Proceedings of the 2000 Winter Simulation Conference
J. A. Joines, R. R. Barton, K. Kang, and P. A. Fishwick, eds.

MODEL COMPOSABILITY AS A RESEARCH INVESTMENT: RESPONSES TO THE FEATURED PAPER

Paul C. Davis

RAND Graduate School of Policy Studies
1700 Main Street
Santa Monica, CA 90407-2138, U.S.A.

Paul A. Fishwick

University of Florida
CISE Department, PO. Box 116120
Gainesville, FL 32611, U.S.A.

C. Michael Overstreet

Computer Science Department
Old Dominion University
Norfolk, VA 23529-0162, U.S.A.

C. Dennis Pegden

Rockwell Software, Inc.
504 Beaver Street
Sewickley, PA 15143, U.S.A.

ABSTRACT

Responses to the featured paper are provided by four authors who represent different elements of the simulation research community: industry, private research laboratory, and university. As is evident from the reactions given, these perspective provide both shared and distinct observations on model composability as an opportunity for research investment.

1 COMMENTS ON THE KASPUTIS-NG PAPER (PAUL C. DAVIS)

I applaud heartily the main thrusts of the Kasputis-Ng paper. My observations add ammunition to the authors' general arguments, question various particulars, and offer alternatives.

1.1 Purposes of Composability

The authors argue well for composable systems. I would add an important argument. A high-priority issue in the "transformation" of the U.S. military for the demands of future battlefields is developing highly composable military *operations*. A future CINC should be able quickly to construct a plan suited to details of the at-the-time situation, which might include, for example: needing quickly to: deploy with minimal warning; conduct forced-entry; suppress and destroy air defenses and weapons of mass destruction; and magnify the effectiveness of allied forces. It might not have been clear in advance what capabilities would be most effective, nor even where C^2 functions would best be located (e.g., ashore, on ship, or on aircraft). Moving to such an adaptive system for joint operations will be no mean feat (Davis, Gompert, and Kugler 1996). A principal enabler will be simulation—for conceiving, exploring, constructing, training, and testing at all levels of hierarchy. That simulation will need many of the features called for by Kasputis and Ng.

1.2 Refining the Paradigm

Where I perhaps take issue with the paper relates more to details. It seems from my reading of draft material that the authors' vision is a system with so extensive a library of compatible modules and machinery for assembly as to allow automated system construction in any circumstance. My preferred vision emphasizes the man-machine interface. This anticipates that it will *usually* be necessary to do a fair amount of at-the-time tailoring and stitching together. Just as a craftsman may chip away at bricks to accommodate odd corners or decorations (i.e., the building blocks are never exhaustive), so also will future simulators need to add to their models and create ad hoc interfaces. An important measure of effectiveness in this paradigm is the number of expert-days needed to assemble the simulation needed. The goal then becomes efficient man-machine work, rather than automation. This affects priorities and emphasis.

1.3 Cautions About the Composability Concept

The authors have in mind nothing of the sort I fear in this regard, but their imagery of a standard library of fully tested modules and eliminating redundancy needs an attached warning notice. Empirically, bureaucracies love

standardization and "accepted" models, but appear to care much less about true validity—much less a rich competition of ideas. Some recent aspects of DoD's approach to modeling and simulation have been compared to Soviet-style Central Planning.

This is not merely academic: the integrated composable system the authors seek cannot be achieved in a fully portable form. Instead, based on my experience with RAND's RSAS and JICM systems, and discussions with scientists at Los Alamos National Laboratories regarding these matters, it seems to me that the compatibility of building blocks typically depends on their having been developed with certain low-level features of modeling and computing environment frozen. This may relate to operating-system issues, language, or, for example, atomic characterization of terrain information. There are also many semantic subtleties that can reasonably be controlled only within a walled system (e.g., recall the story about how different Services interpret the mission to "secure that building"). This leads me to suggest that the composable systems that we need should probably be conceived and defined at a specification-language level. This would include a fair amount of free-form text, picture-drawing, and example-giving; the ideas could then be quickly transported to different environments. In contrast, if the composable system were tied to a particular system—and especially if access to that system and its data were tightly controlled—the barriers to innovation and rethinking could be high.

By "specification-language level," I don't necessarily mean a specific specification language as discussed in earlier years. Indeed, higher-level modeling environments make it possible to design and implement simultaneously, and to build good documentation along the way. One example of this is the Analytica system, which my colleagues and I have been using heavily in the last few years. I believe it to be the case that it is much easier for someone to receive, *comprehend*, and recode for his own environment a model built properly in such a system—than one written in C++ with usual documentation. The recipient has data dictionaries, data-flow diagrams, and algorithms in a high-level language to help. Even with differences in convention from one high-level depiction to another, the time spent comprehending and then recoding could be trivial in comparison with the time required to think out the problem and design an appropriate model. Thus, I caution against equating reusability of concepts and models with off-the-shelf reusability of computer code. Both have their place. On the one hand, the staff of the CINC's I mentioned at the outset would want to be working within a standardized and well-controlled environment. So also would many study and analysis organizations, or weapon-system developers. But they would often prefer their own environments, not a single standard system imposed upon them that lacks some of the special features they need. The community as a whole needs such flexibility to work in their own environments. While some standards are enabling and liberating (e.g., the original internet standards and the HLA), more detailed standards can be quite dangerous to good work and innovation.

1.4 Additional Issues in Composable Systems

I would like to mention one special challenge in composability. That is the need to address seriously—in the structuring of model components and the development of databases—the massive and ubiquitous uncertainty that *often* exists in problems of strategy and tactics. In my own work, I have emphasized *exploratory analysis* as a fundamental concept for dealing with uncertainty. The variable- or multi-resolution modeling that the authors mention is, in my view, an essential element of being able to do exploratory analysis well. Many related principles are emerging, as well as suggestions for enabling tools (Davis 2000).

A difficult but crucial aspect of working with multiple levels and choices of abstraction is the need to use all the information available—at all levels and choices of detail—to develop *mutually* consistent families of models (Davis and Bigelow 1998). Few tools exist for this and most that do exist are biased more toward statistical fits than phenomenological modeling. We should anticipate the need for at-the-time tailoring, because abstractions (and disaggregation schemes) are simply not generally valid. An essential part of this tailoring will be a mutual calibration process using information, including information about uncertainty, suitable to the specific context. Proceduralizing such work will require great strides in theory, tools, and education.

1.5 Final Comments

To conclude, I found a great deal to applaud in the Kasputis-Ng paper and hope that my comments add ammunition to their own arguments, as well as posing even more challenges consistent with their aim.

2 A RESPONSE TO "COMPOSABLE SIMLUATIONS": FIVE KEY ISSUES (PAUL A. FISHWICK)

Kasputis and Ng have chosen a very broad and interesting topic to address: composability within simulation. The article takes the approach of an elicitation of requirements and issues connected with composable simulations. My response will be based on five areas that cut across their issues, by expanding on some of the issues they present. There are not as many answers as I like, since this topic

covers a vast territory. However, with this paper and responses, we can try to address the major issues.

2.1 Representation

We need to address the type of representations that we will accept for components, which will serve the composability requirement. There are roughly two philosophical approaches to making representations: universal versus plethora. Either you believe that one type of representation needs to be used, or that a plethora of representations are acceptable. There are simulationists on both sides of this argument. While theories based on a systems view provide an excellent foundation for understanding the techniques of composability, the simulation community requires multiple representations since no one group is satisfied with a singular view. The representation will likely be of an atomic form that has a clearly defined input/output interface. That much, we can probably all agree on. We need to foster model-based thinking when creating component networks. To those with a modeling bent, this may seem like a foregone conclusion, but many component makers view components as units of source code, say, in Java or C++. It is not enough to work at this level. We need to rise above source code and embrace graphically oriented metaphors that can be defined at multiple abstraction levels (Fishwick 1991, 1995). Even in most University-level introductory courses on "programming," we still do not instill the importance of modeling into a student's mental framework of what it means to do computation via composition, and yet, many good theories and modeling frameworks have been around for at least two decades. Thus, there is also an educational barrier that we need to cross to make components and composability part of the mainstream.

2.2 Requirements

The authors are correct to spend time on requirements. Without knowing the requirements for the simulation, we are hard-pressed to invent compositions. The problem is that a lot more research and tool development needs to take place to translate requirements, which will most likely be in natural language form, into a set of components to do the job. If requirements can be specified in "model form" then this translation step may not be necessary. A lot of researchers, like me, just ignore this problem, not because it is trivial, but because it is extremely difficult. But we need to place heavier emphasis upon it. Some quantitative requirements, such as elapsed time (Lee and Fishwick 1999), do have partial solutions but the larger problem needs further elaboration (perhaps within another response).

2.3 Marketplace

If composable simulations are to ever take flight, there needs to be a place where people buy and sell components. Otherwise, they will not grow in number, or be accepted. Even though DoD mandates do not always produce the desired response (i.e., the adoption of Ada), DoD has a crucial role to play in nurturing this marketplace. It works something like this. DoD acquires new technology in the form of hardware. Upon delivery of this hardware, DoD should require the contractor to produce a digital version of the hardware. In what form? At this point, any form is better than none, and this returns us to the issues of component representation. This may serve as a catalyst for increasing the number of models and components. The idea is that eventually, if DoD fosters the digital object requirement for acquisition, that everyone will require it eventually, even individual consumers before they agree to purchase a product. For those industries that refuse to deliver the components, competitors will be waiting in the wings.

2.4 Plug and Play

We can learn a lot from the way that electronic components hook together to compose networks and integrated installations. The components have to hook together perfectly, which means that the inputs and outputs must be of the specified data types. Having imposed geometric constraints on components will also aid the composability process since the connections are surfaced through visualization (Hopkins and Fishwick 2000). Figures 1 and 2 demonstrate the use of visualization when dealing with components. An agent-based metaphor is applied so that the model is composed of individual agents traversing paths that lead from one resource to the next. In this particular example, from (Hopkins and Fishwick 2000), a model is created to represent an Operating System (O/S) kernel.

2.5 Finding Components

The Web should be a central vehicle for discussions and implementations of composability. We use browsers for almost all our interactions, and much product development focuses on execution in the browser or, at least, through the browser. We need a component search engine, which means that we need standards for components. Tag languages such as XML may address this need since, in XML, one can express components naturally with domain-specific elements.

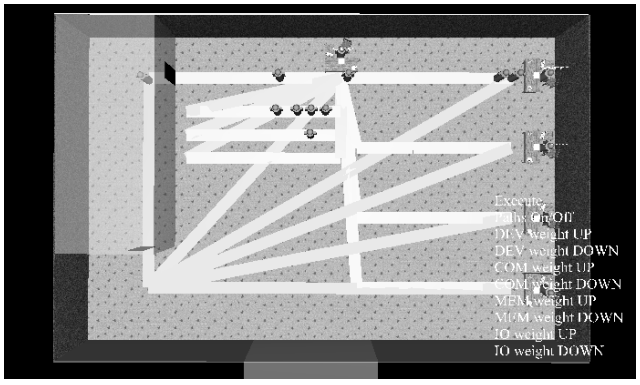

Figure 1: Top View of the O/S Task Scheduler

The CPU facility is shown at top center, with its priority queues just below and to the left of it. DEV, COM, MEM, and I/O service facilities are on shown on the right, top to bottom respectively, with queues to the left of each.

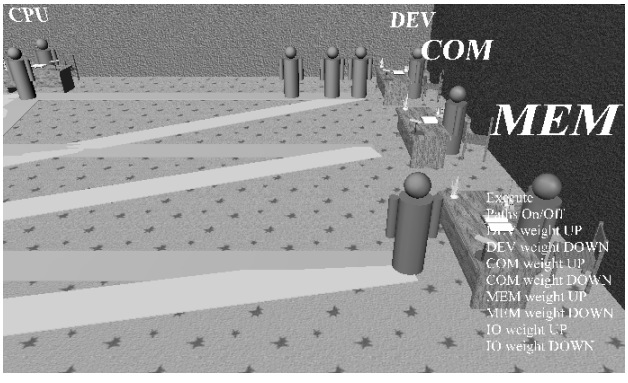

Figure 2: Close-Up View of Task Scheduler Operation

3 COMPOSABLE SIMULATION SYSTEMS: ANOTHER PHILOSOPHER'S STONE? (C. MICHEAL OVERSTREET)

Remember the Philosopher's Stone? It could be used to turn lead (or other base metals) into gold ("alchemy"). Many believed it would be useful. But neither desire nor need implies possibility.

Or perhaps a better, more recent, analogy is Newell and Simon's Generalized Problem Solver (GPS) that many in the artificial intelligence community attempted to build in the past (Newell, Shaw, and Simon 1957, 1959). You give it a problem (a set of requirements), and it uses its reasoning capabilities and store of knowledge to build a solution. If a GPS could be built, its benefits would be significant and it would be perhaps the ultimate in reusable software components. One might argue that a truly general Composable Simulation System requires something as complex as a GPS: a user describes a problem (it terms of requirements), the system then finds and builds a solution.

Kasputis and Ng provide an excellent discussion on the scope of issues related to Composable Simulation Systems. I choose to comment on just a few.

In the software development community in general and specifically in the simulation tool development community, we're fighting two fundamental truths:

- **Our desires are unbounded:** The desire for new system capabilities is never-ending. Some of these desires reflect real needs. Some real needs cannot be met with current capabilities, some will not be met in the near-term, and some will never be met. It is often impossible to tell the difference. It is often impossible to prove that a proposed task is impossible (maybe at any cost, but certainly given scope, time and budget constraints).
- **Some systems we attempt will always be just barely doable:** Systems are built when it is feasible to do so. Some systems (for the research community, the most interesting) are always just barely feasible with current technologies. Regardless of our development capabilities, some desirable systems will remain just beyond our current capabilities.

I mention this because I believe that even if we build a functional Composable Simulation System, we will still find that many desirable models are not quite feasible given time and budget constraints: we will attempt bigger models.

Other current realities (in my view) also make the creation of effective, general-purpose composable simulation systems impossible. These observations are strongly influenced by experiences in working with the U.S. Department of Defense's ModSAF, now OneSAF (ModSAF):

- Hardware improvements are the salvation and the bane of simulations. The speed improvements continue to make feasible this year that which was infeasible last year. They are a bane because these speed improvements make last year's models obsolete. ModSAF, and the level of detail modeled in each participating entity, would not have been attempted without the hardware improvements and costs reductions of the past decade.
- For many uses of simulation, particularly within the U.S. Department of Defense, the simulation system would be a virtual reality (VR) system if technology would allow it. Assuming that the hardware improvement trend of the past 5 decades continues, this implies that most models built today will be obsolete tomorrow. Again, it is

primarily my experiences with ModSAF that lead me to this conclusion.

This merging of simulations and virtual reality systems is bothersome. It seems to make traditional approaches to building effective simulations ineffective in some fundamental ways. In the past, the essence of simulation was modeling: simplify reality by identifying abstractions (for example, queueing system components) in real situations. Models were built by combining these widely recognized and frequently reoccurring abstractions. In addition, the recognition that some abstract components occur frequently in many models allows the construction of simulation languages (like GPSS and scores of more recent, graphics-based derivatives) to be used successfully to build many models in widely different application domains.

Virtual reality requires something different: A component's contribution to the system characteristics of interest may ultimately be that of an M/M/1 queue, but for VR, different M/M/1 queues will likely require different physical representations. And depending on the capabilities of future VR technologies, perhaps require different smells, textures, radiate different amounts of heat, or vibrate at different frequencies when touched. The effort required to build reusable components is significantly different in the VR case. Assuming an adequately parameterized component exists that is valid for this use and can be located, the effort and knowledge required to get all of the specification details right is significantly increased.

Remember the Philosopher's Stone? While impossible, the search for it contributed to development of chemistry as a real science. And while a generalized problem solver may never be built (the SOAR community might argue this (Laird, Newell, and Rosenbloom 1987)), good automatic theorem provers and expert systems were, although their effective scope is much narrower than initially envisioned. Likewise, the quest for a system for building new simulations out of old will make significant contributions to simulation science even if it the ultimate vision gets redefined along the way.

4 PURPOSE-BUILT VERSUS GENERIC MODELING (C. DENNIS PEGDEN)

Since the beginning days of simulation, the conventional wisdom has been that a successful simulation begins with a clear statement of the purpose of the model. This point is hammered home in nearly every introductory textbook on simulation methodology. One begins with a statement of the purpose and develops a model to meet that purpose. This statement of purpose includes the specific questions that need to be answered (e.g., predict daily production capacity) and the accuracy required (e.g., within 5%). The stated purpose then drives the level of detail (and the amount of work) that is put into the model.

A good example of the importance of model purpose on model detail is the application of simulation to predict or compare performance. There is a significant difference in the level of detail required in the model based on whether the model is being used to predict performance of a single system or to compare two or more systems. If we are attempting to predict the production capacity of a factory, we must include all aspects of the system that impact the capacity. This includes details such as restroom breaks, equipment breakdowns, material shortages, and so on. On the other hand, if we simply want to compare system X against system Y to pick the best, all we really care about is the difference in performance between the two systems. In this case, we can omit elements of the system that impact both systems in approximately the same way, and typically we can get by with a much simpler model. This is fortunate, in this instance, since we are modeling two different systems.

The wisdom of beginning with a clearly defined purpose and using this to drive model detail has served the modeling community extremely well and is a key factor in the success of simulation applications. Many novice users who fail in their first big simulation project do so because they do not take this advice to heart. They start off building a model with no clear purpose in mind or no understanding of the questions they want to answer. Their models have no basis for establishing the level of detail required in the model. Often the level of detail within the model varies dramatically based on either the sophistication of the modeling tool or the application knowledge of the modeler. The parts of the system that are well understood and have strong tool support are modeled in detail, and other elements of the model are captured in less detail.

The ideal model is one that will answer the questions within the desired level of accuracy with minimal effort. In many cases, a very simple model is all that is needed; in other cases, a very detailed model is required. The objective of the simulation user is to decide on the minimal level of detail that is adequate to answer the questions posed by the study.

Note that a model with greater fidelity than needed for the purpose of the study is not only more costly to build, but is also undesirable. An overly detailed model is difficult to verify, expensive to modify/maintain, and most importantly, takes significantly longer to run, a factor that generally leads to fewer replications. The gains achieved from the improved model accuracy are more than offset by decreases in statistical accuracy resulting from fewer replications of the model.

As we look to the future in simulation, one of the promising ideas is the concept of having pre-built models or model components that can be plugged together to form a model of our system. The idea is that we simply select

these components from a library and use them directly. For example, we might build a model of our entire supply chain by simply connecting together pre-built, generic models of our plants, distribution centers, and transportation centers. The goal is to build each model component once, verify its operation, and then make it available in a library to be used in many different applications.

There are some significant problems that must be addressed to create a framework that supports the idea of composing models from pre-built, generic models/components. One critical issue is model fidelity.

To make this concept work, we need to rethink completely the concept of a purpose-built model. Our generic model components must be built without knowing the specific questions that they will be used to answer. How do we decide on the level of detail to incorporate into these generic models? If we build a highly detailed model of our plant, then it will be useful for accurately predicting our plant system performance, but much too detailed for incorporation into an enterprise-wide supply chain model. On the other hand, if we build a rough-cut capacity model of our plant, it will be useful in our enterprise-wide supply chain model, but useless for predicting detailed plant system performance.

The challenge is to build model components that have multiple levels of fidelity that can be changed by the user based on the purpose of the model. The generic model must include high-level representations as well as detailed representations of the same system. When a model or model component is selected, the user specifies the level of detail required, which causes the appropriate model representation to be used.

To accomplish this, we must anticipate and accommodate a wide range of questions that might be asked using the generic model that we are building. Our task is shifted from developing a single purpose-built model to one of building a generic model that is multi-purpose built.

The basic idea of composing large models from pre-built components is compelling. However, there are some significant issues to address to make this work in practice. One of these is the challenge of supporting multi-purpose models.

REFERENCES

"alchemy," *Encyclopedia Britannica Online,* <http://search.eb.com>, referenced July 6, 2000.

Davis, P.K., and J.H. Bigelow. 1998. *Experiments in Multiresolution Modeling*. RAND, MR 1004, Santa Monica, CA.

Davis, P.K. 2000, Exploratory Analysis Enabled by Multiresolution Modeling. In *Proceedings of the Winter Simulation Conference 2000*.

Davis, P.K., D. Gompert, and R. Kugler. 1996. *Adaptiveness in National Defense: the Basis for a New Framework,* RAND Issue Paper IP 155, Santa Monica, CA.

Fishwick, P. 1991. Heterogeneous Decomposition and Inter-Level Coupling for Combined Modeling. In *Proceedings of the 1991 Winter Simulation Conference*.

Fishwick, P. 1995. *Simulation Model Design and Execution: Building Digital Worlds*, Prentice Hall.

Hopkins, J. and P. Fishwick. 2000. Synthetic Human Agents for Modeling and Simulation. In *Proceedings of the IEEE,* submitted for publication, May 2000.

Laird, J.E., A. Newell and P.S. Rosenbloom. 1987. SOAR: An Architecture for General Intelligence. *Artificial Intelligence,* 33: 1-64.

Lee, K. and P. Fishwick. 1999. OOPM/RT: A Multimodeling Methodology for Real-Time Simulation, *ACM Transactions on Modeling and Computer Simulation*, 9(2), April 1999, pp. 141-170.

ModSAF, <www.modsaf.org>, referenced July 6, 2000.

Newell, A., J. C. Shaw, and H. A. Simon. 1957. *Preliminary Description of a Generalized Problem Solver.* Pittsburg: Carnegie Institute of Technology, CIP Working Paper No. 7, 1957.

Newell, A., J.C. Shaw and H. A. Simon. 1959. *Report on a General Problem Solving Program,* RAND Corporation, Report No. P1584

AUTHOR BIOGRAPHIES

PAUL K. DAVIS is a senior scientist at RAND and a professor in the RAND Graduate School of Policy Studies. He has published extensively on defense planning, analysis, modeling, and simulation. In the 1980s he led development of the RAND Strategy Assessment System (RSAS), a large-scale analytical war gaming system with a mix of combat models and agents representing political and military leaders. He was the principal author of a 1997 National Research Council 1997 book on Modeling and Simulation. Dr. Davis received a B.S. from the U. of Michigan and a Ph.D. in chemical physics from MIT. Before moving to RAND in 1981, he was a senior executive in the office of the Secretary of Defense. Dr. Davis may be reached via email at <pdavis@rand.org> and his website is located at <www.rand.org/personal/pdavis>.

PAUL A. FISHWICK is Professor of Computer and Information Science and Engineering at the University of Florida. He received the PhD in Computer and Information Science from the University of Pennsylvania. He also has six years of industrial/government production and research experience working at Newport News Shipbuilding and Dry Dock Co. (doing CAD/CAM parts definition research)

and at NASA Langley Research Center (studying engineering data base models for structural engineering). His research interests are in computer simulation modeling and analysis methods for complex systems. He is a senior member of the IEEE and a Fellow of the Society for Computer Simulation. Dr. Fishwick founded the comp.simulation Internet news group (Simulation Digest) in 1987, which now serves over 20,000 subscribers. He has chaired workshops and conferences in the area of computer simulation, and will serve as General Chair of the 2000 Winter Simulation Conference. He was chairman of the IEEE Computer Society technical committee on simulation (TCSIM) for two years (1988-1990). He has published over 40 journal articles, written one textbook, co-edited two Springer Verlag volumes in simulation, and published six book chapters. Dr. Fishwick's WWW home page is <www.cise.ufl.edu/~fishwick> and his E-mail address is <fishwick@cise.ufl.edu>.

C. MICHAEL OVERSTREET is an Associate Professor of Computer Science at Old Dominion University. He received his B.S for the University of Tennessee, an M.S: for Idaho State University, and an M.S. and Ph.D. from Virginia Polytechnic Institute and State University. His current research interests include model specification and analysis, static code analysis and support of interactive distance instruction.

C. DENNIS PEGDEN received his bachelors in Aeronautics, Astronautics, and Engineering Sciences from Purdue University in 1970. He worked in the aerospace industry at the National Aeronautics and Space Administration and the Matrix Corporation. He returned to Purdue in 1973 and received his Ph.D. in mathematical optimization from the Industrial Engineering Department in 1976. After graduation, he taught at the University of Alabama in Huntsville where he began his work in simulation and led in the development of the SLAM simulation language. In 1979, he joined the faculty at the Pennsylvania State University where he completed the development of the SIMAN simulation language. He is currently Director of Development of Rockwell Software, Inc., which markets SIMAN and Arena simulation products; the Tempo scheduling product; and vertical market products in the areas of call centers, business processing, manufacturing, high-speed processing, and real-time control.