Interoperability and Composability: A Journey through Mathematics, Computer Science, and Epistemology

Andreas Tolk
The MITRE Corporation
903 Enterprise Pkwy #200, Hampton, VA 23666
UNITED STATES

atolk@mitre.org

ABSTRACT

The paper highlights the mathematical constraints of information exchange by explaining the first principles of model theory, Robinson's Consistency Theorem and Łoś Theorem. These theorems build the mathematical foundation for truth representation in distributed, potentially heterogeneous systems. When using computer simulation, additional constraints should be addressed, computability and computational complexity. This leads to some general principals on how computer simulations can generally be used to gain knowledge on interoperable and composable services, as they are envisioned for new concepts, like M&S as a Service, cloud-based distribution of M&S services for easy reuse, and more. The compilation of these mathematical, computational, and epistemological constraints became the bedrock of interoperability and composability ensuring "fair fight" and efficient training.

1.0 INTRODUCTION

The M&S community understands *interoperability* quite well as *the ability to exchange information and to use the data exchanged* in the receiving system [1]. Interoperability can be engineered into a system or a service after definition and implementation. Alternative data representations can be mediated into each other if the constraints are understood. Only when data must be disaggregated, which requires that the information that got lost in the aggregation process be reinserted, the engineer has the problem from where to extract this needed information, but often heuristics can be applied that lead to satisfactory results.

Composability is different from interoperability. *Composability is the consistent representation of truth in all participating systems* [1]. It extends the ideas of interoperability by adding the pragmatic level to cover what happens within the receiving system based on the received information. In contrast to interoperability, composability cannot be engineered into a system after the fact. Composability requires often significant changes to the simulation to ensure that a research question is either answered equivalently in all participating simulation systems, or it is not answered at all. Inconsistent versions of truth are not allowed.

Current simulation interoperability standards, such as IEEE 1278 Distributed Interactive Simulation [2] or IEEE 1516 High Level Architecture [3], focus on the information exchange. Composability, the consistent representation of truth in all participating systems, needs to be ensured by additional governance, which needs to be provided by technology aware management.

This paper highlights the main research results regarding the mathematical, computational, and epistemological constraints for composability and derives good practices for the governance of distributed, federated solutions. They can be used as guides for new standards, as augmentations to existing simulation specific guidance, such as the Distributed Simulation Engineering and Execution Process (DSEEP) [4] with its extension to provide a Multi-Architecture Overlay (DMAO) [5], or it can be used as additional guidance in support of general interoperability standards, such as semantic web standards [6].

This paper compiles several ideas and research results presented in various conference proceedings, book chapters, and journal contributions under the common question of how they influence interoperability and composability. The interested reader is encouraged to look up the original work for additional references and relevant related material.

2.0 MATHEMATICAL CONSTRAINTS

It is not the intention of this paper to introduce the reader to the details of the mathematical constraints. Instead, the reader is referred to more detailed literature on this topic, such as given in [7]. Detailed descriptions how to apply model theory in support of modelling and simulation is presented in [8, 9]. This section will just describe the high-level concepts necessary to understand the implications for interoperability and composability challenges.

2.1 Tenets of Model Theory

Model theory is a subset of mathematics that applies logic to formal structures, such as defined by sets, enumerations, or formal languages (such as programming languages). A model collects all the information needed to decide if a statement is true in its context, i.e., if the statement is a member of the applicable sets or the enumeration, or of it can be generated by the formal language. If the statement is true, it is satisfied in the model. This allows model theory to treat mathematical truth as relative: the same statement may be true or false, depending on how and where it is interpreted.

This effect is well known to developers of simulation federations. Depending on which simulation systems are tasked to compute the outcome of an operation, the result may be different. The reasons for this are manifold, and several of them will be addressed in the following sections on computational and epistemological constraints. In any case, the software developer must be aware of such inconsistent domains within the simulation to select those behaviours that are best to fulfil the desired simulation result in the light of the commander's intent for the supported exercise.

As stated above, model theory applies logic to the evaluation of truth represented using mathematical structures. As computer languages are formal languages, and as simulation systems are programed in computer languages, the results regarding truth representation in formal languages can be applied to consistent representation of truth within computer simulations. As truth regarding the same facts and interpretations need to be consistent within M&S applications, the research findings are significant for understanding interoperability and composability challenges in order to address them when selecting simulations to be federated in support of an exercise, operation, or any other simulation application domain.

Using the definition given in [7], a *language* $\mathcal{L}$ is a set consisting of all the logical symbols with perhaps some constant, function and/or relational symbols included. A *model*, sometimes also called a *structure*, $\mathcal{U}$ for a language $\mathcal{L}$ is an ordered pair of the *universe* $\mathcal{A}$, which is a nonempty set, and an *interpretation function* $\mathcal{I}$ with its domain being the set of all constant, function and relation symbols of $\mathcal{L}$. The interpretation function maps each constant symbol to a constant, each function symbol to a function, and each relation symbol to a relation. A *sentence* is an assertion that can be assigned the Boolean value of true or false. And, finally, if $\mathcal{U}$ is a model of $\mathcal{L}$, the *theory* of $\mathcal{U}$ is defined to be the set of all sentences of $\mathcal{L}$ which are true in $\mathcal{U}$.

What does this have to do with interoperability and composability? The symbols used in the simulation, which includes all symbols used in the formal language, are captured in the language $\mathcal{L}$. Applying syntax to these symbols allows to formulate sentences. Using the interpretation function as defined as part of the model, sentences can be evaluated to be true or false. The theory of a model is the set of all true sentences, or the enumeration of all possible states of the simulation system for a given time. Hence, if two simulations need to be consistent, they must have a consistent representation of truth. If two simulations result in different states, they are not consistent. They also are not consistent under these definitions of model theory, as the same statement has different interpretations in the participating simulation systems, and that means the simulation systems are computing different results.

2.2 Important Theorems

These observations motivate one to have a closer look at two results of model theory, captured as theorems. *Robinson Consistency Theorem* simply states that the union of two theories is satisfiable under a model if and only if their intersections are consistent, in other words: there is only one interpretation of truth valid in both models where they overlap. If this is not the case, there will be inconsistencies! As it is possible that two theories are using different languages and the resulting sentences are not directly comparable, *Łoś Theorem* generalizes the idea of expanding a universe through the Cartesian product and defines filters that allow the comparison in a common equivalent representation. Simulation practitioners know this process as data mediation: if two simulation systems use different terms and symbols, data mediation maps those different representations onto each other. The second theorem states therefore that two simulation systems using different data to represent the simulated entities and their actions can be mediated into a common language to make them comparable, and if they are inconsistent in their overlap, the federation will show inconsistencies as well.

Practitioners apply heuristics to overcome such inconsistencies, but the epistemological constraints will show that is often extremely complicated to identify them without a formal apparatus, as provided by model theory and its applications.

3.0 COMPUTATIONAL CONSTRAINTS

The second block of constraints is in significant parts also based on mathematics. A important part of the interoperability challenge is driven by computer and electrical engineering challenges, such as the choice of signals and their interpretation. Many of these have been solved and are now governed by applicable standards that regulate our network, wireless, and other related connections between those sending the signals and those receiving and interpreting as meaningful symbols. While these technical challenges that enable a common infrastructure are important, the focus of this section are the computer science related challenges that are often more elusive.

3.1 Limits of Logic

Computer systems are based on mathematical logic, and as such constrained by the limits of logic. In 1931, Kurt Gödel’s incompleteness theorem shocked the academic world. Up until his proof, science philosophers considered mathematical logic to be the key for unambiguous, consistent, and complete description of knowledge. Gödel showed that a logical system that is powerful enough to allow for mathematical reasoning will necessarily comprise axioms that are true, but that cannot be proven to be true within the system. Another interpretation is that complex and powerful logical systems can be either complete or consistent, but not both. If the system is complete, it comprises statements that make the system fail. If we exclude these statements to reach consistency, the system is no longer complete.

This well-known research result was used in [9] to motivate the introduction of a reference model, which serves to collect all information available to describe the real-world referent to be simulated. The reference model strives to be complete, which makes it in most cases inconsistent. Simulation systems require consistency, so conceptual models are derived from the reference model that address facets of the system of interest. Each conceptual model contributes to capture a facet of the overall system. Each of these conceptual models can now be implemented in different languages, different modelling paradigms, and on different computer architectures. That alone these variety can lead to various additional challenges has been featured in the research led by Oberkampf [10]. As important as Oberkampf's observations regarding the variety of implementations and resulting possible errors, uncertainties, and inconsistencies is, the observations that the same reference model must result in several conceptual models that due Gödel's incompleteness theorem will produces inconsistencies if federated into a common solution is equally interesting. Although the paper on reference modelling [9] is highly referenced and often discussed on academic conferences, the practical implications of this research are not yet sufficiently understood by simulation practitioners: it is not only possible but highly likely that two simulation systems used to represent the same real-world reference are producing inconsistent results!

3.2 Limits of Computational Support

When simulation experts develop a federation, the following steps must be performed: first, the objectives of the training or study must be articulated; secondly, a set of possible simulation systems that can provide the needed functionality is identified. Then, from this set, the best sub-set is selected that provides all the functionality needed. This subset is then federated and the execution of the federation is orchestrated. It seems natural to think about computer support for these tasks. However, two computational constraints are standing in the way of such support: decidability and computational complexity.

Decidability was first addressed in the Church-Turing thesis, and even more so in the Church Turing Theorem. The thesis deals with computable functions and establishes the equality of algorithmically and Turing machine computable. The theorem addresses that certain classes of decisions cannot be solved by an algorithm; hence they are never going to be generally solvable by a computer, such as "Will the system terminate?", "Are two modelled actions order independent or do I have to orchestrate them?", "Is the specification complete?", "Is the specification minimal?", or "Are two specifications functionally equivalent, in other words, do they deliver the same functionality?" But even if an algorithm can be built, we may still run into problems. *Computational complexity* addresses the challenge to determine if a solution can be found by a computer in reasonable time, using reasonable resources. It classifies several groups of problems that are increasingly harder to solve, e.g., as the time needed to solve a problem grows polynomial or even exponential with the amount of entities needed for a solution. A solution may work for small numbers, but would simply use too much time or too many resources when generally applied. In such case, only heuristics or numerical approximations can help, and brings us back to the problems identified in [10].

Two ground breaking papers were dealing with the implications. Overstreet and Nance [11] who described the Condition Specification (CS) formalism that formally and implementation independently captures the idea of conceptual blueprints (condition) and implementation blueprint (specification) as introduced here as well. In their work, Overstreet and Nance demonstrate that any CS has an equivalent Turing Machine (TM) specification. In other words, our principles followed in our federation designs and implementations are constrained by the same rules as algorithms, including decidability.

Page and Opper [12] extend this work. They observe that intuitively, component-oriented design offers a reduction in the complexity of system construction by enabling the designer to reuse appropriate components without having to re-invent them. However, in their paper they show that this assumption is wrong when applied in the context of CS, or conceptualization and implementation, as defined in [11], as we are introducing new complexity to the problem. Although determining if a collection of components satisfies a set of requirements becomes feasible under certain assumptions, we still must solve a potentially computationally intensive problem, i.e., the selection problem is computationally complex.

Selecting the right component to fit into a federation is a non-trivial task that cannot be generally solved or left to technology. We can apply heuristics, but we cannot provide a general computer based solution.

3.3 Limits of Numerical Solutions

As already discussed in [10], a lot of compromises may happen during the implementation of a model as a simulation system. Systems resources, available tools and compilers, and many other practical constraints often require to use heuristics. Furthermore, numerical approximations are needed to solve differential equations as often used to describe changes in complex systems. The results of such numerical approximations may differ significantly, and be highly dependent on the approximated system. Figure 1 shows an example of the same equation solved using the Euler method and the Runge Kutta method.

This effect can lead to significantly different simulation as results, as the systems we are training with and training for are highly complex in their own right. They have many interfaces, and their relations internally and externally are usually non-linear. This makes them highly sensitive to even slight changes in the initial conditions as well as exchanged parameters, including rounding or truncating of values during the information exchange, or different errors resulting from different numerical approximations.

Furthermore, a system may exhibit chaotic characteristics, normally resulting when mapping the range of a highly non-linear function back to a limited range. An easy example is the logistic map, shown on the right of Figure 1. The exact same program is executed with 32bit and 64bit accuracy, and only due to this different resolution, the results are drifting apart.

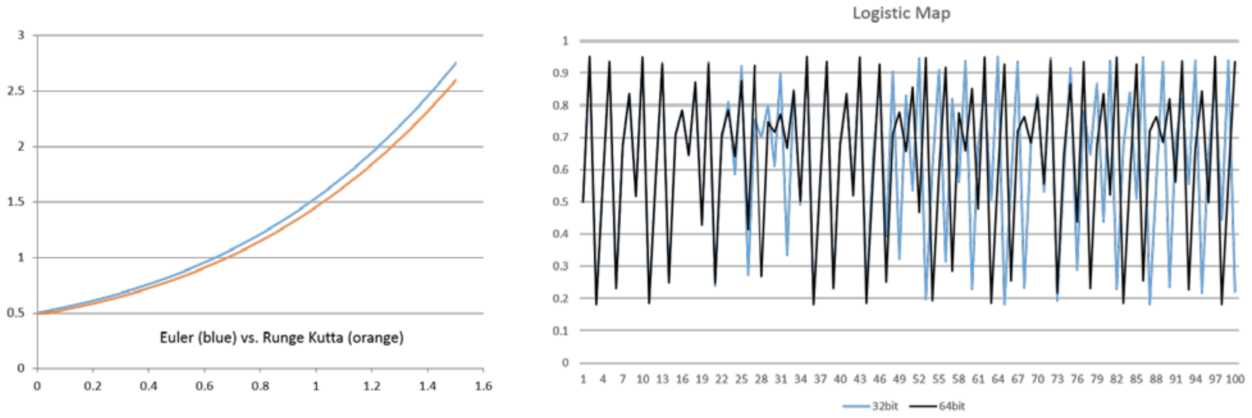

Figure 1: Numerical Approximations and Tracks of a Chaotic Systems [13]

Even fully interoperable systems can therefore show very different results due to such numerical and heuristic effects. No interoperability standard can solve this systemic problem of different system behaviour resulting from these effects. Roy and Oberkampf [14] therefore proposed to include such uncertainty measures into the verification and validation processes.

4.0 EPISTEMOLOGICAL CONSTRAINTS

Although these technical challenges and constraints we addressed so far are not easy to overcome, we have in addition conceptual challenges as well, that are at least as hard to solve. The conceptual level of a discipline is usually addressed by *epistemology,* which deals with gaining knowledge in a scientific discipline. As such, the epistemology of simulation is very important, as it provides the philosophical foundation for simulationist. An important discussion within this realm is what exactly modelling is, as it builds the tenet in our efforts. *Modelling is a task-driven purposeful simplification and abstraction of a perception of reality* [15]. Let's have a closer look at the components of this definition:

- *Task-driven:* a model is generated for a task, such as to answer a question within the domain of analysis or providing a certain functionality, such as supporting training. Like the question that initiates the scientific method, the task drives the modelling process.
- *Purposeful*: modelling is a creative act. The various activities are driven by the task and are done knowingly and purposefully to reach the goal to the greatest extent possible.
- *Simplification and abstraction*: as in scientific experiment described in [13], elements that are not important and only distract from the main event are eliminated from the model. Furthermore, components that may have an effect but are considered secondary or less important can be combined as a form of data reduction techniques. Overall, choosing the right abstraction level is an important decision.
- *Reality*: our work shall be rooted in empirical data, no matter if we assume a positivistic or post-modern world view.
- *Perception*: our perception is shaped by physical-cognitive aspects and constrains. The physical aspect defines what attributes of an object are observable with the sensory system of the observer, or more generally, the information about the object that can be obtained (this can include gaining insight from literature, discussions with colleagues, using instruments, etc.) Cognitive aspects are shaped by the education and the knowledge of the observer, their paradigms and even knowledge of related tools associated with the tasks. Furthermore, legal and moral constraints may limit the obtainability of data.

The result of the modelling phase is a conceptualization shaped by the task, as well as by the physical-cognitive abilities of the modeler. This conceptualization is then captured using a modelling method and subsequently transformed into an executable simulation. These principles are well captured in the *Semiotic Triangle*, which has been introduced in [16] to cope with the challenge that the same real world referent can be perceived differently, leading to different conceptual representations thereof. The symbolic representation describes these concepts, but is often perceived to stand for the real-world referent. Thus, we often speak "past each other," as we agree on the real-world referent and the symbols we use to describe it, but we do not align our concepts we use to think about them. We talk about concepts of the real-world referent in our mind, not about the referent itself. These principles are captured in the left picture of Figure 2.

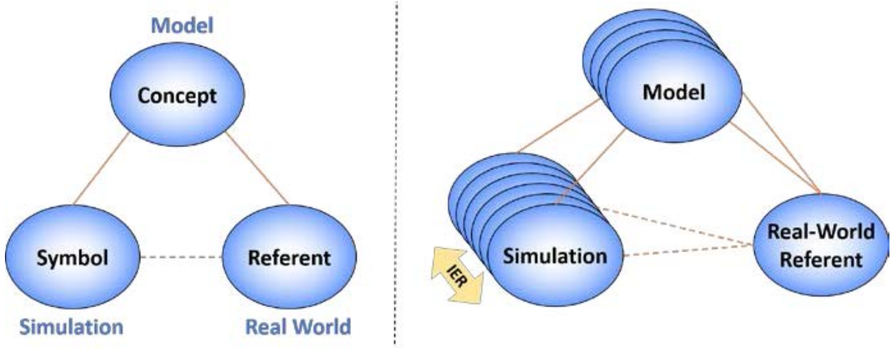

Figure 2: Semiotic Triangle for Modelling and Simulation and Interoperability Standards

When we are using simulations, the same principles apply. As discussed above, the models are the conceptualizations of our real-world referents. They are the "reality" of the implementing simulations. However, when we build and design our federations, we use the real-word referents as our yardstick (concretely, we use yet another conceptualization that represents what we expect for training events that we want to support). However, our simulation interoperability standards [2, 3] focus on the information exchange requirements (IER) derived from the real-world referents for the simulation systems on the level of the simulations, not the underlying conceptualizations. As such, they are not designed to resolve – or even discover – conceptual misalignments. Thus, even with simulation interoperability standards in place, our simulation systems can talk "past each other" as observed [16] for human actors. These epistemological constraints must be addressed to avoid that we are not trying to federate systems with each other that conceptually were never meant to be combined.

5.0 INTEROPERABILITY AND COMPOSABILITY

To provide an easy but powerful framework to address the various challenges of interoperability and composability, the Levels of Conceptual Interoperability Model (LCIM) was first introduced in slightly different form in [17] and continuously developed further with input from many applications domains – such as defence and security, health and human sciences, energy and more – to the current form [18] shown in Figure 3.

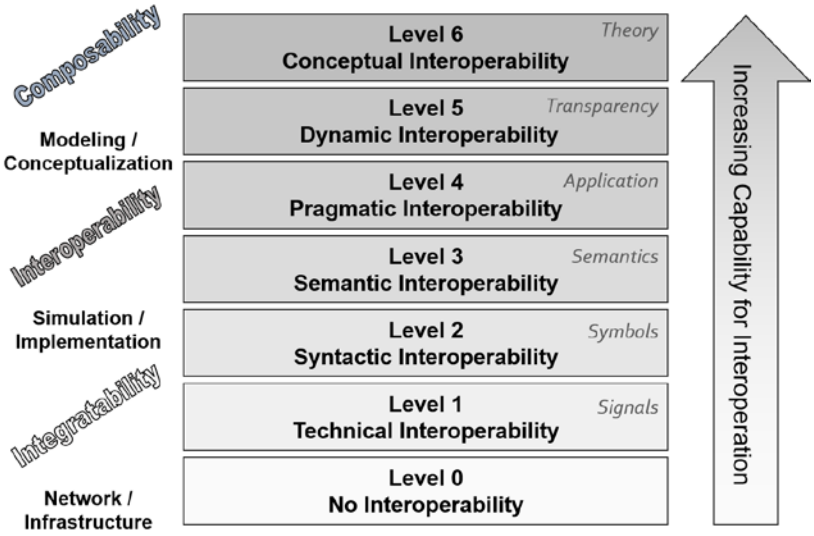
Composability
Modeling / Conceptualization
Interoperability
Simulation / Implementation
Integratability
Network / Infrastructure
Level 6 Conceptual Interoperability
Theory
Level 5 Dynamic Interoperability
Transparency
Level 4 Pragmatic Interoperability
Application
Level 3 Semantic Interoperability
Semantics
Level 2 Syntactic Interoperability
Symbols
Level 1 Technical Interoperability
Signals
Level 0 No Interoperability
Increasing Capability for Interoperation

Figure 3: The Levels of Conceptual Interoperability Model

The current version of the LCIM exposes six layers of interoperation, namely:

- The *technical layer* deals with infrastructure and network challenges, enabling systems to exchange signals, the carriers of information.
- The *syntactic layer* deals with challenges to interpret and structure the information to form symbols within protocols.
- The *semantic layer* provides a common understanding of the information exchange. On this level, the pieces of information that can be composed as objects, messages, and other higher structures are identified.
- The *pragmatic layer* recognizes the patterns in which data are organized for the information exchange, which are the inputs and outputs of procedures and methods to be called. This is the context in which data are exchanged as applicable information. These groups are often referred to as business objects, as they are identified in business process models or comparable architecture artefacts.
- The *dynamic layer* recognizes various system states, including the possibility for agile and adaptive systems. The same business object exchanged with different systems can trigger very different state changes. It is also possible that the same information sent to the same system at different times can trigger different responses. This layer provides transparency.

- Finally, assumptions, constraints, and simplifications need to be captured. This happens in the *conceptual layer*. This layer addresses the alignment of conceptualizations.

These levels are well aligned with the three governing concepts of interoperation proposed in [19]. The concept of *integrateability* contends with the physical/technical realms of connections between systems, which include hardware and firmware, protocols, networks, etc. *Interoperability* contends with the software and implementation details of interoperations; this includes exchange of data elements via interfaces, the use of middleware, mapping to common information exchange models, etc. Finally, *composability* contends with the alignment of issues on the modelling level. The underlying models are purposeful abstractions of reality used for the conceptualization being implemented by the resulting systems. *Successful interoperation of solutions requires integrateability of infrastructures, interoperability of systems, and composability of models*.

There are significant implications for current efforts, such as Modelling and Simulation as a Service (MSaaS) [20] or Modelling and Simulation in the Cloud [21]. For successful cloud projects, five views are necessary:

- *Technical View* focuses on the technical requirements of infrastructures, protocols, information exchange formats, etc.
- *Governance View* defines consistent management, cohesive policies, guidance, processes and decision processes. In the cloud environment, it must be defined who is responsible for which action, such as component updates, introduction of new resources, maintenance, etc.
- *Business View* support the fair share of the financial burden of setting up and conducting a distributed simulation event, clear rules and value assessment are needed. Clear contracts that regulate the use of components and resources are needed as much as the value added by organizational contributions to the cloud.
- *Security View* addresses cyber security as a growing concern. It is insufficient to know how someone can conduct certain changes, it must also be assured that the person, organization, or service has the authentication required, via which access points such actions are allowed, etc.
- *Conceptual View* addresses the need for conceptual alignment of the models to support composability in addition to the means required for interoperability of the simulation components. This viewpoint is unique to model-based solutions and needs to be addressed for cloud-based M&S.

A coherent approach that aligns technical necessities, constraints and possibilities with supporting governance of all components and resources harmonized with the appropriate business model ensuring security and conceptual consistency is needed and requires a multidisciplinary approach.

MSaaS is envisioned to allow for the rapid deployment of services that can easily be identified, selected, and composed into a new set of functionality. Their use of standard interfaces and descriptions broadly utilized in commercial information technologies allows the developer to combine them relatively easily on the technical level, as successfully demonstrated in [20]. The conceptual alignment, however, is yet unsolved. The use of metadata to support semantic and higher interoperability has been identified, but which metadata is necessary to support the higher levels of interoperability is a topic of ongoing research [1, 18].

6.0 SUMMARY

The purpose of this paper was to provide a landscape of challenges in the mathematical, computational, and epistemological domain of modelling and simulation. If these challenges are not address, the result of current efforts may include inconsistences and contradictions that may lead to system crashes, but the worst case is that the systems will not crash, the user doesn't realize that his supporting simulation federation is flawed, and the resulting training is unfair and inefficient, or his/her decision is based on unjustified data and derived recommendations.

There are no technical solutions for conceptual problems! NATO needs to establish governance bodies to ensure interoperability and composability – specifically, the successful exchange of data that can be used in the receiving system without resulting in inconsistent representations of truth in any of the participating systems. The governance body must manage all conceptual challenges, including definition and use of metadata, all rooted in rigorous mathematical principles.

REFERENCES

[1] Taylor, Simon J. E, Azam Khan, Katherine L. Morse, Andreas Tolk, Levent Yilmaz, Justyna Zander, and Pieter J. Mosterman. "Grand Challenges for Modeling and Simulation: Simulation Everywhere - From Cyberinfrastructure to Clouds to Citizens," *SIMULATION* vol.91, (2015): 648-665.

[2] IEEE 1278: *Distributed Interactive Simulation (DIS)*, multiple volumes, latest version accessible via https://standards.ieee.org

[3] IEEE 1516: *Standard for Modeling and Simulation (M&) High Level Architecture (HLA)*, multiple volumes, latest version accessible via https://standards.ieee.org

[4] IEEE 1730-2010: *Recommended Practice for Distributed Simulation Engineering and Execution Process (DSEEP)*, accessible via https://standards.ieee.org/develop/wg/DSEEP.html

[5] IEEE 1730.1-2013: *Recommended Practice for Distributed Simulation Engineering and Execution Process Multi-Architecture Overlay (DMAO)*, accessible via https://standards.ieee.org/develop/wg/DSEEP.html

[6] Hoppe, Thomas, Bernhard Humm, Ulrich Schade, Timm Heuss, Matthias Hemmje, Tobias Vogel, and Benjamin Gernhardt. "Corporate semantic web–applications, technology, methodology." *Informatik-Spektrum* 39, no. 1, (2016): 57-63.

[7] Weiss, William and Cherie D'Mello. *Fundamentals of Model Theory.* Department of Mathematics, University of Toronto, Canada, 1997. Accessible via http://www.math.toronto.edu/weiss/model_theory.pdf.

[8] Diallo, Saikou Y., Jose J. Padilla, Ross Gore, Heber Herencia-Zapana, and Andreas Tolk. "Toward a formalism of modeling and simulation using model theory," *Complexity*, Vol. 19, Iss. 3, (2014): 56-63.

[9] Tolk, Andreas, Saikou Y. Diallo, Jose J. Padilla, and Heber Herencia-Zapana. "Reference Modeling in support of M&S–Foundations and Applications," *Journal of Simulation* Vol. 7 no. 2, (2013): 69-82

[10] Oberkampf, W. L., S. M. DeLand, B. M. Rutherford, K. V. Diegert, and K. F. Alvin. "Error and Uncertainty in Modeling and Simulation." *Reliability Engineering & System Safety* Vol. 75 Iss. 3 (2002): 333-357.

[11] Overstreet, C. Michael, and Richard E. Nance. "A specification language to assist in analysis of discrete event simulation models." *Communications of the ACM* 28, no. 2 (1985): 190-201.

[12] Page, Ernest H. and Jeffrey M. Opper. “Observations on the complexity of composable simulation.”. *Proceedings of the Winter Simulation Conference*, Vol. 1. ACM, New York, NY, USA, 553-560, 1999.

[13] Tolk, Andreas. “Bias ex Silico – Observations on Simulationist’s Regress,” *Proceedings of the Annual Simulation Symposium,* SCS Spring Simulation Multi-Conference, Virginia Beach, VA, 314-322, 2017.

[14] Roy, Christopher J., and William L. Oberkampf. "A comprehensive framework for verification, validation, and uncertainty quantification in scientific computing." *Computer methods in applied mechanics and engineering* 200, no. 25 (2011): 2131-2144.

[15] Tolk, Andreas. “Interoperability, Composability, and their Implications for Distributed Simulation: Towards Mathematical Foundations of Simulation Interoperability,” *Proceedings of the ACM/IEEE DS-RT Conference*, Delft, The Netherlands, pp. 3-9, IEEE Press, 2013.

[16] Ogden, Charles Kay, and Ivor Armstrong Richards. *The meaning of meaning: A study of the influence of thought and of the science of symbolism.* University of Cambridge, UK, 1923.

[17] Tolk, Andreas, and James A. Muguira. "The Levels of Conceptual Interoperability Model." *Proceedings of the Fall Simulation Interoperability Workshop*, Orlando, Florida, 2003.

[18] Tolk, Andreas, Lisa J. Bair, and Saikou Y. Diallo. “Extending the Levels of Conceptual Interoperability Model towards an Interoperability Maturity Model,” *Journal for Defense Modeling and Simulation*, Vol. 10 No. 2, (2013): 145–160.

[19] Page, Ernest H., Richard Briggs, and John A. Tufarolo. "Toward a family of maturity models for the simulation interconnection problem." *Proceedings of the Spring Simulation Interoperability Workshop*, Orlando, Florida, 2004.

[20] van den Berg,Tom, Wim Huiskamp, Robert Siegfried, Jon Lloyd, Amy Grom, and Robbie Phillips. “Modelling and Simulation as a Service: Rapid deployment of interoperable and credible simulation environments – an overview of NATO MSG-136.” *Proceedings of the Fall Simulation Interoperability Workshop*, Orlando, Florida, 2017.

[21] Cayirci, Erdal. "Modeling and Simulation as a Cloud Service: A Survey." Proceedings of the Winter Simulation Conference, pp. 389-400. IEEE Press, 2013.

This paper has been approved for public release, distribution unlimited, by The MITRE Corporation Case No. 17-2588. The author's affiliation with The MITRE Corporation is provided for identification purposes only, and is not intended to convey or imply MITRE's concurrence with, or support for, the positions, opinions, or viewpoints expressed by the author.