THE EXPERIMENT MODEL AND VALIDITY FRAME IN M&S

Joachim Denil
University of Antwerp
Flanders Make, Belgium
Joachim.Denil@uantwerpen.be

Stefan Klikovits
Université de Genève
CERN, Switzerland
Stefan.Klikovits@unige.ch

Pieter J. Mosterman
MathWorks, USA
Pieter.Mosterman@mathworks.com

Antonio Vallecillo
Universidad de Málaga, Spain
av@lcc.uma.es

Hans Vangheluwe
University of Antwerp & Flanders Make, Belgium
McGill University , Canada
hv@cs.mcgill.ca

ABSTRACT

Modelling and Simulation approaches use system models to conduct simulation experiments. Experimental frames have been applied in this context to formally define a system's context. During the creation of an experimental frame for a simple spring model it becomes clear that experimental frames in their current definition lack certain properties and omit relevant information. Our approach describes the process of capturing the context of models and systems to provide truly reproducible experiment descriptions. The information, captured as *experimental setups*, can then be used for different purposes and in different scenarios, in particular for checking the validity of a model, the discovery of suitable models for the design of a system, and for calibrating models.

Keywords: Cyber-Physical Systems (CPS), Multi-paradigm Modelling (MPM), Model-Based Systems Engineering (MBSE)

1 INTRODUCTION

In the Modelling and Simulation (M&S) community it is widely accepted that the validity of a model depends on its purpose and the context that it is used in. Depending on the application and the requirements, a model that is valid for one use case can produce invalid results for another. A well-known example for this behaviour can be found in gravity theory: While Newton's law of gravity is perfectly applicable to many systems, for certain applications (such as inter-stellar movements) it will provide incorrect predictions and the more precise—but also more complex—General Theory of Relativity must be used.

Zeigler explained in (Zeigler et al. 2000) how the context of a *source system* can be defined by its experimental frame (EF), which specifies the conditions under which the system is observed and experimented with. This leads to the conclusion that a valid model should also adhere to the EF of the system. Using a formally well-defined EF this adherence property can be validated.

SpringSim-TMS/DEVS, 2017 April 23-26, Virginia Beach, VA, USA;

Our research focusses on analysing four different use cases surrounding the concept of EFs and their requirements.

- One scenario is the above mentioned test for the *validity* of a model. The requirements for EFs that allow for easy substitution of models are defined based on their interactions with the context.
- Another purpose of EFs is the discovery of suitable models for the *design* of a system. Often it is a burdensome process to find a component that matches a set of properties and constraints. The task of the modeler can be eased by letting them define an EF that captures the required properties. The frame can then be automatically used to perform experiments on a library of models, revealing the ones that prove to be valid.
- One of the most important uses of EFs is the creation and *calibration* of a model. In this case the frame is first employed to capture the properties of the source system. As a next step it is used to iteratively refine the model until it matches the requirements defined by the EF.
- The last use case is the definition of an EF to formally define the experimental process. This *reproducibility* allows the reproduction of experiments and sustains trustworthiness in the model, the system and the entire setup.

Each of these use cases puts forward different requirements that render an implementation as one single frame impossible. In this paper we show that the application of the current state-of-art to a simple example fails to capture all of the aforementioned properties. Therefore, we split up the concept of an experimental frame into several frames for these different properties. Furthermore, we describe the elements of a framework to implement such frames. Finally, we look at one individual frame that captures the validity of a model.

This paper is structured as follows. Section 2 reviews the concept and current understanding of experimental frames. In Section 3 an introductory example displays the drawbacks that were encountered while defining EFs and also describes the new concept of experimental setup (ES). Section 4 displays the versatility of experimental setups (ES) using several application use cases. Section 5 describes one of these use cases, the validity frame, in detail. Section 6 elaborates on applications of such validity frames. Section 7 analyses related work and Section 8 concludes.

2 EXPERIMENTAL FRAMES

Bernard Zeigler defined the experimental frame in the System-Model-Simulator view as the environment that surrounds the source system (Zeigler et al. 2000). This EF is the "operational formulation of the objectives that motivate a modeling and simulation project". It consists of three components, namely a generator, an acceptor and a transducer. They are responsible for generating inputs, observing outputs and analysing outputs, respectively.

Formally defined, an EF can be represented by its time base T, input variables I, run control variables C, set of output variables O, set of input event segments Ω_I (i.e. a description of the input over a time period), set of control event segments Ω_C, and the summary mapping SU of all IO (Input/Output) pairs observed within the frame (Zeigler 1984). Within this structure a model or its respective source system can be executed. While execution in the former case (*acceptance mode*) can be used to *validate* a model (using an appropriate simulator), the latter case (*generation mode*) produces original output that can be used for the creation and *calibration* of a model.

Traoré and Muzy extended Zeigler's concept by recognising the duality between these two modes (Traoré and Muzy 2006). Their work shows the versatility of EFs by listing various ways in which they can be

viewed and used. Based on their findings, they suggest to generalise EFs to *frame*s, which describe the context a system or model is embedded in. Their research displays the similarity between the relationships of systems and their contexts, models and their frames, and simulators and their experimenters.

They continue by giving a guideline for the definition of such a frame following the Discrete Event System Specification (DEVS) formalism's specification hierarchy. This leads them to identify three structures, each being an extension of the underlying, previous one. One of their contributions is to specify the interfaces between the individual frame components and the model, additionally to the specification of the EF.

As a first step the frame interface (FI) is described through the time base and the input and output variables to the frame and the model. The FI is then extended by the frame behaviour (FB), adding information about the valid segments for model inputs, frame inputs and frame controls, and a mapping between the control inputs and the IO pairs. Lastly, the structure is extended to a frame system (FS) by adding information about the individual components of the frame, including the interfaces and couplings.

This extension leads to the final frame system being defined as the structure

$$\begin{aligned} FS = &\langle T, I_M, I_E, O_M, O_E, \Omega_M, \Omega_E, \Omega_C, D, \{C_d, d \in D\}, \\ &CPIC, EICC, POCC, CEOC, CCC\rangle \end{aligned} \tag{1}$$

where T is a time base; I_M the (set of) input variables of the plug-in (the model inside the FS); I_E the input variables to the frame (controls); O_M the output variables of the plug-in; O_E the output variables of the frame (summary set); Ω_M the admissible input segments for I_M; Ω_E the admissible input segments for I_E; Ω_C the admissible output segments for O_M; D the component names; $C_d, d \in D$ the models for each component D; $CPIC$ Control-to-Plug-in-Input coupling; $EICC$ External-Input-to-Control coupling; $POCC$ Plug-in-Output-to-Control coupling; $CEOC$ Control-to-External-Output coupling, and CCC Control-to-Control coupling.

We refer to the original publication (Traoré and Muzy 2006) for a detailed explanation of this structure and its correspondence to the DEVS specification hierarchy.

3 MOTIVATING EXAMPLE

In this section we describe, apply and evaluate the experimental frame for a spring. The spring is a well known mechanical component that has real-world realisations and component catalogues that make its properties explicit.

3.1 The Spring Model

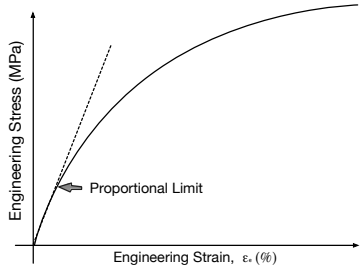

Figure 1: Stress-Strain curve of a material

A spring is a mechanical object that stores mechanical energy. The spring exerts an opposing force from its original resting position when it is extended or compressed. This restoration force is proportional to the extension or compression of the spring. Hooke's Law, $F = -kx$, where F is the force, x is the length of compression or extension and k is the spring constant, is a well-known model used to describe this behaviour.

Hooke's Law is applicable for many elastic deformation scenarios but only holds under certain loading conditions in most engineering applications. Figure 1 shows the stress-strain curve of a spring (Roy-

lance 2001). As can be seen, Hooke's Law is only applicable in the region before the proportional limit point.

3.2 Defining an Experimental Frame

Using Zeigler's approach and building on the advances made by Traoré and Muzy, we specify an experimental frame for the spring and its model. The complete experiment/simulation consists of a spring with a spring constant k, which is placed between the ground and a mass m. Additionally, a force source (e.g. gravity) works on the mass, exerting a pull force F onto the spring-mass system. This leads the mass to be displaced, which is the output parameter x of the system. Both m and the ground are not part of the system, as we aim to model the spring by itself. Figure 2 depicts the schema of this setup.

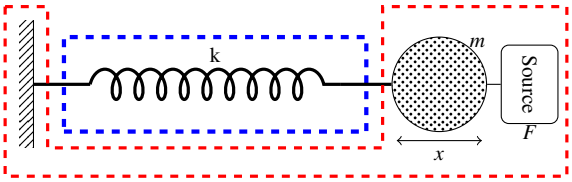

Figure 2: Structure of the Experimental Setup for the Spring model/system.

Based upon the information described in the previous section, we define a frame system (FS) for our spring as follows:

T	$[0, \texttt{MaxT}], T \in \mathbb{R}$,
I_M	$\{F\}, F \in \mathbb{R}$, the force,
I_E	$\{\texttt{start/stop}\}$,
O_M	$\{x\}, x \in \mathbb{R}$, the displacement in metres,
O_E	$\{execution\ time,$ $model\ validity,$ $calibration\ error\}$
Ω_M	$\{(F,t)/t \in \langle t_{init}, t_{end}\rangle\}$, the valid input F at time t
Ω_E	$\{(start, t_{init}), (stop, t_{end})\}$
Ω_C	$\{error < \epsilon\}$, ϵ is the error threshold

D	$\{Gen,\ Trans,\ Acc\}$
C_d	the model of each $d \in D$
$CPIC$	$\{((Trans, data), (EF, data))\}$
$EICC$	$\{((EF, start/stop), (Gen, start/stop))\}$
$POCC$	$\{((EF, displacement), (EF, displacement))\}$
$CEOC$	$\{((Trans, exectime), (EF, exectime)),$ $((Trans, error), (EF, error)),$ $((Acc, yes/no), (EF, yes/no))\}$
CCC	$\{((Gen, data), (Trans, data)),$ $((Trans, exectime), (Acc, exectime)),$ $((Trans, error), (Acc, error))\}$

This is the specification of an EF as described in (Traoré and Muzy 2006). Note that in our understanding it is not possible to define the structural aspect of the mass m in the behavioural description of the experimental frame. We believe, using standard EFs, it would be best defined as part of the Gen component' model. It defines the input and output variables, valid input segments for each variable and the structure of the generator, acceptor, transducer and the model (plug-in).

Our FS consists of three components, generator, transducer and acceptor (*Gen, Trans, Acc*), that operate in a time base *T*. The components are connected via the couplings *CPIC, EICC, POCC, CEOC, CCC*. Each coupling is described by the two ports and the type of information passed. For example, *((Trans, data), (EF,data))* states that *Trans*'s *data* port is connected to *EF*'s *data* port. The system is executed via inputs to the components I_M, I_E whose input validity is validated by the acceptor according to the information specified in Ω_M, Ω_E. The outputs O_M, O_E are read from the system. The acceptor further measures Ω_C for validity.

3.3 Drawbacks of Experimental Frames

After analysing the spring example we identified five main properties that have to be supported and asserted by the EF: 1. validity, 2. substitutability, 3. compositionality, 4. comparability, and 5. reproducibility.

The first property (*validity*) specifies that a model has to be valid. This means that it faithfully represents the source system in its given context. This context is encoded in the EF and can be used for two distinct verification purposes: a) The EF is based on source system observations, a match of the model outputs validates the model; b) the EF is used to create a context for the source system, a comparison of a valid model to the source system validates the EF.
Substitutability is a collection of several properties of an EF. It asserts that two models which are both valid in an EF can be exchanged without a change in functionality or correctness. It also means that a model can be placed into a different EF, given that the new EF is a subtype of the original EF.
Compositionality demands that frames can be connected in the same way their embedded models are. Reasoning about this model should be independent of whether it is looked at as an individual system or as a part of a larger composition. The connection of such EFs can be seen as a description of the composition, assembled of the components' descriptions.
Comparability states that an EF is a black-box description of a model's functionality. This property allows system designers to start specifying their requirements as an EF and subsequently choose a model from a catalogue that contains models and their respective EFs. By comparing the catalogue EFs to the designer's "blueprint" the search can be narrowed down to appropriate candidate models, which will then be executed for comparison. This leads to a customised benchmark, showing each model's preciseness, performance and resource requirements.
Lastly, an EF introduces deterministic *reproducibility*. By formally specifying a source system's behaviour, the EF should precisely capture all information that is required to reproduce the source system experiment. Further, a model can be simulated repeatedly in this context to test its validity and execution properties.

Trying to construct an EF for the spring setup that fulfils all above requirements, we observed that some vital information is missing. Furthermore, while the frame concept is very broad, implementing a tool that supports all of the uses of an experimental frame seems infeasible. We therefore split the the concept of an experimental frame into several individual frames for different purposes. Note that, e.g. composition, comparison and substitution of simulations have been proven to be NP-hard (Page and Opper 1999). Though this does not limit the applicability of the framework itself, highly composed systems may introduce complexity issues.

4 PURPOSE-DRIVEN FRAMES

EFs, as defined by Zeigler, Traoré and Muzy, omit some of the information required to fully reproduce experiments. In our example this missing information includes the following data:

- The model/system behaviour is governed by constants `restLength` and `k`, which represent the length of the spring at which it does not exert a force and its spring constant, respectively.
- the initial condition (IC) condition of the plug-in (i.e. the initial length of the spring $x(0)$),
- the frame has just one parameter, `m`, representing the mass of the object,
- the frame does not have ICs, hence $\varnothing$,
- the time span of the experiment (`MaxT`),
- the valid range of the forces (`[MinF,MaxF]`),
- the ICs and parameters of the solver(s). Note that (Rozenblit 1985) shows the specification of ICs using control segments. We see these concepts as fundamentally different and hence should be treated as such.

A frame system should store all required information to fully validate, reproduce and compare the contexts of models and source systems. We extend EFs to Experimental Setups (ES) for overcoming this problem. Our approach is process-driven and uses the activity diagram formalism to visualise the creation of an ES that satisfies the required properties.

For the purpose of validity and reproducibility, the EF does not define information about the individual parts' detailed structure, setups, solvers, ICs, and parameters. This means that the EF does not reveal the parameters of the model under test, the initial conditions of the frame, nor the configuration of the solver(s). Furthermore, a model of a spring can only exhibit spring-like characteristics when there is a resultant force being applied to it. In an experimental context this can be done by fixing the spring on one side and applying a displacement of a mass on the other side. This has to be captured in the frame as well. The next sections will introduce the *validity frame* concept descriptions for the creation and the validity assertion of models and frames.

As described in Section 3.3, we wish for our model-frame combinations to be easily substituted and compared. Finding an appropriate model can be done by executing a model within the *design frame*, or by comparing both models' frames. In many cases, the replacement can be a refinement of another model, leading to a type of sub-type relationship, as defined by Liskov's substitution principle (Liskov and Wing 1994). *Combination of frames* is needed for model compositions. This means that if the model/system itself is an assembly of several models, the frame for this assembly should also be a combination of the parts' frames. This is an important property, as in many applications the model is actually a composition of various submodels. The combination of frames is closely related to the concept of contracts in systems theory. We similarly require our frames to follow the concepts of composition, conjunction, refinement and abstractions, as described for general systems in (Benveniste et al. 2012).

Another possible application is to use the ES in form of a *property checking setup*. This application provides the opportunity to test individual properties of a model to verify its appropriateness. Property setups are similar to validation setups, their intent however lies in testing individual properties rather than the entire model/system. In the rest of the paper, we dive deeper into the *validity frame* which supports the validity and reproducibility of a model and the experiments on that model, respectively.

5 THE VALIDITY FRAME

Our validity frame defines the experimental context of a model in which that model gives predictable results. For example, a model of a spring is valid before the physical spring its proportionality point. After this point, the spring will start to plastically deform and the results of the model are no longer valid.

5.1 Experimentation, Modelling and Validity

Experimentation defines the experimental system as the object under study, the experimentation apparatus and their interactions. When experimenting, the experimenter actively intervenes in the environment of this system under study to produce new objects, phenomena, substances and processes (Radder 2009). However, the scientific method does not categorise all active interventions as experiments. Reproducibility of the experiment is needed to be classified as such. Karl Popper acknowledged this and instructs scientists to replicate their own experiments and to be able to instruct other scientists to repeat the experiment with similar results (Popper 2005). This reproducibility of experimental results is a precondition to construct and test scientific theories. Radder further elaborates on the reproducibility of scientific experimentation. He distinguishes the material realisation of the experiment from the theoretical description of the experiment. The theoretical description consists of: (a) the preparation of the object and experimental equipment; (b) the staging of the processes of interaction and detection; and (c) the closedness of the experiment. The

material realisation is defined as the experimental actions that are carried out by the experimenter as per instructions of the initial experiment definition. The language used to communicate between the reproducing experimenter and the initial experimenter has to be a common language (Radder 1996).

This reproducibility of the experiment and its results must occur both in the physical as well as in the computational world. Figure 3 shows a commuting diagram about experimentation in both the real-world as in the computational world. The results obtained by doing the experiment in the real-world have to commute with results obtained from doing a completely virtual experiment. These commuting diagrams can further be extended with intermediate experiments. For example, a hardware-in-the-loop-like setup can use the model of the system in the real context of the world. The results between the complete physical experiment and the hardware-in-the-loop experiment should also commute. The commuting diagram shows two paths (a and b) that have to produce similar results. Therefore, the results of both the real-world experiment and the virtual experiment have to be compared to each other. Quantitative comparison requires a distance metric and a tolerance for measuring the goodness-of-fit (Zeigler et al. 2000). In the real world, a set of abstractions have been chosen to conduct the experiment. For example, in the case of our spring, we measure the displacement of the spring. However, we do not measure the increase or decrease of the spring temperature due to our actions in the spring environment or the temperature of the environment when conducting the physical experiment. The experimental context models this both in the real-world as in the virtual-world. Other experiments could include measures of this temperature fluctuations. Multiple contexts can thus exist and a context can be applied to many models (n-to-m relation) as already pointed out by Zeigler (Zeigler et al. 2000).

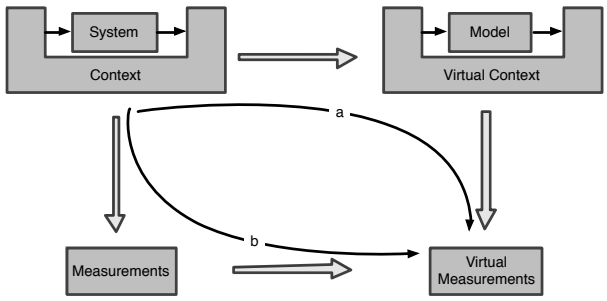

Figure 3: commuting diagram of validity of experiments and virtual experiments

5.2 Experiment Model

An experiment model is the model of the material realisation of the experiment. It is thus the process that an experimenter has to follow. Note that these experiment models can become very complex. For example, to set an initial condition of our model, we first need to run several other experiments to obtain steady state values for the initial position of our spring.

To model this process of setting up the experiment correctly we use UML 2 activity diagrams (Object Management Group 2015). Examples of activities that relate to experiment model are the setting of the initial conditions of our spring model and defining the spring's *k*-value.

5.3 The Experimental Setup

The experimental setup (ES) provides the theoretical description of an experiment. Our ES extends the experimental frame defined by Zeigler, and Traoré and Muzy. The concepts of an EF are incorporated into our ES and extended where needed.

Figure 4 displays a diagram showing the individual parts of the ES. The ES describes the context of the system and the conditions of the experiment. Moreover, in case of simulations it interacts with a (set of) *solver(s)* that provide the execution platform for the simulations. There might be more than one solver

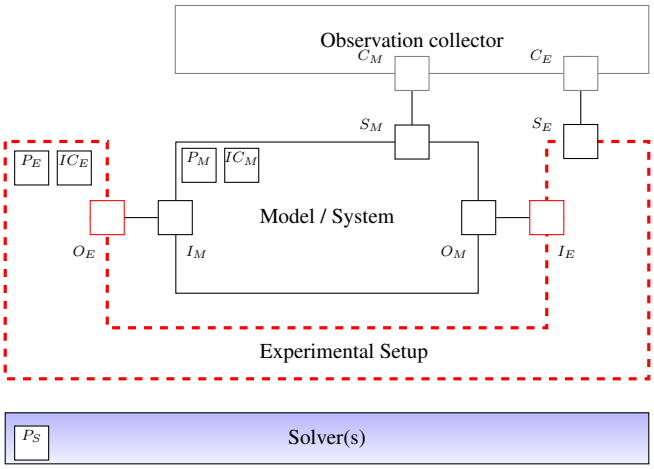

Figure 4: Revised structure of an Experimental Framework.

in situations where the plug-in and the ES-components require different configurations. Examples of this setup occur when co-simulation is used to orchestrate a combined simulation of different subsystems. The model/system, the ES and the solver(s) together form an independent system, in the sense that no external inputs are required to run the system or to simulate the model.

The *observation collector* is an element that is required in many executions to verify inner states or signals of the plug-in and ES. Note that while the system/model and the ES are treated as black-boxes in order to respect the required encapsulation, it is common to introduce (artificial) probes or controlling elements to measure/monitor/control some internal properties that are relevant to the experiment. These are usually removed in production mode but should be reflected in an ES if the plug-in is indeed used for experimentation. The observation component also takes care of the metric and tolerance that is present when validating a model to the real-world system. The observation collector lies outside the EF as an orthogonal concept to the frame as defined by Zeigler. Depending on the application, it will also observe model-internal values which's ports are not be exposed to the surrounding frame.

All interactions among these elements are modelled by means of interfaces. In addition, both the system/-model and the experimental setup usually have associated parameters and initial conditions, which have to be reflected in the experimental setup. Similarly, solvers may have associated parameters that need to be specified for conducting the experiments.

The interfaces in Figure 4 are acceptors for the following values:

- C_M and C_E are the types of the signal values that the *observation collector* reads from the model and the frame, respectively, in order to observe the experiment. In many occasions they coincide with the types of the corresponding output signal values of the model and the frame, specified by S_M and S_E, respectively.
- P_M and P_E represent the parameter types of the model and the frame.
- IC_M and IC_E represent the types of the initial conditions of the model and the frame.
- Finally, P_S represents the parameter types of the solver(s) used in the experiment.

6 APPLICATIONS OF THE VALIDITY FRAME

While the Validity frame does not change, its usage does so significantly in different scenarios.

Hereafter we look at two important uses of this validity frame. Both uses occur multiple times in an iterative design process. First, a *validation frame* takes a model and tests its validity. This solution can be used to assure that a model faithfully represents the original system. This scenario is especially of importance when comparing several models with different abstraction levels to find the one with lowest complexity but still produces results with enough precision. Second, a *calibration frame* takes empirical results from real-world physical experiments and tries to calibrate the model with this data. It thus results in a (set of) valid experimental setups that can be used for other purposes. Other scenarios are also possible but not further specified in this paper, e.g., we have a model that we want to apply this model in an unknown context. New experiments need to be created and modelled in the real-world and in the computational world. Another application might be the experimentations for exploring the sensitivity of the different parameters in the model.

6.1 The Validity Frame for Validity of Models

In the first case, we use the validity frame for the modelling and validation of an experiment. The process is described in Figure 5. Each step produces a set of intermediate artefacts, containing relevant information about the experiment. That information will serve as inputs to subsequent steps of the process, and also to precisely document the experiment. Thus, the information contained in the collection of produced artefacts will permit both the repeatability of the experiment, since it has all the details to faithfully repeat it, and its reproducibility, given that the final results are also recorded. The process starts by defining the model and its interfaces, i.e. the tuple $\langle T, I_M, O_M, \Omega_I, \Omega_O, R_{IO} \rangle$, as defined in (Traoré and Muzy 2006). Then, in step 2 the experimental setup needs to be specified including its internal structure, inner components, observation collector (if needed) and their interfaces $\langle O_E, I_E, S_M, S_E, C_M, C_E \rangle$ (see Figure 4). Step 3 is in charge of determining the solvers that will be used to simulate the experiment. Then, the parameters, initial conditions and constants for all elements are determined in step 4, and the solvers configured in step 5. Finally, the simulation is run and the results are analysed. Depending on the values of the results obtained, the experimenter may decide to revisit any of the previous steps of the process to readjust some parameters or reconsider any of the decisions made (e.g. replace the solvers, change the ES configuration, etc.).

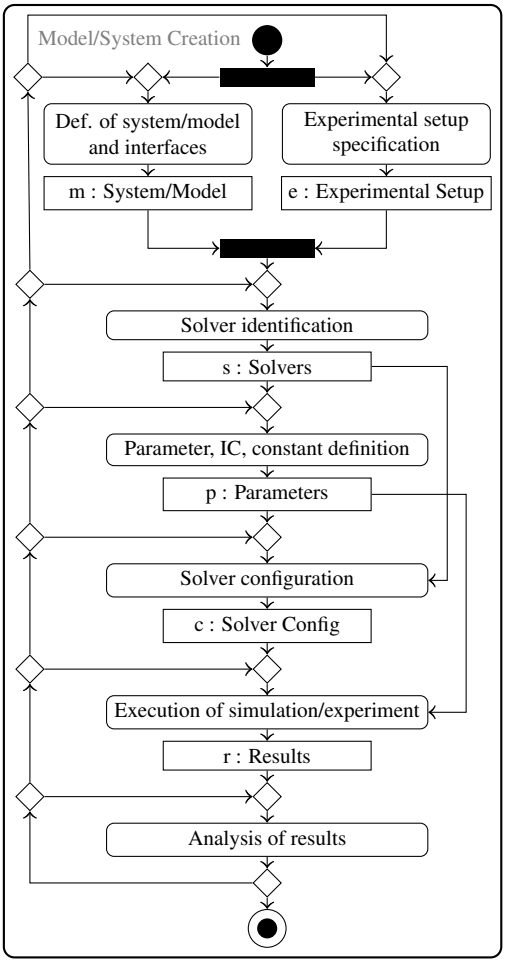

Figure 5: Activity diagram describing the experiment.

6.2 Validity Frame for Calibration of Models

When using the validity frame for calibration, we try to configure the parameters and initial conditions of the experimental setup so that the model and its configuration faithfully represent the source system behaviour. Starting from a model of a system, its parameters are adjusted so that its behaviour matches the source

Table 1: Experimental spring results, with mass *m* in kg and displacement *x* (± 0.0001) in cm .

m	x	m	x	m	x	m	x	m	x
1	2.100	3	6.3749	5	10.4915	7	14.6081	9	19.0012
2	4.3166	4	8.4332	6	12.5489	8	16.7774		

system of interest. Figure 6 shows the activity diagram describing the process in this case. The process generates a (set of) experiment setups as output.

The input of a calibration frame is the experimental data of the source system and a model of the system. After the initial setup of combining the model of the system with an experimental setup and selecting an appropriate solver, an initial guess of the parameters, initial conditions, etc. is needed. Finally the independent system is executed and the results are analysed, that is compared to the results of the experimental data of the source system.

We apply the process of calibration on our motivating example. Therefore, we couple the spring model to a mechanical ground and an ideal force source to models a disturbance in the environment of the system under study. The force source will, depending on the amount of created force, extend the spring. An equal experiment has been done in the real-world, where we attach different masses to our spring under study. Depending on the mass, a different force is generated under the influence of gravity. The experimental results of the hypothetical study are shown in Table 1. In our example we are looking to calibrate the *k*-value of the spring model based on the values of a real-world experiment. However, because of measurement errors in the real-world (calliper scale, mass uncertainty, etc.), and because of the approximation and abstraction mechanisms used in the modelling world, we need to take both a distance metric and tolerance into account. In our case, the observation collector calculates a sum of squares error between the virtual experiment and the real-world experiment.

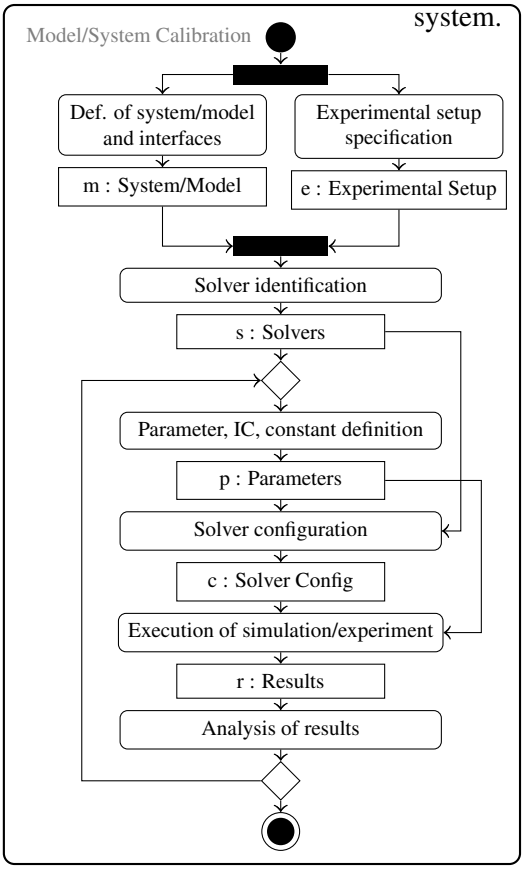

Figure 6: Activity diagram describing how a model is calibrated.

We made an implementation of the activity diagram of Figure 6 using a MATLAB script (Klikovits et al. 2017). In our calibration process, we use a linear regression to estimate the k-value of the spring. For each of the real-world experimentation points, we execute a simulation. A possible experimental setup specification for the spring in Table 1 is:

$r^2 = 0.9998413$ | $k = 4.759093$ | $\Omega_{in} = [1, 9]$ (force) | $\Omega_{out} = [2.101241, 18.91116]$ (displacement)

7 RELATED WORK

Zeigler's approach has been used in several works by Ören, where he studies advanced simulation methodologies (Ören and Zeigler 1979) and the acceptability of simulations (Ören 1981). Pegden describes the use of EFs in SIMAN to model manufacturing systems (Pegden 1984). Schultz et. al. describe the usage of a catalogue of EFs which they call "experimental frame base" in (Schulz et al. 1998). (Spiegel et al. 2005) document the non-trivial challenge of re-using EFs due to the complexity of their constraints. Recently (Schmidt et al. 2016) showed how they use EFs for model-based testing of MATLAB/Simulink models.

(Ihrig 2016) describes a very similar approach to simulation and modelling to ours, but does not show its application. Similarly, (Gore and Diallo 2013) report on the validation and verification of MS-SDF, as presented in (Tolk et al. 2013). None of these works, however, characterize the current limitations of EFs in their expected contexts of use, nor provide solutions to overcome these limitations, as we do in this paper.

In several scientific fields, there are examples of languages to allow reproducibility of computational experiments, e.g. in (Rahmandad and Sterman 2012). These approaches differ from our approach as they focus on the computational aspects of reproducibility. Ehwald and Uhrmacher go a step further by also including other aspects of scientific experimentation, e.g. the random number generators as inputs (similar to the generator in EF) (Ewald and Uhrmacher 2014). Our framework goes further and captures all necessary information for validity during the M&S process.

8 CONCLUSIONS AND FUTURE WORK

This paper investigates various usage scenarios of experimental frames. First an introductory example reflects on the current state of the definition, while showing drawbacks of these current systems. In a next step, we introduce the five requirements we demand of an experimental frame: 1. validity, 2. substitutability, 3. compositionality, 4. comparability and 5. reproducibility. The need for these properties is supported by the analysis of several use cases. Therefore we first introduce *experimental setups*, which serve as an extension, fulfilling these properties. Based on this concept we show the process of their creation and usage for model calibration and validity verification. We introduce further uses of experimental setups to point out their versatility and applicability, but also the data that is additionally required. As part of our future work, we plan to expand our analysis and describe further use cases to show the versatility of experimental setups and the processes described above.

REFERENCES

Benveniste, A., B. Caillaud, D. Nickovic, R. Passerone, J.-B. Raclet, P. Reinkemeier, A. Sangiovanni-Vincentelli, W. Damm, T. Henzinger, and K. G. Larsen. 2012, November. "Contracts for System Design". Research Report RR-8147, INRIA.

Ewald, R., and A. M. Uhrmacher. 2014. "SESSL: A domain-specific language for simulation experiments". *ACM Transactions on Modeling and Computer Simulation (TOMACS)* vol. 24 (2), pp. 11.

Gore, R., and S. Diallo. 2013. "The Need for Usable Formal Methods in Verification and Validation". In *Proceedings of the 2013 Winter Simulation Conference: Simulation: Making Decisions in a Complex World*, WSC '13, pp. 1257–1268. Piscataway, NJ, USA, IEEE Press.

Ihrig, M. 2016. *A New Research Architecture for the Simulation Era*, pp. 47–55. Cham, Springer International Publishing.

Klikovits, S., J. Denil, P. Mosterman, A. Vallecillo, and H. Vangheluwe. 2017. "ExperimentalSetup_SpringModel: Matlab Script". *Zenodo* (http://doi.org/10.5281/zenodo.292613).

Liskov, B. H., and J. M. Wing. 1994, November. "A Behavioral Notion of Subtyping". *ACM Trans. Program. Lang. Syst.* vol. 16 (6), pp. 1811–1841.

Object Management Group 2015, March. *Unified Modeling Language (UML) Specification. Version 2.5*. OMG Document formal/2015-03-01.

Ören, T. I. 1981. "Concepts and Criteria to Assess Acceptability of Simulation Studies: A Frame of Reference". *Communications of the ACM* vol. 24 (4), pp. 180–189.

Ören, T. I., and B. P. Zeigler. 1979. "Concepts for Advanced Simulation Methodologies". *Simulation* vol. 32 (3), pp. 69–82.

Page, E. H., and J. M. Opper. 1999. "Observations on the Complexity of Composable Simulation". In *Proceedings of the 31st Conference on Winter Simulation: Simulation—a Bridge to the Future - Volume 1*, WSC '99, pp. 553–560. New York, NY, USA, ACM.

Pegden, C. D. 1984. "Introduction to SIMAN". In *Proceedings of the 16th Conference on Winter Simulation*, pp. 34–41, IEEE Press.

Popper, K. 2005. *The logic of scientific discovery*. Routledge.

Radder, H. 1996. *In and about the world: Philosophical studies of science and technology*. SUNY Press.

Radder, H. 2009, October. "The philosophy of scientific experimentation: a review". *Automated Experimentation* vol. 1 (1), pp. 1–8.

Rahmandad, H., and J. D. Sterman. 2012. "Reporting guidelines for simulation-based research in social sciences". *System Dynamics Review* vol. 28 (4), pp. 396–411.

Roylance, D. 2001. "Stress-strain curves". Technical report, Massachusetts Institute of Technology study, Cambridge.

Rozenblit, J. 1985. *Experimental Frames For Distributed Simulation Architectures*. 2 ed, Volume 15, pp. 14–20. Soc for Computer Simulation.

Schmidt, A., U. Durak, and T. Pawletta. 2016, August. "Model-Based Testing Methodology Using System Entity Structures for MATLAB/Simulink Models". *SIMULATION* vol. 92 (8), pp. 729–746.

Schulz, S., J. W. Rozenblit, M. Mrva, and K. Buchenriede. 1998. "Model-Based Codesign". *Computer* vol. 31 (8), pp. 60–67.

Spiegel, M., P. F. R. Jr., and D. C. Brogan. 2005. "A case study of model context for simulation composability and reusability". In *Proceedings of the 37th Winter Simulation Conference, Orlando, FL, USA, December 4-7, 2005*, pp. 437–444.

Tolk, A., S. Y. Diallo, J. J. Padilla, and H. Herencia-Zapana. 2013. "Reference modelling in support of M&S—foundations and applications". *Journal of Simulation* vol. 7 (2), pp. 69–82.

Traoré, M. K., and A. Muzy. 2006. "Capturing the dual relationship between simulation models and their context". *Simulation Modelling Practice and Theory* vol. 14 (2), pp. 126–142.

Zeigler, B. P. 1984. *Multifaceted Modelling and Discrete Event Simulation*. London, Academic Press.

Zeigler, B. P., H. Praehofer, and T. G. Kim. 2000. *Theory of Modelling and Simulation, Integrating Discrete Event and Continuous Complex Dynamic Systems*. 2 ed. USA, Academic Press.

AUTHOR BIOGRAPHIES

JOACHIM DENIL is currently a post-doc at the university of Antwerp and is associated with the Flanders Make Research Centre.

STEFAN KLIKOVITS Stefan Klikovits is a PhD Student at the University of Geneva and CERN. He researches in the domains of Software Modeling, Verification and Testing.

PIETER J. MOSTERMAN is Chief Research Scientist and Director of the MathWorks Advanced Research & Technology Office .He also holds an Adjunct Professor position at McGill University.

ANTONIO VALLECILLO is Professor of Computer Science at the University of Málaga. His research interests include model-based software engineering, open distributed processing and software quality.

HANS VANGHELUWE is a Professor at the University of Antwerp / Flanders Make and an Adjunct Professor at McGill University.