JSIME *Journal of Simulation Engineering*, Volume 1 (2018). Article URL: https://articles.jsime.org/1/4

Panel Discussion: On the Unity and Diversity of Computer Simulation

Alexis Drogoul
alexis.drogoul@ird.fr
UMMISCO, IRD, France and Vietnam

Paul Fishwick
paul.fishwick@utdallas.edu
University of Texas at Dallas, United States

Nigel Gilbert
n.gilbert@surrey.ac.uk
University of Surrey, United Kingdom

Dennis Pegden
cdpegden@simio.com
Simio LLC, United States

Gerd Wagner ✉
G.Wagner@b-tu.de
Brandenburg University of Technology, Germany

Levent Yilmaz
yilmale@auburn.edu
Auburn University, United States

Abstract

The term *Computer Simulation* subsumes different simulation paradigms, languages and implementation technologies as well as many different application areas each with its own scientific communities. So, there is clearly a lot of conceptual, methodological, technological and application diversity in the area of Computer Simulation. From its start in 1967, the Winter Simulation Conference managed to get four scientific communities involved: computer scientists, electrical engineers, industrial engineers and mathematicians (operations researchers). Only later, in 2011 and 2012, an attempt was made to get environmental and social scientists involved who have been adopting the idea of "individual-based" or "agent-based" simulation. Today, two American, a European and an Asian social simulation conference have been established. How much unity exists between the scientific areas and communities represented by the Winter Simulation Conference? How much unity exists between the scientific areas and communities represented by the newer social science simulation conferences? And how much unity exists between Discrete Event Simulation and the newer forms of social science simulation? These and other questions about the unity and diversity of Computer Simulation have been discussed via email from April 17 to May 17, 2018, by five leading experts: Alexis Drogoul, Paul Fishwick, Nigel Gilbert, Dennis Pegden and Levent Yilmaz, moderated by Gerd Wagner.

Introduction of Panelists

Alexis Drogoul graduated in Artificial Intelligence in 1990, became full professor in 2000 and joined the Institut Français de Recherche pour le Développement (IRD) as a Senior Researcher in 2004. He is mostly working on agent-based modeling and simulation with applications to Environmental Decision-Support Systems. He has co-designed, and still contributes to, the Java-based GAMA platform, which combines multi-agent simulation with Geographic Information Systems (GIS) capabilities.

Paul Fishwick is Distinguished University Chair of Arts, Technology, and Emerging Communication, and Professor of Computer Science at the University of Texas at Dallas, where he is active in forging new bridges between the arts, engineering and computer science, and involved in teaching undergraduates at the intersection of humanities and Modeling & Simulation. While in the 1980s and 90s, he has written articles about many issues of simulation, being one of the first who investigated the potential of web-based simulation, he increasingly focused on the connections between arts and simulation in the 2000s, coining the term *Aesthetic Computing*.

Nigel Gilbert read for a first degree in Engineering, initially intending to go into the computer industry. However, he was attracted into sociology and obtained his doctorate on the sociology of scientific knowledge from the University of Cambridge. He is the founder and director of the Centre for Research in Social Simulation at the University of Surrey and (co-)authored several textbooks in the area of computational social sciences. He was the founding editor of the peer-published Open Access Journal of Artificial Societies and Social Simulation. He counts as one of the founders of modern *Computational Sociology*, a discipline that merges social science research with simulation techniques with the goal of modelling complex policy issues and fundamental aspects of human societies.

Dennis Pegden is the founder and CEO of Simio

LLC, one of the leading vendors of Discrete Event Simulation software packages. Prior to this position, in the 1970s and 80s, he earned a Ph.D. from the School of Industrial Engineering at Purdue University, held faculty positions at the University of Alabama in Huntsville and The Pennsylvania State University and founded Systems Modeling Corporation, which developed the famous Arena simulation software package. He also led in the development of the SLAM and SIMAN simulation languages, which have been milestones in the history of Discrete Event Simulation. He is the author/co-author of three textbooks in simulation and a frequent speaker at the Winter Simulation Conference.

Gerd Wagner, who moderates the panel discussion, is Professor of Internet Technology at Brandenburg University of Technology, Germany, and a member of the *Consortium for True Open Access in Modeling and Simulation*, which publishes the *Journal of Simulation Engineering*.

Levent Yilmaz is Professor of Computer Science and Software Engineering at Auburn University with a joint appointment in Industrial and Systems Engineering. His research interests are in agent-directed simulation, cognitive computing, and model-driven science and engineering for complex adaptive systems. He co-authored a 2016 book with the title “Distributed Simulation: A Model-Driven Engineering Approach". He is the founding organizer and general chair of the Agent-Directed Simulation Symposium series. ”

Panel Discussion

Gerd Wagner: As you can see from this list of biographies, we do have a quite diverse expertise and background set in our panel. I would say that it includes Computer Science, Industrial Engineering, Sociology and Artificial Intelligence. Is this type of diversity unique for (the area of) Computer Simulation? Do you feel that you benefit from this type of diversity when attending general simulation conferences or reading simulation journals?

Paul Fishwick: One of the things that drew me toward modeling and simulation (M&S) as a field in the late 1980s was that M&S had a significant variety of theorists and practitioners. I found this refreshing compared to other communities that I “tested” during the same period. One of the reasons for the variety seems to come from modeling as an interdisciplinary subject – everyone models, but we may define “model” differently.

Levent Yilmaz: I find the Modeling and Simulation (M&S) discipline unique in terms of both the degree and the types of diversity that it affords. Given the transdisciplinary nature of model-building, there is always the opportunity to have cross-disciplinary interactions with others from a wide range of scientific and engineering fields. In my experience, there also appears to be sustained diversity even within specific fields of the M&S discipline. For instance, at a recent Winter Simulation Conference, two different tracks focused on the use of computational agents, yet with distinct objectives. The cross-fertilization of ideas among the members of both the Social Behavioral Simulation and the Agent-Directed Simulation communities were noteworthy. Such diversity merits attention, at least subjectively, because it makes the interactions more interesting. After all, if members of a scientific community offer similar views and have the same preferences, no one would have the incentive to trade ideas. On the other hand, diversity, alone though, may not be sufficient. Interestingness requires some level of uniformity to sustain meaningful communication.

The Summer Simulation Conference, which is organized by the Society for Modeling and Simulation International, has been sustainable for over six decades. Similarly, in 2017, the Winter Simulation Conference celebrated its 50th anniversary. Can the diversity afforded by the M&S discipline explain the resilience of these conference series? Why do we see diversity within the M&S discipline? Is it merely because of the transdisciplinary nature of the discipline, or are there other mechanisms in play? On the other hand, can too much diversity impede meaningful and useful progress?

Gerd Wagner: Like Paul, I also find the “richness” of M&S fascinating. But as noted by Levent, meaningful communication among researchers from different disciplines requires some degree of shared understanding of issues, or conceptualization, of problem domains and methodological approaches. For those of you who have attended multi-disciplinary simulation conferences like the Winter (or Summer) Simulation Conference, did you experience sufficient shared understanding or have you experienced situations where researchers live/work in parallel worlds?

Alexis Drogoul: As a computer scientist building and maintaining a simulation platform, whose role is mainly to operationalise and “objectify” the concepts manipulated by different disciplines in order for these “objects” to be later manipulated easily by modellers, I am constantly facing situations (in conferences, projects, panels, etc.) where researchers from different domains seem to live in parallel worlds – while they actually (in my biased perception) talk about the same things, at least the same abstractions of structures and phenomena. I find it normal, as I see my role as that of a “facilitator” or “translator”: i.e. someone who does not force people to use the same language but instead builds and provides abstractions that help them to express their own concepts in models. Diversity is therefore more a necessity than a problem, at least for me. And this is what makes this field so interesting to work in.

Gerd Wagner: Speaking about M&S abstractions, I have the impression that the M&S research discipline suffers from too much conceptual and terminological diversity, and not enough unity, compared, e.g., with Statistics, which has a similar large diversity in application domains, but much more conceptual and terminological unity. Consider, for instance, the term "agent-based" M&S and the proposed abstraction of "agents". It is being used with at least two fundamentally different meanings: as an ontological category for capturing those special entities of a problem domain that are able to interact with each other and their environment ("actors"), and as a computational concept similar to "active objects" in object-oriented programming.

Dennis Pegden: As someone involved in simulation modeling tool development, the challenge has been to abstract the common behavior from a broad set of diverse systems. When properly viewed, a manufacturing facility, hospital emergency department, supply chain, call center, cloud-based computer system, restaurant, mining operation, etc. can all be described using a common set of constructs. Although these systems appear to be very different at first glance, they are actually very similar.

Gerd Wagner: Dennis, I guess you are referring to what I prefer to call *processing networks* (PNs) consisting of

1. entry nodes (in Simio called "Source" objects)
2. processing nodes ("Server" objects)
3. exit nodes ("Sink" objects)

when you mention a "common set of constructs" for modeling the systems that you have listed where certain objects (work pieces, trucks, passengers, patients, etc.) move through a system that is constrained by processing resources (machines, pathways, security check points, doctors, etc.). PN models are an important class of simulation models, but I don't think that they represent a general paradigm that can be used for modeling any kind of system. And PN modeling suffers from conceptual/terminological diversity: entry nodes are called "Source" in Simio and AnyLogic, "Create" in Arena and "Start Point" in Simul8, processing nodes are called "Server" in Simio, "Service" in AnyLogic, "Process" in Arena and correspond to a combination of "Queue" and "Activity" in Simul8. Why do we (researchers, students and potential users of PN modeling tools) have to struggle to come to grips with this confusing terminological diversity? Modeling tools in other areas (e.g., software modeling with UML, statistical modeling), even from different vendors, do not have this conceptual/terminological diversity.

Paul Fishwick: We are discussing abstraction, and I agree that this is something of vital importance to modeling. I also find it in my academic environment, to be something difficult to teach. Mathematics usually offers us a way of continuing up the abstraction ladder as far as we need to go. For example, the "networks" being discussed are graphs and in Computer Science, we often speak of data vs. control flow graphs. This is a fairly broad understanding of how discrete systems can be modeled. One might even combine both types of graphs into one, even more, abstraction: graph. This becomes an issue of vocabulary and, more generally, language.

The vocabulary of particular disciplines and software packages tends to obscure high level discussions of concepts. I find disciplines most confining. For example, if inside of an Operations Research Department, we may speak of certain processes appropriate for industry. In a Computer Science department, we speak of processes appropriate for computer hardware and software. However, there is a high level of abstraction that extends beyond the two departments. This level needs to be communicated, not to the exclusion of domain knowledge, but to complement that knowledge.

Yesterday, in my modeling class, I challenged students to take a modeling concept found in Computer Science – a *Finite State Machine (FSM)*. But I asked them to find (non-computer) things in the class room whose behavior could be captured by an FSM. The problem is that we frequently fail to teach the Computer Science students that an FSM is a mathematical construct, and so, a broad interdisciplinary one. If we get stuck within our disciplines, we'll fail to find FSMs in the park, car wash, or operating in a magnolia tree.

Gerd Wagner: Mathematical abstractions, like FSMs, ODEs, Petri Nets or Cellular Automata, can be used for making mathematical models, but for making simulation models of real world systems, we need computational abstractions that capture the ontological/metaphysical categories allowing us to conceptualize real world systems and turn them into computational simulation models. An important milestone in the history of computer simulation and in the history of computing was Simula's (1966) introduction of a computational object concept that allowed representing real world objects, which are one of the two most fundamental ontological categories (also called "endurants"). The other fundamental ontological category are events/processes (also called "perdurants"). Events have been introduced to computer simulation by SIMSCRIPT (in 1962), and a formal semantics for event-scheduling-based simulation models, *Event Graphs*, has been proposed by Schruben (1983). It seems to me that an Event Graph is not a typical mathematical abstraction, but rather a computational abstraction, like UML Class Diagrams or Colored Petri Nets, because they go beyond using a simple set of variables for state structure modeling.

Paul Fishwick: On Gerd's response, there seems to be some questions of vocabulary. Everything is a math-

ematical model if we acknowledge that mathematics is comprised of mental frameworks (with everything else, including standard lexical notation) being representation (i.e. modeling). Computer Science is applied mathematics – mainly seeming from discrete mathematics, which was originally taught (like CS) in math departments and, with other subjects like numerical analysis, has shifted to CS departments.

I am not trying to take away from disciplinary research accomplishments, but merely trying to establish that at the highest abstraction level, we have mathematical thought (regardless of how the mathematics is represented). For your point about *Object-Orientation (OO)*, virtual everything in OO is based on ideas of set theory laid out in the 19th century. Encapsulation (set membership), hierarchies (partial sets), and inheritance (grafted from evolutionary biology). This is why OO can be taught completely independently of the machine or its software – it is a set of mathematical abstractions.

Event graphs are the duals of state machines. In my 1994 text, I define the event graph as a finite event automaton. Event graphs are a very useful graph application and approach to automata. Layers upon layers of abstractions.

Levent Yilmaz: As we are on the topic of abstraction, let me follow up on the significance of connecting abstractions to the constructs/phenomena that we observe in our environment. Such phenomena tend to be complex due to its multiple aspects, facets, and scales, which call for a diverse set of abstractions. Sometimes, as Paul indicated, an OR expert's representation/terminology of a process may be distinct than a Computer Scientist's vocabulary. Yet, both of which may be instances of an overarching meta-model (an abstract definition of process). The provision of such meta-models help mediate and bring uniformity. This is an example of how uniformity and diversity is balanced in practice. There are other effective recurring strategies that stem from seeking a balance between diversity and uniformity.

For instance, in cases where representations focus on distinct aspects, there is a need for coherence and synergy to accurately capture the behavioral richness and complexity of the things that we want to build or explain/understand. As we ground our models in theory and aim to align them with empirical observations, we tend to be confronted by a set of problems that cannot be addressed by a single formalism. The interdisciplinary nature of M&S appears to stem from such situations that require mediation across multiple diverse representations, each one of which may focus on aspects that are pertinent to a specific disciplinary point of view. The diversity in M&S may have something to do with the complexity of the world we are trying to understand and influence.

Gerd Wagner: Paul and Levent, you have been both pointing to the important issue of modeling processes. Can you suggest any approach or, at least, requirements for a process modeling language? Arena/Simio/AnyLogic etc. support “processing processes” where certain objects are processed along the nodes of a network (called “queueing network” in OR). But not every process is a processing process. Especially in social simulation, this abstraction isn't used at all.

Paul Fishwick: There are numerous languages, and software packages supporting capturing the idea of process. Most of the work for discrete processes is laid out in automata languages, enhanced and made more specific in systems theory. For continuous systems, we have the calculus. On social simulation, perhaps Nigel has the best answer. I would say, processes don't mind where they are located. A process can be defined at a top-level for a plant operation, but it also can lurk inside an animal. In ecology, they refer to “individual-based modeling”. Regardless of whether we use “agent”, “object”, “human”, the process is both inside the objects as well as outside (in the environment as in field theories of physics – gravitational attraction, for example).

Gerd Wagner: Of course, different concepts are needed for modeling discrete and continuous state changes in dynamic systems. I assume that we are not really concerned much with modeling continuous state changes (we leave this to mathematics and science). But we cannot be content with having “numerous languages capturing the idea of [discrete] process”, because this is a state of a scientific area paralyzed by conceptual diversity.

Levent has been asking for “an overarching meta-model”. Do you have any ideas for this? And would it make sense to ask for an abstraction of discrete processes that subsumes discrete manufacturing processes (exemplifying discrete event processes), ecological processes (with “individual-based” models) and social processes (with “agent-based” models)?

Levent Yilmaz: I am not sure if it is reasonable to expect a single overarching meta representation to subsume a diverse set of formalisms and languages. However, it is probably still effective to continue to increase the level of abstraction while also maintaining transformations that map abstract models onto not only different languages and formalisms but also target platforms that continue to evolve. To cope with the increasing complexity and diversity of platforms and formalisms, we probably need innovations in meta simulation programming infrastructures – simulation programs that generate other simulation programs in a staged manner – that use abstract concepts for specification along with generators to produce simulation code appropriate for a specific purpose. Clearly, though, there will be challenging trade-offs among dealing with complexity, productivity, and performance, and these trade-offs need to be carefully balanced. So, I expect to see the emergence of a loosely

coupled ecology of such innovations growing and evolving with selective pressures coming from the developers as well as end users of such applications. From that perspective, the diversity of the domain (practices, knowledge, and skills) appears to be closely coupled to the diversity of the participants (field) of a discipline. They appear to feedback to each other.

Dennis Pegden: Although it is difficult to define a single abstract framework for discrete models, I think it is possible to view objects (agents), processes, and event frameworks as related abstractions for defining discrete system behavior. Events are typically defined in code or state diagrams and provide the base framework for defining state transitions that occur at a single instant of time. Processes are described with flow charts and define a collection of events that execute state transitions that span time. Objects (agents) are a collection of processes or events that define the behavior of a physical component of a system. A discrete simulation model is a collection of events, processes, and/or objects that describe the system.

Gerd Wagner: Notice that classical flow chart languages have been used in business process modeling (BPM), but without a well-defined semantics. Today, the *BPM Notation (BPMN)* has been adopted by the BPM community as a modeling standard, while in M&S we still don't have any comparable standards for discrete process simulation modeling. In most M&S textbooks, you still see sketchy old-style simplistic flow charts. Why didn't the M&S community make any progress in developing (widely adopted) standards?

@Alexis and Nigel: do you have any views on the question of common abstractions, like meta-models or modeling concepts and languages, that can serve as a foundation for modeling various kinds of discrete dynamic systems, including ecological and social systems? Or would you like to bring up any other issue/question, e.g., on the relationship between these different kinds of systems?

Nigel Gilbert: Or UML? See http://jasss.soc.surrey.ac.uk/15/1/9.html which describes the use of UML for agent-based modelling.

Levent Yilmaz: Our students use UML in their Software Modeling and Design class. They find the provision of different UML model types (i.e., class, state, interaction models) quite useful in describing the distinct aspects of both the structure and the behavior of systems. On the other hand, achieving consistency and coherence across multiple models (e.g., system sequence diagrams versus statecharts) continues to be a challenge in practice. Also, the lack of a consensus on the formal semantics of UML models appears to stifle progress in developing executable UML models.

Gerd Wagner: Nigel, the paper by Hugues Bersini rightly argues that UML provides useful diagram languages, not only for modeling OO software systems, but also for making (platform-independent) agent-based models at a higher abstraction level than OO programming languages like Java and NetLogo. However, he doesn't explain that the most important/useful types of models are class models/diagrams, which allow modeling the state structure of a discrete dynamic system, and which can be directly transformed into class definitions of a target OOP language (Java classes or NetLogo breeds). UML Class Diagrams are widely used in the IT industry, especially for model-driven software development. The process/behavior modeling languages (Sequence Diagrams, State Machine Diagrams and Activity Diagrams) are not as widely used and they have a fundamental weakness: unlike BPMN, they do not include a general concept of events. This is the reason why only BPMN can be viewed as an extension of Event Graphs, which makes BPMN especialy suitable for modeling discrete event processes like discrete manufacturing processes or message-based inter-agent communication processes.

@Levent: UML Class Diagrams do have a formal semantics. In fact, they can be transformed into OWL Ontologies, which are a formal logic representation.

Alexis Drogoul: In the numerous training sessions and classes we organise on or using GAMA, UML has always been a good candidate for semi-formal descriptions of the components and processes found inside models. One of its main advantages, in my opinion, is that it is now widely used in different domains. And the lack of formal semantics (that is, if you do not translate them into OWL ontologies) does not hurt. As a matter of fact, I would even argue that it is precisely because it doesn't carry too much formal underpinnings that it has become so popular. A bit like “agents” if you want. I have always found that in order to make people from different disciplines work together and agree on a shared representation, it is better to start with a weakly formalised language or representational framework. Usually, we base the first steps of a modelling process on a combination of ODD (for the overall logic) and UML (for a more precise description of the “inside”), and then we progressively refine it down to the code.

Gerd Wagner: For those who don't know the acronym: the *ODD (Overview, Design concepts, and Details)* “protocol” was proposed by Grimm et al (2006) as a standard model documentation format for ecological simulation models, and it has also been used in social simulation. It defines a fixed structure of seven documentation sections: (1) Purpose, (2) Entities, state variables, and scales , (3) Process overview and scheduling, (4) Design concepts, (5) Initialization, (6) Input data, and (7) Submodels (see https://www.ufz.de/export/data/2/100066_ODD_Update.pdf).

It seems that in the areas of ecological and social simulation, there is more agreement on the use of stan-

dard formats and modeling languages compared to the area of traditional Discrete Event Simulation (DES) in manufacturing and services industries.

@Dennis: Is my observation correct that DES software vendors, including Simio, prefer to sponsor their own proprietary process diagram languages and do neither promote the use of UML Class Diagrams for modeling entity types (state variables) nor the development of simulation industry standards for process modeling?

Paul Fishwick: I'd like to introduce the idea of balance between polarities. On one hand, we have a need to talk about concepts that rise above disciplines. On the other, we have different modeling cultures. By modeling culture, consider the Petri net or System Dynamics communities. These groups promote particular world views on modeling – different text-based or visual languages to accompany those views. Standardization is a push in "one method for many" and they help us define concepts. However, I don't think we should always err on the side of standardization since we must recognize that, as with natural language, multiple modeling cultures exist. Speaking of UML, I am an adherent; however, it does not go far enough up the abstraction ladder. Ideas expressed in UML are applicable outside of software and hardware. For example, the class diagram idea is built off formal logic and can describe virtually anything from the structure of a bird's nest to how a cathedral is architected. That we limit our discussion outside of such realms indicates our current vocational emphasis in academia. What is more important – preparing students for jobs or teaching them fundamental, discipline-free, concepts? Both are important.

Gerd Wagner: I would rather describe standardization as a push to agree on "one language for one paradigm/method". I think having different research cultures based on different simulation paradigms is a good thing. But for being able to compare and evaluate them, they need to be well-defined, preferably by means of an established standard. For instance, the System Dynamics community has recently made an effort to develop such a standard called SMILE (and XMILE for its XML serialization/interchange format). And we've heard that the ecological/social simulation community is using the ODD standard. However, I am not aware of a similar effort by the Discrete Event Simulation community. Does this mean that after 50 years of Discrete Event Simulation research, the field is still not mature enough for being able to agree on standards?

Levent Yilmaz: Another example for multi-polarity is the tension between precision, generality, and accuracy. This conflict was originally observed by Levins in his seminal paper on the strategy of model building. It is often desirable to develop models with a reasonable degree of generality, precision, and accuracy (realism); however, because it is difficult in practice to achieve all three criteria together, alternative strategies for model building appear to have emerged:

1. By sacrificing generality to precision and realism, some model developers choose to construct high-fidelity/resolution models that aim to mimic the activities in the problem domain as they are observed. Trace-driven simulations are good examples for such modeling applications. A virtual flight simulator and many simulation games are intended to improve realism at the expense of generality.
2. Models that sacrifice realism to generality and precision are usually in the form of meta-models (regression, classifiers etc.) or equation-based models that are inductively derived from data. Many machine learning models that are based on rule induction may also be classified under this category. These models may be general enough to account for the observed data, but they do not often capture the underlying casual behavior; hence, it is a challenge to use these models for predictive or exploratory analysis.
3. The models that sacrifice precision are akin to flexible and highly abstract agent models such as the Schelling model, the Prey-Predator models that explain competitive access to resources etc. The use of artificial assumptions in such models results in doubt about whether the results are due to the essentials of the model or the simplifying assumptions. Such problems do not exist in precise and accurate models such as maps where continuity and relative distance in the representation is consistent with the distances in reality.

I think this tension between precision, accuracy, and generality is why we need alternative models with distinct simplifications, yet underlined with a common assumption. There will be other models that vary these simplifying assumptions. If all these models result in similar regularities, we can have a more robust understanding of the phenomena of interest.

Dennis Pegden: I think the key challenge to standardization of modeling frameworks is the fact that they are still evolving. Although some aspects of modeling terminology could certainly be standardized, it is more difficult to standardize the framework without handcuffing creativity in terms of modeling constructs.

As an example, consider the basic concept of an Entity that flows through the system. This construct has been around for a half of a century using differing terminology (e.g. a transaction in GPSS), but has different meanings in different modeling tools. In Arena the Entity has no intelligence and its only purpose is to flow from block to block, executing the logic associated with a process flow, and thereby change the system state. In Simio an Entity is a full-fledged object and can interact

with other Entities and make decisions of its own. It does not need to execute a process flow to change the state of the system. Entities have user-definable behaviors and can form the basis of an agent-based model. Hence the term Entity in Simio has a completely different meaning than Entity in Arena (Tokens in Simio are analogous to Entities in Arena).

Many modeling tools incorporate the concept of a Resource that can be used by Entities, however it's very difficult to standardize the meaning of a Resource. In many modeling systems Resources are separate constructs that are seized and released by Entities and they have no ability to make decisions on their own. They are strictly acted on by Entities. However, in Simio we expanded the concept to allow any object in the system (including Entities) to be viewed as a Resource by other objects in the system. Resource objects can have behaviors, travel around the system, and make decisions such as rejecting a seize request.

The equivalence of an Entity, Resource, Agent, Object, and Model is an evolving and powerful construct that would not have fit nicely into a previously standardized framework. Although it is true that other fields have been more successful in standardizing frameworks, I think standardization of simulation models is far more challenging because of the complex behaviors that we attempt to model. This is particularly the case when we are modeling human behaviors such as operators or physicians, and their complex decision making. The same is true with programming languages in general – although each particular dialect has a standard – there is no standard that covers all programming languages.

Perhaps the best we can do at this point is to agree on some standard definition of terms (e.g. event, state, process, object, input parameter, etc.) when describing a modeling framework, but continue to evolve our modeling frameworks without the limitations imposed by a standard framework.

Gerd Wagner: I agree that a simulation modeling standard should not handcuff the creativity of developers in advancing simulation technologies. But I think it should be possible to define a standard modeling language for the paradigm of *processing networks*, as sketched on the web page https://sim4edu.com/pn-models, without preventing creative extensions of all sorts. This is what we can observe with UML modeling tools. They are all based on the UML standard, but different tools from different vendors compete with each other by offering different user interfaces and different extensions. Having a standard for the core modeling concepts/language gives customers some protection against vendor lock-in by allowing them to export their models from vendor A's tool and import them into vendor B's tool.

It was a conceptualization mistake (made in many simulation and workflow management tools) to strictly distinguish between Entity and Resource, instead of conceptualizing entities (or work items) as objects and resources as special roles played by objects at certain times in the context of activities. Such a distinction between object types and roles is well-known in information systems modeling.

Dennis Pegden: I think we all agree that it was a conceptualization mistake to view Entities and Resources as different constructs. It's more powerful to think of any object as being able to take on the role of a Resource, and to have decision making ability in that role. Had a standard been written before this conceptual error was corrected this mistake would be part of the standard. The question is do we think we have the framework perfected to the point where a standard can be written. I suspect that we do not.

Perhaps we are ready to standardize some aspects of modeling, but not others. For example, event modeling concepts are well defined, and have not changed in many years. In fact, Schruben's event graphs to some extent provide a standard way to think about event modeling. However, things are a bit more open-ended when it comes to process and object-based modeling approaches. Here things are still evolving, and perhaps it's a bit early for standardizing the framework. An important start would be to agree on the basic terminology for describing these frameworks.

Levent Yilmaz: The comments made by Dennis and Gerd highlight the tradeoffs between over standardization and the flexibility needed to conduct studies based on assumptions that may not be foreseeable in advance. The use of standardized and complex federated campaign models in the DoD context had proved to result in monolithic models that are extremely narrow, incomprehensible, and insensitive to analysis under uncertainty. The standardization was to a large extent achieved by suppressing uncertainty; yet, dealing with uncertainty (both parametric and structural) was extremely critical in dealing with analytical problems. I think we can avoid the fallacies of over standardization by focusing in developing abilities to bring together modeling concepts appropriately for a given context and objective.

During the standardization process, there are often strong pressures and incentives to agree on and hold the parameters of component models as well as relations among them fixed. Yet, this approach is often a recipe for failure when modeling complex socio-cultural phenomena. The notion of objects playing multiple roles, as raised by both Gerd and Dennis, appear to fit here quite nicely. Variability management, role-driven modeling, and the ability to develop models that can be customized to adapt to a new context are indeed essential. The provision of multiple models that allow disagreement, competition, and uncertainty may be a better strategy for dealing with wicked problems in analyzing complex phenomena. In

the past decade, there has been significant progress in the theory and methodology of programming language development, including metaprogramming, context-oriented programming, multi-paradigm languages, aspect-oriented modeling, and feature-oriented languages that together may help avoid the need for over standardization by managing flexibility in a tractable manner.

Gerd Wagner: Thanks to all of you for participating in this discussion. We must stop at this point. I think we have addressed many important issues. But the discussion must go on...

Copyright Information