Environmental Modelling & Software 93 (2017) 180–192

Contents lists available at ScienceDirect

Environmental Modelling & Software

journal homepage: www.elsevier.com/locate/envsoft

# The rocky road to extended simulation frameworks covering uncertainty, inversion, optimization and control[☆]

Daniel Wirtz [a, *], Wolfgang Nowak [b]

[a] Institute of Applied Mechanics (CE), University of Stuttgart, Pfaffenwaldring 5a, 70569, Stuttgart, Germany
[b] Institut für Wasser- und Umweltsystemmodellierung, University of Stuttgart, Pfaffenwaldring 61, 70569, Stuttgart, Germany

ARTICLE INFO

*Article history:*
Received 2 November 2015
Received in revised form
25 September 2016
Accepted 14 October 2016

*Keywords:*
Scientific computing
Extended software frameworks
No free lunch
Reproducibility
Software trust

ABSTRACT

In the past decades, simulation frameworks have greatly increased in complexity, due to coupling of models from various disciplines into so-called integrated models. Recently, the combination with tools for uncertainty quantification, inverse modelling, optimization and control started a development towards what we call *extended* simulation frameworks. While there is an ongoing discussion on quality assurance and reproducibility for simulation frameworks, we have not observed a similar discussion for the extended case. Particularly for extended frameworks, the need for quality assurance is high: The overwhelming range of options and algorithms is unmanageable by a domain expert and opaque to decision makers or the public. The resulting demand for 'intelligent software' with automated configuration can lead to a blind trust in simulation results even if they are incorrect. This is a threatening scenario due to potential consequences in simulation-based engineering or political decisions. In this paper, we analyze the increasing complexity of scientific computing workflows, and discuss the corresponding problems of extended scientific simulation frameworks. We propose a paradigm that regulates the allowable properties of framework components, supports the framework configuration for complex simulations, enforces automatic self-tests of configured frameworks, and communicates automated algorithm choices, potentially critical user settings or convergence issues with adaptive detail level and urgency to the end-user. Our goal is to start transferring the quality assurance discussion in the field of integrated modeling and conventional software frameworks to the area of extended simulation frameworks. With this, we hope to increase the reliability and transparency of (extended) frameworks, framework use and of the corresponding simulation results.



## 1. Introduction

Modeling of natural and technical processes or behavior has become an important part of both research and industry. Simulations have become a large part of everyone's lives, as already envisioned by Gallopoulos et al. (1994):

> "Some of the capabilities of future problem-solving environments seem like science fiction, but whatever form they eventually take, their scientific and economic impact will be enormous."

Simulations are used more and more as prognostic tools. They help designing new cities or traffic guidance systems, forecast the weather and support decisions for nuclear waste storage in deep geological formations. Thus, simulations can have a large influence in large-scale engineering projects or even in policy making.

Even science and engineering are experiencing a paradigm shift towards simulation-based approaches, as discussed by Oden et al. (2006); Glotzer et al. (2009); Gallopoulos et al. (1994), see also Section 2. For example, simulations in industry are used to improve experimentation and rapid prototyping in crash simulations (Mei and Thole, 2008; Flidrova et al., 2010; Ryckelynck et al., 2011). Sometimes, simulations in research are even believed to help in discovering new physical laws and natural phenomena (due to the complex interaction of already known laws), see King et al. (2009).

Often, the tools used for those simulations are so-called *simulation frameworks*, i.e. compiled suites of software designed to solve a huge class of problems. Such frameworks often claim true-to-life

☆ The authors would like to thank the German Research Foundation (DFG) for financial support of the project within the Cluster of Excellence in Simulation Technology (EXC 310/1) at the University of Stuttgart.

* Corresponding author.
*E-mail addresses:* daniel.wirtz@simtech.uni-stuttgart.de (D. Wirtz), wolfgang.nowak@iws.uni-stuttgart.de (W. Nowak).

http://dx.doi.org/10.1016/j.envsoft.2016.10.003
1364-8152/

simulation for complex multi-physics and/or multi-scale systems in a large variety of scientific fields at a highly automated level. As the conceptual models grow more and more complex, so do the simulation frameworks: COMSOL (COMSOL, 2012), DUNE (Ohlberger et al., 2012), DuMu$^x$ (Flemisch et al., 2011), espresso (Holm et al., 2012), FEniCS (The FEniCS project, 2012), HydroGeoSphere (Brunner and Simmons, 2012), FEFLOW (Trefry and Muffels, 2007), OpenGeoSys (Kolditz et al., 2012a), ParFlow (Kollet and Maxwell, 2008), to name just a few. Examples of individual modeling tools combined into coupled simulation frameworks can be found in the area of integrated (environmental) modeling (e.g., Laniak et al., 2013; Bierkens et al., 2014).

**However, the typical present-day simulation-based scientific workflow is generally more complex than merely running a series of simulation codes**. There are important additional components like uncertainty quantification, inverse modelling, optimization and control. The corresponding algorithms are beginning to be (or will sooner or later be) included in frameworks. Until now, tools for these additional workflow tasks are usually available as separate toolboxes. Examples are the sensitivity analysis, parameter estimation and uncertainty quantification toolboxes PEST, UCODE and DAKOTA (Doherty and Hunt, 2010; Hill et al., 2005; Eldred et al., 2007), the decision support framework MADS (Vesselinov et al., 2012), the data impact analysis tool PreDIA (Leube et al., 2012a) and the optimization toolbox OSTRICH (Matott, 2003). In a more general sense, the optimization and control toolboxes in programming environments such as MATLAB or SIMULINK (The MathWorks, 2014a, b) are readily available and cover most of these aspects.

Fusing these additional components with simulation frameworks leads to what we call *extended* simulation frameworks. In fact, one can already observe the first developments in that direction, for example the iTOUGH2 version of the TOUGH2 simulator for non-isothermal multiphase flow in fractured porous media (Finsterle, 2004), the combination of LS-Dyna with LS-Opt (Dynamore, 2014), the Python Multivariate Land Data Assimilation Framework DasPy (Han et al., 2015) or the extension of ModelWeb to "UncertWeb", where uncertainty information is communicated between chained components (Bastin et al., 2013). These examples demonstrate the increasing awareness of individual research communities of the need for overarching frameworks that cover more complex workflows around the basic (forward-) simulation task. In our study, we focus on extended simulation frameworks. These highly integrated frameworks, however, need to be distinguished from lean, made-for-purpose simulation and research tools, which are sometimes preferred depending on the user's environment and philosophy.

The above developments mark a growth in complexity of models, simulation-based workflows and corresponding software frameworks. Along with this increasing complexity, the probability of misconfiguration, failed convergence and other errors grows in particular for extended frameworks. If simulations are performed inaccurately or are applied wrongly in any simulation-assisted engineering, research or experimental setting, the following consequences are imminent:

1. Liability questions will arise when systems, political decisions or structures developed with the aid of simulations fail and lead to severe damage in material or even human life.
2. Conversely, economical, ecological and financial consequences can be faced if simulation results suggest over-conservative decisions with too many preventive measures.
3. In the context of translational science (Council, 2012) or post-normal science (e.g., Funtowicz and Ravetz, 1994), there is rapid and evolving feedback between science, engineering, decision making, policy making and long-range societal decisions. Clearly, political decisions based on false simulations would imply a loss of credibility. The social impact of simulations (and their uncertainties and errors) is a delicate topic which has just begun to be investigated, e.g. (Brugnach et al., 2007).

Clearly, to avoid these consequences, it is essential to ensure the right choice of models and quality of simulation results. This can be supported by providing standards for quality, transparency and reproducibility of simulation frameworks (Peng, 2011; Atmanspacher et al., 2014) as well as complete documentation of input data, simulation tools, simulations settings and logs, simulation results, and the workflows that produced and used them.

Additionally, a high coding standard needs to be ensured (Morris, 2008; Faulk et al., 2009; Kelly, 2007). Rigorous application of software engineering techniques (e.g., Freeman et al., 2004), quality management, quality-of-service concepts (Gil et al., 2007), code validation and code verification (Post and Votta, 2005a, b) and other paradigms stemming from computer science (e.g., Nejmeh, 1988; Knupp and Salari, 2003; Wilson, 2006; Panas et al., 2007; Campbell and Papapetrou, 2013) can help to alleviate those problems. At least, they can ensure the *algorithmic* correctness of the simulation results and support the transparency of the used algorithms and settings to the user. The above quality assurance mechanisms have been used, to a varying extent, for conventional frameworks and will have to be applied to extended frameworks even more rigorously.

Still, algorithmic correctness is insufficient to guarantee accurate results, because the framework may have been configured with inadequate algorithmic choices or parameter settings. In extended frameworks, the number of such choices grows combinatorially with the available components and algorithms. In this situation, the folkloric "no free lunch theorem" (Wolpert, 1996; Wolpert and Macready, 1997) applies: there will be no pre-configuration or fixed combination of algorithms that will work best over the large range of all possible combined modelling/simulation/optimization tasks. As there is no unique pre-configuration, the choice of algorithms to combine will necessarily have to be done by the user and/or in an intelligent automated fashion.

Clearly, this increased complexity can hardly be overseen by a single domain expert, who has to count on each component's reliability and on well-made algorithm choices. It is already being recognized that this leads to an unjustifiable blind trust in the software by users, although each of the possible algorithmic combinations named above contains multiple sources of error.

Thus, we strongly suggest that extended simulation frameworks must support the user at all phases of the simulation workflow. Also, they must prevent failure of the simulations due to mathematical, numerical, algorithmic or software technical reasons including automation of important adjustments or possibly dangerous user settings. Possible measures include proper documentation exceeding the scope of "class based" JavaDoc-style (Kramer, 1999) documentation, automatic self-configuration and active configuration support, forced self-tests for the resulting configuration, and automated communication between framework and user about these processes at adaptive complexity levels through adequate user interfaces. The user should also be informed about the consequences of settings and possible trade-offs, such as the balance between expected simulation time and overall simulation quality (Leube et al., 2013).

The paper is organized as follows: In Section 2, we review the most critical steps within the workflow of simulation-based engineering and science that are relevant to our discussion. In Section 3, we present three examples of complex extended simulation workflows taken from (environmental) engineering applications. Based on these examples, we highlight the evolving complexity of scientific workflows that focus not only on simulation, but also on