuse an experiment across simulation studies for cross-validation. Afterward, we discuss the advantages of our approach in the fields of sociology and cell biology. To reproduce the case studies, please refer to the accompanying Zenodo publication[2].

Since in this paper we can only show a selected set of rules and patterns at work, with our source code we provide an additional, abstract simulation study to illustrate further reuse cases based on a predator-prey model.

6.1 Configuration of the Rule Set

For the case studies, we define two exemplary rules as shown in Figure 6. The first rule ($r1$) describes the generation of a sensitivity analysis. It takes a model refinement step as a trigger and the sensitivity analysis pattern for

[1]https://git.informatik.uni-rostock.de/mosi/exp-generation
[2]https://doi.org/10.5281/zenodo.6792025

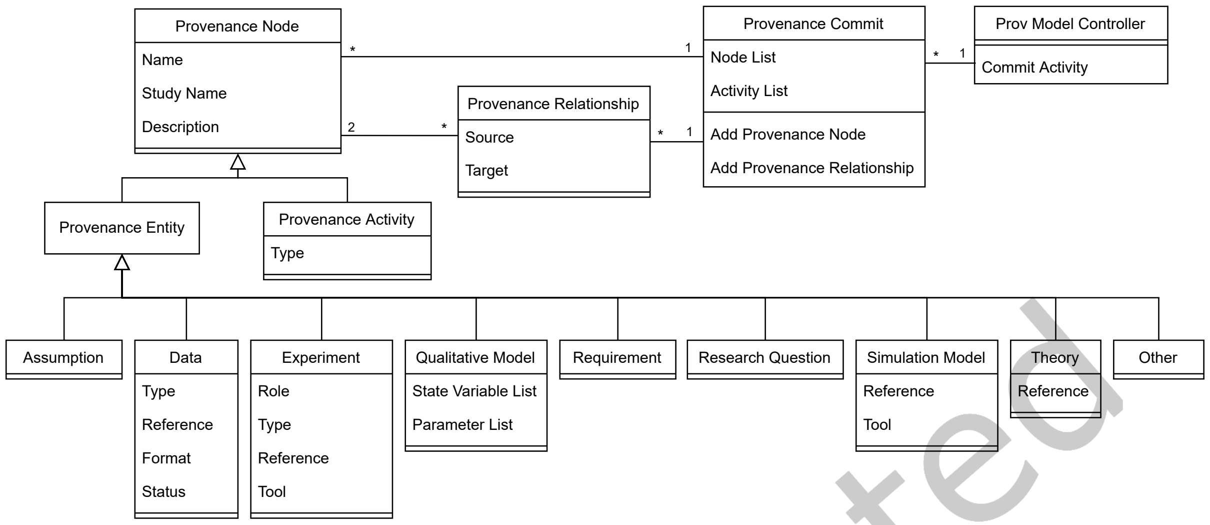


Fig. 5. API for capturing provenance. Various kinds of provenance nodes and provenance relationships between two nodes can be created and grouped via provenance commits. These can be send to the graph database via the Prov Model Controller, which may trigger some reuse rules to fire. Provenance nodes can be either activities or entities and have to be filled with metadata. For instance, the entity type Experiment provides the fields Role (e.g., for calibrating or validating the simulation model), Type (e.g., sensitivity analysis or steady-state analysis), Reference (i.e., the path to the experiment specification), and Tool (i.e., the tool for running the experiment).

finding previous experiments. As a condition for searching in the provenance graph, the simulation model used in the refinement has to be the same model that was used in the sensitivity analysis. From this information, a new activity is generated that takes the latest simulation model (from the trigger pattern), as well as other inputs from the previous experiment activity (accumulated in the multi-entity Y). To denote that an experiment was reused during this step, also the old simulation experiments SE is taken as input. The output of the new activity is a simulation experiment entity with type sensitivity analysis. Moreover, a simulation data entity is generated.

The second rule ($r2$) example presents the cross-validation scenario. Here, model calibration is used as a trigger activity. Suitable experiments to reuse are identified using the analysis pattern. However, further conditions have to apply, i.e., the simulation model used in the trigger pattern has to be based on the simulation model used in the analysis pattern, they have to belong to different studies, and the model used in the analysis has to be validated. The generated activity corresponds to the validation pattern. In contrast to the generation part in $r1$, here the output data of the previous experiment is taken as input to allow for comparison of the two models.

6.2 Repeated Analysis of a Migration Model

The first case study refers to an agent-based model of asylum migration to Europe [28] focusing on the formation of migration routes in response to spread of information. It aims to connect individual decisions (information transfer) on the micro-level to processes observed at the macro level (variability and optimality of migration routes). Models are successively refined and analyzed[3] in response to developing theoretical lines of inquiry and the introduction of different data sources, which presents various opportunities for our framework to automatically reuse and generate simulation experiments. The provenance of the study was described in [55]. Here, we

[3]M1–M2: https://github.com/mhinsch/RoutesRumours, M3–M5: https://github.com/mhinsch/rgct_data

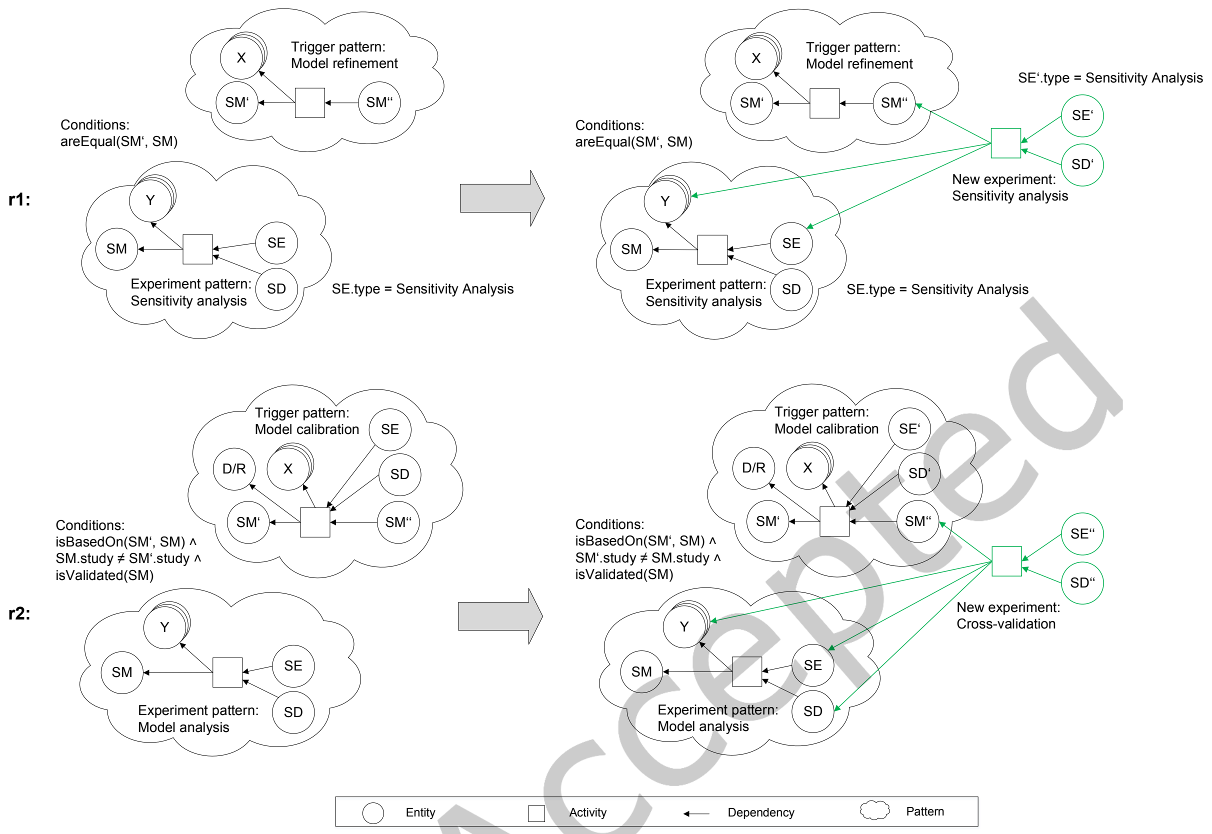

r1:
Trigger pattern: Model refinement
X
SM'
SM"
Conditions: areEqual(SM', SM)
Y
SM
SE
SD
Experiment pattern: Sensitivity analysis
SE.type = Sensitivity Analysis
SE'.type = Sensitivity Analysis
SE'
SD'
New experiment: Sensitivity analysis
r2:
Trigger pattern: Model calibration
D/R
SE'
SD'
Conditions: isBasedOn(SM', SM) ∧ SM.study ≠ SM'.study ∧ isValidated(SM)
Conditions: isBasedOn(SM', SM) ∧ SM'.study ≠ SM.study ∧ isValidated(SM)
Experiment pattern: Model analysis
SE"
SD"
New experiment: Cross-validation
Entity
Activity
Dependency
Pattern

Fig. 6. Rule r1 describes the repetition of a sensitivity analysis. Rule r2 shows the cross-validation scenario. In both rules, the left-hand side specifies a trigger pattern, a pattern of a previous simulation experiment, and additional conditions. The right-hand side extends the provenance graph by a new experiment activity (shown in green).

only show the parts that focus on the modeling and analysis activities (Figure 7). Everything concerning psychological experiments and data processing is omitted. Furthermore, we modified the provenance slightly to make the experiment specifications explicit ($E1 - E4$).

The simulation study begins with the creation of a model M1, which already includes agent knowledge, social networks, and information exchange with discrete-time behavior, and model M2 which alters the simulated world from a grid-based layout to a more general and less densely connected graph topography [27].

During the model building steps m2' and m3, different versions of the originally time-stepped model (M2) are created to obtain more realistic time courses. Model M3 then presents a continuous-time version of M2, implemented in Julia [8]. With this continuous-time model M3, now first experiments are conducted.

In a1, a parameter scan is employed to reduce the 17 parameters to the 6 most influential factors. Then, in a2 a sensitivity analysis is conducted on the selected parameters. The analysis uses a Latin Hypercube sample to build Gaussian process emulators and carry out the uncertainty and sensitivity analysis. In the original publication, the

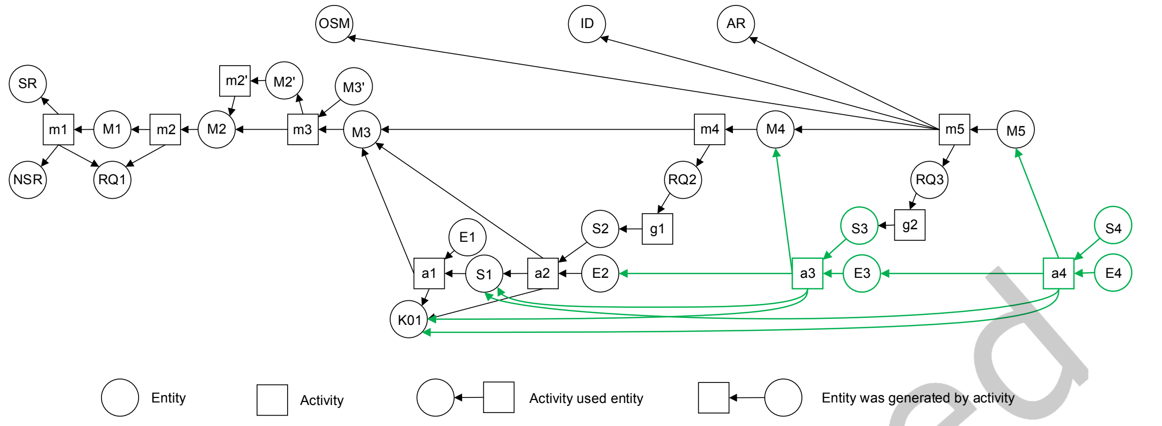
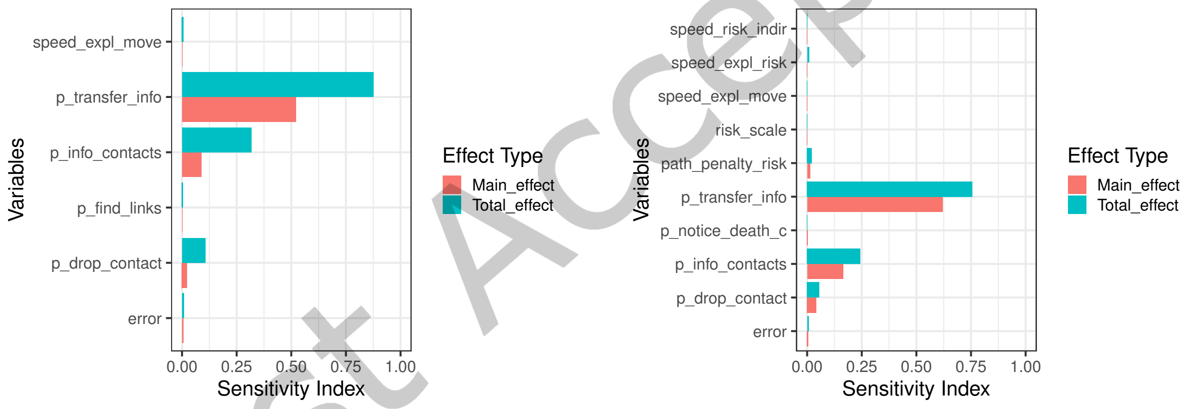


Fig. 7. Provenance graph of the migration study published in [55]. Activities, entities, and dependencies shown in green are generated automatically by our approach.



(a) Main and total effects for M3

(b) Main and total effects for M4

Fig. 8. Sensitivity indices calculated for the output mean_freq_plan with (a) the original experiment and model M3, and (b) the automatically reused experiment and model M4.

analysis was conducted with the (GUI-based) tool GEM-SA. For this case study, we prepared the same experiment as R scripts to have the experiment specifications accessible to our approach [4].

Following the experiment, the next model version is developed to include additional empirical data on information exchange and risk behavior, i.e., RQ2 and RF are used during the model refinement step m4. This model refinement now triggers the rule matching of our approach. Rule *r*1 can be applied at activity m4 and the

[4]https://github.com/jasonhilton/screen_run

following match is found:

$$\begin{aligned} match_1 = \{&SM' \leftarrow M3, X \leftarrow \{RQ2, RF\}, SM'' \leftarrow M4, \\ &SM \leftarrow M3, Y \leftarrow \{K01, S1\}, SE \leftarrow E2, SD \leftarrow S2\}, \end{aligned}$$

where the left-hand side of the arrows represents the variable names given in the rules, and the right-hand side represents the names of the matched entities from the provenance graph in Figure 7.

Executing the rule generates a new activity a3, and used-dependencies are drawn to the new simulation model M4, the previous experiment E2, and other input entities involved in a2, i.e., K01 and S1. As output, according to the "sensitivity analysis" pattern, an experiment entity E3, and a simulation data entity S3 are generated. When generating the new experiment specification for entity E3, the model name has to be adapted from "M3" to "M4". In addition, the list of parameters is updated to include also the factors concerning risk behavior. Running the experiment produces main and total order sensitivity information which is added to the information model of S3. Figure 8 shows the sensitivity data for M3 (left) and M4 (right) for the output mean_freq_plan, which captures the proportion of time for which agents are following their plan for transiting the space. In both model versions, the parameter p_transfer_info was identified as the key driver for planning behavior, indicating that information transfer was crucial for plan choice. However, it is noticeable that the addition of risk behavior in M4 does not have any substantial influence on this particular output.

Following the experiment results, the simulation model is refined further. New data entities and research questions are included in the next model refinement step, which produces the model version M5. Again, triggered by the refinement activity, our approach automatically repeats the previous sensitivity analysis experiment. This time the activity m5 is taken as the anchor point for matching the rule r1. The activity a4 is automatically generated including the new entities E4 and S4, and new used-connections to the inputs K01, S1, and the previous experiment E3.

6.3 Cross-Validation of two Models of the Wnt/β-Catenin Signaling Pathway

The Wnt/β-Catenin signaling pathway is a central pathway in the development and homeostasis of cells [48]. Degenerated forms of this pathway are involved in a number of cancers and neurological disorders [11]. The study by Haack et al. (2015) analyses the regulation of Wnt signaling in the initial cell fate commitment phase of neuronal progenitor cells [24]. In-vitro experiments indicated that raft- and redox-dependent signaling events play a crucial role in the regulation of Wnt signaling during this phase. A previous simulation model of the Wnt signaling pathway by Lee et al. (2003) [37] does not contain the corresponding model entities to accurately represent these regulatory mechanisms. Therefore the Lee model was extended with a membrane model component as well as additional intracellular model entities. The extended model was calibrated against new in-vitro data and subsequently cross-validated with simulation data from the Lee study to ensure that the basic model behavior was not changed due to the extension and (re-)calibration of the model.

Provenance for both simulation studies was documented by [10]. Figure 9 (left) depicts the entire provenance graph of the Lee study. For the Haack et al. study, we only show the initial phase with the first few activities (model building BSM1 and calibration CSM1) until the first automatic experiment generation. The connection between the models is shown by a used relationship between SM2 of the Lee model and BSM1 of the Haack model. Following the model extension, a calibration experiment is conducted. This is recognized as a trigger by our approach[5].

In particular, the cross-validation rule *r*2 can be applied at CSM1. Note that to save space, in Figure 9 activities of the same type are displayed in aggregated form (e.g., ASM1 of the Lee study aggregates four experiment

[5]The resources for the Wnt case study are provided with the main repository: https://git.informatik.uni-rostock.de/mosi/exp-generation

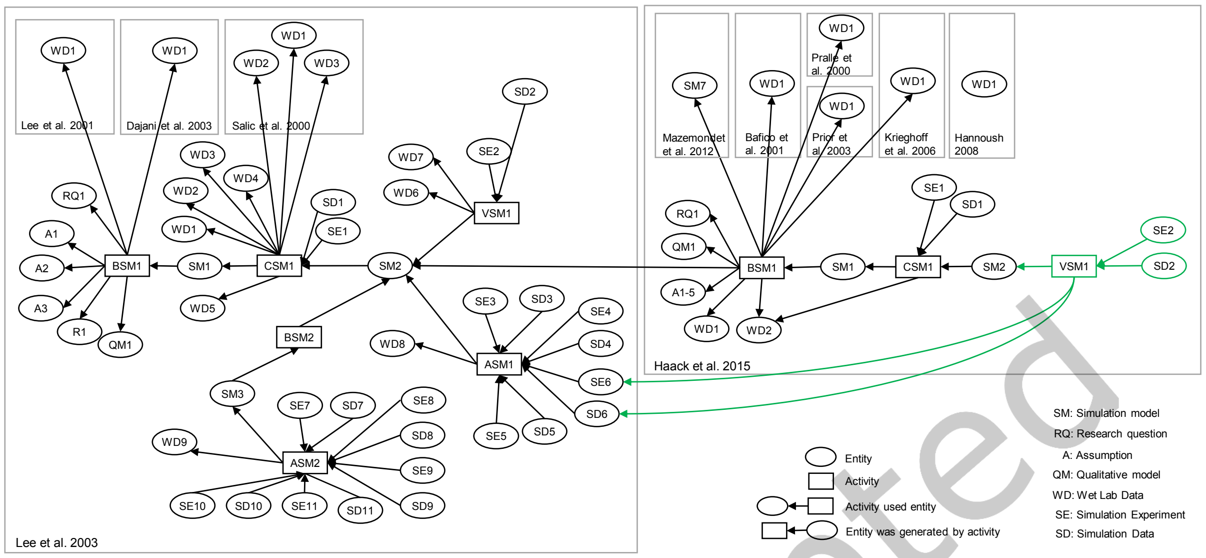


Fig. 9. Provenance graph showing the simulation study by Lee et al. (2003) and the initial phase of the Haack et al. (2015) study. Both studies are related via the model building activity BSM1. Following the calibration activity CSM1, cross-validation is triggered for the new model SM2. Thus, a new validation activity VSM1 is generated which reuses the simulation experiment SE6 and the simulation data SD6 from the Lee study and produces an adapted simulation experiment SE3 and the simulation data SD3.

activities). Thus, this yields four different matches of the rule. E.g., $match_1$ matches SE5, SD5, and input data WD8 as suitable candidates for reuse, and $match_2$ matches SE6 and SD6:

$$\begin{aligned} match_1 = \{&SM' \leftarrow SM1, D \leftarrow WD2, X \leftarrow \emptyset, SE' \leftarrow SE1, SD' \leftarrow SD1, SM'' \leftarrow SM2, \\ &SM \leftarrow SM2, Y \leftarrow \{WD8\}, SE \leftarrow SE5, SD \leftarrow SD5\} \\ match_2 = \{&SM' \leftarrow SM1, D \leftarrow WD2, X \leftarrow \emptyset, SE' \leftarrow SE1, SD' \leftarrow SD1, SM'' \leftarrow SM2, \\ &SM \leftarrow SM2, Y \leftarrow \emptyset, SE \leftarrow SE6, SD \leftarrow SD6\}. \end{aligned}$$

Figure 9 exemplarily shows the generated validation activity based on SE6 and SD6 of the Lee et al. study and the new model SM2 of the Haack et al. study. The simulation experiment SE6 was specified in SED-ML and the corresponding model SM2 of the Lee et al. study in SBML [31]. The files[6] are available on the BioModels database [41]. However, as the Wnt model by Haack et al. was specified using the rule-based modeling language ML-Rules [26], and the experiments are conducted using the experiment specification language SESSL [17], during the adaption and generation step, the experiment specification has to be translated.

During the translation, to receive an executable experiment specification, it is checked whether the observed species are still known under the same name in the extended model. Here, we facilitate the qualitative models (both denoted QM1) based on which the simulation models were built. The qualitative models contain a list of species, each annotated with proteome identifiers using the Uniprot proteome identifier (UPID) [12]. Table 1

[6]https://www.ebi.ac.uk/biomodels/BIOMD0000000658

Table 1. UniProtKB IDs (prefixed by UniProtKB) are used to identify the species involved in the Lee model, and to map them to species in the Haack model.

Lee Model	Haack Model	Ontology Tag
W	Wnt	UniProtKB - P31285 (WNT3A)
Axin	Axin	UniProtKB - Q9YGY0 (AXIN1)
Beta-catenin	Bcat	UniProtKB - P26233 (CTNB1)
Dsh	Dvl	UniProtKB - P51142 (DVL2)

shows a mapping of selected model species from the Lee and Haack models. In the case of Beta-catenin, which is the variable of interest in this experiment, a translation is required.

But also other aspects of the experiments or models might have to be considered when reusing an experiment from a different simulation study. For instance, in our case study, both models are subject to different time scales. This is because the parameters of both models were fitted against experimental data from Xenopus egg extracts and human neural progenitor cells, two cell-biological systems with different time scales. Therefore, when comparing simulation results between both models, experiment outputs need to be translated by a certain factor.

After these adaptations, the new experiment SE3 is generated and can finally be executed. Figure 10 shows the cross-validation results. It compares the trajectories of the key protein β-catenin, an indicator of the pathway's activity, produced by the Lee and the Haack model (with adapted time scale) when stimulated with a transient stimulus. After applying the correct transformation factor, that corrects for the varying time scales, the β-catenin curves show the same maximum at the same time. This means that the extensions applied in the study of Haack et al. do not alter the dynamics of the pathway.

6.4 Results

In both case studies, we successfully demonstrated how provenance information can be exploited in conducting simulation studies in a more systematic and effective manner. Changes (or lack of changes) in the sensitivity of simulation outputs to parameters following substantive alterations to the model provide important information about the mechanisms at work in a simulated system. In the case of the migration model above, changes in the decision-making process of agents did not hugely affect the output in question. The ability to automatically identify the experiment needed for sensitivity analysis (and subsequently to trigger this experiment following a change in the model) significantly shortens the modeling cycle and reduces the burden on the modeler. This means that the focus can be on analyzing and interpreting simulation results and considering additional modeling steps.

In addition to shortening the modeling cycle, automated experiment specification is also valuable to increase the models' validity and reusability. For instance, in the Haack study, an existing model was extended by further model components that significantly altered the structure of the model. However, despite the changes, the model should still be able to reproduce basic dynamics of the pathway that were either obtained in wet-lab experiments or by simulations of a previous model version. The ability to automatically identify and reuse all experiments that are necessary to establish the validity of a model allows for easier and more rigorous validation of extended models.

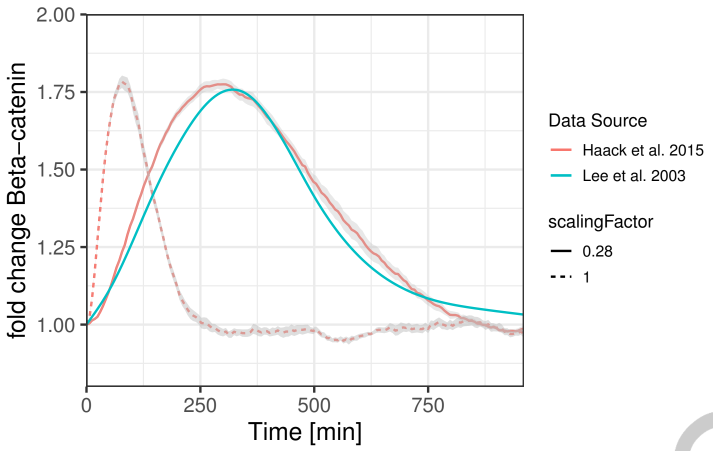


Fig. 10. Results of the cross-validation experiment. Beta-catenin values are compared between the Lee model and the Haack model before and after applying the scaling factor.

6.5 Discussion

Our approach relies on two requirements: 1) on documenting simulation studies as graphs following the provenance standard PROV-DM and 2) on executable simulation experiments, simulation models, and input data as part of or referenced in the graphs. Referring to 2), the last decades have seen increasing efforts of documenting simulation studies and making the diverse artifacts, including the simulation experiments, available to ensure the reproducibility and credibility of study results. These can now be exploited by our approach. E.g., the systems biology community made substantial progress in standardizing their specification formats as well as the way they package and share simulation models [63, 68]. Explicit simulation artifacts are, e.g., made available as COMBINE archives [7], which comprise separate model and simulation experiment files as well as data. Also in the agent-based modeling community executable artifacts are shared, e.g., via OpenABM [32] and Jupyter notebooks [4].

Regarding requirement 1), the provenance standard PROV-DM comes with a lot of benefits, such as improved visualization and analysis of the simulation study, but is currently not widely used in simulation. However, provenance may also come in a different form. For instance, documentations based on reporting guidelines, such as TRACE [22] and STRESS [44], provide crucial information about a simulation study in the form of verbal narratives. They describe the inputs, outputs, and versions of a model as well as the analyses conducted. Often they refer to platforms such as GitHub, where the described artifacts are made available. To apply our pattern-based approach to studies that use non-standardized provenance, two ways exist. First, the provenance could be transformed to PROV-DM. Here, recent approaches for automatically capturing provenance could be facilitated as a pre-processing step to our framework. These include, e.g., the abstract syntax tree analysis and execution trace analysis of scripts [46], the interpretation of electronic lab notebooks using ontologies [64], and the extraction of facts from scientific articles [36]. The second way would be to recognize patterns in these documentations directly. In TRACE, e.g., it is already annotated whether an experiment was run for calibration or validation, which would facilitate the recognition. COMBINE archives, as an other example, could be interpreted as a single experimentation activity that can be reused to generate a new experiment, i.e., a new COMBINE archive.

In future work, we plan to look more closely into dealing with these other forms of documenting simulation studies, outside of the provenance standard. We anticipate that in the near future more helper tools for provenance transformation and provenance pattern recognition will become available, and thus will make provenance pattern-based support mechanisms even more feasible. However, we are aware that there are domains that use closed-source, all-in-one tooling for which integration with our framework would be difficult.

7 RELATED WORK

Following the reproducibility crisis [50], changes in how simulation studies are conducted (and thus how simulation experiments are carried out) were required. Therefore, in recent years, simulation researchers investigated the reuse and/or generation of simulation experiments or parts thereof. Table 2 summarizes the related work on reusing and/or generating simulation experiments with regards to their overall objective, the central methodology applied, the experiment types supported, the simulation tools supported, and the scope of the automation (i.e., how the generation process is initiated, and whether the approach can be used within or across simulation studies). These are compared to the features of the approach presented in this paper (RASE).

The model-driven engineering (MDE) approach by Teran-Somohano et al. [66] targets the generation of experiment designs for single simulation experiments anywhere in the modeling life cycle. However, the experiments are generated based on user inputs, without considering provenance information.

Yilmaz et al. demonstrated the potential of MDE for generating experiment designs for hypothesis testing as part of a goal-hypothesis-experiment framework [75]. For generating experiment designs from hypotheses, the formal specification of hypotheses using a domain-specific language (DSL) is central, as shown by Lorig [39]. Again, they do not target specific steps in the modeling and simulation life cycle and the generation has to be triggered by the user.

The approach by Peng et al. [51] targets the reuse of statistical model checking experiments after model composition. Logic formulas, specified in Metric Interval Temporal Logic (MITL) [40], are merged to express and check the properties of the composed model. The generation must be triggered by a user, providing the formulas and adaptation information.

The cardiac electrophysiology wet lab designed by Cooper et al. [13] targeted the comparison and validation of simulation models. Similar to our approach, all data and experiments are stored. When a user uploads a finished model to the database, it is automatically cross-checked with other models by running all available experiments on the new model. Due to the limitation to cardiac electrophysiology, and the use of well-defined formats and naming conventions a high degree of automation can be achieved. However, the process of developing a model, including the various iteration of model building, calibration, and validation, is not supported.

The artifact-based workflow by Ruscheinski et al. [58] can guide users through the various building, calibration and validation steps of a model, and to repeating simulation experiments if certain milestones are achieved or invalidated. The concept of toolboxes enables the flexible use of different simulation tools and experiment types inside the workflow. However, the experiments are not automatically specified, adapted and executed by the workflow. Therefore, we suggest combining the artifact-based workflow with our approach where it would function as a provenance recording tool.

In contrast to the existing research, we provide a provenance-based mechanism for automatically reusing, adapting, and executing simulation experiments during the simulation study. Exploiting provenance information of the simulation study given in the provenance standard PROV-DM allows us to reason about the various model building and experimentation activities that can serve as triggers to the experiment generation, and for automatically selecting suitable simulation experiments to reuse. Thus, the approach is not restricted to specific settings and can cover any experiment type, be it factorial sweeps, statistical model checking or optimization.

Table 2. Comparison of related work on generating simulation experiments. n.a. = not applicable.

Publication	Objective	Central Methodology	Experiment Types	Supported Formats, Tools	Generation Trigger	Across Studies
Teran-Somohano et al. (2015)	Generation of factorial designs	Model-driven engineering	Factorial designs	Repast	User input (GUI-guided)	no
Yilmaz et al. (2016)	Generation of experiment designs from hypotheses	Model-driven engineering	Hypothesis testing	n.a.	User input (goals)	no
Lorig (2019)	Generation of experiment designs from hypotheses	Formal hypothesis specification	Hypothesis testing	NetLogo	User input (hypotheses)	no
Cooper et al. (2016)	Comparison and validation of electrophysiological cell models	Online database of models, experiments and data	Flexible	CellML, SED-ML	User input (model upload)	yes
Peng et al. (2017)	Reuse and adaption of simulation experiments for model composition	Composition of experiment specifications	Statistical model checking	SESSL, MITL	User input (logic formulae, adaption information)	yes
Ruscheinski et al. (2019a)	Guidance for re-running experiments based on artifact milestones	Artifact-based workflow	Flexible	Flexible	User input (GUI-guided)	no
RASE	Reuse and adaptation of simulation experiments based on provenance information collected throughout the simulation study	PROV-DM patterns and inference rules, Model-driven engineering	Flexible	Flexible	Automatically recognized model building activities	yes

In addition, provenance contains valuable information that allows for an automatic adaption of the reused experiment specifications. In contrast to Peng et al., where the adaption information needed to be provided by the user specifically for each experiment generation, this information can now be automatically extracted from the provenance of the simulation study. Integrating this with an MDE approach for generating and translating simulation experiments allows us to flexibly support a variety of experiment types and a variety of modeling and simulation tools. In contrast to Ruscheinski et al., MDE also facilitates automatic reuse across simulation studies that use different tooling as experiment specifications can be translated automatically.

8 CONCLUSIONS

The repetition of simulation experiments forms a salient feature of the model development process. Regression testing of successive model refinements and cross-validation of related models are just some examples of this. In this paper, we exploit provenance information of simulation studies to automatically identify suitable experiments to reuse depending on the last model development steps and to automatically adapt the experiment specifications accordingly for the new model version. The central methods of our approach are the definition of provenance patterns and the construction of reuse rules. Our collection of patterns and rules captures knowledge about simulation studies explicitly and thereby contributes to the conduction of more systematic and effective simulation studies. We successfully demonstrated the applicability of our approach in the context of human migration models as well as cell biology models. The software prototype of our Reuse and Adapt framework for Simulation Experiments (RASE) is publicly and permanently available.

To expand the scope of our framework, it could be integrated, e.g., with approaches for automatically analyzing and interpreting the simulation output [18], or approaches for experiment prioritization if limited resources are available, analogously to approaches for test prioritization known in software engineering [47]. Also, we are planning to improve the API and add (graphical) support for specifying new patterns and rules to allow for easier usability and customizability of our framework. So far, our work concentrates on the reuse of existing simulation experiments. However, another interesting step forward would be exploiting provenance for generating new experiments from scratch (i.e., without reusing a previous experiment specification as a blueprint).

ACKNOWLEDGMENTS

Funding: This research was funded by the German Research Foundation (DFG) via the grants 320435134 (P.W.), 258560741 (P.W.) and SFB 1270/1–299150580 (F.H.), and by the European Research Council (ERC) via grant CoG-2016-725232 (J.H.).
Author contributions[7]**:** Conceptualization: P.W., A.M.U.; Methodology: P.W., A.W., A.M.U.; Software: A.W., P.W., J.H., F.H.; Investigation: P.W., J.H., F.H.; Data Curation: J.H., F.H.; Writing original draft: P.W., A.M.U., Writing review and editing: P.W., A.M.U., J.H., F.H., A.W.; Visualization: P.W., J.H., F.H.; Supervision: A.M.U.; Project administration: P.W.; Funding Acquisition: A.M.U.
We would like to thank Martin Hinsch of the University of Southampton for his assistance in executing the sensitivity analyses. We would also like to show our gratitude to Marcus Dombrowsky for his early work on the software prototype, and we thank the reviewers for their constructive feedback.

REFERENCES

[1] Gul Agha and Karl Palmskog. 2018. A survey of statistical model checking. *ACM Trans. Model. Comput. Simul.* 28, 1, Article 6 (2018), 39 pages. https://doi.org/10.1145/3158668
[2] M David Allen, Len Seligman, Barbara Blaustein, and Adriane Chapman. 2010. *Provenance capture and use: A practical guide.* Technical Report. MITRE CORP MCLEAN VA MCLEAN.
[3] Robert Axtell, Robert Axelrod, Joshua M Epstein, and Michael D Cohen. 1996. Aligning simulation models: A case study and results. *Computational & Mathematical Organization Theory* 1, 2 (1996), 123–141.
[4] Daniel Ayllón, Steven F. Railsback, Cara Gallagher, Jacqueline Augusiak, Hans Baveco, Uta Berger, Sandrine Charles, Romina Martin, Andreas Focks, Nika Galic, Chun Liu, E. Emiel van Loon, Jacob Nabe-Nielsen, Cyril Piou, J. Gareth Polhill, Thomas G. Preuss, Viktoriia Radchuk, Amelie Schmolke, Julita Stadnicka-Michalak, Pernille Thorbek, and Volker Grimm. 2021. Keeping modelling notebooks with TRACE: Good for you and good for environmental research and management support. *Environmental Modelling & Software* 136 (2021), 104932. https://doi.org/10.1016/j.envsoft.2020.104932
[5] Osman Balci. 2012. A life cycle for modeling and simulation. *SIMULATION* 88, 7 (2012), 870–883. https://doi.org/10.1177/0037549712438469
[6] Khalid Belhajjame, Reza B'Far, James Cheney, Sam Coppens, Stephen Cresswell, Yolanda Gil, Paul Groth, Graham Klyne, Timothy Lebo, Jim McCusker, et al. 2013. PROV-DM: the PROV data model. *W3C Recommendation* (2013).
[7] Frank T Bergmann, Richard Adams, Stuart Moodie, Jonathan Cooper, Mihai Glont, Martin Golebiewski, Michael Hucka, Camille Laibe, Andrew K Miller, David P Nickerson, et al. 2014. COMBINE archive and OMEX format: one file to share all information to reproduce a modeling project. *BMC Bioinformatics* 15, 1 (2014), 1–9.
[8] Jeff Bezanson, Alan Edelman, Stefan Karpinski, and Viral B Shah. 2017. Julia: a fresh approach to numerical computing. *SIAM Rev.* 59, 1 (2017), 65–98. https://doi.org/10.1137/141000671
[9] Benjamin J. Bornstein, Sarah M. Keating, Akiya Jouraku, and Michael Hucka. 2008. LibSBML: an API library for SBML. *Bioinformatics* 24, 6 (02 2008), 880–881. https://doi.org/10.1093/bioinformatics/btn051
[10] Kai Budde, Jacob Smith, Pia Wilsdorf, Fiete Haack, and Adelinde M. Uhrmacher. 2021. Relating simulation studies by provenance—Developing a family of Wnt signaling models. *PLOS Computational Biology* 17, 8 (2021), 1–27. https://doi.org/10.1371/journal.pcbi.1009227
[11] Hans Clevers and Roel Nusse. 2012. Wnt/β-Catenin Signaling and Disease. *Cell* 149, 6 (Jun 2012), 1192–1205. https://doi.org/10.1016/j.cell.2012.05.012
[12] The UniProt Consortium. 2018. UniProt: a worldwide hub of protein knowledge. *Nucleic Acids Research* 47, D1 (11 2018), D506–D515. https://doi.org/10.1093/nar/gky1049
[13] Jonathan Cooper, Martin Scharm, and Gary R Mirams. 2016. The cardiac electrophysiology web lab. *Biophysical journal* 110, 2 (2016), 292–300. https://doi.org/10.1016/j.bpj.2015.12.012
[14] Mélanie Courtot, Nick Juty, Christian Knüpfer, Dagmar Waltemath, Anna Zhukova, Andreas Dräger, Michel Dumontier, Andrew Finney, Martin Golebiewski, Janna Hastings, et al. 2011. Controlled vocabularies and semantics in systems biology. *Molecular Systems Biology* 7, 1 (2011), 543. https://doi.org/10.1038/msb.2011.77
[15] Peter V. Coveney, Derek Groen, and Alfons G. Hoekstra. 2021. Reliability and reproducibility in computational science: implementing validation, verification and uncertainty quantification in silico. *Philosophical Transactions of the Royal Society A* 379 (2021).

[7] Contributor Roles Taxonomy (CRediT) http://credit.niso.org/

[16] Orçun Dayıbaş, Halit Oğuztüzün, and Levent Yılmaz. 2019. On the use of model-driven engineering principles for the management of simulation experiments. *Journal of Simulation* 13, 2 (2019), 83–95. https://doi.org/10.1080/17477778.2017.1418638

[17] Roland Ewald and Adelinde M. Uhrmacher. 2014. SESSL: a domain-specific language for simulation experiments. *ACM Trans. Model. Comput. Simul.* 24, 2, Article 11 (2014), 25 pages. https://doi.org/10.1145/2567895

[18] Niclas Feldkamp, Soeren Bergmann, and Steffen Strassburger. 2020. Knowledge Discovery in Simulation Data. *ACM Trans. Model. Comput. Simul.* 30, 4, Article 24 (2020), 25 pages. https://doi.org/10.1145/3391299

[19] Mingbin Feng and Jeremy Staum. 2017. Green simulation: reusing the output of repeated experiments. *ACM Trans. Model. Comput. Simul.* 27, 4, Article 23 (2017), 28 pages. https://doi.org/10.1145/3129130

[20] Nadime Francis, Alastair Green, Paolo Guagliardo, Leonid Libkin, Tobias Lindaaker, Victor Marsault, Stefan Plantikow, Mats Rydberg, Petra Selmer, and Andrés Taylor. 2018. Cypher: an evolving query language for property graphs. In *Proceedings of the 2018 International Conference on Management of Data (SIGMOD '18)*. 1433–1445. https://doi.org/10.1145/3183713.3190657

[21] Cláudio Gomes, Casper Thule, David Broman, Peter Gorm Larsen, and Hans Vangheluwe. 2018. Co-Simulation: A Survey. *ACM Comput. Surv.* 51, 3, Article 49 (2018), 33 pages. https://doi.org/10.1145/3179993

[22] Volker Grimm, Jacqueline Augusiak, Andreas Focks, Béatrice M. Frank, Faten Gabsi, Alice S.A. Johnston, Chun Liu, Benjamin T. Martin, Mattia Meli, Viktoriia Radchuk, Pernille Thorbek, and Steven F. Railsback. 2014. Towards better modelling and decision support: Documenting model development, testing, and analysis using TRACE. *Ecological Modelling* 280 (2014), 129–139. https://doi.org/10.1016/j.ecolmodel.2014.01.018

[23] Volker Grimm, Steven F Railsback, Christian E Vincenot, Uta Berger, Cara Gallagher, Donald L DeAngelis, Bruce Edmonds, Jiaqi Ge, Jarl Giske, Juergen Groeneveld, et al. 2020. The ODD protocol for describing agent-based and other simulation models: A second update to improve clarity, replication, and structural realism. *Journal of Artificial Societies and Social Simulation* 23, 2 (2020). https://doi.org/10.18564/jasss.4259

[24] Fiete Haack, Heiko Lemcke, Roland Ewald, Tareck Rharass, and Adelinde M Uhrmacher. 2015. Spatio-temporal model of endogenous ROS and raft-dependent WNT/beta-catenin signaling driving cell fate commitment in human neural progenitor cells. *PLOS Computational Biology* 11, 3 (2015). https://doi.org/10.1371/journal.pcbi.1004106

[25] David Hales, Juliette Rouchier, and Bruce Edmonds. 2003. Model-to-model analysis. *Journal of Artificial Societies and Social Simulation* 6, 4 (2003).

[26] Tobias Helms, Tom Warnke, Carsten Maus, and Adelinde M. Uhrmacher. 2017. Semantics and efficient simulation algorithms of an expressive multilevel modeling language. *ACM Trans. Model. Comput. Simul.* 27, 2, Article 8 (2017), 25 pages. https://doi.org/10.1145/2998499

[27] Martin Hinsch and Jakub Bijak. 2019. Rumours lead to self-organized migration routes. In *2019 International Workshop on Agent-Based Modelling of Human Behaviour (ABMHuB), Artificial Life Conference, ALife*. http://abmhub.cs.ucl.ac.uk/2019/papers/Hinsch_Bijak.pdf

[28] Martin Hinsch and Jakub Bijak. 2022. Towards more realistic models. In *Towards Bayesian Model-Based Demography: Agency, Complexity and Uncertainty in Migration Studies*. Methodos Series, Vol. 17. Springer, Dordecht, Chapter 8. https://doi.org/10.1007/978-3-030-83039-7_8

[29] Alexandre Hocquet and Frédéric Wieber. 2021. Epistemic issues in computational reproducibility: software as the elephant in the room. *European Journal for Philosophy of Science* 11, 2 (2021), 1–20.

[30] Stefan Hoops, Sven Sahle, Ralph Gauges, Christine Lee, Jürgen Pahle, Natalia Simus, Mudita Singhal, Liang Xu, Pedro Mendes, and Ursula Kummer. 2006. COPASI—a COmplex PAthway SImulator. *Bioinformatics* 22, 24 (2006), 3067–3074. https://doi.org/10.1093/bioinformatics/btl485

[31] M. Hucka, A. Finney, H. M. Sauro, H. Bolouri, J. C. Doyle, H. Kitano, A. P. Arkin, B. J. Bornstein, D. Bray, A. Cornish-Bowden, A. A. Cuellar, S. Dronov, et al. 2003. The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models. *Bioinformatics* 19, 4 (2003), 524–531. https://doi.org/10.1093/bioinformatics/btg015

[32] Marco A Janssen, Lilian Na'ia Alessa, Michael Barton, Sean Bergin, and Allen Lee. 2008. Towards a community framework for agent-based modelling. *Journal of Artificial Societies and Social Simulation* 11, 2 (2008), 6.

[33] JSON 2017. *ECMA-404 The JSON Data Interchange Standard, 2nd Edition.* Retrieved August 5, 2021 from https://www.json.org/

[34] JSON Schema 2018. *JSON Schema Draft-07 Release Notes.* Retrieved August 5, 2021 from https://json-schema.org/draft-07/json-schema-release-notes.html

[35] Thomas Kluyver, Benjamin Ragan-Kelley, Fernando Pérez, Brian Granger, Matthias Bussonnier, Jonathan Frederic, Kyle Kelley, Jessica Hamrick, Jason Grout, Sylvain Corlay, Paul Ivanov, Damián Avila, Safia Abdalla, Carol Willing, and Jupyter development team. 2016. Jupyter Notebooks - a publishing format for reproducible computational workflows. In *Positioning and Power in Academic Publishing: Players, Agents and Agendas*. IOS Press, 87–90. https://eprints.soton.ac.uk/403913/

[36] Frank Krüger and David Schindler. 2020. A Literature Review on Methods for the Extraction of Usage Statements of Software and Data. *Computing in Science Engineering* 22, 1 (2020), 26–38. https://doi.org/10.1109/MCSE.2019.2943847

[37] Ethan Lee, Adrian Salic, Roland Krüger, Reinhart Heinrich, and Marc W Kirschner. 2003. The roles of APC and Axin derived from experimental and theoretical analysis of the Wnt pathway. *PLOS Biology* 1, 1 (2003). https://doi.org/10.1371/journal.pbio.0000010

[38] Junxue Liang, Zhaowen Lin, and Yan Ma. 2012. OF-NEDL: an openflow networking experiment description language based on XML. In *International Conference on Web Information Systems and Mining*. Springer, 686–697.

[39] Fabian Lorig. 2019. *Hypothesis-Driven Simulation Studies: Assistance for the Systematic Design and Conducting of Computer Simulation Experiments*. Springer. https://doi.org/10.1007/978-3-658-27588-4_5

[40] Oded Maler and Dejan Nickovic. 2004. Monitoring Temporal Properties of Continuous Signals. In *Formal Techniques, Modelling and Analysis of Timed and Fault-Tolerant Systems*, Yassine Lakhnech and Sergio Yovine (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 152–166.

[41] Rahuman S Malik-Sheriff, Mihai Glont, Tung V N Nguyen, Krishna Tiwari, Matthew G Roberts, Ashley Xavier, Manh T Vu, Jinghao Men, Matthieu Maire, Sarubini Kananathan, Emma L Fairbanks, Johannes P Meyer, et al. 2020. BioModels — 15 years of sharing computational models in life science. *Nucleic Acids Research* 48, D1 (2020), D407–D415. https://doi.org/10.1093/nar/gkz1055

[42] Justin J Miller. 2013. Graph database applications and concepts with Neo4j. In *Proceedings of the Southern Association for Information Systems Conference, Atlanta, GA, USA*, Vol. 2324.

[43] Paolo Missier, Bertram Ludäscher, Shawn Bowers, Saumen Dey, Anandarup Sarkar, Biva Shrestha, Ilkay Altintas, Manish Kumar Anand, and Carole Goble. 2010. Linking multiple workflow provenance traces for interoperable collaborative science. In *The 5th Workshop on Workflows in Support of Large-Scale Science*. 1–8. https://doi.org/10.1109/WORKS.2010.5671861

[44] Thomas Monks, Christine S. M. Currie, Bhakti Stephan Onggo, Stewart Robinson, Martin Kunc, and Simon J. E. Taylor. 2019. Strengthening the reporting of empirical simulation studies: Introducing the STRESS guidelines. *Journal of Simulation* 13, 1 (2019), 55–67. https://doi.org/10.1080/17477778.2018.1442155

[45] Luc Moreau and Paul Groth. 2013. Provenance: an introduction to PROV. *Synthesis Lectures on the Semantic Web: Theory and Technology* 3, 4 (2013), 1–129. https://doi.org/10.2200/S00528ED1V01Y201308WBE007

[46] Leonardo Murta, Vanessa Braganholo, Fernando Chirigati, David Koop, and Juliana Freire. 2015. noWorkflow: Capturing and Analyzing Provenance of Scripts. In *Provenance and Annotation of Data and Processes*. Springer International Publishing, Cham, 71–83.

[47] Tanzeem Bin Noor and Hadi Hemmati. 2015. A similarity-based approach for test case prioritization using historical failure data. In *2015 IEEE 26th International Symposium on Software Reliability Engineering (ISSRE)*. 58–68. https://doi.org/10.1109/ISSRE.2015.7381799

[48] Roel Nusse. 2008. Wnt signaling and stem cell control. *Cell Research* 18, 5 (2008), 523–527. https://doi.org/10.1038/cr.2008.47

[49] Cyrus Omar, Jonathan Aldrich, and Richard C. Gerkin. 2014. Collaborative Infrastructure for Test-Driven Scientific Model Validation. In *Companion Proceedings of the 36th International Conference on Software Engineering (ICSE Companion 2014)*. 524–527. https://doi.org/10.1145/2591062.2591129

[50] K. Pawlikowski, H.-D.J. Jeong, and J.-S.R. Lee. 2002. On credibility of simulation studies of telecommunication networks. *IEEE Communications Magazine* 40, 1 (2002), 132–139. https://doi.org/10.1109/35.978060

[51] Danhua Peng, Tom Warnke, Fiete Haack, and Adelinde M Uhrmacher. 2017. Reusing simulation experiment specifications in developing models by successive composition — a case study of the Wnt/β-catenin signaling pathway. *SIMULATION* 93, 8 (2017), 659–677. https://doi.org/10.1177/0037549717704314

[52] R Core Team. 2019. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. https://www.R-project.org/

[53] Ranjit Randhawa, Cliff Shaffer, and John Tyson. 2010. Model Composition for Macromolecular Regulatory Networks. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 7, 2 (2010), 278–287. https://doi.org/10.1109/TCBB.2008.64

[54] Oliver Reinhardt, Tom Warnke, Andreas Ruscheinski, and Adelinde M. Uhrmacher. 2019. *Valid and Reproducible Simulation Studies—Making It Explicit*. Springer International Publishing, Cham, 607–672. https://doi.org/10.1007/978-3-319-70766-2_25

[55] Oliver Reinhardt, Tom Warnke, and Adelinde M. Uhrmacher. 2022. Agent-Based Modelling and Simulation with Domain-Specific Languages. In *Towards Bayesian Model-Based Demography: Agency, Complexity and Uncertainty in Migration Studies*. Methodos Series, Vol. 17. Springer, Dordecht, Chapter 7. https://doi.org/10.1007/978-3-030-83039-7_7

[56] Stewart Robinson, Gilbert Arbez, Louis G. Birta, Andreas Tolk, and Gerd Wagner. 2015. Conceptual modeling: definition, purpose and benefits. In *2015 Winter Simulation Conference (WSC)*. 2812–2826. https://doi.org/10.1109/WSC.2015.7408386

[57] Andreas Ruscheinski and Adelinde M. Uhrmacher. 2017. Provenance in modeling and simulation studies — bridging gaps. In *2017 Winter Simulation Conference (WSC)*. 872–883. https://doi.org/10.1109/WSC.2017.8247839

[58] Andreas Ruscheinski, Tom Warnke, and Adelinde M. Uhrmacher. 2019. Artifact-based workflows for supporting simulation studies. *IEEE Transactions on Knowledge and Data Engineering* 32, 6 (2019), 1064–1078. https://doi.org/10.1109/TKDE.2019.2899840

[59] Andreas Ruscheinski, Pia Wilsdorf, Marcus Dombrowsky, and Adelinde M. Uhrmacher. 2019. Capturing and reporting provenance information of simulation studies based on an artifact-Based workflow approach. In *Proceedings of the 2019 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation (SIGSIM-PADS '19)*. 185–196. https://doi.org/10.1145/3316480.3325514

[60] Susan M. Sanchez, Paul J. Sanchez, and Hong Wan. 2020. Work smarter, not harder: a tutorial on designing and conducting simulation experiments. In *2020 Winter Simulation Conference (WSC)*. 1128–1142. https://doi.org/10.1109/WSC48552.2020.9384057

[61] R G Sargent. 2013. Verification and validation of simulation models. *Journal of Simulation* 7, 1 (2013), 12–24. https://doi.org/10.1057/jos.2012.20

[62] Martin Scharm, Olaf Wolkenhauer, and Dagmar Waltemath. 2015. An algorithm to detect and communicate the differences in computational models describing biological systems. *Bioinformatics* 32, 4 (2015), 563–570. https://doi.org/10.1093/bioinformatics/btv484

[63] Falk Schreiber, Björn Sommer, Tobias Czauderna, Martin Golebiewski, Thomas E. Gorochowski, Michael Hucka, Sarah M. Keating, Matthias König, Chris Myers, David Nickerson, and Dagmar Waltemath. 2020. Specifications of standards in systems and synthetic biology: status and developments in 2020. *Journal of Integrative Bioinformatics* 17, 2-3 (2020), 20200022. https://doi.org/doi:10.1515/jib-2020-0022

[64] Max Schröder, Susanne Stählke, J. Barbara Nebe, and Frank Krüger. 2020. Towards in-situ knowledge acquisition for research data provenance from electronic lab notebooks. In *DaMaLOS - First Workshop on Data and Research Objects Management for Linked Open Science, Co-located at the International Semantic Web Conference ISWC 2020, Virtual Conference, November 2-3, 2020*. 1–12. https://doi.org/10.4126/FRL01-006423288

[65] Gregory A Silver, John A Miller, Maria Hybinette, Gregory Baramidze, and William S York. 2011. DeMO: an ontology for discrete-event modeling and simulation. *SIMULATION* 87, 9 (2011), 747–773. https://doi.org/10.1177/0037549710386843

[66] Alejandro Teran-Somohano, Alice E. Smith, Joseph Ledet, Levent Yilmaz, and Halit Oğuztüzün. 2015. A model-driven engineering approach to simulation experiment design and execution. In *2015 Winter Simulation Conference (WSC)*. 2632–2643. https://doi.org/10.1109/WSC.2015.7408371

[67] The Gene Ontology Consortium. 2018. The gene ontology resource: 20 years and still GOing strong. *Nucleic Acids Research* 47, D1 (2018), D330–D338. https://doi.org/10.1093/nar/gky1055

[68] Krishna Tiwari, Sarubini Kananathan, Matthew G Roberts, Johannes P Meyer, Mohammad Umer Sharif Shohan, Ashley Xavier, Matthieu Maire, Ahmad Zyoud, Jinghao Men, Szeyi Ng, Tung V N Nguyen, Mihai Glont, et al. 2021. Reproducibility in systems biology modelling. *Molecular Systems Biology* 17, 2 (2021), e9982. https://doi.org/10.15252/msb.20209982

[69] Dai Hai Ton That, Gabriel Fils, Zhihao Yuan, and Tanu Malik. 2017. Sciunits: Reusable Research Objects. In *2017 IEEE 13th International Conference on e-Science (e-Science)*. 374–383. https://doi.org/10.1109/eScience.2017.51

[70] T.G. Trucano, L.P. Swiler, T. Igusa, W.L. Oberkampf, and M. Pilch. 2006. Calibration, validation, and sensitivity analysis: what's what. *Reliability Engineering & System Safety* 91, 10 (2006), 1331–1357. https://doi.org/10.1016/j.ress.2005.11.031

[71] Dagmar Waltemath, Richard Adams, Daniel A. Beard, Frank T. Bergmann, Upinder S. Bhalla, Randall Britten, Vijayalakshmi Chelliah, Michael T. Cooling, Jonathan Cooper, Edmund J. Crampin, Alan Garny, Stefan Hoops, et al. 2011. Minimum Information About a Simulation Experiment (MIASE). *PLOS Computational Biology* 7, 4 (04 2011), 1–4. https://doi.org/10.1371/journal.pcbi.1001122

[72] Dagmar Waltemath, Richard Adams, Frank T Bergmann, Michael Hucka, Fedor Kolpakov, Andrew K Miller, Ion I Moraru, David Nickerson, Sven Sahle, Jacky L Snoep, et al. 2011. Reproducible computational biology experiments with SED-ML-the simulation experiment description markup language. *BMC Systems Biology* 5, 1 (2011), 198. https://doi.org/10.1186/1752-0509-5-198

[73] Mark D. Wilkinson, Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, Jan-Willem Boiten, Luiz Bonino da Silva Santos, Philip E. Bourne, Jildau Bouwman, Anthony J. Brookes, et al. 2016. The FAIR Guiding Principles for scientific data management and stewardship. *Scientific Data* 3, 1 (2016), 160018. https://doi.org/10.1038/sdata.2016.18

[74] Pia Wilsdorf, Jakob Heller, Kai Budde, Julius Zimmermann, Tom Warnke, Christian Haubelt, Dirk Timmermann, Ursula van Rienen, and Adelinde M. Uhrmacher. 2021. A Model-Driven Approach for Conducting Simulation Experiments. *TechRxiv* (2021).

[75] Levent Yilmaz, Sritika Chakladar, and Kyle Doud. 2016. The goal-hypothesis-experiment framework: a generative cognitive domain architecture for simulation experiment management. In *2016 Winter Simulation Conference (WSC)*. 1001–1012. https://doi.org/10.1109/WSC.2016.7822160

[76] S. Yoo and M. Harman. 2012. Regression testing minimization, selection and prioritization: a survey. *Software Testing, Verification and Reliability* 22, 2 (2012), 67–120. https://doi.org/10.1002/stvr.430