On reproducibility and traceability of simulations

Olivier Dalle

ON REPRODUCIBILITY AND TRACEABILITY OF SIMULATIONS

Olivier Dalle

University of Nice Sophia Antipolis
I3S Laboratory – UMR CNRS 7172 &
INRIA – CRI-SAM
2004 route des Lucioles
F-06903 Sophia Antipolis, FRANCE

July 25, 2012

ABSTRACT

Reproducibility of experiments is the pillar of a rigorous scientific approach. However, simulation-based experiments often fail to meet this fundamental requirement. In this paper, we first revisit the definition of reproducibility in the context of simulation. Then, we give a comprehensive review of issues that make this highly desirable feature so difficult to obtain. Given that experimental (in-silico) science is only one of the many applications of simulation, our analysis also explores the needs and benefits of providing the simulation reproducibility property for other kinds of applications. Coming back to scientific applications, we give a few examples of solutions proposed for solving the above issues. Finally, going one step beyond reproducibility, we also discuss in our conclusion the notion of traceability and its potential use in order to improve the simulation methodology.

1 INTRODUCTION

A scientific method defines the "principles and procedures for the systematic pursuit of knowledge involving the recognition and formulation of a problem, the collection of data through observation and experiment, and the formulation and testing of hypotheses" (Merriam-Webster Dictionary 2012.) Computer simulations have long proven to be useful for working on the latter activities, to collect data from a computerized representation of the system under study (called the simulation model), and to test hypotheses by varying the parameters and inputs of the model.

Advanced and commercial simulation software are generally designed for making the life of the experimenter as easy as possible. Unfortunately, by simplifying the process of running an experiment, this software often hides important parameters and experimental design questions. For example, as reported by Perrone, Yuan, and Nicol (2003), these hidden details end up missing in the published description of the experiment, which prevents a third party from reproducing the experiment with another simulation software, or even with a different version of the same simulation software.

The pressure to publish or perish, combined with the short time allotted for reviewing, and in some cases the unavailability of the source code, almost established as a standard the acceptance of a simulation result without any attempt at reproducing the actual simulation experiments.

Ince, Hatton, and Graham-Cumming (2012) claim, in their editorial article for the journal *Nature*, that "[they] have reached the point that, with some exceptions, anything less than release of actual source code is an indefensible approach for any scientific results that depend on computation, because not releasing such code raises needless, and needlessly confusing, roadblocks to reproducibility." Going further than raising the issue, some journals such as *Science* already mandate that authors supply the source code along with their paper submission, while others, such as *Biostatistics*, have appointed an "Associate Editor for Reproducibility" (Niemeyer 2012).

In this paper, we discuss reproducibility in the case of simulation-based experiments. We first give the multiple definitions of reproducibility in this particular context, including its degenerate forms, in section 2. Then, we identify and discuss a number of issues that need to be solved in order to achieve this property, in section 3. We discuss the various usages of reproducibility depending on the type of the simulation application, in section 4, and then we propose some solutions in section 5. Finally, in section 6, we present some related works, before we conclude in section 7.

2 DEFINITIONS

As a generic science concept, reproducibility designates the ability to confirm the results of a previous experiment by means of another similar experiment. In physics, for example, the criteria for reproducibility and repeatability are standardized with respect to results of measurements (NIST 1994). However, the previous definitions do not apply well to the particular context of in-silico experiments. For example, the conditions for the repeatability of results measurements, as defined by U.S. National Institute of Standards and Technology (NIST, op. cit.), requires the same observer, the same instrument used in the same conditions and the same location. These conditions and definitions clearly need to be adapted to the particular case of simulations.

Applied to in-silico experiments (ie. computer simulations), we consider reproducibility as the ability to produce similar data by running a similar simulation. The similarity of data does not mean that data must be strictly identical, but that they must exhibit the same properties according to some metric of interest, and usually in accordance with the goal of the study (some of the conditions defined by NIST could certainly apply here, such as the 95% similarity threshold). For example, data samples collected with two similar experiments may have different numerical values but have the same statistical properties or belong to the same curve with different sampling coordinates.

Before proceeding further with the definition(s) of reproducibility of simulations, we need first to introduce a few elements of terminology to set up the landscape of this paper. These definitions are intended to clarify the content of this paper rather than the body of knowledge of Modeling & Simulation at large or the many variants of the general terms that could be found in literature:

- A **simulation** is short for *simulation run*, which is the process of executing a computer program that runs a *computer simulation*;
- A **computer simulation** is composed of the following elements:
  - a **model** defines the structure and behavior of the *system* that is the subject to the study; the previous system can take many forms including computer and networking systems, biological systems, transportation networks, factory plants, crowds, battlefields, etc.
  - a **scenario** defines a possible history of the system defined by the model; the scenario can be produced either using a dedicated means (ie. a scripting language or some formatted inputs provided to the computer simulation program) or using the same elements as those used for definition of the model (a scenario is a model of the exogenous stimuli applied to the system);

- a **simulator** refers to a reusable simulation *engine* and its *Application Programming Interface* (API); the engine is reusable for the simulation of many models and scenarios;
- an **instrumentation** defines which variable(s) of interest to observe during the simulation execution and what computation(s) to apply at run-time to the data samples produced by these variables (including their logging to result files).

When referring to computer simulations, we suggest that reproducibility should be considered according to the four levels of requirement presented hereafter:

L1. **deterministic, identical computation**: this level of compliance is also known as the *repeatability* property; it designates the ability to re-execute the exact same simulation history as previously in terms of computation, including the inherently non-deterministic computations. In the case of a discrete-event simulation, for example, repeatability means that the simulations produce the same series of events and process them in the same order. In the case of distributed simulations, in particular, repeatability implies that parallel activities (eg. event occurring at the same simulation time) must always be processed in the same order, which requires some additional synchronization (Fujimoto 2000). Although not strictly necessary, repeatability usually comes with the idea of reusing the same simulation source code in order to produce the same computation; this level implicitly requires an identical instrumentation;

L2. **non-deterministic, identical computation**: this level of compliance is a relaxed form of the previous one in which the non-deterministic situations are not resolved. This lighter form may either result from the lack of additional synchronization, or in some extreme cases, from the inability to fully control the experimental conditions, such as with real-time distributed simulation (see McLean and Fujimoto 2000, for example). Note that it is not considered good practice to produce significantly different results due to non-determinism in the model. Indeed, in practice, if two events occur at the same time and their processing order is not specified by the model, the order of their execution should not influence the simulation result, given that the result should only be correlated to the model, never to the simulator. For example, if two customers A and B arrive and get served at the same time at two different counters, then simulating customer A before customer B or the opposite should not change the total amount of transactions processed by that bank at the end of the day. Assuming that different instrumentation produces different outputs, which have resulted from different computations, achieving reproducibility at this level requires instrumentation to be identical;

L3. **identical scenario and instrumentation**: this level consists in reproducing the simulation experiment based on the detailed specifications of its scenario and model rather than re-using the same program source code. By neglecting some implementation details (eg. the sequence of random numbers produced by a random number generator), reproducibility at this level may lead to the production of different data, even though the instrumentation is assumed to be identical. Considering that the scenario should have been designed to observe a phenomenon, then the various possible executions of that scenario should reproduce the same phenomenon, and therefore the data, although different, should have identical meaning from a statistical point of view;

L4. **similar scenario and instrumentation**: this level is the most general and relaxed form of reproducibility. In general science, experiments may be considered similar even if they do not apply to the exact same subjects (qualitatively or quantitatively), especially when it is impossible to reproduce the exact same experimental conditions. Although computer simulation offers a fine control on the experimental conditions, it is still relevant to consider similar cases.

Compared to other forms of experiments, Modeling & Simulation has two distinctive features with respect to reproducibility:

- **ability to replay:** the ability for computers to reproduce in a deterministic way the exact same computation (the repeatability property discussed above at level L1), is a distinctive feature of simulation. For example, simulation allows for a replay of the same history of the system under study, but with a different instrumentation, in order to make different observations. Not only can the risk of interference (the Heisenberg principle) be avoided with computer-based experiments, but with the help of proper techniques, the instrumentation can even be totally separated from the model and scenario (Dalle and Mrabet 2007). This leads to a different form of reproducibility, in which the experiment is reproduced, but with different outputs.

- **fully virtual nature:** when building in-silico experiments using simulation, the whole experiment, including the system under study, is artificially created. While this virtual nature offers a great flexibility, it also introduces new risks, that makes it a distinctive feature of simulation. Indeed, in other forms of experiments, the physical reality of the system under study means that at least that part of the system can be trusted and considered error-free, while the experiment process and the observation should be considered error-prone. In simulation, not only the same sources of error exist, but the system itself can be modeled incorrectly and be a source of error. This situation leads to another form of reproducibility, in which the experiment is reproduced, but with a different system, that intrinsically produces different outputs.

3 ISSUES FOR REPRODUCIBILITY

In this section we discuss issues for reproducibility that are assumed to have some solution. Indeed situations may exist in which reproducibility is not possible due to some intrinsic property of the simulation. For example, as will be discussed later in section 4, the fact that a Virtual Reality simulation is driven by user interactions in real time makes it difficult to reproduce and is therefore not considered as an issue worth to address.

If we except the case of repeatability, issues for reproducibility fall into two main categories: human factors and technical issues.

3.1 Human factors

A number of human factors may impede reproducibility. Although negligence is certainly a primary concern, we won't elaborate further on this particular aspect and will assume that human-related operations are handled by operators always willing to contribute to the best extent of their current knowledge. Other human factors left to consider include the following:

- **"genuine" ignorance:** Being unaware of some hidden parameters, such as which random number generator is being used by default by the simulation engine or library;

- **insufficiently detailed publications:** Despite the fact that this issue is widely acknowledged as being major, it may be considered a result of the publication process as a whole, making responsibility very difficult to attribute. Some details may be skipped by authors either inadvertently or on purpose, because the publication space is limited. Additionally, there is very little incentive to provide reproducible content because this aspect is almost never checked by reviewers, either because the reviewing deadlines are too tight, or because publishers do not give that instruction;

- **business related limitations:** As mentioned in introduction, availability of the source code seems necessary to ensure reproducibility. However, providing an Open Source simulation software is not always an option. Firstly, some commercial simulators are commercial products with a business model that is not designed for Open Source. Furthermore, as will be explained in greater details in the next section, simulation has

many applications that are not aimed toward producing science and, therefore, do not necessarily require a reproducibility based on the source code availability. Secondly, when using an open source simulator, some additional elements, such as the models and scenario used to produce the results, may not be fully disclosed at source level, for intellectual property reasons;

- **manipulation errors:** These errors result from the manual handling of some of the tasks in the simulation study work-flow. Trying to build a detailed typology of such errors is pointless (and certainly endless). However, assuming that their probability of occurrence is in proportion with the amount of human intervention required in simulations, human involvement should be minimized as much as possible and replaced by automation, as will be later discussed in section 5.

3.2 Technical issues

A variety of technical issues may also arise, including the following ones:

- **software bugs:** Because of software bugs in simulation engines and libraries, perfectly valid experiments and models can produce erroneous results that are unreproducible in newer versions of the simulator once the bugs are fixed. Being able to reproduce these erroneous results with old versions, however, is still useful, because that can help to explain other discrepancies or track erroneous results in time. Unfortunately, the various versions of a simulation software, especially the ones with bugs, are not always available;

- **software availability:** Multiple reasons may explain software unavailability, including losing track of old versions, commercial reasons (eg. discontinued or unaffordable products) or confidentiality and competitive reasons (eg. providing a good simulator of a system to a competitor may not be desirable).

- **floating points numbers:** Although very useful, floating point numbers (FP) are reputed to be an endless source of complications, as explained by Goldberg (1991). For example, in the popular networking simulator ns-2, the time of occurrence of events is encoded using FP and it is easy to observe discrepancies when scheduling repetitive tasks during a simulation: Due to FPs errors, repeatedly adding a little delay does not produce the same date as multiplying an incremental delay by the number of iterations. Indeed, although semantically identical, ns-2 considers the dates 0.99999999999 and 1.0000000001 as different. This kind of error is likely to introduce a bias in the results given that all events occurring at time $0.9-9$ end up having more priority than those occurring at time $1.0-01$. Such a bias may be unreproducible with another simulator in which time is encoded using integers;

- **computer and Operating Systems evolutions:** Ensuring the long term reproducibility of a simulation may be a challenging issue, given the continuous evolution of computer systems and architectures. Depending on the language used for writing the simulation, a significant dependency may exist with the operating system or additional libraries.

4 USAGE EXAMPLES & APPLICATIONS

In this section, we discuss the various usages of reproducibility depending on the type of the simulation application. In section 4.1, we first use the example of a network performance evaluation as a typical case of scientific study. Then through the following sections, 4.2 to 4.5, we discuss Road Traffic simulations, Training, Gaming and Virtual Reality as examples of non scientific studies. The summary of these various cases is then presented in section 4.6. These examples show that reproducibility at various levels may be considered and even have value for simulations beyond the limited scope of scientific studies.

4.1 Network performance evaluation

This type of simulation is normally used as part of a scientific study and usually consists in assessing the performance of a protocol or network architecture subject to a typical traffic workload. The simulation consists in reproducing the protocol(s) operations while the simulated traffic is flowing through the network. Multiple simulations are necessary to estimate the metrics of interest, such as the level of Quality of Service achieved end-to-end, or the average size of the packet queues in the network routers. Parallel simulations may be employed for large or complex networks, which explain the need for reproducibility at level L1. A special form of network simulation, called emulation, consists in coupling real hardware or software with the simulation. In this case, the non-determinism due to the real-time operation of the network may limit the reproducibility to level L2. Compared to other types of simulation studies, a network simulation study usually requires to develop new models. Reproducibility at Level L3 is sometimes used as a technique of verification and validation: The same simulation is run using different simulators to assess the correctness of the new model implementation. Since network simulation is mostly used for scientific studies, reproducibility at level L4 is relevant.

4.2 Road traffic

This type of simulation is typically used in an operational context: For example, it may be used by a traffic management center, to compute the dynamics of traffic within a road network over a period of time. The road network is modeled in terms of lanes and geographical elements such as crossings, traffic lights, traffic circles, etc. The traffic is usually defined by an Origin-Destination matrix that describes the starting and ending points of each vehicle on the road network. A possible goal of this type of simulation is to identify congestion points either when the traffic changes (eg. following an event) or when the network changes (eg. a street or lane is closed). Parallel simulation may be employed, which explains the possible need for reproducibility at level L1. Reproducibility at level L2 does not seem to be necessary, even though road traffic simulators may be fed with real-time traffic information, because in this case the simulation still remains deterministic. Reproducibility at level L3 is not necessary in operational context because the simulator is often used to make decision in real time and because writing new models for this type of simulation is not common, once the initial development stage is complete. Reproducibility at level L4 is not necessary because scientific studies are not the main use for this type of simulation.

4.3 Training

This type of simulation is designed to put users in operational situations according to some predefined scenario; they are used to train operators in particular tasks or resolve situations using a virtual copy of their working environment. For example, such simulator can be used to train nuclear plant operators, train or subway conductors, or to train plane and helicopter pilots to fly with no visibility using their instruments. Parallel simulation is not expected on a large scale and therefore issues related to reproducibility at level L1 are irrelevant. Repeatability at level L2 is a highly desirable feature, since repeating identical actions is an actual process of learning. Repeatability at level L3 is of marginal use, because it is not common practice to offer trainees several simulators for the same training purpose, however, a given simulator may be subject to the release of new versions, which then would require reproducibility at level L3. This type of simulator produces interactivity rather than outputs, and therefore reproducibility at level L4 is not applicable.

4.4 Gaming

Although very similar to the previous one, this type of interactive simulation has different requirements. Reproducibility at level L1 is irrelevant for the same reasons as for Training; reproducibility at level L2 is possible as part of the game design (many games follow strictly a pre-established scenario), but is not mandatory; Reproducibility at level 3 is not relevant, because most games have only one (commercial) implementation; and reproducibility at level L4 is not applicable.

4.5 Virtual Reality

This type of simulation is another form of interactive simulation in which a view on the virtually simulated world is computed in real-time, depending on the user's physical actions. In many such simulations, the user is a passive observer and the virtual world is either static or has its own dynamics, independent of the user's actions. Since the view generated in real-time by each simulation is driven by the user's action in real-time, each simulation is meant to be unique and therefore reproducibility at levels L1 or L2 is irrelevant. Reproducibility at level L3 is relevant only with more advanced forms of VR simulations in which the user becomes active and fully interacts with the VR world. These advanced forms may then be considered as either Gaming or Training depending on their intended use. Since reproducibility is impossible at level L2, it must be provided at level L3 if required. For example, a VR-based flight-combat simulator obviously requires the VR world to react to users' actions; it may be used for Training, and therefore requires the simulation to follow a pre-existing scenario. Since the simulation cannot be reproduced at level L2 because of the user's real-time interactions, however, only the simulation scenario can be reproduced. Reproducibility at level L4 is not applicable to VR simulations.

4.6 Summary

Table 1 gives a summary of the usage (and benefits) for reproducibility in terms both of the level of reproducibility and the type of the simulation application. It shows that despite the main use by and stringent requirements of the scientific field, there are still some potential applications of reproducibility in other types of simulations. Nevertheless, in the remaining sections of this paper, we will only concentrate on reproducibility in the context of scientific studies.

Simulation application	Level of Usage			
	Level L1	Level L2	Level L3	Level L4
Network performance evaluation	yes, if parallel exec.	yes, if RT emul.	yes, verification	yes, 3d party validation
Road traffic	possibly, if parallel exec.	no	no	no
Training	no, irrelevant	yes	marginal, with new versions	N/A
Gaming	no, irrelevant	possibly, by design	no, irrelevant	N/A
Virtual reality	no, irrelevant	no, irrelevant	yes, if active user	no

Table 1: Usage for each level of reproducibility depending on the application type

5 SOLUTIONS FOR REPRODUCIBILITY

In this section we discuss possible solutions for reproducibility issues in scientific studies based on simulations. In section 2, we distinguished among four levels of reproducibility, which in turn suggested specific solutions at corresponding levels, which we presented in sections 5.2 to 5.4. However, some issues are level-independent, which we will first discuss in section 5.1.

5.1 Level independent issues

Human errors are an example of level-independent issues. As already suggested in introduction, a pragmatic way to solve these issues is to reduce human intervention by relying on automation as much as possible. The *Simulation Automation Framework for Experiments* project (Perrone, Main, and Ward 2012), is an example of the ongoing development of such a solution for the ns-3 network simulator.

5.2 Solutions for Repeatability (levels L1 & L2)

In the introduction we mentioned several works on protocols to solve the non-determinism issues of parallel execution (Fujimoto 2000) or real-time interactions (McLean and Fujimoto 2000). If we except these determinism-related issues, repeatability is mainly a question of ensuring the long-term execution of a program. This issue is challenging given that computer architectures and operating systems and their accompanying libraries are both continuously evolving. The following two pragmatic solutions can be proposed for this evolution issue:

- **use a platform-independent solution:** platform independence can be provided at various levels, but the most efficient is certainly at the language level, assuming the language is stable due to its large user community. This level of independence can be achieved by adopting a simulation formalism, such as Petri Nets or DEVS (Zeigler, Praehofer, and Kim 2000), especially in association with one their well maintained software platform (eg. CPNTools, Jensen and Kristensen 2009) or in accordance with a standard (eg. DEVS standardization, Wainer et al. 2011).
- **ensure the availability of the computing solution:** virtual appliances provide a timely and straight-forward implementation of this solution. A virtual appliance is a virtual machine fully configured to run a particular application or service, including the application software and all of its dependencies. Since the virtual appliance is a self-contained snapshot of a functional system, it only requires the continuous support of virtual machine. This solution can also solve the issue related to the availability of the source code: Since with a virtual appliance, the execution is fully handled by the virtual machine at the binary level, the source code is not necessary.

5.3 Solutions for level L3

At this level, the challenge is to reproduce the model and scenario of a previous simulation study. For this purpose, an obvious requirement is the ability to retrieve the corresponding models and scenario specifications, which become more and more challenging to retrieve as time goes on. Long-term archives of the model, the scenario, and to a larger extent of any simulation experiment is a critical issue when using simulation for scientific purposes. A solution to this problem could be to place the responsibility in the hands of the publishers and editors of scientific publications. As already mentioned in introduction, a few publications already enforce such a policy. However, this policy comes at a cost and the real difficulty is therefore to offer enough incentives to both overcome this overhead and deeply change the practices. A summary of the changes required to enforce such a policy is given hereafter:

- The sources of the code required to reproduce the simulations should be provided with each paper submission that contains a scientific result obtained by means of simulations;
- The previous requirement should be enforced by editors and publishers;
- This additional material should be part of the publication and, as such, archived and redistributed by the publishers along with the printed material of the publication;
- The review process should include a review of the provided material and at least one test of reproducibility.

Enforcing such a policy comes with a number of positive and negative side-effects listed hereafter:

− Publishers would need to adapt their archival and diffusion means to support the provided source code;

− Editors would have to account for this new form of submitted material in their decisions; for example, it might be necessary to ensure that the reviewers are sufficiently familiar with the type of source code provided and able to check the reproducibility;

− The peer-review process becomes more complex and takes more time;

\+ As part of the publication, the provided source code can be cited;

\+ The general availability of the source code makes the reproduction and extension of previous published works much easier, which is a clear benefit to the whole scientific community;

+/− This policy enforces the use of Open Source software, which may not be acceptable for commercial or strategic reasons, but would certainly be largely supported within the scientific community;

Note that the source code required to reproduce the simulations may extend well beyond the code of the models and experiments. Depending on the dependencies existing among the results of the simulations and other elements, it may be necessary to include the code of the simulator engine and libraries, as well as those of the operating system. However, given the additional amount of space required for the storage of these elements, we can assume that publishers would encourage reusing previously published ones, or would put restrictions on attempts at publishing new ones (eg. request a justification from the authors explaining why they need a new configuration instead of the one already published).

5.4 Solutions for level L4

At this level we assume that we want to reproduce the results of a previous experiment, but not necessarily using the source code. The issues about using Floating Points numbers and insufficiently detailed publications are typically observed at this level. Obviously, avoiding Floating Point as much as possible is a way of solving the former, while the latter leads again to the publication process, as discussed in the previous section. However, the requirement on the publications becomes lighter at this level, compared to the level L3: if we assume that providing the source code is no longer necessary, then any sufficiently detailed specification of the experiment may be accepted. Unfortunately, this lighter requirement would make the task of reviewers more difficult than it was in the previous level. Not asking to do this, however, would leave the issue of insufficiently detailed publications unsolved.

6 RELATED WORKS

The idea of reproducible research applied to computations seems to be first presented in Schwab, Karrenbach, and Claerbout (2000), which opened a new research direction (see Fomel and Claerbout 2009). Rahmandad and Sterman (2012) discuss the concepts of "Minimum Model Reporting Requirement" and "Preferred Model Reporting Requirement" as a means of ensuring reproducibility in simulation-based research in Social Science, at level L3, even though the authors do not explicitly distinguish multiple levels. An interested reader will also find a number of related works in various domains in the previous paper, including biochemistry, biology, computational research, and mathematics.

However, many published works based on simulation still fail to meet the minimum conditions to ensure reproducibility. For example, Kurkowski et al. (2005) surveyed the results of MANET simulation studies published between 2000 and 2005 in the ACM MobiHoc Symposium: 75% of these papers used simulation but less than 15% only of the simulations were repeatable and only 7% addressed such important issues as initialization bias.

This failure to ensure reproducibility is one of the reasons that concerns about credibility are raised in simulation. About this general issue of credibility: Pawlikowski, Jeong, and Ruth Lee (2002) surveyed over 2200 publications on telecommunications networks in proceedings of the IEEE INFOCOM and such journals as the *IEEE Transactions on Communications*, the *IEEE/ACM Transactions on Networking*, and the *Performance Evaluation Journal* and their conclusion was that "[...] the majority of recently published results of simulation studies do not satisfy the basic criteria of credibility." The two main criteria they consider in order to assess credibility were the use of appropriate pseudo-random numbers generators and the appropriate analysis of simulation outputs.

This credibility issue is surprising given that the many steps followed, and decisions taken in simulation studies have been quite extensively covered in the literature and, in particular, in the simulation reference books (Banks et al. 2010, Law 2007, etc.) These steps and decisions are usually described by means of a typical work-flow that starts with the definition of the goal of the study and proceeds, with possibly several iterations, through the major steps of modeling, validation, verification, and experiment planning, and concludes with an analysis of the results produced by the simulation(s). Although some of these work-flows depicted in literature are quite detailed (eg. Wainer 2009), they all choose the same activity perimeter that starts with the goal definition and ends with data analysis. This already large, but still limited perimeter, explains why the issues of reproducibility are seldom covered: they occur at a higher level.

Indeed, reproducibility is part of a larger scientific process that extends well beyond the boundaries of Modeling & Simulation, both in terms of scope and duration. In terms of scope, for example, a physical experiment can be reproduced using simulations in order to confirm the physical results. In terms of duration, for example, a simulation could be reproduced a long time after a study was completed (and possibly published), either as part of another study, or to confirm published results.

7 CONCLUSIONS

In this paper, we discussed the issues and current status of simulation reproducibility. We first made explicit the multiple levels of reproducibility that can be achieved using computer simulations. We explored their potential use in various forms of simulation, and found a few in which they might be relevant. Then, we reviewed a number of issues that need to be solved in order to achieve reproducibility in the particular case of scientific applications and we discussed some solutions to solve them at the various levels we identified.

Our analysis reached similar conclusions to Ince, Hatton, and Graham-Cumming (2012), which advocate the use of Open Source software for scientific publications. We also discussed the publication process itself and how it should evolve in order to ensure the reproducibility of simulations based on Open Source software. Although

these recommended change would have significant impact on the current practices for all the actors involved, we strongly believe that they would result in greater benefits for the whole scientific community.

Last but not least, solving the reproducibility issues also opens new perspectives, such as allowing for the traceability of simulations results.

Traceability is the ability to trace the dependencies of results and publications to the software elements that were used to produce them. In conjunction with reproducibility, this ability is of particular interest when a bug is discovered, because it can help identify (trace) which results and publications are impacted by the bug. Since the simulation software and models are continuously subject to improvements and new versions, ensuring traceability rapidly becomes a problem of version and dependency management. In these cases, the most appropriate level of reproducibility is level L2, so that the code of the impacted simulation can be re-executed after the bug is fixed. However, in order to cope with more complex forms of errors, it is certainly desirable to allow the re-execution of whole experiments, which may involve a number of complex tasks. Solutions have emerged in the last few years, such as the Apache Foundation's Maven project for management of dependencies and versions or the MyExperiment and Bonita projects for the management of work-flows (Dalle 2011).

Therefore, used conjointly, these two properties of reproducibility and traceability would allow for the proper detection and correction of errors in published results obtained using simulations.

ACKNOWLEDGMENTS

This work has been co-funded by the DISSIMINET Associated Team initiative (INRIA Sophia Antipolis and Carleton University) and the French Agence Nationale de la Recherche (ANR) SONGS Project. I would like to address my special acknowledgements to Joe Peters and the Faculty of Applied Sciences at the Simon Fraser University for hosting me while writing this paper and for their funding support through an Ebco Eppich Fellowship grant. Last but not least, I would like to kindly acknowledge the three anonymous reviewers of this paper for their insightful comments and suggestions, and my wife for her careful and professional English corrections.

References

Banks, J., J. S. Carson, B. L. Nelson, and D. M. Nicol. 2010. *Discrete-Event System Simulation*. 5th ed. Upper Saddle River, New Jersey: Prentice-Hall, Inc.

Dalle, O. 2011, July. "Should Simulation Products Use Software Engineering Techniques or Should They Reuse Products of Software Engineering? Part 1". *Modeling & Simulation Magazine* 11 (3): 12p. Online publication.

Dalle, O., and C. Mrabet. 2007, September. "An instrumentation framework for component-based simulations based on the Separation of Concerns paradigm". In *Proc. of 6th EUROSIM Congress on Modelling and Simulation (EUROSIM2007)*, 10p. Ljubljana, Slovenia: EUROSIM.

Fomel, S., and J. Claerbout. 2009, Jan./Feb.. "Introduction to: Special Issue on Reproducible Research". *Computing in Science and Engineering* 11 (1): 5–7.

Fujimoto, R. M. 2000. *Parallel and Distributed Simulation Systems*. Wiley Interscience.

Goldberg, D. 1991. "What Every Computer Scientist Should Know About Floating-Point Arithmetic". *ACM Computing Surveys* 23:5–48.

Ince, D. C., L. Hatton, and J. Graham-Cumming. 2012, February. "The case for open computer programs". *Nature* 482:485–488.

Jensen, K., and L. Kristensen. 2009. *Coloured Petri Nets – Modeling and Validation of Concurrent Systems*. Springer-Verlag Berlin.

Kurkowski, S., T. Camp, and M. Colagrosso. 2005. "MANET Simulation Studies: The Incredibles". *ACM's Mobile Computing and Communications Review* 9 (4): 50–61.

Law, A. M. 2007. *Simulation Modeling & Analysis*. 4th ed. New York: McGraw-Hill, Inc.

McLean, T., and R. Fujimoto. 2000, May. "Repeatability in Real-Time Distributed Simulation Executions". In *Proceedings of the 14th Intl. Workshop on Parallel and Distributed Simulation (PADS'00)*, 23. Bologna, Italy: ACM/IEEE/SCS: IEEE.

Merriam-Webster Dictionary 2012. "Definition of scientific method". Accessed May 5, 2012. http://www.merriam-webster.com/dictionary/scientific-method.

Niemeyer, Kyle 2012. "Nature Editorial: If you want reproducible science, the software needs to be open source". Accessed May 5, 2012. http://arstechnica.com/science/news/2012/02/science-code-should-be-open-source-according-to-editorial.ars.

NIST 1994. "TN-1297- Guidelines for Evaluating and Expressing the Uncertainty of NIST Measurement Results – Appendix D". Accessed June 3, 2012. http://physics.nist.gov/Pubs/guidelines/appd.1.html.

Pawlikowski, K., H.-D. Jeong, and J.-S. Ruth Lee. 2002, January. "On Credibility of Simulation Studies Of Telecommunication Networks". *IEEE Communications Magazine* 40 (1): 132–139.

Perrone, L. F., C. S. Main, and B. C. Ward. 2012. "SAFE: Simulation Automation Framework for Experiments". In *Proceedings of the 2012 Winter Simulation Conference*, edited by C. Laroque, J. Himmelspach, R. Pasupathy, O. Rose, and A. M. Uhrmacher, 11p. Piscataway, NJ, USA: IEEE: IEEE Press.

Perrone, L. F., Y. Yuan, and D. M. Nicol. 2003. "Simulation of large scale networks II: modeling and simulation best practices for wireless ad hoc networks". In *Proceedings of the 35th conference on Winter simulation: driving innovation*, WSC '03, 685–693. IEEE: Winter Simulation Conference.

Rahmandad, H. and Sterman, J. 2012. "Reporting Guidelines for Simulation-based Research in Social Sciences". Accessed June 3, 2012. http://jsterman.scripts.mit.edu/Reporting_Guidelines_for_Simulation-based_Research_in_Social_Sciences.htm.

Schwab, M., M. Karrenbach, and J. Claerbout. 2000. "Making Scientific Computations Reproducible". *Computing in Science and Engineering* 2 (6): 61–67.

Wainer, G. A. 2009. *Discrete-Event Modeling and Simulation: a Practitioners approach*. CRC Press. Taylor and Francis.

Wainer, G. A., K. Al-Zoubi, O. Dalle, D. R. Hill, S. Mittal, J. R. Martín, H. Sarjoughian, L. Touraille, M. K. Traoré, and B. P. Zeigler. 2011. *Standardizing DEVS model representation*, Chapter 17, 427–458. Taylor and Francis.

Zeigler, B. P., H. Praehofer, and T. G. Kim. 2000. *Theory of Modeling and Simulation*. Academic Press, Inc.

AUTHOR BIOGRAPHIES

OLIVIER DALLE is Maître de Confrences in the Computer Sciences department of the Faculty of Sciences at the University of Nice-Sophia Antipolis (UNS). He received his B.Sc. from the University of Bordeaux 1 and his M.Sc. and Ph.D. from UNS. From 1999 to 2000, he was a postdoctoral fellow at the French Space Agency center in Toulouse (CNES-CST), where he started working on component-based discrete-event simulation of multimedia telecommunication systems. In 2000, he joined the MASCOTTE joint research group of UNS, CNRS, and INRIA, working on telecommunication systems. In 2012, he moved to the OASIS group, which is also a joint research group of UNS, CNRS, and INRIA, working on distributed computing and component middleware. His current research interests are discrete-event simulation (mainly methodology support), very large-scale networked systems, and component-based software engineering.